VICTORIA UNIVERSITY OF WELLINGTON
*Te Whare Wananga o te Upoko o te Ika a Maui*

School of Mathematical and Computing Sciences
Computer Science

# Reflections on CRC Cards for OO Design

Robert Biddle, James Noble, Ewan Tempero

Technical Report CS-TR-01/9
August 2001

School of Mathematical and Computing Sciences
Victoria University
PO Box 600, Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Email: Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research

VICTORIA UNIVERSITY OF WELLINGTON
*Te Whare Wananga o te Upoko o te Ika a Maui*

School of Mathematical and Computing Sciences
Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341, Fax: +64 4 463 5045
Email: Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research


# Reflections on CRC Cards for OO Design


Robert Biddle, James Noble, Ewan Tempero



Technical Report CS-TR-01/9
August 2001



**Abstract**

We recently had the opportunity to introduce object-oriented design to a number of teams, and used CRC cards as one of the key techniques. The teams members had varied backgrounds, and we had the opportunity to observe many teams tackle the same design exercises. This allowed us the opportunity to observe the effectiveness of the CRC cards, and reflect on the strengths and weaknesses. This paper documents our observations and reflections, and presents our advice on the strengths of the technique, and strategies we found useful for addressing the weaknesses.

**Author Information**

The authors are lecturers in Computer Science at Victoria University of Wellington, New Zealand.

# Reflections on CRC Cards for OO Design

Robert Biddle, James Noble, Ewan Tempero
School of Mathematical and Computing Sciences
Victoria University of Wellington
Wellington, New Zealand
(robert,kjx,ewan)@mcs.vuw.ac.nz

## ABSTRACT

We recently had the opportunity to introduce object-oriented design to a number of teams, and used CRC cards as one of the key techniques. The teams members had varied backgrounds, and we had the opportunity to observe many teams tackle the same design exercises. This allowed us the opportunity to observe the effectiveness of the CRC cards, and reflect on the strengths and weaknesses. This paper documents our observations and reflections, and presents our advice on the strengths of the technique, and strategies we found useful for addressing the weaknesses.

## 1. INTRODUCTION

CRC cards form the basis of one of the most venerable practical techniques for facilitating object-oriented design. We have used CRC cards for some time, and felt confident CRC cards had significant advantages. We recently had the experience in introducing the technique to a number of diverse teams in a short period of time, giving us the opportunity to study the strengths and weaknesses of the approach. This paper documents our observations and reflections on this experience.

CRC stands for Class, Responsibilities, and Collaborators, and each of these are recorded on index cards which are then used to facilitate team-based roleplay. The technique was described in Beck and Cunningham's 1989 paper [1] as a way to help beginners learn object-oriented design. More recently, the technique has become regarded as useful beyond the learning stage because of the way it subtlety supports critical characteristics of design [12, 2]. Surveys such as the *Open Toolbox of Techniques* [8] list CRC cards as a "well tried" technique, and acknowledge applicability of CRC cards beyond simply learning.

The rest of this paper is organised as follows. In the next section, 2, we briefly describe the audiences and settings of our recent work in introducing CRC cards. Then in section 3 we document our observations, and discuss some adjustments we made to the technique, and their consequent effect. Finally in section 4 we present our conclusions.

## 2. OUR RECENT EXPERIENCE

We have both used and introduced the CRC card technique for a long time, but last year we found ourselves introducing it on a larger scale we had before. In particular, we introduced the technique both to students in a university course, and to professionals working in industry.

The university course had approximately 150 students, and we ran the course twice in the past eighteen months. The focus of the course was an introduction to object-oriented analysis and design for programmers, and these students had already done programming in an OO object-oriented language. The course format consisted of lectures and tutorials, both with the whole group, as well as team programming projects, all within one semester.

