Modeling Interaction and Behavior

Behavioral Modeling

- Where are people coming from?
- Where are they going?
- How do they move from one space to the other?

Interaction Diagrams

Interaction diagrams are used to model the dynamic aspects of a software system

- They help you to visualize how the system runs.
- An interaction diagram is often built from a use case and a class diagram.

  —The objective is to show how a set of objects accomplish the required interactions with an actor.

Interactions and messages

- Interaction diagrams show how a set of actors and objects communicate with each other to perform:
  - —The steps of a use case, or
  - —The steps of some other piece of functionality.
- The set of steps, taken together, is called an *interaction*.
- Interaction diagrams can show several different types of communication.
  - —E.g. method calls, messages send over the network
  - —These are all referred to as *messages*.

Elements found in interaction diagrams

- Instances of classes
  - —Shown as boxes with the class and object identifier underlined
- Actors
  - —Use the stick-person symbol as in use case diagrams
- Messages
  - —Shown as arrows from actor to object, or from object to object

Analysis Model Structure

- Key Abstraction folder

For each business package:

  - Boundary classes
  - Control classes
  - Entity classes

- Use Case Realization folder

For each major flow of the use case, draw a **sequence diagram**

Creating interaction diagrams

You should develop a class diagram and a use case model before starting to create an interaction diagram.

- There are two kinds of interaction diagrams:
  - —*Sequence diagrams*
  - —*Communication diagrams*

Sequence Diagram

What is a Sequence Diagram

- This diagram is a model describing how groups of objects collaborate in some behavior over time.
- The diagram captures the behavior of a single use case.
- It shows objects and the messages that are passed between these objects in the use case.

When to use a sequence diagram

- A good design can have lots of small methods in different classes. Because of this it can be difficult to figure out the overall sequence of behavior.
- Simple and visually logical, so it is easy to see the sequence of the flow of control.
- Clearly shows concurrent processes and activations.

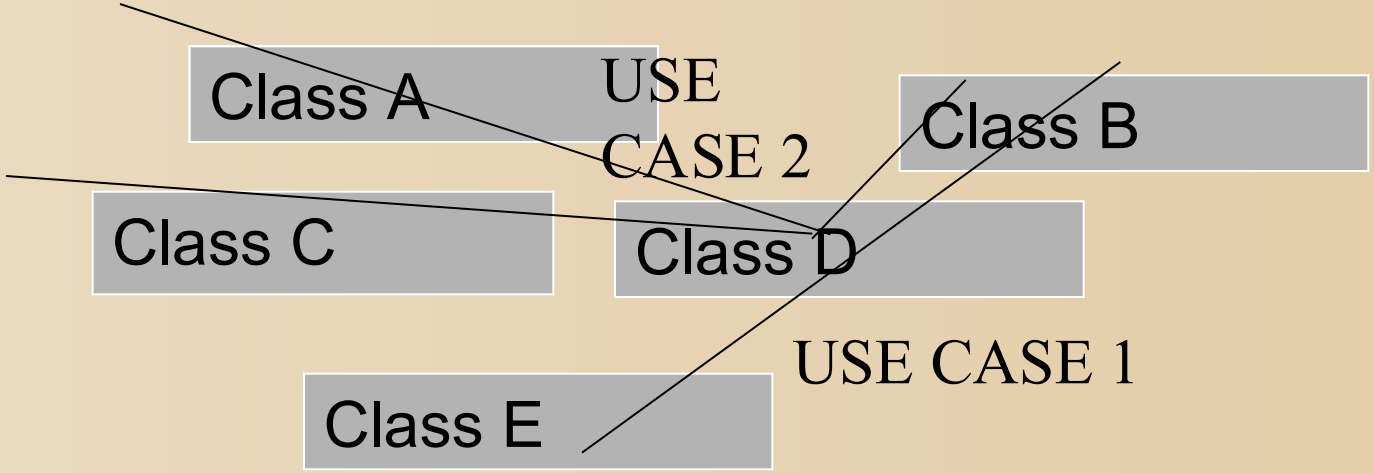
Characteristics of Sequence Diagram

- Represents an interaction (messages) exchanged among collaborating objects for a specific result.
- It shows time sequence
- It does not show relations between objects
- It can show general forms that do not deal with objects but with class interaction used more for real time types of application

Behavioral Modeling

Building a Sequence Diagram:

Sequence diagrams capture the use-case behavior using the foundation of the classes.



Sequence =Objects+messages

Analysis Model: Key abstractions

Boundary classes

- Model the interaction of the system with its actors
- For human actors – maps to web pages/HTML forms
- For external systems – maps to adapters

Control classes

Capture the control logic (presentation and business logic) of one or more use cases

Entity classes

Represent the business concepts manipulated by the system

No attributes or operations are specified for key abstraction classes in analysis model

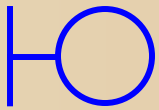
Domain Modelling

- Represents concepts or objects appearing in the problem domain.
- Also captures relationships among objects.
- Three types of objects are identified
  - Boundary objects
  - Entity objects
  - Controller objects

Class Stereotypes

Three different stereotypes on classes are used: <<boundary>>, <<control>>, <<entity>>.

Boundary

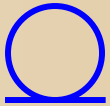


Cashier Interface

Control

Withdrawal

Entity



Account

Boundary Objects

- Interact with actors:
  - User interface objects
- Include screens, menus, forms, dialogs etc.
- Do not perform processing but validates, formats etc.

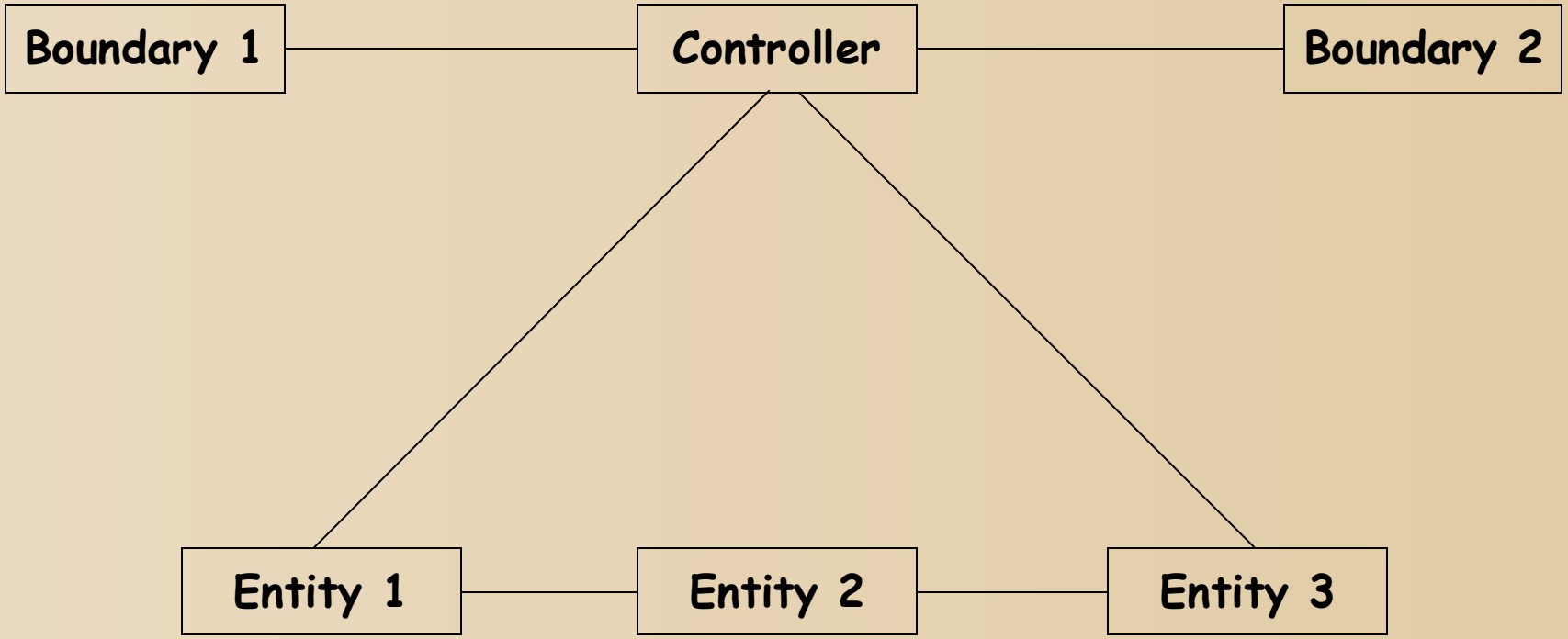
Entity Objects

- Hold information:
  - Such as data tables & files, e.g. Book, BookRegister
- Normally are dumb servers
- Responsible for storing data, fetching data etc.
- Elementary operations on data such as searching, sorting, etc.
- Entity Objects are identified by examining nouns in problem description

Controller Objects

- Coordinate the activities of a set of entity objects
- Interface with the boundary objects
- Realizes use case behavior
- Embody most of the logic involved with the use case realization
- There can be more than one controller to realize a single use case

