Test Driven Development and Refactoring (plus an eclectic flyover)

Shawn Bohner
Office: Moench Room F212
Phone: (812) 877-8685
Email: bohner@rose-hulman.edu

ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

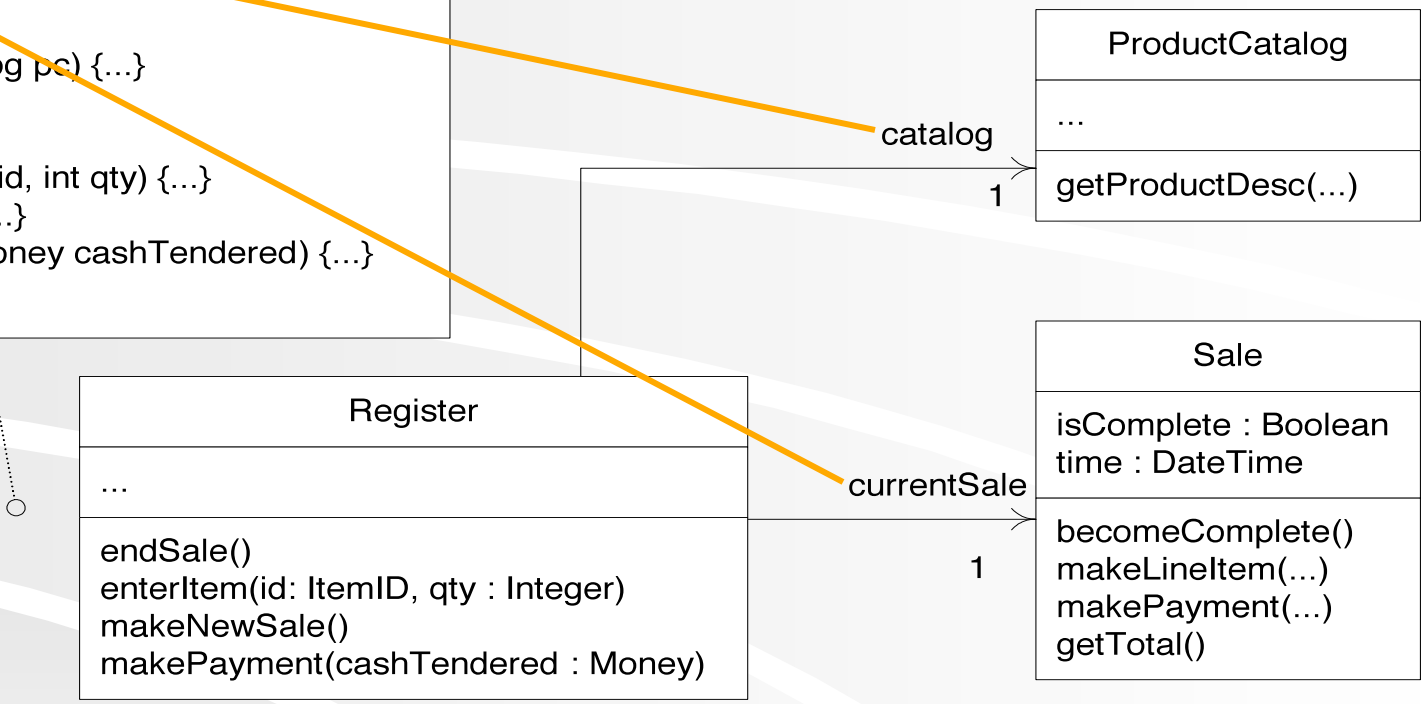
Q1

Agenda

- Test-Driven Development
- Refactoring
- Transition to Iteration 2
  - Analysis Refresh
  - Discuss Milestone 4