The industry course had approximately 15 students each, and we ran the course six times in the same eighteen month period we ran the university course. The focus of the course was introducing object-oriented analysis and design for general use. The audience consisted of both experienced programmers and experienced business analysts with no programming experience. The course format consisted of six half-day sessions, each comprising a lecture presentation and a practical team exercise, and leading to a team exercise to be done and reviewed at the next session.

In both the university course and the industry course, there are many topics and issues involved. In this paper, we only address object-oriented design using CRC cards. This was an important topic in both courses, and one that can leave students enlightened and empowered, or deeply frustrated.

Our general teaching approach accommodates the CRC card technique easily within our preferred "active learning" approach [5]; indeed, we also use other earlier techniques of our own that were inspired in part by CRC cards [6].

## 3. OBSERVATIONS AND ADJUSTMENTS

Our presentation of the CRC card technique was based strongly on Beck and Cunningham's paper, which we also encourage the students to read. We often also make further points made based on the two widely known books on CRC cards [12, 2], and provide references to the books for further reading. With these as starting points, we believe our presentation of the CRC technique began as fairly conventional.

Our course design allowed us to easily observe student practice in the early stages, and later allowed us to assess the effect of the technique. Throughout both university and industry courses, we tended to use the same set of example designs and design exercises. This allowed us to see many teams tackle the same exercise, to see the different designs that resulted, and to see the effect of small adjustments we made in our presentations.

We would claim that the CRC card technique has three basic strengths, and we discuss each of these below, with a commentary on our experience, noting especially any weaknesses, and strategies we found to address the weaknesses. The three strengths we see are:

- CRC cards facilitate open discussion of static structure of a system
- CRC cards have built-in heuristics that guide design, in the nature of determining the Class Name, the Responsibilities, and the Collaborators
- CRC cards facilitate open discussion of dynamic structure, the "what-if" exploration, by use of object roleplay

## 3.1 Cards, Static Structure, and Discussion

The first benefit of CRC is that having cards represent classes facilitates discussion of the static structure of the system. We did sometimes see this benefit in action, but we also found many teams reluctant to work this way.

One reason for reluctance seemed to be a desire for note-taking. In many teams, there were people who wanted to record whole arrangements in their own notes, for later reference. Some teams wanted to record arrangements as a group, and so generated several sheets of paper recording alternative arrangements. Where a whiteboard or flip-chart was present, many teams preferred to use it to record possible class arrangements instead of using the cards. In all these cases, teams responded to suggestions to use cards by suggesting they would transcribe the result to cards later.

All these motivations are perfectly understandable. Especially for beginners, it may be very important to record alternatives, either individually or as a group, for later study. Also, a whiteboard or flip-chart may be easier to see than cards on a table.

We wondered if anything we had done might have encouraged this behaviour. On possibility was that we had not made spare cards sufficiently available, but we addressed this later with little change. Another possibility is that our prior introduction of UML class diagrams had created an in initial focus on class diagrams instead of cards. We were reluctant to change this, as understanding class diagrams of sample systems is part of our initial presentation on OO systems. We did notice that people preferred to make the eventual cards neatly written, and so emphasised that messiness that fine in the early stages, and that we had plenty of spare cards. We experimented with Post-it notes, smaller cards, and loose coloured notepaper. But people still often preferred notebooks and whiteboard, and wanted to leave the cards for later.

The critical question is whether using notes instead of cards actually impairs the technique. The important factor is whether the discussion and participation is impaired. Our observations are that this can be the case, because notes and whiteboards can be more easily dominated by one person than a set of cards. We observed this several times, where the note-taker, or note-dictator, controlled the discussion and limited consideration of design alternatives. On the other hand, we also observed teams where centralised notes did not lead to this effect: discussion was still fair, and discussion flowed freely. At the same time, some teams that did use the cards also had very dominant members who controlled the card layout.

One the whole, we became sympathetic with teams wanting to use notes or whiteboards. They are more suited to later study of structure, and are easier to maintain and photocopy. Interestingly, they also make it easier to observe when one or two individuals are dominating the process. Accordingly, we allow initial use of sketch diagrams, and then suggest moving to cards, and encourage use of word-processors, drawing tools, and CASE tools be left until the design is more well determined.