Use Case Realization

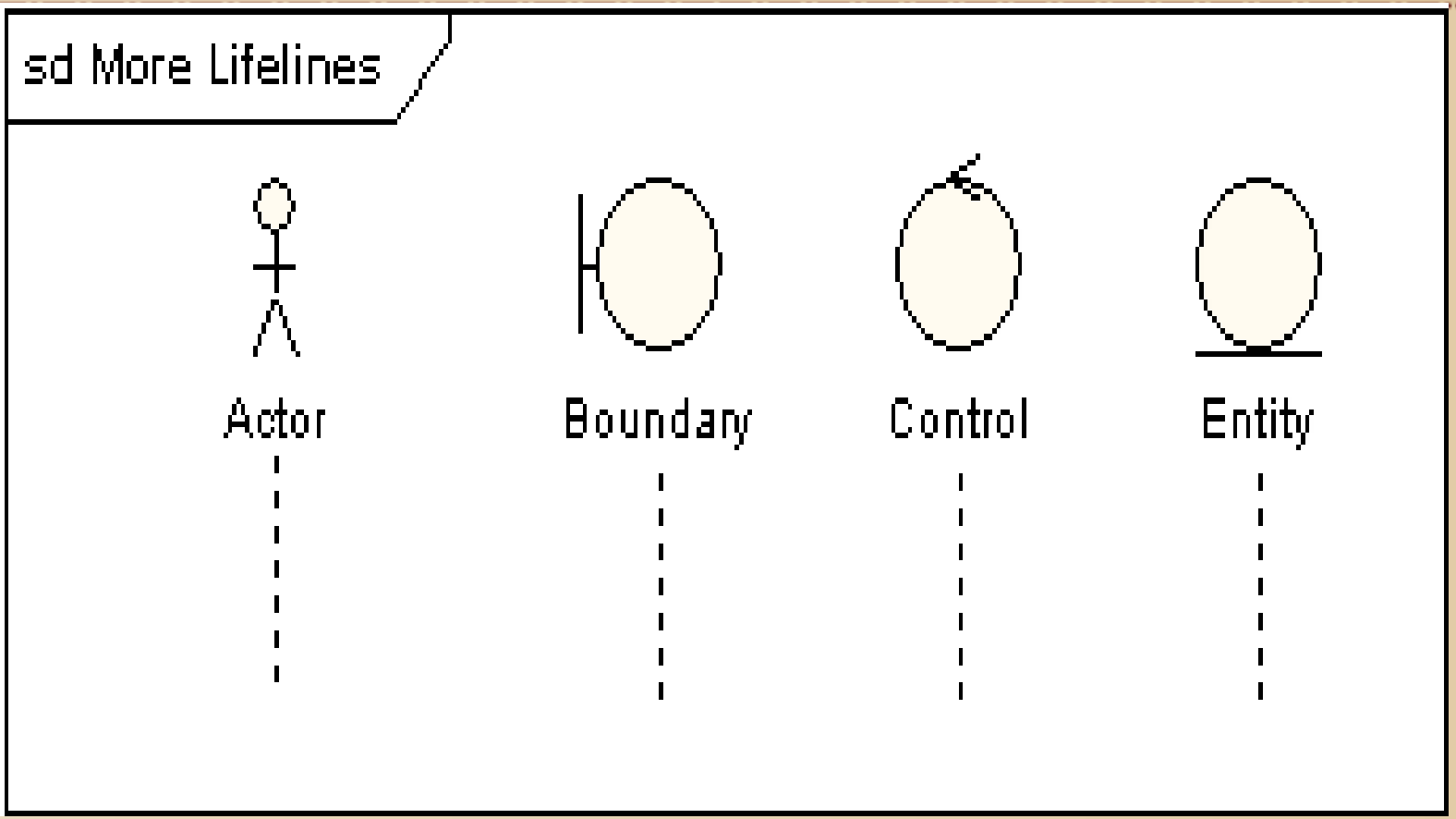


Realization of use case through the collaboration of Boundary, controller and entity objects

Sequence diagrams

A sequence diagram shows the sequence of messages exchanged by the set of objects performing a certain task

- The objects are arranged horizontally across the diagram.
- An actor that initiates the interaction is often shown on the left.
- The vertical dimension represents time.
- A vertical line, called a *lifeline*, is attached to each object or actor.
- The lifeline becomes a broad box, called an *activation box* during the *live activation* period.
- A message is represented as an arrow between activation boxes of the sender and receiver.
  - —A message is labelled and can have an argument list and a return value.



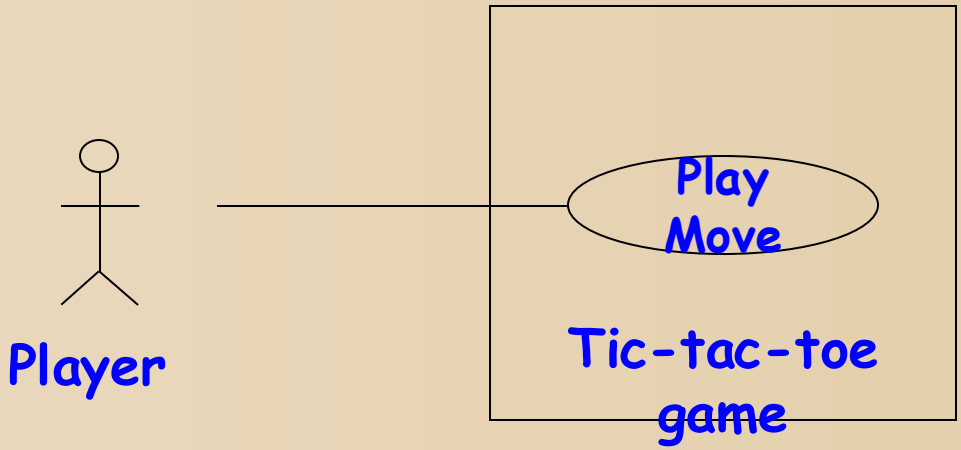
Example 1: Tic-Tac-Toe Computer Game

- A human player and the computer make alternate moves on a 3 3 square.
- A move consists of marking a previously unmarked square.
- The user inputs a number between 1 and 9 to mark a square
- Whoever is first to place three consecutive marks along a straight line (i.e., along a row, column, or diagonal) on the square wins.

Example 1: Tic-Tac-Toe Computer Game

- As soon as either of the human player or the computer wins,
  - A message announcing the winner should be displayed.
- If neither player manages to get three consecutive marks along a straight line,
  - And all the squares on the board are filled up,
  - Then the game is drawn.
- The computer always tries to win a game.