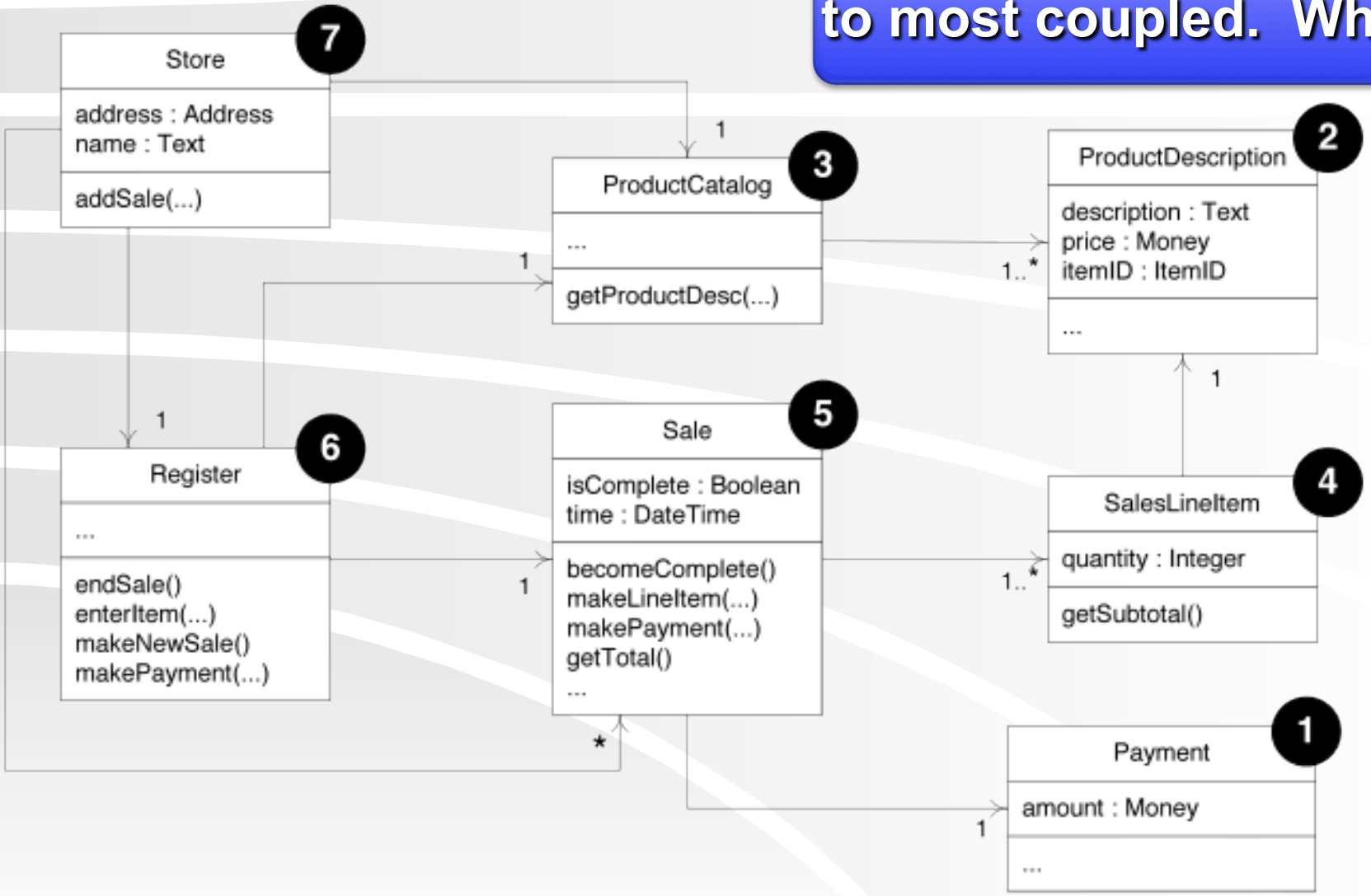
Moving from Design to Code Example

```
public class Register
{
private ProductCatalog catalog;
private Sale currentSale;

public Register(ProductCatalog pc) {...}

public  void endSale() {...}
public  void enterItem(ItemID id, int qty) {...}
public void makeNewSale() {...}
public  void makePayment(Money cashTendered) {...}
}
```



What Order?

Typically, least coupled to most coupled. Why?



ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Test-Driven Development: Key Ideas

- Tests get written first (not postponed)
- Stub in method, then write tests for method before writing the actual method
- Quickly alternate between testing and implementation (i.e., one method at a time)
- Build up a library of test cases

Advantages of TDD

- Unit tests actually get written
- Programmer satisfaction is increased
- Tests serve to clarify the interface and document behavior
- As test suite grows, it serves as an automated verification
- Gives developers confidence to make changes

Refactoring

Structured, disciplined method to rewrite/restructure existing code *without changing its external behavior*