## 3.2 Class, Responsibilities, Collaborators

When a rough general approach has been decided, the cards are used to record the three elements of CRC: the class name, the responsibilities, and the collaborators. In the subsections below, we discuss each of these components.

### *3.2.1 Classes*

Giving class names to cards was largely straightforward and positive. One problem that sometimes arises is that learners give names that are verb, rather than nouns, and this usually shows misunderstanding of the approach, or myopic overemphasis on some procedural element. But the naming allows this to be detected and addressed early, which is an advantage.

A more difficult problem concerns the distinction between classes and objects. We are disappointed to find that this is still a common problem for many learners, and CRC cards do not always help. The freedom of the CRC technique often means that the distinction between class and object is blurred, but this can be bad for beginners.

Especially in later roleplay, having two objects played by the same person with the same card can cause learners to loose the distinction. Our approach was to increasingly say that the cards represented objects, not classes, despite the term CRC. We found this helped in later roleplay, and also helped us reinforce the nature of a class by showing how different objects could be generalised with a class. (This also helped when we later introduced abstract classes).

A related difficulty is that many systems involve singleton classes, where there is only one object in a class. We can explain this well, but the need for separate object and class names seems a confusing irritant to beginners.

One large difficultly we experienced also involves names. As is common with any system strongly related to the real world, our examples lead easily to direct metaphorical classes and objects, where a class in the system represents a concept or artifact in the real world. For example, a library system has a "book" class. In the discussion and roleplay, we often noticed learners confusing the class or object in the system with the real world artifact.

For example, teams might start discussing how information that needed to be recorded was going to be written in the back of the book. This confusion was most problematic when the class or object had the same name as one of the actors in the system. For example, many teams invent a "librarian" object to help manage the system, and then confuse the role of this object with the role of the human librarian who may use the system. This can seem humorous, and sometimes only affects some learners in a team, and can be difficult to detect. However, we found it important to address and correct early and strongly, as the confusion otherwise leads to frustration.

#### 3.2.2 Responsibilities

Responsibility is at the core the CRC card technique, and we have found it to be a strong and successful concept. We do sell the concept fairly heavily, and explain in on several levels. For example, we do give the simple explanation that responsibilities are "things to do, and things to know", but we also explain at a higher level. In particular, we stress that responsibilities should outline *what*, rather than *how*. We feel this is in line with precepts of responsibility-driven design [13, 14], and so leads to the insulation of the interface from the details.

We also discuss how "responsibility" is a good heuristic. For example, responsibility denotes both duty and power, which helps us remember that an object should fulfill its obligations, and should have what it needs to accomplish that. Responsibility does permit delegation, and they can help: this is where collaborators come in.

Our experience with the term has been successful. It seems that a strong reason is that many people find it easy to think of occupational or business roles that remind them how responsibility works. We have often found it useful, for example, to give business examples to inspire selection of class responsibilities. We point out how workers need cogent descriptions of their responsibilities, but can be allowed some freedom in determining how to accomplish them, delegating where appropriate. This also can help show the benefits of broader responsibilities, heading to the "smarter" objects that Wirfs-Brock shows lead to better designs [15].

The emphasis on responsibility also allows us to begin by encouraging teams to consider the responsibilities of the whole system required for the system use cases. We can then consider any system design as constituting a set collaborating objects that together fulfill the same responsibilities as the whole system. In the same way, for example, a business unit may have responsibilities that will be met when responsibilities are distributed to a set of collaborating people working within the business unit.

We have found that this approach works well in connecting use cases with OO design. In particular, it works especially well using Constantine and Lockwood's "essential" use cases [7], where system responsibilities are explicitly recorded. This allows explicit decomposition of responsibilities from use cases to responsibilities of objects, so resulting in improved traceability [4].