Example 1: Use Case Model

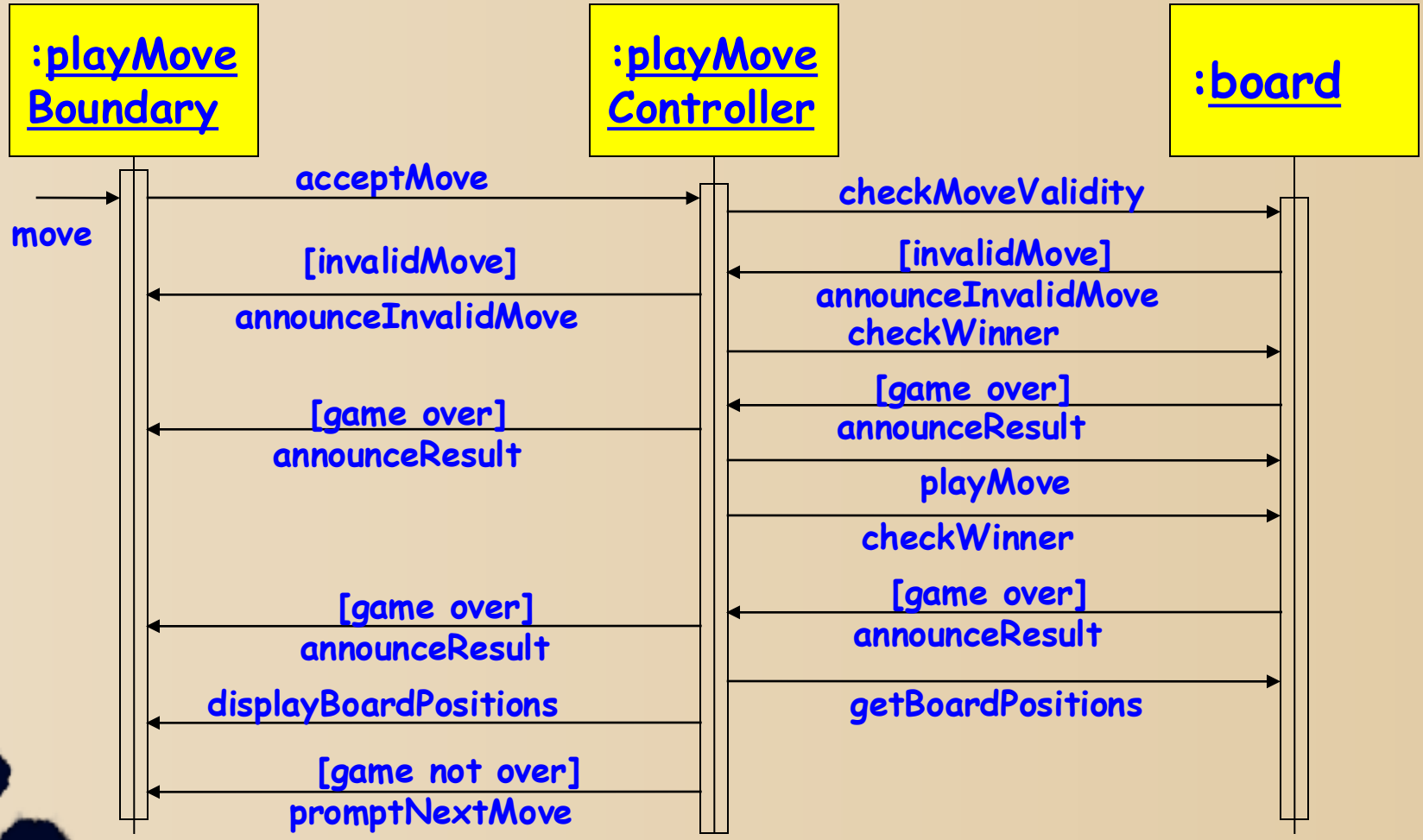


Example 1: Initial and Refined Domain Model

Initial domain model

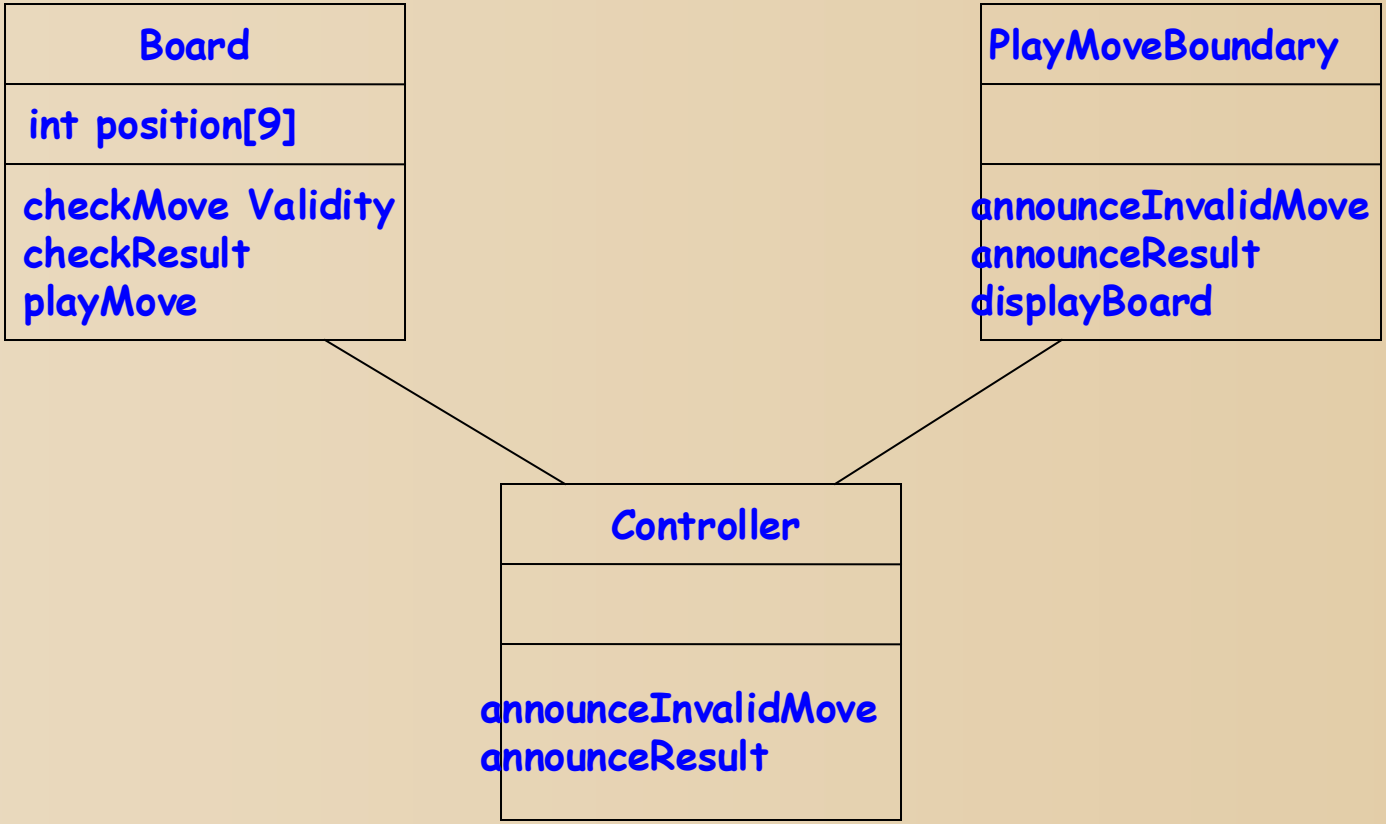
Refined domain model

Example 1: Sequence Diagram



Sequence Diagram for the play move use case

Example 1: Class Diagram



Example 2: Supermarket Prize Scheme

- Supermarket needs to develop software to encourage regular customers.
- Customer needs to supply his:
  - Residence address, telephone number, and the driving licence number.
- Each customer who registers is:
  - Assigned a unique customer number (CN) by the computer.

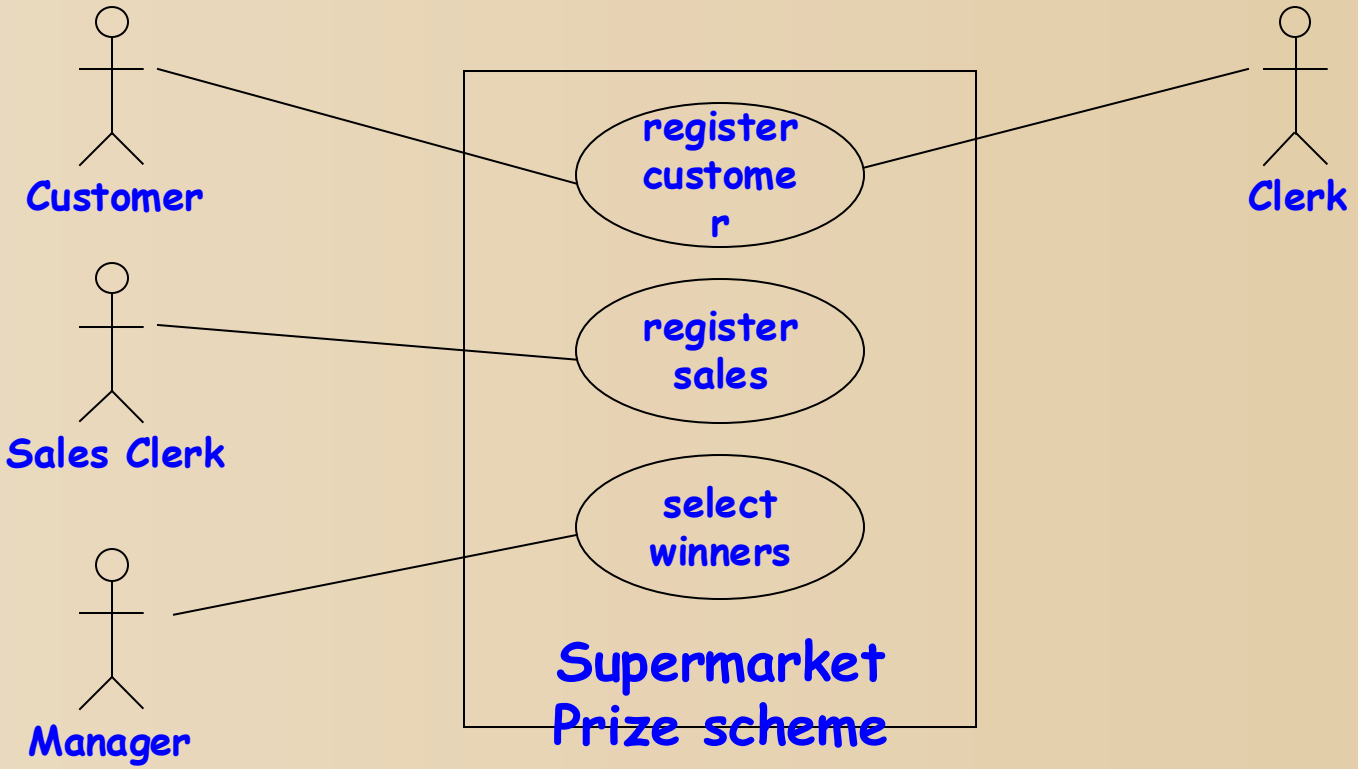
Example 2: Supermarket Prize Scheme

- A customer can present his CN to the staff when he makes any purchase.
- The value of his purchase is credited against his CN.
- At the end of each year:
  - The supermarket awards surprise gifts to ten customers who make highest purchase.

