Introduction to CRC Cards

by David M. Rubin

Access the Word Document version, or the Acrobat PDF version.

Softstar Research, Inc. - Methodologies & Practices

Table of Contents

Introduction

CRC (Class-Responsibility-Collaborator) Card Modeling is a simple yet powerful object-oriented analysis technique. CRC modeling often includes the users, analysts, and developers in the modeling and design process, bringing together the entire development team to form a common understanding of an OO development project. It is one of many tools that should be used in the collaborative design of a system.

Kent Beck and Ward Cunningham first introduced CRC cards at OOPSLA ’89 in their paper "A Laboratory for Teaching Object-Oriented Thinking". Originally their purpose was to teach programmers the object-oriented paradigm. The technique has since been refined to become valuable beyond the education curriculum.

A CRC Model is a collection of cards (usually standard index cards or larger) that are divided into three sections.

1. Class
2. Responsibility
3. Collaborator

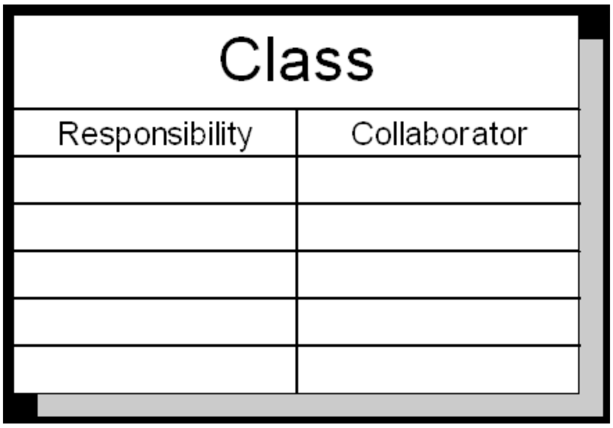


Figure 1: CRC Card Layout

The back of the CRC card is often used for a more detailed description of the class. Along with any other notes captured during the CRC session. Many times these include the actual attributes of the class.

Class

A Class represents a collection of similar objects. Objects are things of interest in the system being modeled. They can be a person, place, thing, or any other concept important to the system at hand. The Class name appears across the top of the CRC card.

Responsibility

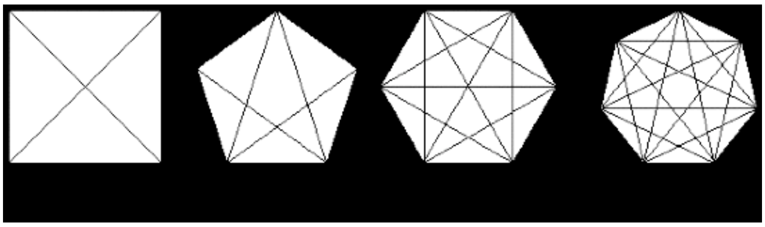
A Responsibility is anything that the class knows or does. These responsibilities are things that the class has knowledge about itself, or things the class can do with the knowledge it has. For example, a person class might have knowledge (and responsibility) for its name, address, and phone number. In another example an automobile class might have knowledge of its size, its number of doors, or it might be able to do things like stop and go. The Responsibilities of a class appear along the left side of the CRC card.

Collaborator

A Collaborator is another class that is used to get information for, or perform actions for the class at hand. It often works with a particular class to complete a step (or steps) in a scenario. The Collaborators of a class appear along the right side of the CRC card.

The CRC Team

The CRC team is comprised of different types of people involved in the project. The overall size of the CRC team is important, and active participants should be limited to 6. Interaction between active participants grows experientially with the number candidates. The figures below show the lines of communications between 4, 5, 6, and 7 participants.



There are generally three types of skill sets for active participants in the CRC session.

1. Domain Users
2. OO Design Analyst
3. Facilitator

Additional non-active participants can include a scribe and observers.

Domain Users (3-5)

These members of the CRC team are the users of the system being designed. They should have a good amount of business domain knowledge for which the system is being modeled.

Good domain users also have the following characteristics:

- Know the business being modeled.
- Think logically.
- Good communication skills.
- Are willing to invest the time in systems designs.

OO Design Analyst (1-2)

These members of the CRC team are analysts or developers familiar with OO methodologies and techniques. Generally the design analysts have responsibilities on the project well beyond the CRC modeling sessions. These may include fleshing out the CRC cards to create a formal object model, and interpreting the CRC collaborators to document the OID's (Object Interaction Diagrams).

Good Design analyst have the following characteristics:

- Understand the CRC modeling process and methodology.

- Understand the OO modeling process and methodology.