We have also found that exercises work better when team members are familiar with some of the principles of the domain. This is hardly surprising, but working with our industry teams, we realised something new. Many of the industry teams specialised in business analysis, and especially business process. For these people, we often referred to the applicability of the object-oriented approach to business analysis and design, such as outlined by Jacobson [9].

We found it easy to motivate understanding of many principles using business process examples. Moreover, we realised that many people, even our university students, are sufficiently familiar with simple business processes to benefit from the same examples and exercises. For example, everyone is familiar with processes involved in borrowing books from a library, buying groceries in a supermarket, booking theatre tickets, and so on. Moreover, everyone also has some understanding of how these work from the business point of view. This means that in designing new systems of a similar kind, learners have an understanding that makes the design work more accessible and more satisfying. This suggests to us that system exercises based on familiar domains, such as libraries or supermarkets, may be more successful than those based on unfamiliar domains, such as the inner workings of coffee machines.

#### 3.2.3 Collaborators

With collaborators, we found no great difficulties, but some important points did arise. One common point was uncertainly about the term "collaborator". In particular, whether one class using another should result in each class noting the other as a collaborator. Our answer is no, and we suggest that the class doing the using should list the other as a collaborator, and not the other way around. We point out that some people prefer the simpler term "helper", and this makes the direction of help clear. We also mention how this promotes reuse, because the helper class can remain independent.

Probably our biggest surprise about the term "collaborator" was that some people see strong negative meaning in the word. For example, we were reminded, in wartime occupations, civilians who help the invading forces are known as collaborators. This increased our emphasis of the more neutral word "helper".

### 3.3 Cards, Dynamic Structure, and Roleplay

The third key element in the CRC card technique is object roleplay. Our experience with roleplay has been quite positive, and the strengths clearly show. As a facilitation technique, roleplay really forces all team members to participate and really engage with the design. Moreover, team members do identity with the object they are playing, and think carefully about the responsibilities and collaborators, often leading to "what-if" exploration and improvement. As a design technique, roleplay really creates a focus on how the design makes objects collaborate to work through a use case.

These strengths are real and valuable, but we did notice some weaknesses. The main theme we detected is how easy it is for teams to roleplay a use case, yet neglect or gloss over difficulties. For example, they omit critical behaviour, or they assume knowledge unreasonably. The result is that a team may decide a roleplay is successful, when in fact it may well be incomplete or impossible. We tried to address these problems in several ways. Some particular problems involve object creation, lookup by key in objects acting as lists or directories, and so on.

One approach we took is to require teams to perform roleplay in front of a larger group: often other teams. The immediate benefit comes from simple peer pressure: when a team knows a roleplay presentation will be asked for, they are considerably more careful in checking that the roleplay is sensible. Moreover, each team member makes very certain that they know their part.

Roleplay presentations also allow us to emphasise the importance of auditing and review. We encourage the audience to watch for slips and inconsistencies, and after a roleplay, we facilitate a large group discussion with the team. Our focus is on formative evaluation, where iterative improvement is the aim. We encourage an "actors studio" ambiance, where review and evaluation is done in supportive way focused on support for improvement, rather than assessment. We have found that this also feeds back to the roleplay in the team environment, where auditing and fast review are also sound practice.

This also leads in well to our later practice of explicitly addressing design evaluation [3]. Using a technique inspired by usability evaluation [10], we often use heuristics (such as some of those documented by Riel [11]), in conjunction with roleplay presentation of use cases.

Another approach we use to improve roleplay is to focus on the user of the system. Roleplay already provides a focus on use cases, but we emphasise that use cases are all about users and their goals in usage of the system. We often suggest that team roleplay feature one team member as the user, so that the user characteristics and goals are considered, and the object collaboration can be reviewed to determine whether it meets these goals.

We also have found it important to help teams record roleplay, both to ensure correct recall later, as well as to facilitate review. We encourage use of rough sequence diagrams to record the roleplay, and to consider the steps in conjunction with the actual CRC cards, and class diagrams.

