

E-Film Hiring Project

HW4. From Requirements to Design

Deliveries: Design Model (Interaction Diagrams)

THIRD MACRO-ACTIVITY: OBJECT DESIGN

To review the previous macro-activities, relationships between some software development artifacts are listed:

- The use case suggests the system events that are explicitly shown in system sequence diagrams.
- Details of the effect of the system events in terms of changes to domain objects may optionally be described in system operation contracts.
- The system events represent messages that initiate interaction diagrams, which illustrate how objects interact to fulfill the required tasks, that is the use case realization.
- The interaction diagrams involve message interaction between software objects whose names are sometimes inspired by the names of conceptual classes in the Domain Model, plus other classes of objects.

Why Object Design?

The requirements and object-oriented analysis has focused on learning to do the right thing; that is, understanding the goals for our system, and related rules and constraints. The following design macro-activity will stress do the thing right; that is, skillfully designing a solution to satisfy the requirements.

During object design, a logical solution based on the object-oriented paradigm is developed. The heart of this solution is the creation of **interaction diagrams**, which illustrate how objects collaborate to fulfill the requirements. Drawing interaction diagrams is a reflection of making decisions about the software system design.

In parallel with drawing interaction diagrams, **design class diagrams** can be drawn. These summarize the definition of the software classes (and interfaces) that are to be implemented in software and will be added to the classes inspired by the domain model.

These artifacts are part of the **Design Model**. In the followings, we will use a linear approach, for sake of simplicity and clarity, but you have to carry concurrently out both activities.

Keep in mind that the principles of responsibility assignment and design patterns are the fundamental knowledge for object design!

First Step: Identifying interactions for use-case realization

After identifying requirements, creating a domain model, and identifying system operations you should add methods to the software classes, and define the messaging between the objects to fulfill the requirements.

Responsibility is "a contract or obligation of a class". Responsibilities are related to the obligations of an object in terms of its behavior. These responsibilities are of the following two types:

- knowing
- doing

Knowing responsibilities of an object include:

- knowing about private encapsulated data
- knowing about related objects
- knowing about things it can derive or calculate

Doing responsibilities of an object include:

- doing something itself, such as creating an object or doing a calculation
- initiating action in other objects
- controlling and coordinating activities in other objects

Responsibilities are assigned to classes of objects during object design. A responsibility is not the same thing as a method, but methods are implemented to fulfill responsibilities.

For assigning responsibility, use typically five fundamental patterns:

1. Information Expert
2. Creator
3. High Cohesion
4. Low Coupling
5. Controller

There are others (Polymorphism, Fabrication, Indirection, Protected Variations etc.) but it is worthwhile mastering these five first because they address very basic, common questions and fundamental design issues.

Procedure

1. Design the interaction (sequence or collaboration) diagrams for each use-case and system operation identified in the previous homework. Use also system operation contracts when they are available. Do not hesitate to introduce new classes when needed:
 - ? Introduce a controller for each use case: a use case will be managed by at least one controller and each controller manages at least one use case.
 - ? Introduce graphical interface classes such that each time when an actor interacts with the system he/she does through a GUI.
2. Give reason for your design decisions by indicating the used patterns.
3. Complete the static model with the classes created in this step.