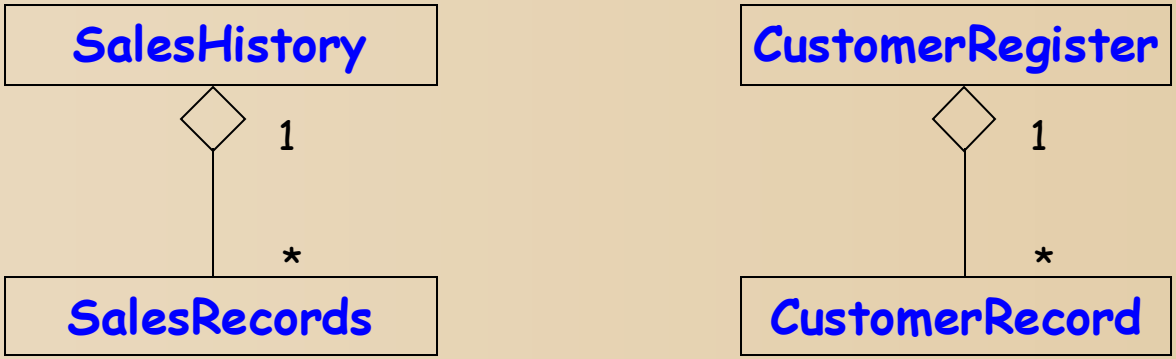
Example 2: Supermarket Prize Scheme

- Also, it awards a 22 carat gold coin to every customer:
  - Whose purchases exceed Rs. 10,000.
- The entries against the CN are reset:
  - On the last day of every year after the prize winner's lists are generated.