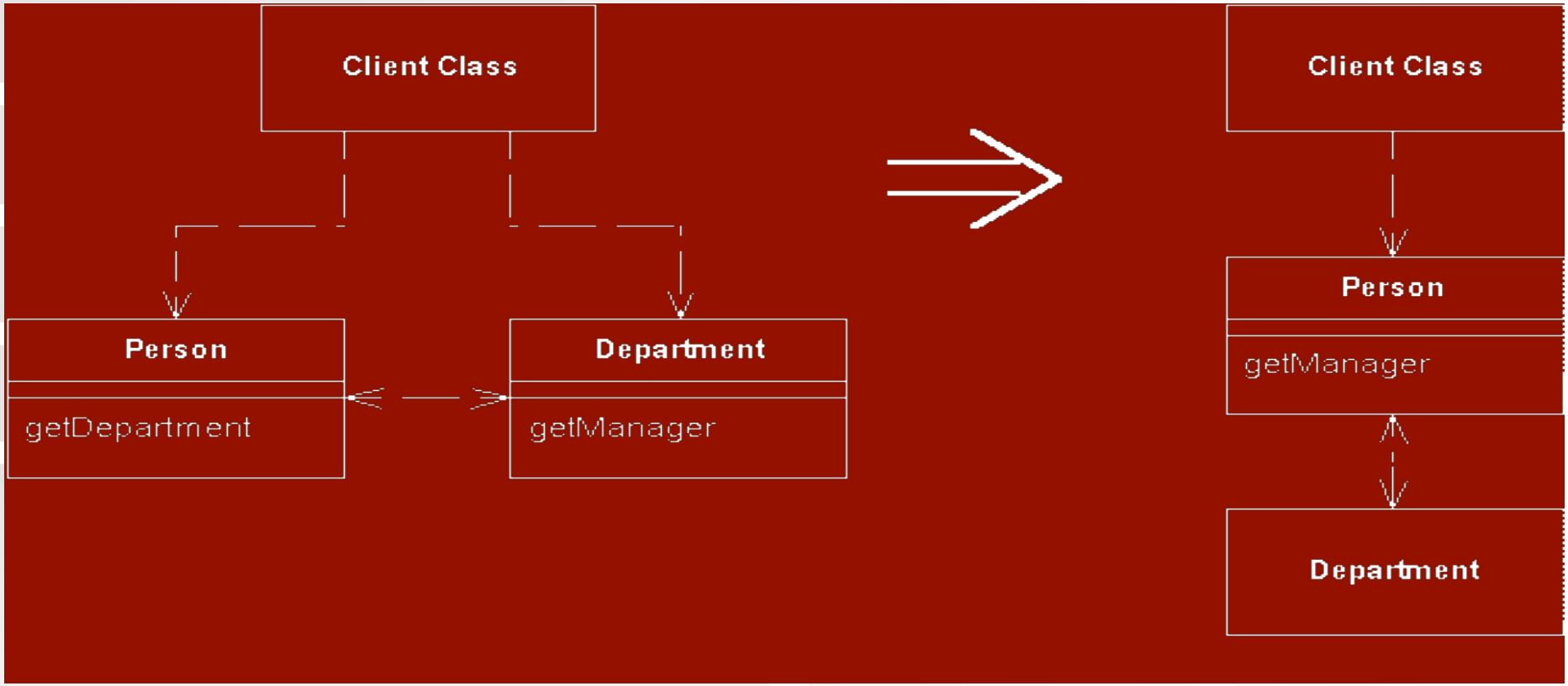
- Recognized importance of refactoring *Kent Beck* and *Ward Cunningham*
- *Ralph Johnson's* work with refactoring and frameworks has also been an important contribution
- *Martin Fowler's* book on Refactoring is a must read…
  - One of the texts for CSSE 375

Some Example Refactorings

- Add Parameter
- Change Association
- Reference to value
- Value to reference
- Collapse hierarchy
- Consolidate conditionals
- Procedures to objects
- Decompose conditional
- Encapsulate collection
- Encapsulate downcast
- Encapsulate field
- Extract class
- Extract Interface
- Extract method
- Extract subclass
- Extract superclass
- Form template method
- Hide delegate
- Hide method
- Inline class
- Inline temp
- Introduce assertion
- Introduce explain variable
- Introduce foreign method

Example: Hide Delegate

When a client is calling a delegate class of an object



Bad Code Smells

- Duplicated code
- Long methods
- Class with many instance variables
- Class with many methods
- Little or no use of interfaces
- …

Not every bad smell indicates a problem

Refactorings, …Code Deodorant?

