Last GRASP (for now) and Use Case Realization

Curt Clifton

Rose-Hulman Institute of Technology

General,
Responsibility **A**ssignment
Software ~~Patterns~~ **P**rinciples

Coupling

- A measure of how strongly one element is connected to, has knowledge of, or relies on other elements
- Want low (or weak) coupling

Cohesion

- A measure of how strongly related and focused the responsibilities of a class (or method or package…) are
- Want high cohesion

Information Expert

- **Problem**: What is the most general principle of assigning responsibilities?
- **Solution**: Assign a responsibility to the class that has the necessary information

Creator

- **Problem**: Who should be responsible for creating a new instance of some class?
- **Solution**: Make *B* responsible for creating *A* if…
  - *B* contains or is a composition of *A* ← Most important
  - *B* records *A*
  - *B* closely uses *A*
  - *B* has the data to initialize *A*

The more matches the better.

Team Creativity

Q11

Controller

Controller

What's that?

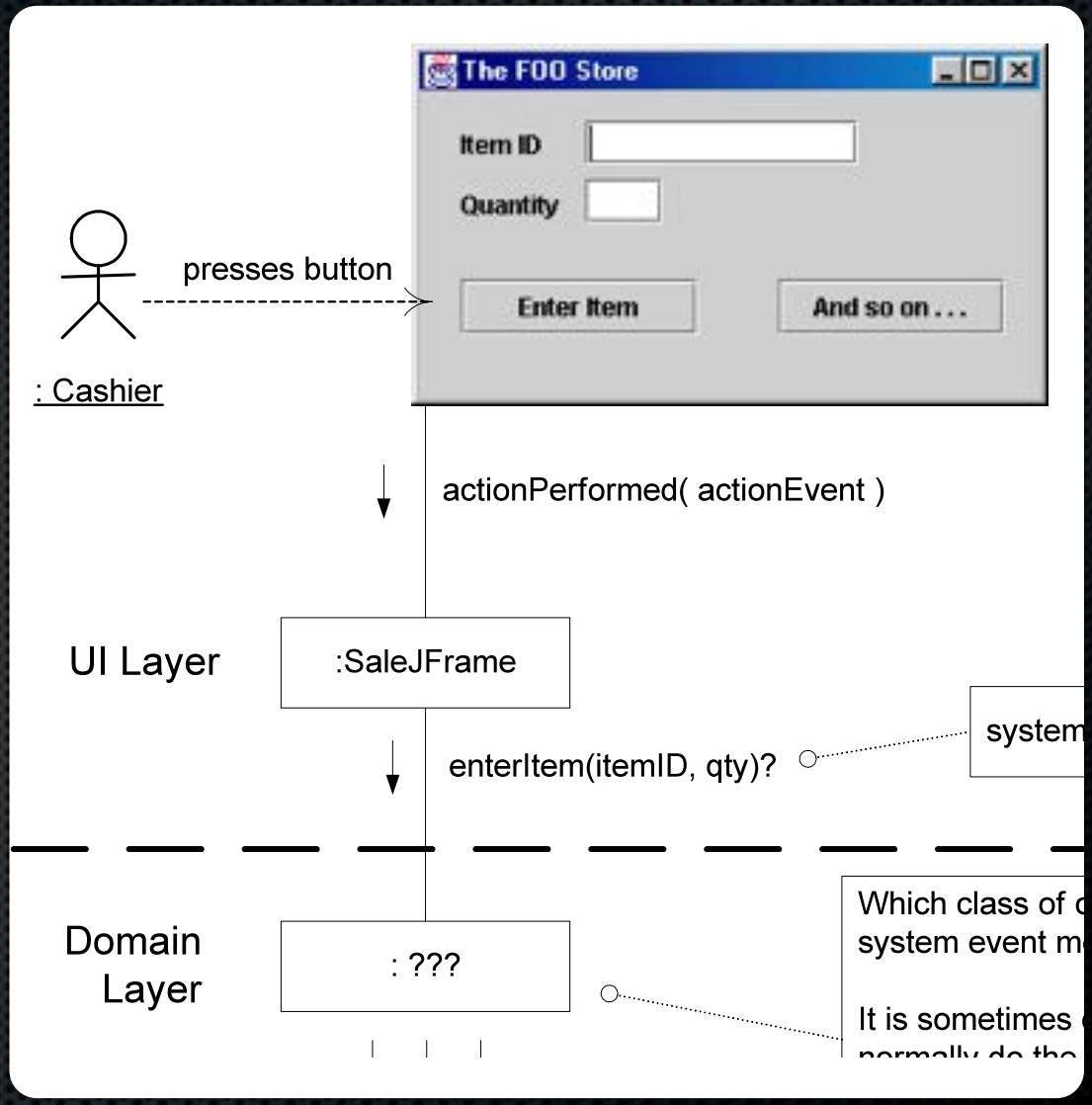
- **Problem**: What first object beyond the UI layer receives and coordinates a **system operation**
- **Solution**: Assign the responsibility to either…
  - A façade controller, representing the overall system and handling all system operations, or
  - A use case controller, that handles all system events for a single use case

Not JFrame, not JPanel, …

Q12,13

Example

- What domain layer class should own handling of the *enterItem* system operation?



Guidelines

- Controller should **delegate** to other domain layer objects
- Use a façade controller when…
  - There are a limited number of system operations, or
  - When operations are coming in over a single “pipe”
- Use a use case controller when a façade would be bloated (low cohesion!)

Controller Benefits

- Increased potential for reuse
- Can reason/control the state of a use case
  - E.g., don’t close sale until payment is accepted

Controller Issues

Switch from façade to use case controllers

- Controller bloat—too many system operations
- Controller fails to delegate tasks
- Controller has many attributes

Delegate!

Team Control

Q14,15

Cartoon of the Day

Jan 4, 2010. Used by permission

Use Case Realization

Use Case Realization

- The process of generating the design model from use cases and other requirements artifacts
- Using GRASP and other patterns to guide decisions

Use Cases…

- Drove the development of
  - Domain Model
  - System Sequence Diagrams
  - Operation Contracts

System Sequence Diagrams

- Helped us identify **system operations**
- Use these to begin interaction diagrams
  - System operations are the starting messages
  - Starting messages are directed at controller objects

Operation Contracts

- Defined **post-conditions** of system operations as changes to objects/associations **in the domain model**
- Use post-conditions to help determine…
  - What should happen in the interaction diagrams
  - What classes belong in the design class diagram

Also often discover classes that were missed in the domain model

Q2

Where to Begin

- In code, begin at the beginning
- In design, defer design of the Start Up use case
  - Why?

Q3

Example (if time permits)

Design *makeNewSale*

Operation:	makeNewSale()
Cross References:	Use Case: Process Sale
Preconditions:	none
Postconditions:	▪ A *Sale* instance *s* was created ▪ *s* was associated with the *Register* ▪ Attributes of *s* were initialized