Example 2: Use Case Model

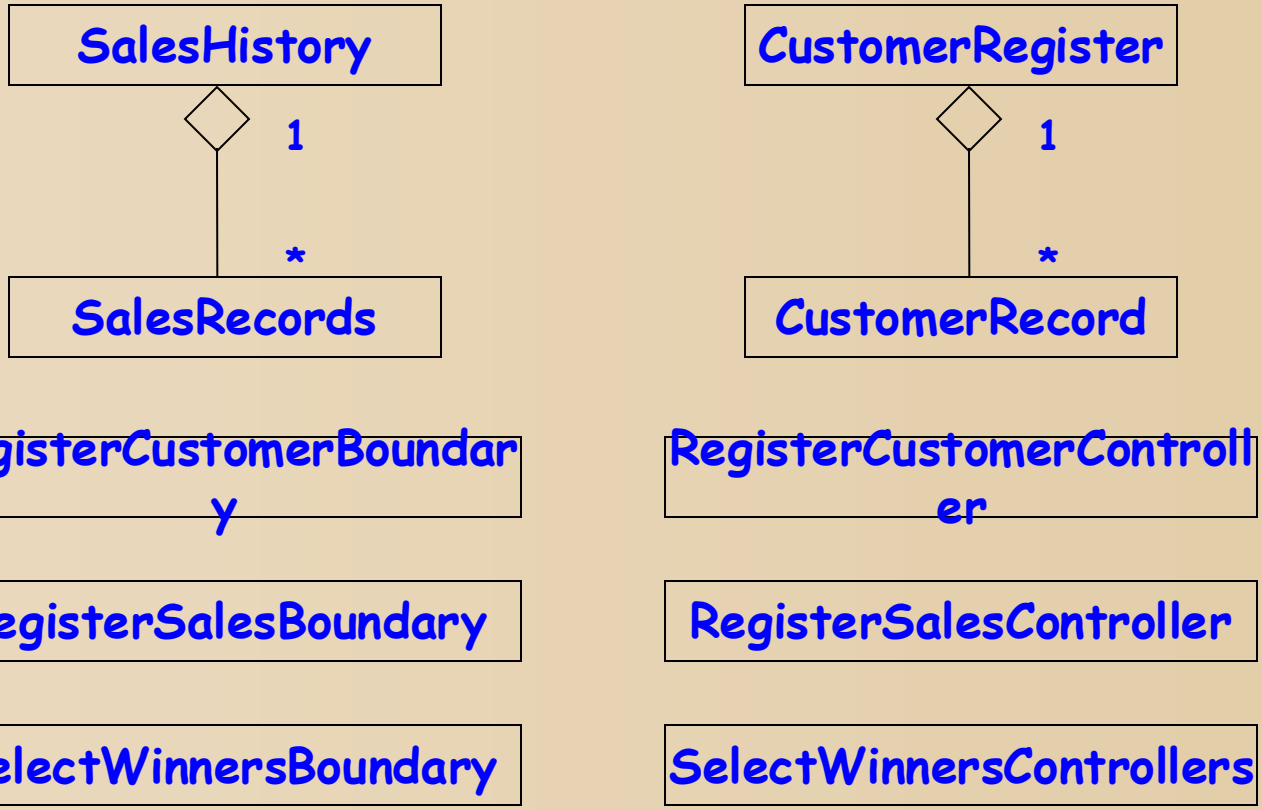


Example 2: Initial Domain Model



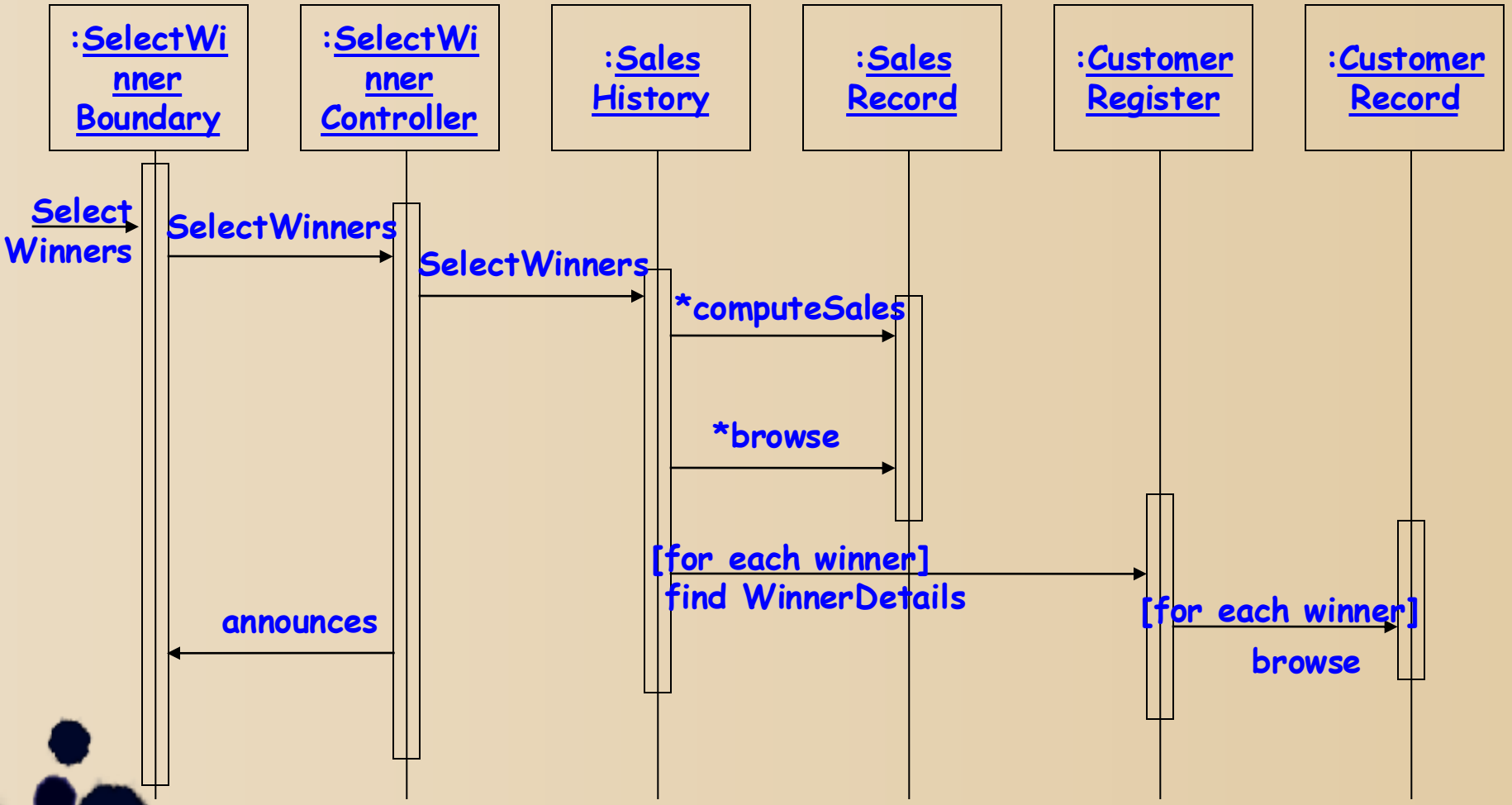
Initial domain model

Example 2: Refined Domain Model



Refined domain model

Example 2: Sequence Diagram for the Select Winners Use Case



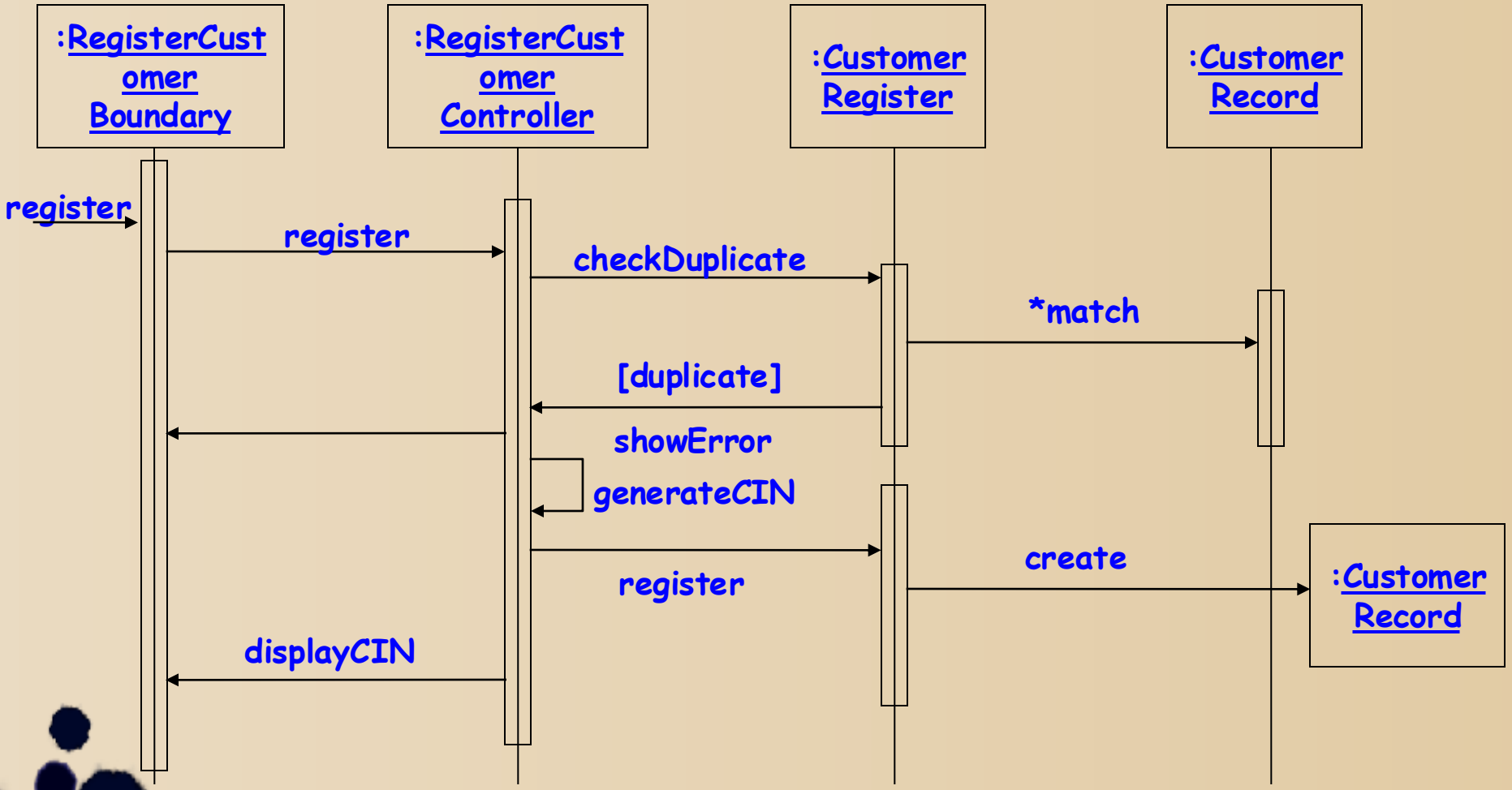
Sequence Diagram for the select winners use case

Full Message Attributes

Full Message Attributes[sequence-expression]

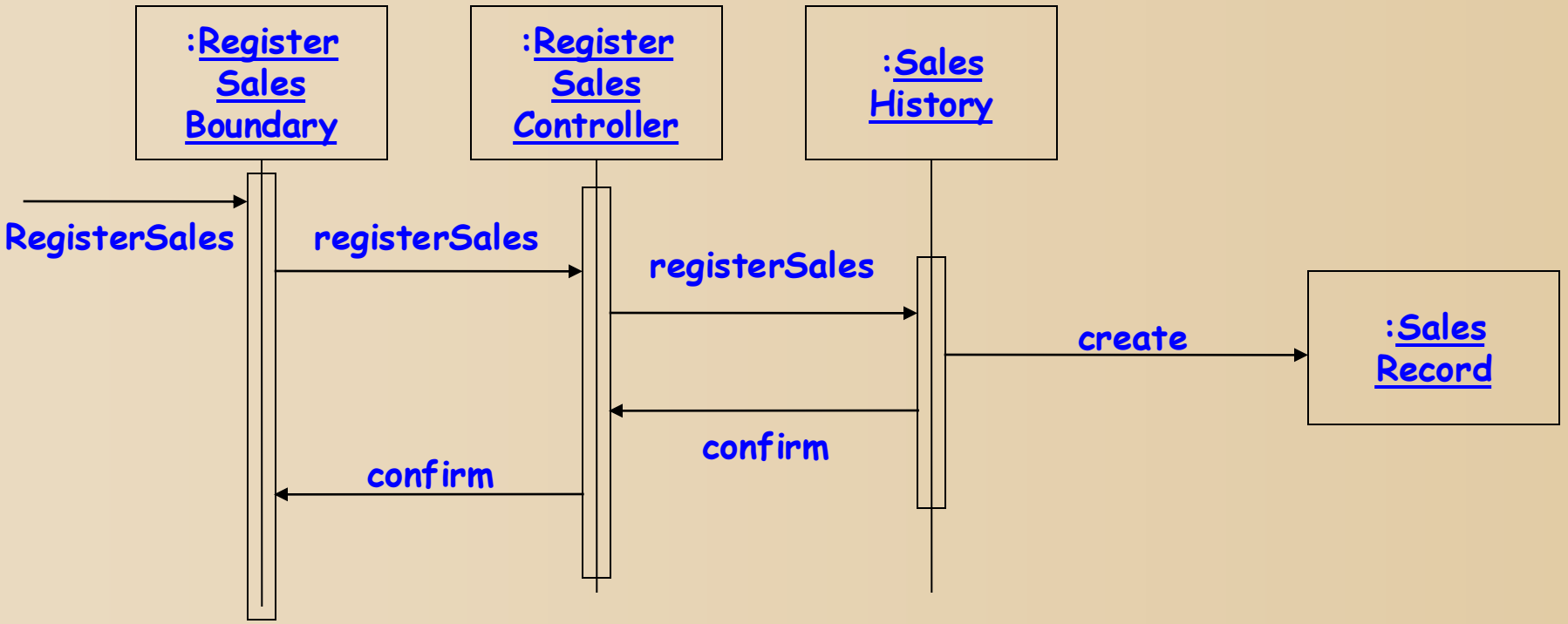
[return-value :=] [message-name] [(argument-list)]

