Responsibility Driven Design

Responsibility Driven Design, Rebecca Wirfs Brock, 1990


Introducing Object-Oriented Design with Active Learning, Rick Mercer, Consortium for Computing in Small Colleges, 2000
Responsibility-Driven Design is a way to design that emphasizes behavioral modeling using objects, responsibilities and collaborations. In a responsibility-based model, objects play specific roles and occupy well-known positions in the application architecture. Each object is accountable for a specific portion of the work. They collaborate in clearly defined ways, contracting with each other to fulfill the larger goals of the application. By creating a "community of objects", assigning specific responsibilities to each, you build a collaborative model of our application.

Responsible: able to answer for one's conduct and obligations—trustworthy, Merriam Webster
Responsibility Driven Design
Summarizing Rebecca in Rick's words

1) Identify candidate objects that model a system as a sensible set of abstractions

2) Determine the responsibility of each object
   — what an instance of the class must be able to do,
   — and what each instance must know about itself

3) Understand the system through role play
   — To help complete its responsibility, an object often needs help from other objects
The Single Responsibility Principle

Classes should have a single responsibility


Why?

- Cohesion, when high, reduces complexity, makes the system more understandable
  http://en.wikipedia.org/wiki/Cohesion_%28computer_science%29
  - Maintenance: Fixing or changing a module should not break other parts of the system
First Design a Model
Note: design is iterative

- Find a set of objects (candidate classes) that model a solution
- Each will be a part of the bigger system
- Each should have a single responsibility
- What are these objects?
Find the Objects

♦ Candidate objects may come from
  — The nouns in the problem statement
    • **Underline the noun phrases** to look for the objects that could model the system
  — An understanding of the problem domain
    • knowledge of the system that the problem specification may have missed or took for granted
  — The words floating around the room Alistair Cockburn