Our focus on roleplay presentations means we also want to share these artifacts, UML diagrams and CRC cards, with larger groups. This can be difficult, especially because we like to dissuade early investment in beautiful diagrams, and advocate low-investment rough sketches that make amendment and improvement less traumatic. The technology we favour is use of document video cameras in the presentation space. Document cameras allow good and large projection of all kinds of diagrams and sketches, even those on back of envelopes or serviettes. We have found this approach very successful, and now always show diagrams, cards, and sketches when doing roleplay presentation or review.

Fortunately, document cameras are becoming very affordable, and are easy to use with video projectors commonly used for presentations. On occasion we have even fabricated our own document camera simply using a web cam and its standard software, although the video resolution is only minimally acceptable.

## 4. CONCLUSIONS AND ADVICE

We have used CRC cards for many years, to help learners of OO design. Recent circumstances allowed us to observe a number of teams learning with the same design exercises all in a short period of time, and reflect on how well the CRC card technique works. This paper has documented our main findings. Our overall conclusion is that CRC cards remain an effect technique for learning and practicing OO design. We did detect weaknesses, but found strategies to compensate. Using these strategies, we found the CRC card technique is sound. We advocate use of the strategies to take advantage of the strengths of the CRC card technique, while addressing the weaknesses.

## 5. REFERENCES


[1] Kent Beck and Ward Cunningham. A laboratory for teaching object-oriented thinking. In *Proc. of OOPSLA-89: ACM Conference on Object-Oriented Programming Systems Languages and Applications*, pages 1–6, 1989.

[2] David Bellin and Susan Suchman Simone. *The CRC Card Book*. Addison-Wesley, 1997.

[3] Robert Biddle, Rick Mercer, and Eugene Wallingford. Report on workshop on evaluating object-oriented design. In *Addendum to Proceedings of 1998 ACM OOPSLA Conference*, Vancouver, Canada, 1999. ACM SIGPLAN.

[4] Robert Biddle, James Noble, and Ewan Tempero. Essential Use Cases and Responsibility in Object-Oriented Development. Technical Report CS-TR-01-1, Victoria University of Wellington, 2001. Further publication details pending.

[5] Robert Biddle, James Noble, and Ewan Tempero. Techniques for active learning of Object-Oriented Development. In *ACM Oopsla Educators Symposium*, 2001.

[6] Robert Biddle, James Noble, and Ewan Tempero. Use case cards and use case roleplay. Technical Report CS-TR-01-4, Victoria University of Wellington, 2001. Further publication details pending.

[7] Larry L. Constantine and Lucy A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage Centered Design*. Addison-Wesley, 1999.

[8] Brian Henderson-Sellers, Anthony Simons, Houman Younessi, and Ian S. Graham. *The Open Toolbox of Techniques*. Addison-Wesley, 1998.

[9] Ivar Jacobson. *The Object Advantage : Business Process Reengineering With Object Technology*. Addison-Wesley, 1995.

[10] Jakob Nielsen. *Usability Engineering*. Academic Press, New York, 1992.

[11] Arthur Riel. *Object-Oriented Design Heuristics*. Addison-Wesley, 1996.

[12] Nancy Wilkinson. *Using CRC Cards - An Informal Approach to OO Development*. Cambridge University Press, 1996.

[13] Rebecca Wirfs-Brock and Brian Wilkerson. Object-oriented design: A responsibility-driven approach. In Norman Meyrowitz, editor, *Proc. of OOPSLA-89: ACM Conference on Object-Oriented Programming Systems Languages and Applications*, pages 71–75, 1989.

[14] Rebecca Wirfs-Brock, Brian Wilkerson, and Lauren Wiener. *Designing Object Oriented Software*. Prentice Hall, 1990.

[15] Rebecca J. Wirfs-Brock. How designs differ. *The Smalltalk Report*, 1(4), 1994.