Example 2: Sequence Diagram for the Register Customer Use Case



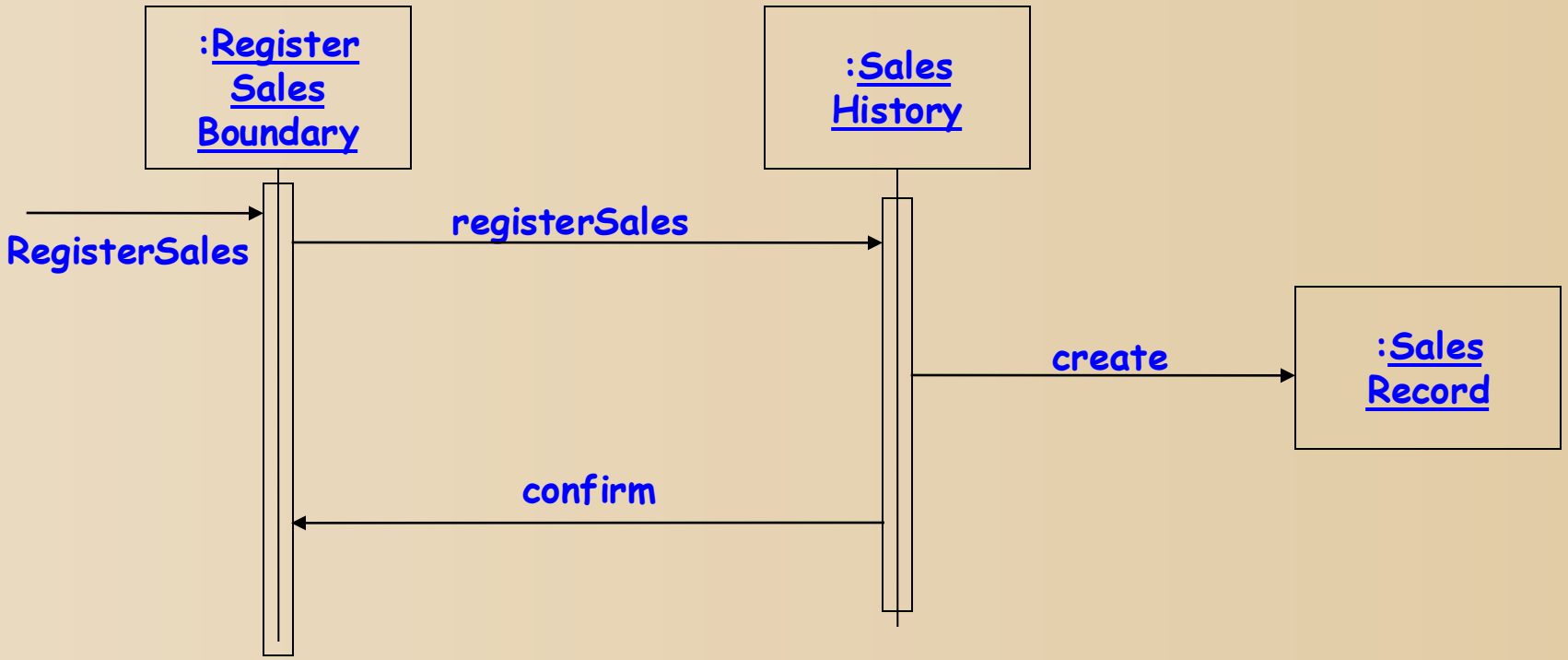
Sequence Diagram for the register customer use case

Example 2: Sequence Diagram for the Register Sales Use Case



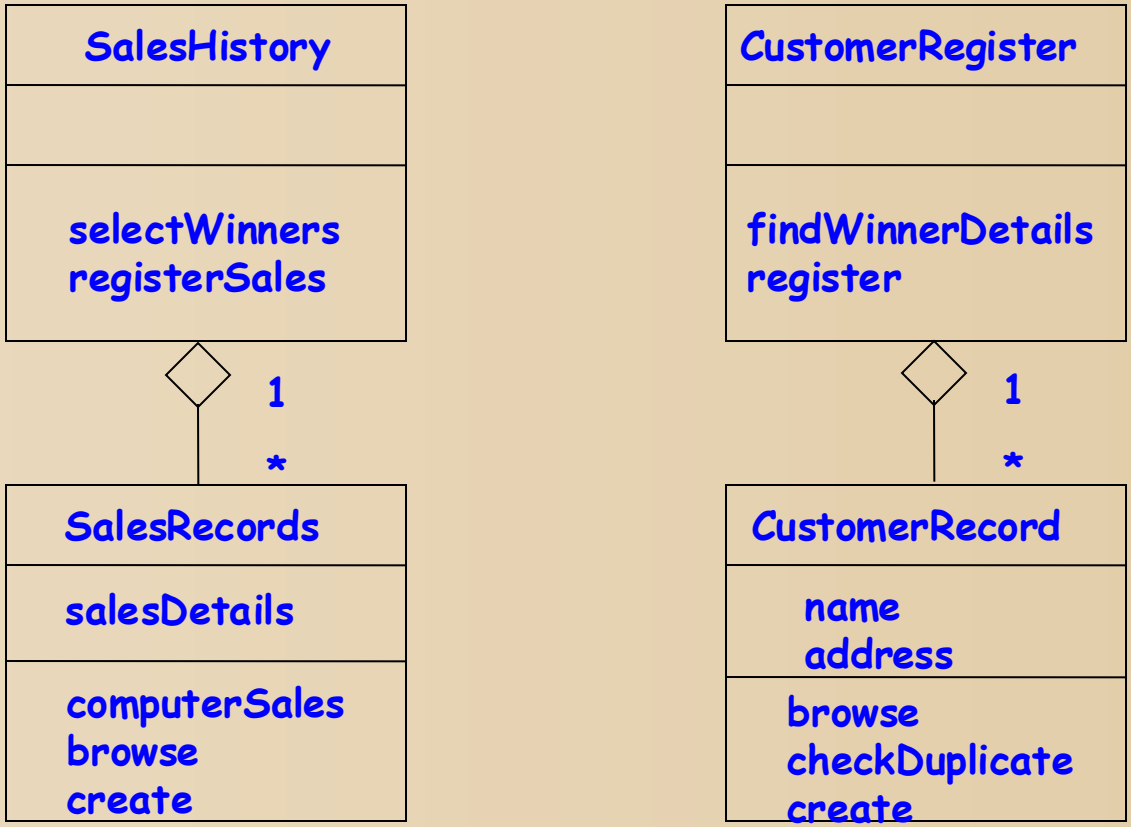
Sequence Diagram for the register sales use case

Example 2: Sequence Diagram for the Register Sales Use Case



Refined Sequence Diagram for the register sales use case

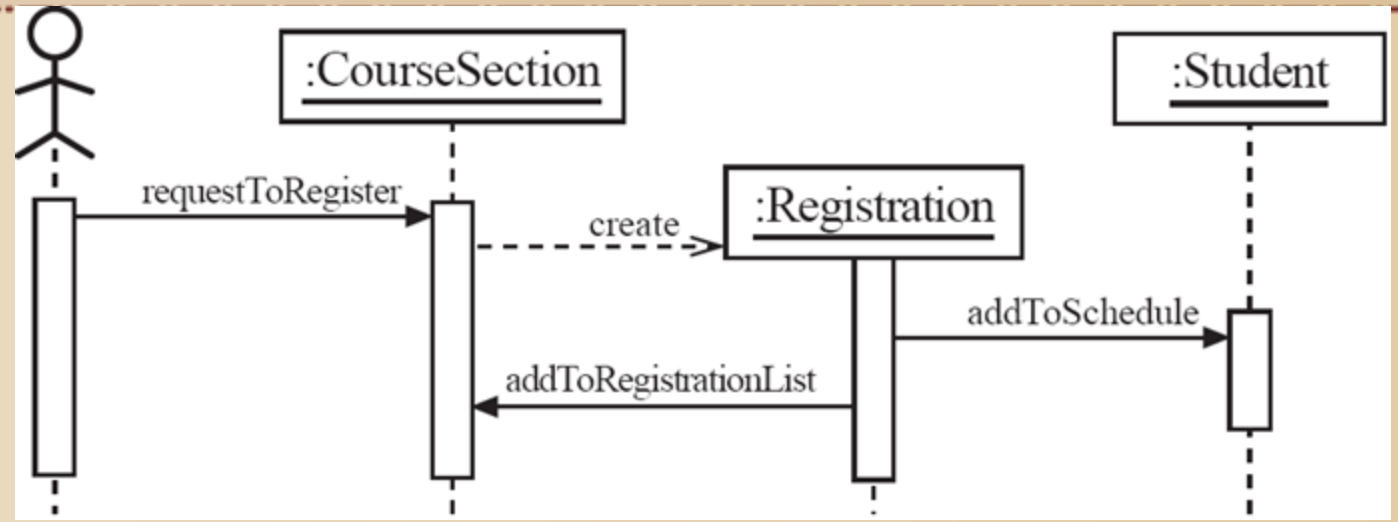
Example 2: Class Diagram

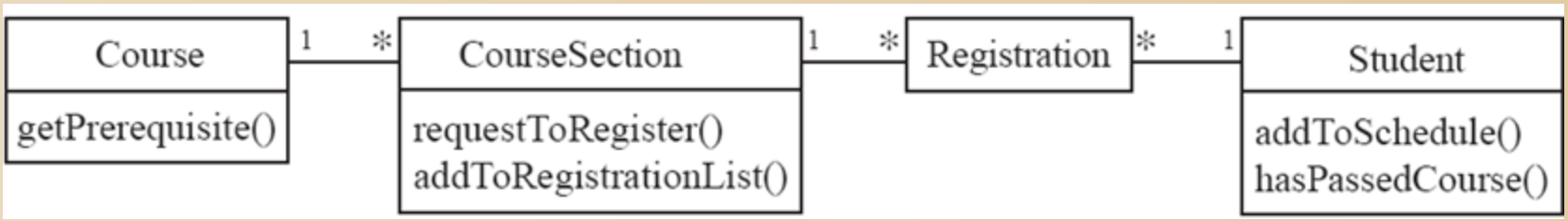


Use of Interaction Operators in Sequence diagram

- Combined fragment may have interaction **operator** also called **guards** in UML 2.4.
- Interaction **Operator** could be one of:
  - **Alt**
  - **Opt**
  - **Loop**
  - **Break**
  - **Par**

