inf.ed.ac.uk

OOP Tutorial Week 4: Class, Responsibilities, Collaborators

4-6 minutes

This tutorial steps back from object-oriented programming to look more broadly at object-oriented design. In the tutorial, you will work through the process of turning a loose specification into a design that could then feed into an implementation at a later stage. The goal is to help you think about defining a set of tasks and analysing them in an object-oriented way. A key part of the tutorial exercise is creating a set of CRC cards. For the origins of this technique, see:

Kent Beck & Ward Cunningham (October 1989), "A Laboratory for Teaching Object-Oriented Thinking", ACM SIGPLAN Notices (New York, NY, USA: ACM) 24 (10): 1-6.

A CRC card lists:

- a Class,

- the Responsibilities of the class, and
- the Collaborators of the class.

We will use index cards for this. Figure 1 shows an idealized CRC card. The class name appears underlined in the upper-left hand corner, a list of responsibilities appears under it in the left two-thirds of the card, and the list of collaborators appears in the right third.

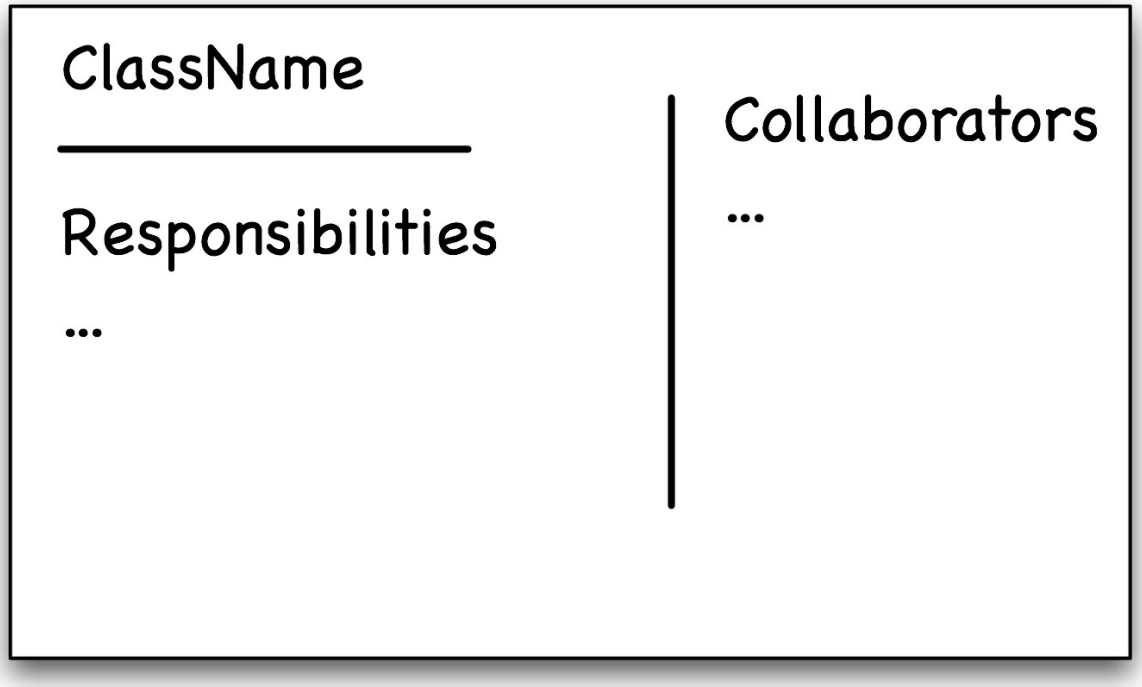

Figure 1: A Class-Responsibility-Collaborator (CRC) index card

The *class name* of an object creates a vocabulary for discussing a design. The *responsibilities* of an object are expressed by a handful

of short verb phrases, each containing an active verb. *Collaborators* are objects which will send or be sent messages in the course of satisfying responsibilities.

Figure 2 shows a couple of example CRC cards using the scenario of a banking ATM.