Refactoring	Description
Extract Method	Transform a long method into a shorter one by factoring out a portion into a private helper method
Extract Constant	Replace a literal constant with a constant variable
Introduce Explaining Variable	Put the result of the expression, or parts of the expression, in a temporary variable with a name that explains its purpose
…	…

Refactoring Indicators: Bad Smells in Code

- Duplicated Code
- Long Method
- Large Class
- Long Parameter List
- Divergent Change
- Shotgun Surgery
- Feature Envy
- Data Clumps
- Primitive Obsession
- Switch Statements
- Parallel Interface Hierarchies
- Lazy Class
- Speculative Generality
- Temporary Field
- Message Chains
- Middle Man
- Inappropriate Intimacy
- Incomplete Library Class
- Data Class
- Refused Bequest

Cartoon of the Day

Used by permission. http://notinventedhe.re/on/2010-1-18

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

From Iteration 1 to Iteration 2

- Iteration 2 corresponds to Milestone 4 in the class
- Take a few minutes to review Milestone 4
- Answer quiz question

Some Typical Iteration 2 Activities

Though not necessarily for our projects, since we took smaller bites in Iteration 1

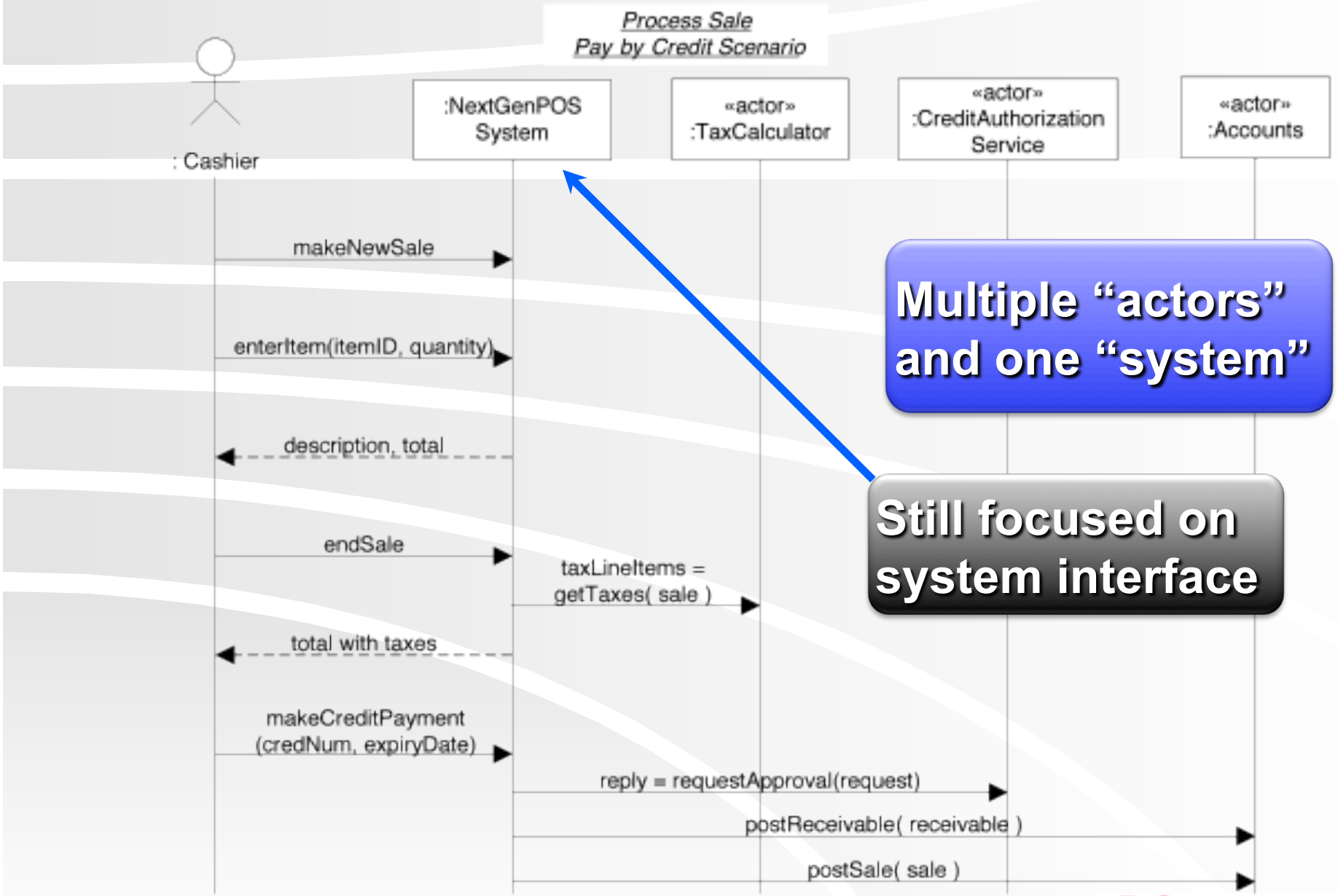
Second Iterations

- Would typically add a few lower risk use cases
  - First iteration focuses on greatest risks
- Would typically do analysis for a significant portion of the system's features—maybe 80%
  - Wouldn't implement all of them yet
- Might implement some alternative scenarios for use cases where we only did the main scenario in Iteration 1