- Experience developing OO systems.

Facilitator (1)

This is the member who runs the CRC session. This person is perhaps the most important member of the team. It is the facilitator’s responsibility to keep the CRC session progressing forward. This may include acting as an intermediary when collaboration debates occur, and generally making sure that the session is on track with regards to the agenda.

Good facilitators have the following characteristics:

- Good meeting skills.
- Understand the CRC modeling process and methodology.
- Understand the OO modeling process and methodology.

Scribe (1-2)

This is the member responsible for documenting any business logic and discussion that isn’t captured on the CRC Cards themselves. Often analysis logic on why a process is the way it is discovered within a CRC session, and should be captured and documented. This documentation is often rolled back into the requirements and business case documents, as well as used by the design analysts to further the systems object model.

Good scribes have the following characteristics:

- Listen extremely well.
- Good written communications skills
- Can identify business logic

Observers

These are members who aren’t directly participating in the CRC session. They are generally other users of the system, or other project team members. It is important that the facilitator ensure that these people are not active during the CRC session.

Good observers have the following characteristics:

- Know how to contain opinions during the modeling session, i.e. shut-up.

The CRC Environment

Room

A meeting room is normally a good place for a CRC session. There should be ample room that all participants can interact with each other equally. Extra room should be considered depending on the number of observers of the CRC session.

Equipment

- CRC Cards, either normal 3x5 index cards or larger 5x7 index cards.
- Writing instruments normally dark markers.
- Overhead (Optional)
- White board (Optional)
- Paper for the scribe.
- Flip chart paper
- Tape
- Refreshments.

The CRC Session

Organizing the Room

Normally participants of a CRC modeling session can be position around a conference room table. This allows open interaction between all members. Observers of the session should remain around the edges of the room.

Explaining the CRC Technique

The facilitator should begin by explaining the purpose of the CRC session. Remember that scenario execution is at the heart of the CRC session. An explanation of the CRC cards, as well as some sample CRC cards themselves can be created.

The concept of role playing techniques should be covered. Role playing involves acting out what a class does in order to facilitate the scenario execution. A person who owns a particular card should 'become' that class when control of the scenario is passed to them. This is often done by holding the card in the air while the class has control of the execution.

This way, a card is an object while being held in the air, and a card is a class while resting on the table.

Selecting a Scenario

One scenario should be selected to begin the CRC session. Often an initial brainstorming time can be set aside at the beginning of the session to discuss and identify an appropriate scenario. The scenario chosen should be well documented and very specific. Any related scenario's should be modeled separately. Often it is desirable to choose an easy scenario to begin with.

If there are already use-case's diagrams or descriptions for the problem domain, a good scenario is any 'normal coarse' through one of these 'use cases'. Using just the 'normal coarse' ensures that the session focuses on one particular scenario, and not the entire use case. It's important to emphasize using only the 'normal coarse' through the use case, and not concentrating on any 'alternative path's'. It will be up to the facilitator to ensure the group doesn't diverge into alternative paths through a use case.

Also during this brainstorming time an initial set of classes can be identified (see Finding Classes on page * for further information).

Creating Initial CRC Card(s)

Initial CRC cards can be created for any class identified during the scenario selection. CRC cards are also created for existing classes in the system. Often these classes are the result of an existing legacy system. Further brainstorming is an option depending on the experience level of the CRC session participants.).

Arranging the CRC Cards

As CRC cards are created, they are often placed on a centralized table for all participants to see. Once collaborations are established between classes, the arrangement of the classes can be moved so that collaborating classes are in close proximity to each other. The more that two classes collaborate throughout the session, the closes they should be placed to each other.

In the initial stages of the CRC session, you can expect that the location of the classes on the table move quite frequently. This is typical of card placement, and will settle down after collaboration become more established.

It is often helpful to put the busy cards towards the center of the table, and those less busy cards around the outside perimeter. While the importance of card placement is subtle, beware of just choosing a card location because there is empty space on the table. If the table fills up quickly during a scenario execution, the quite possibly the scenario chosen is overly complicated, or is not the primary coarse through a 'use case'.

Miscellaneous Techniques

The CRC Interview

It is often useful to start the session using the interview technique, or new employee technique. The interview technique involves identifying a scenario, then identifying a user that asks a 'starting' class to perform the scenario. The class in turns asks questions (interviews) the requester (user) to get enough information to perform the task. The user can respond with the information, or respond that the class should know that information. The class can create a responsibility, or collaborate with other classes to get the information it requires.