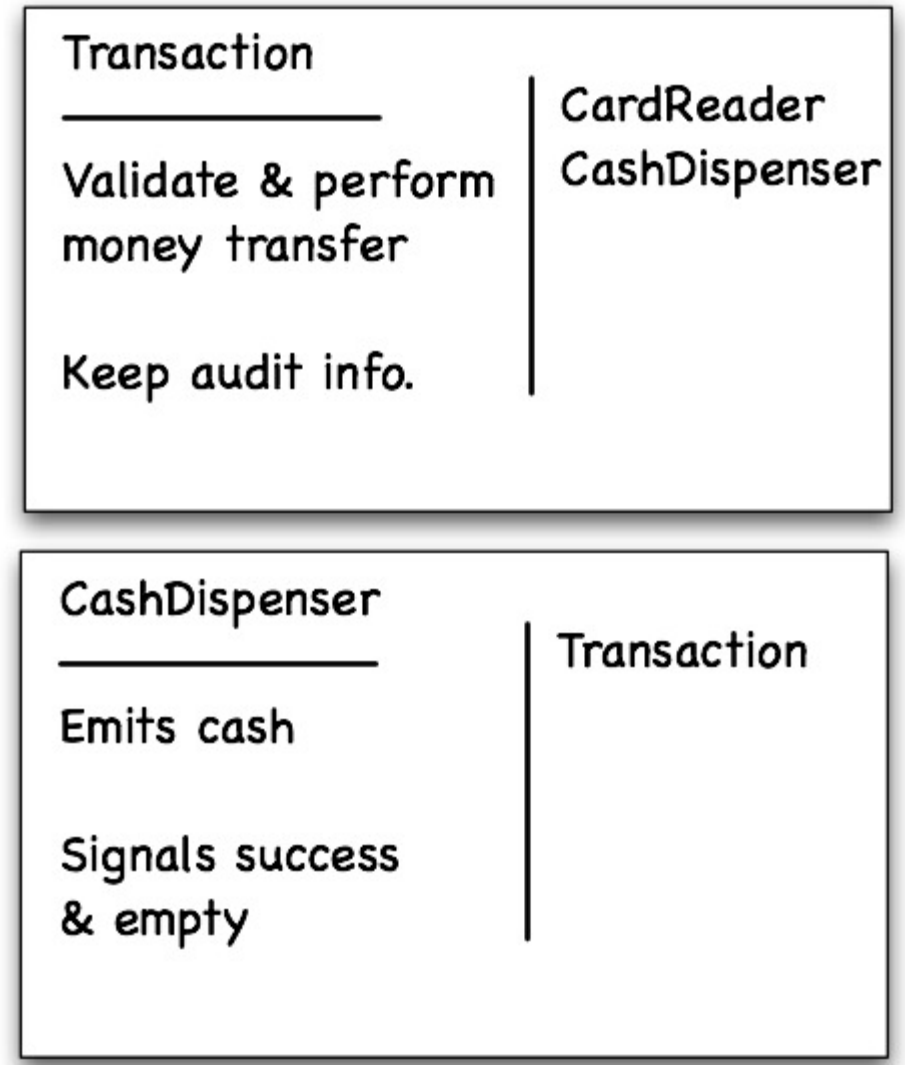

Figure 2: Examples from ATM Scenario

Here is a proposed scenario.

User Jane Doe buys the track "Heartless" from the iTunes store for 1GBP. Jane has an account with iTunes. She has enough credit to buy the track. She does not already have it in her personal library. Once she has purchased it, it gets downloaded to her library and her account is debited accordingly.

Task 1

The scenario above is intended to be simple and relatively well-defined. However, you might want to spend about 5 minutes as a group to see if parts of it need clarification. What kind of data is going to be involved, and how does it need to be organized?

Task 2

Once you feel the scenario is clear enough to get started on, you need to decide as a group what cards should be created. You might want to nominate a particular person to take charge of each card as it is created. In this scenario, you should start off by making a single GUI CRC card that is responsible for all interaction with the user.

A common method to determine what cards should be created is to read a specification for the program being designed and consider if each noun should be a class and if each verb should be a responsibility of the noun or class to which it belongs. Naturally, the existence of a noun or verb does not require a class or responsibility in the program, but it is considered a good starting point.

In thinking about responsibilities of objects, recall that you are trying to build a model in software, not a model of real-world objects. For example, in the software model, an iTunes track might have the responsibility of recording whether it has been paid for.

Task 3

In alternation with Task 2, try to carry out a simulated execution of the scenario by role-play, where messages are passed from one collaborating class to another. As you hit problems or things that you didn't allow for, elaborate the set of CRC cards and then resume the execution.

Start with only one or two obvious cards and start playing "what-if". As the group walks through the scenario, members of the group take on the roles of the objects that are interacting, for example, by

holding up the card so that everyone can see it, and saying what responsibility is being performed. The intention here is to test how responsibilities are distributed across classes, *not* how the classes are implemented.

If the situation calls for a responsibility not already covered by one of the objects, you either add the responsibility to one of the objects, or create a new object to address that responsibility. If one of the object becomes too cluttered during this process, copy the information on its card to a new card, searching for more concise and powerful ways of saying what the object does. If it is not possible to shrink the information further, but the object is still too complex, create a new object to assume some of the responsibilities.