SSDs in Second Iterations

- Often augmented to show some intersystem collaboration
- Update other analysis artifacts as needed…
  - Domain model: might introduce subclasses to deal with clarifying variability
  - Operation contracts: if new system operations warrant detailed post-conditions

Example SSD with Intersystem Collaboration

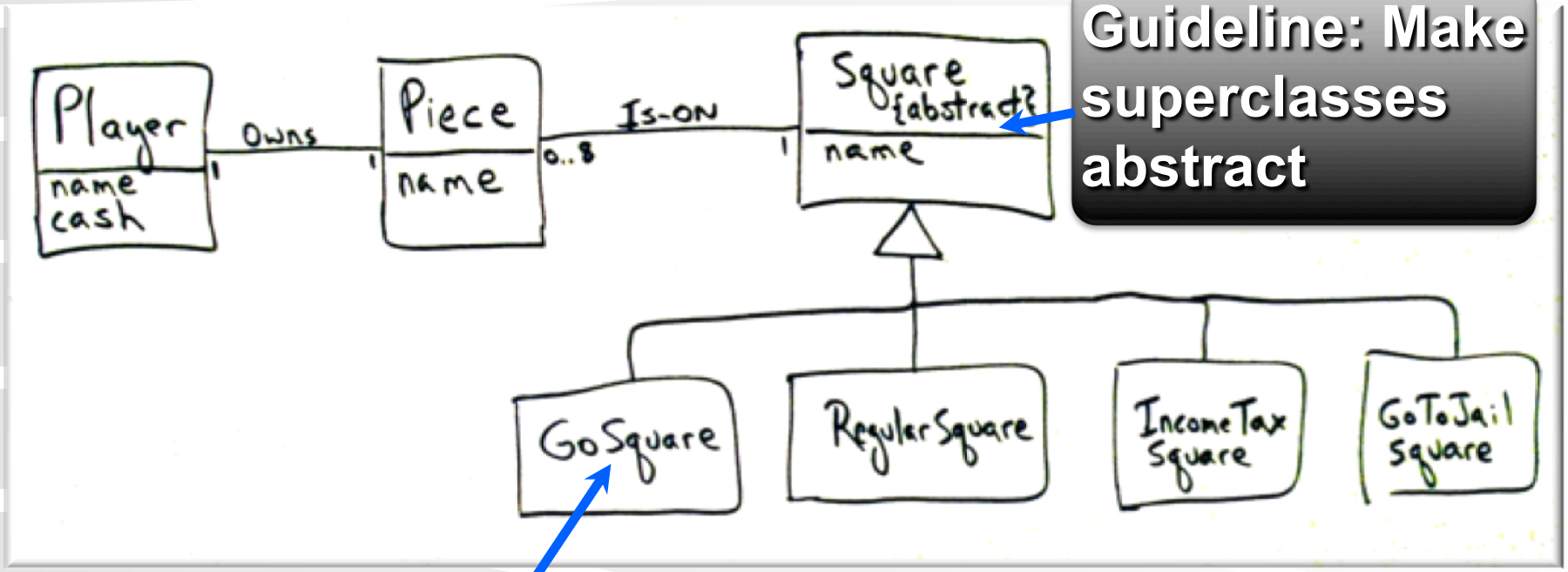


ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Create Conceptual Subclasses in DM when:

- Subclass has additional attributes
- Subclass has additional associations
- Subclass concept “behaves” differently than superclass or other subclasses

Example of Conceptual Subclasses



Guideline: Make superclasses abstract

Guideline: Append superclass name to subclass

Which reason(s) for creating subclasses apply here?

Homework and Milestone Reminders

- Read Chapter 25
- Homework 6 – More GRASP on Video Store Design
  - Due by 5:00pm Tuesday, January 26th, 2010
- Milestone 4: Patterns and Detailed Design, with some Iteration 2 on the Side
  - Due by 11:59pm Friday, January 29th, 2010

Collections


```
public class Sale {
            …
  private List<SalesLineItem> lineItems = new ArrayList<SalesLineItem>
();          …
}
```

Guideline: If an object implements an interface, use the interface type for the variable.

ROSE-HULMAN INSTITUTE OF TECHNOLOGY