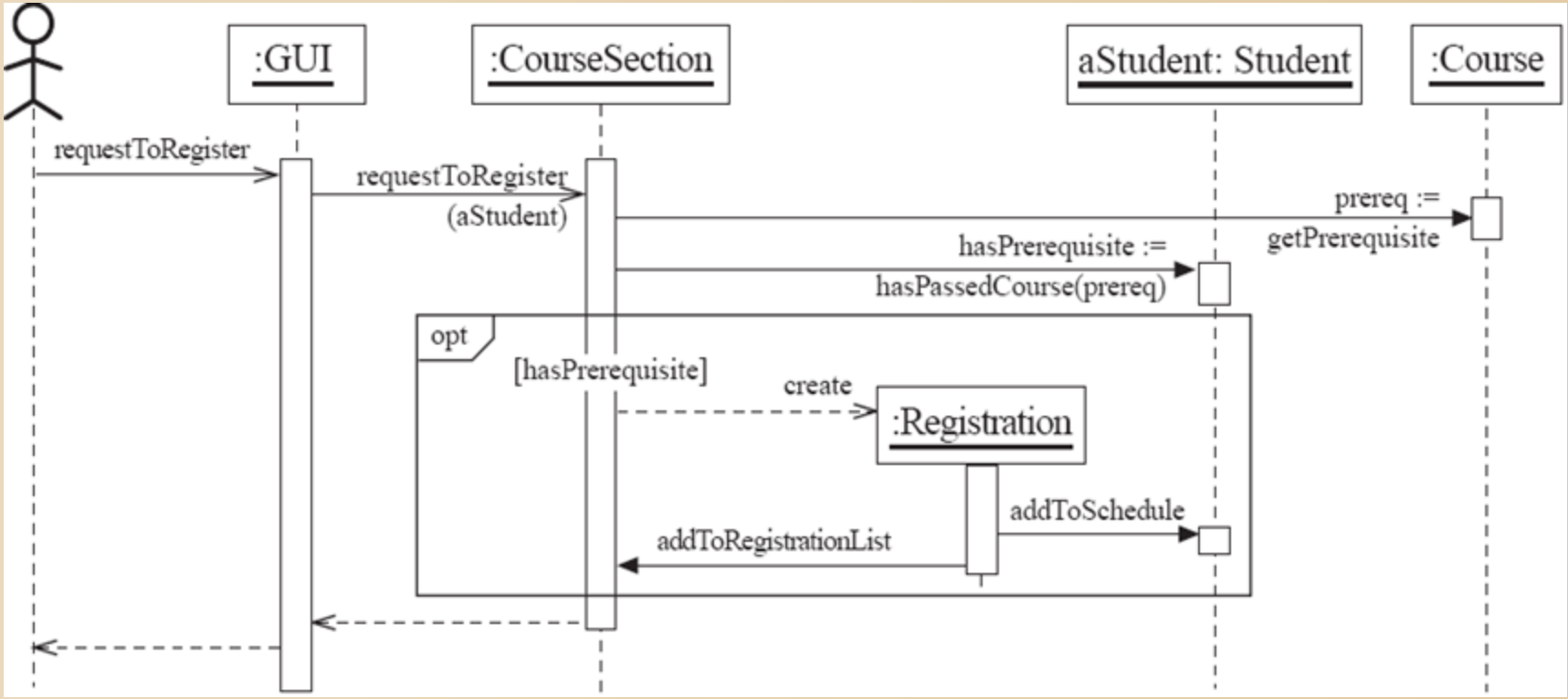
Sequence diagrams – an example of Object Creation





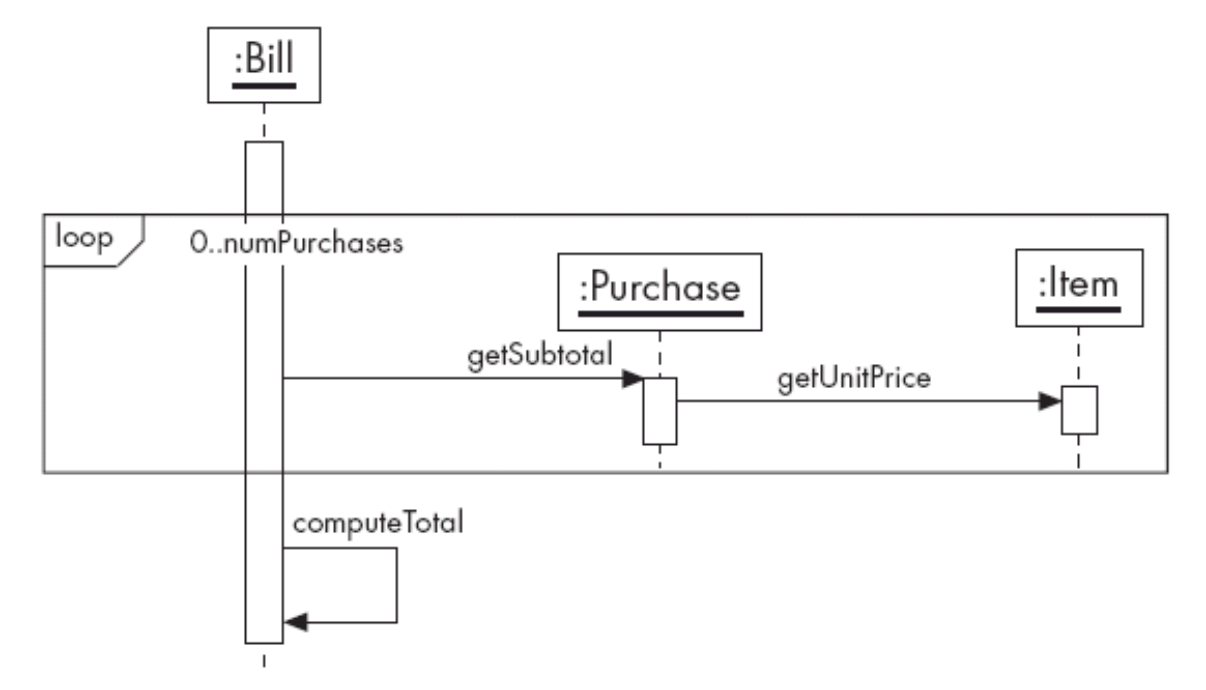
Sequence diagram of use case-Register for Course(s:Student, c:Couse)

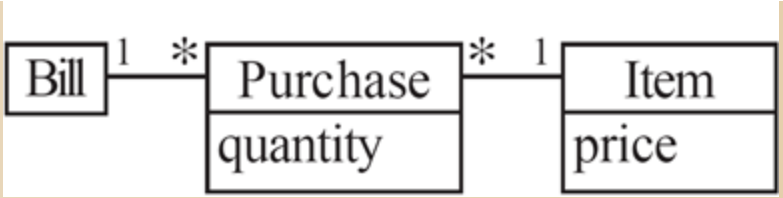
(Interaction operator: opt)



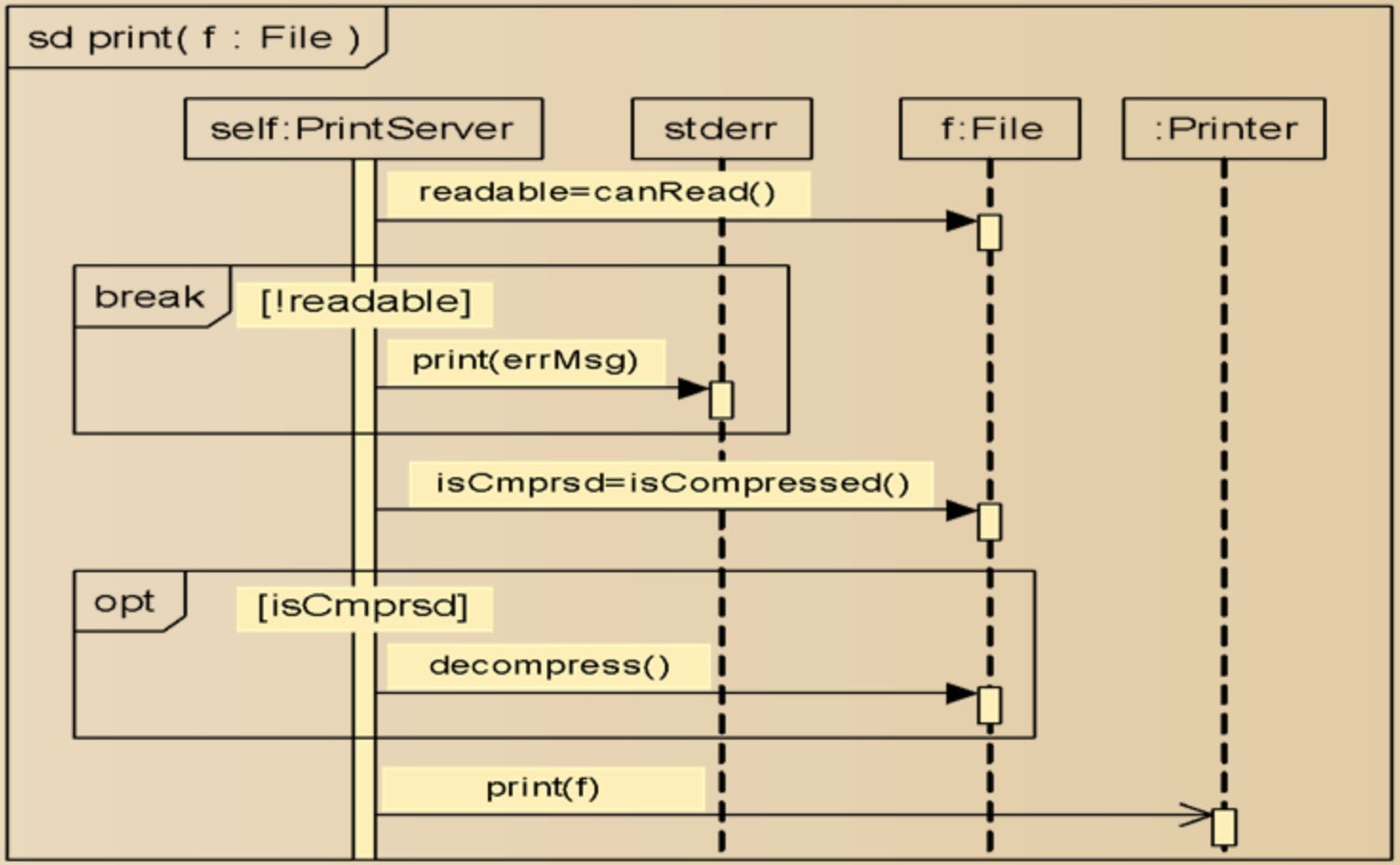
Sequence diagrams – an example with replicated messages

