Today

- Chap. 13 – "operation contracts"
- Chapt. 14-15 – begin *design*

Operation Contracts

- *System Operations*: functions provided by the system to the world
  - The description of these constitutes the system's interface
  - System operations respond to requests from outside the system
- Such a request is called a system event
- System event makeNewSale, causes system operation makeNewSale
- Like OO message sqrt(…) invokes method sqrt(…)
  - It is convenient to give them the same name
  - Conceptually they are very different (but of course quite related)
- See Figure 13.1

Operation Contracts

- **System events** vs. **System operations**
  - These are analogous to:
- **Messages** vs. **Methods**
  - A **makeNewSale** message causes execution of a **makeNewSale** method

Operation Contract Details

- See Figs., p. 179 for an example
- Note: these details are outside the scope of inception
- Use them during elaboration phase
  - Judiciously
    - (i.e. to clarify complicated situations)
- What domain class model portion would correspond to contract CO2?

Operation Contract Postconditions

- A **postcondition** is something guaranteed to hold after the operation
- Three kinds of postconditions
  - Create (& delete) instances of classes
    - (also called whats?)
  - Modify attributes
    - (how are these implemented in OO?)
  - Create & delete associations
    - (how are associations implemented in OO?)

Operation Contract Postconditions II

- Which is a better postcondition?
  - 1: A salesLineItem has been created
  - 2: Create a salesLineItem

Postconditions – 3 kinds

- Examples of each
  - 1) Instance creation: a salesLineItem, call it sli, has been generated
  - 2) Attribute modification: sli.quantity gets appropriate value
  - 3) Association created: sli and currentSale are linked together
- How would these be implemented in OO?

From Requirements to Design

- Reference: chapter 14
- Background:
  - Traditionally, design answers "what functions call what other functions?"
  - In OOA/D, design answers "what objects interact with what other objects?"
- Naturally we will emphasize the OOA/D

Design Vs. Requirements Match Column A to B

- A. Analysis
- D. Design

- 1) Do the thing right
- 2) Do the right thing
- 3) Specifications
- 4) Requirements

The Design Model

- The design model is, essentially, the design
  - In the UP it has two major parts
    - **Class diagrams** (also seen in analysis)
      - See domain class model Figure 10.1
      - This maps nicely to a design class diagram
    - Interaction diagrams
      - Two kinds –
        - **Sequence diagrams** (also seen in analysis)
        - **Collaboration diagrams** (could be seen in analysis)

Interaction Diagrams

Interaction diagrams come in two varieties

- UML **Sequence diagrams**
- UML **Collaboration diagrams**
- They are different ways to say the same thing!
- Why would you want to have that?
  - Sometimes one aspect needs to be clear
  - Sometimes another aspect needs to be clear instead
  - It would be nice, but maybe impossible, to have both clear at once
  - Let's check an example (see 3 elevator Figures)

Sequence Diagrams Vs. Collaboration Diagrams

- Sequence diagrams emphasize
  - chronologies
    - (Does that mean timing or interactions?)
- Collaboration diagrams emphasize
  - connections among actors
- True or false: arrow 1. should point to the elevator button bubble
- Note how the sequence diagram shows the order of things
- Note how the collaboration diagram shows the centrality of the elevator controller

Sequence Diagrams Vs. Collaboration Diagrams II

- Sequence diagrams:
- Time starts at top, proceeds down
- A sequence diagram is based on a scenario
- Scenarios come from what?
- Collaboration diagrams:
- Build from scenarios like sequence diagrams
- It has what the sequence diagram has!
- So why have both?
- - one emphasizes timing,
- - other emphasizes relationships among actors
- - (which is which?)
- We could (should?) convert one diagram to the other

Interaction Diagram Details

- Classes, instances (objects), and instances with names
- Each has its own notation
- See Figure 15.5.
- Which notation is for which of those?