For example, in a billing application, the scenario might call for an invoice to be created for a customer account. The session would begin by a user asking the system to create an invoice. This immediately calls for a class that the user interacts with class [1]. Two classes come to mind, an invoice scheduler and an invoice itself. Questions that the facilitator might ask are what would happen in the real world. Would the user go to an invoice and ask it to create itself, or would the user go somewhere else and ask that an invoice be created.

Next the class selected, say the Invoice scheduler, would ask questions to get the information it required to perform the task. For example, what invoice, what account, what period of time, where are the charges, what is the tax, what is the account address, what are the previous payments for the account, etc..

As you can see, to execute the scenario, much information and action is needed. The starting class either has the information or performs the action (responsibility), or gets the information from another class, or has another class perform some action (collaborator).

[1]. Normally the user would interact with a n interface class, that would then interact with a business class, however for the sake of this session, we choose to ignore specific interface classes and model them using prototyping techniques.

Finding Classes

A Class represents a collection of similar objects. Objects are things of interest in the system being modeled. They can be a person, place, thing, or any other concept important to the system at hand.

There are many ways to identify classes. One of the easiest to start with is noun extraction. Noun extraction identifies all of the Nouns in a problem statement and/or use-case scenario. The nouns extracted make excellent candidate classes.

Other ways to identify classes are to look for items that interact with the system, or things that are part of the system. Ask if there is a customer of the system, and identify what the customer interacts with. Screens and reports represent interface classes, but for the sake of a CRC session, a single GUI class can represent these.

If a class can’t be named with less than three words, then it’s probably not a class but a responsibility of another class.

Finding Responsibilities

A Responsibility is anything that the class knows or does. These responsibilities are things that the class has knowledge about itself, or things the class can do with the knowledge it has.

To compliment the 'Noun Extraction' technique above is verb extraction. Verb extraction identifies all of the verbs in a problem statement and/or use-case scenario. These are usually good indicators of actions that must be performed by the classes of the system.

Other techniques included asking what the class knows, and what information must be stored about the class to make it unique. Ask what the class does, and then flesh out how the class might do it.

Finding Collaborators

Collaboration occurs when a class needs information that it doesn’t have. Classes know specific things about themselves. Very often to perform a task a class needs information that it doesn't have. Often it's necessary to get this information from another class, in the for of a collaboration.

Collaboration can also occur if a class needs to modify information that it doesn’t have. One property of information that a class knows about itself is the ability to update the information. Often a class will want to update information that it doesn't have. When this happens, the class will often ask another class, in the form of a collaboration, to update the information for it.

Generally for a collaboration to occur, one class is the initiator. In other words, there has to be a starting point for a collaboration. Often times the initiating class is doing very little work beyond initialing the collaboration itself. However, be careful that a class doesn't just 'pass the buck', just to have it passed again. For example, if class A collaborates with class B, just to have class B pass the work to class C, consider eliminating the collaboration with class B.

Facilitation Tips

CRC Definitions

It is often helpful to write CRC definitions on flip chart paper and post them in plain view around the room. These definition can include an example of the CRC card, both blank and a completed example.

Scenario Execution

Always go back to the original scenario being modeled when questions arise. The entire CRC session should always focus on the scenario being executed.

OO Terminology

Keep the technical OO terminology to a minimum. Always talk in terms of the users domain and business knowledge.

Low Tech

Always keep the CRC session low-tech (i.e. Index cards, flip chart paper, etc.). Using low-tech techniques is one of the success factors of CRC sessions.

Keep it Fun

Keep the atmosphere fun and playful. This is one of the reasons for using role playing techniques.

Set Expectations

New users to CRC modeling should not expect immediate results. As sessions continue, they become easier. Expect the first few classes and collaborations to be followed by lots of discussion. It’s up to the facilitator to regulate this efficiently.

Get Management Support

CRC Modeling somewhat moves the analysis away from the developers and into the hands of the users. Without management support, the users will maintain other responsibilities during the CRC session. You'll want to block off enough time so that the sessions aren't interrupted with other activities. Some of the most successful CRC sessions are held off-site, away from the users office.

Glossary

Abstract Class	A class that is never instantiated into an object.
Attribute	An interesting property of a class. Usually in the form of a value.
Class	A representation of a collection of similar objects. Classes are often objects, representing a person, place, or thing.
Collaborator	A class that provides information or performs an action for another class.
Object	An instance of a particular class. While a class may represent people, an object represents one particular person.
Responsibility	Anything a class knows or does.
Candidate Class	A class that may become a permanent class in the system after further analysis.

Send mail to Info@Softstar-Inc.com with questions and/or comments.