- An *iteration* over objects is indicated by an asterisk preceding the message name

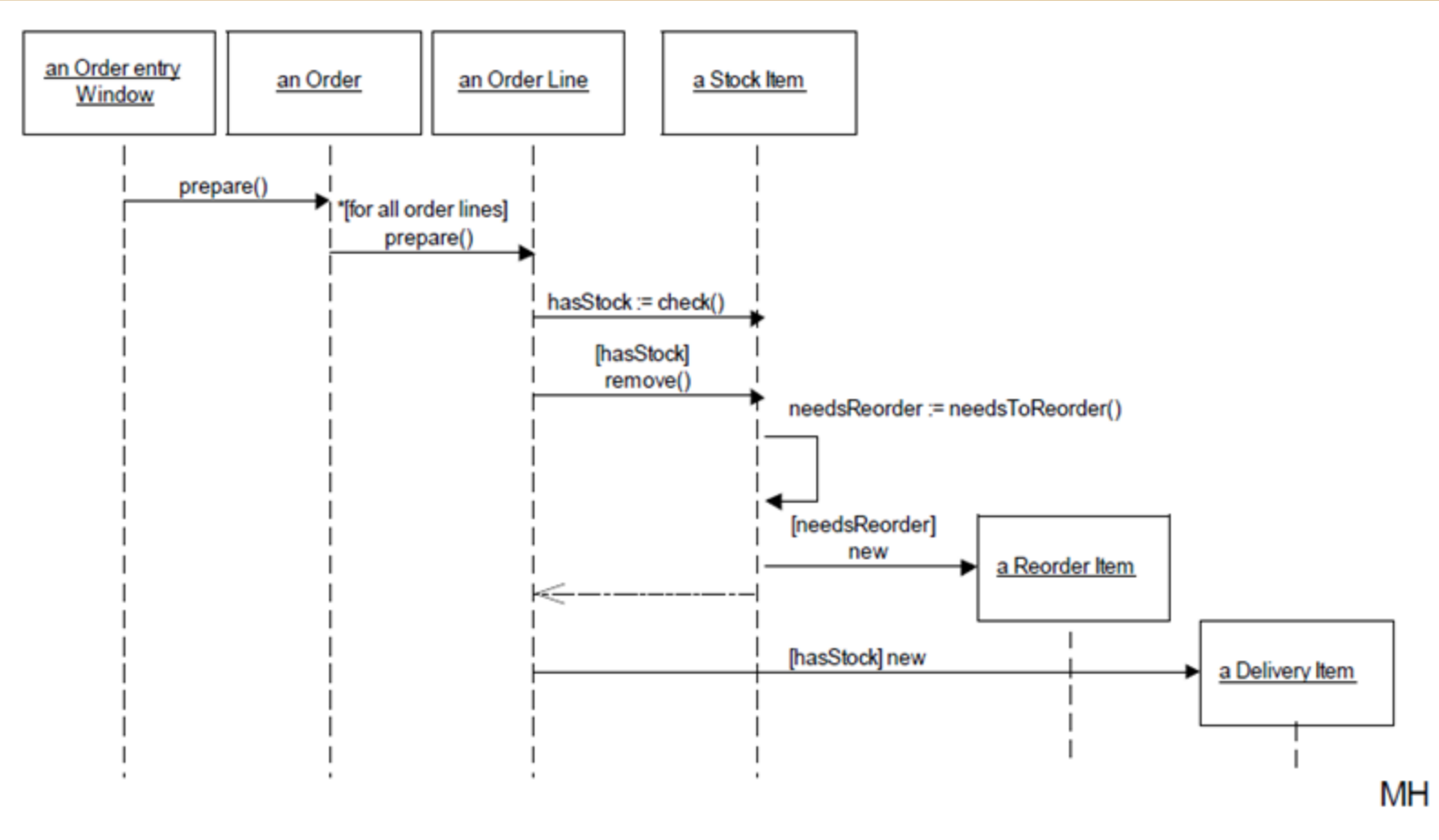




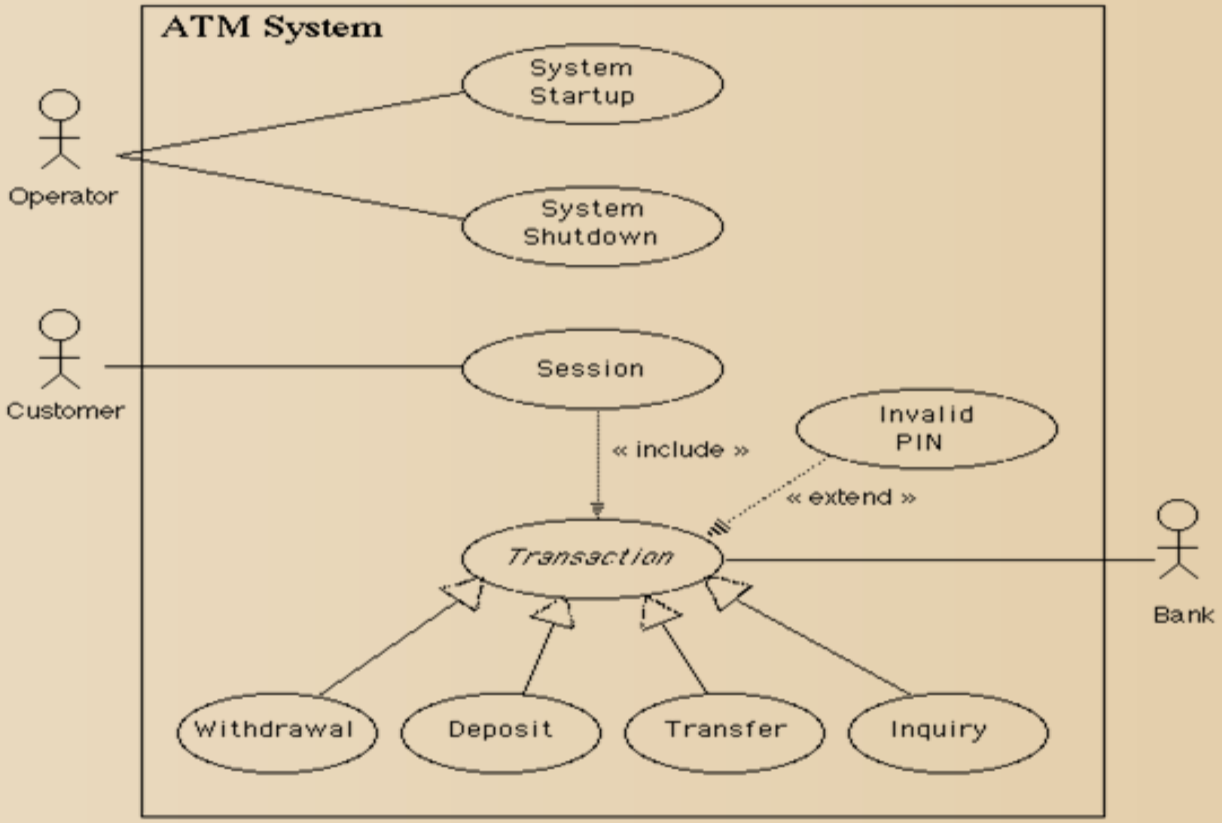
An Example Of Sequence Diagram of use case Print File(file:File)



Sequence Diagram of Order Handling System



ATM USECASE


ATM System
System Startup
System Shutdown
Session
Invalid PIN
« include »
« extend »
Transaction
Withdrawal
Deposit
Transfer
Inquiry
Operator
Customer
Bank

ATM StartUp Sequence Diagram