

# Applying UML & Patterns (3<sup>rd</sup> ed.)

## Chapter 6

### USE CASES

# Use Case Overview

- A UC is a dialogue between an Actor and a system that accomplishes a task.
- The dialogue is presented as a sequence of steps
- A complete sequence of steps is a use case scenario
  - A scenario forms a complete path thru the UC, aka UC instance.

# Use Cases

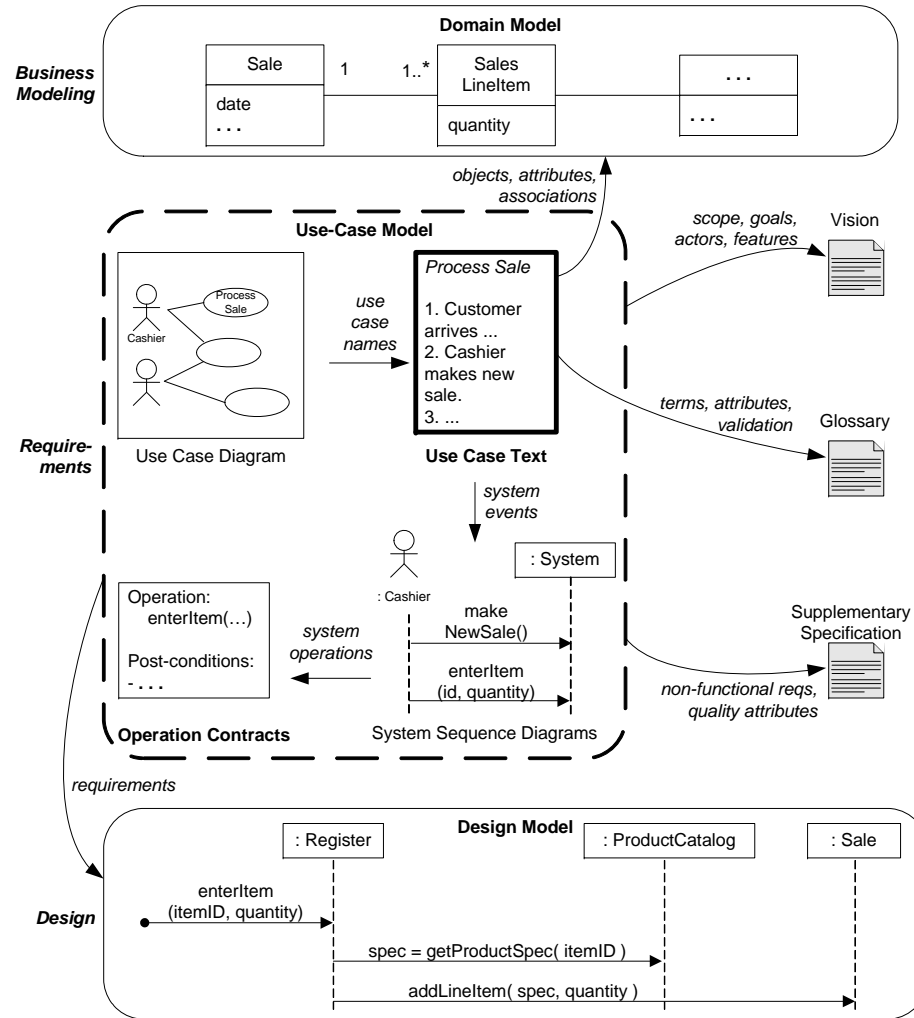
- UC can contains multiple scenarios (i.e., >1 path thru UC)
- Can range from simple (brief summary) to elaborate (detailed steps using adopted document template)
- UCs are NOT object-oriented artifacts!  
They feed into other OO models

# Use Cases

- UCs are written in simple concise text
- Complex UCs should probably be depicted using UML activity diagrams (we'll cover activity diagrams later)

# Fig. 6.1

Sample UP Artifact Relationships



# Use Cases

- UC strengths & uses
  - Emphasize user goals & objectives
  - Decompose system functionality into a set of discrete tasks (divide & conquer)
  - Easy for users to understand
  - Can be reused for user documentation
  - Test cases can be taken directly from UCs
  - Independent of implementing technology

# Use Cases

- Actor is anything with behavior that triggers UC execution
  - Primary Actor accomplishes goal via UC
    - Typically a human user, also concepts like Time
  - Secondary Actor provides a service to UC
    - 3<sup>rd</sup> party system
  - Offstage actor is interested in UC results
    - Regulatory agency

# Use Case Structure

- You can find lots of UC templates in books & on the WWW
- Specific format depends on needs of project
- If company has an official template, use it!
- Here is an acceptable general purpose UC structure

# Generic UC Structure

- UC Name in 'Verb Noun' format
  - Open Account, Enroll Student
- Description
  - 1 paragraph high-level description of the UC
- Actors
  - List all Actors who can trigger use case
- Preconditions
  - What must be true before use case can be triggered
  - Perhaps another UC must precede this one
- Postconditions
  - What must be true upon completion of the UC
- Base Scenario
  - Steps describing the sequence of actions that must occur assuming no alternative flows or error conditions
  - 'Sunny day scenario'

# Generic UC Structure

- **Alternative Scenarios**
  - State condition that precipitates branching
  - Specify sequence of steps that occur as a result
- **Additional Information**
  - Any other information or constraints that impact the UC

# Use Cases-Base Scenario

- UC steps are a dialogue between user and system, for example...
  1. User stimulates the system
  2. System responds to the user
  3. User stimulates the system again....
- Keep text simple and clear
  - Minimize articles ('a', 'an', 'the')
  - Maintain present tense
  - Always active voice
  - Ideally, 1 action per step

# Use Cases

- Larman's Process Sale UC (p. 68) is very elaborate and complex
- Contains many UC scenarios
- Difficult to comprehend in its entirety
- Should either decompose into smaller UCs
- Or, depict in an activity diagram

# Larman's UC-Alternative FYI

## **UC: Process Sale**

1. User selects new sale option
2. System requests item identifier
3. User enters item identifier
4. System records sale of item, and
5. System displays item description, price, current total  
*Steps 2-5 repeated until user finished*
6. User selects sale finished option
7. System displays total and taxes due
8. User selects payment option
9. System requests payment information
10. User enters payment information
11. System handles payment
12. System logs completed sale and sends sale information to Accounting System and Inventory System
13. System generates receipt

# Use Cases

- UCs can invoke other UCs
  - Very similar to calling a function
  - Permits reuse of UCs
  - Avoids duplication & simplifies maintenance
  - Use hyperlinks in UC doc
  - See example on p. 77

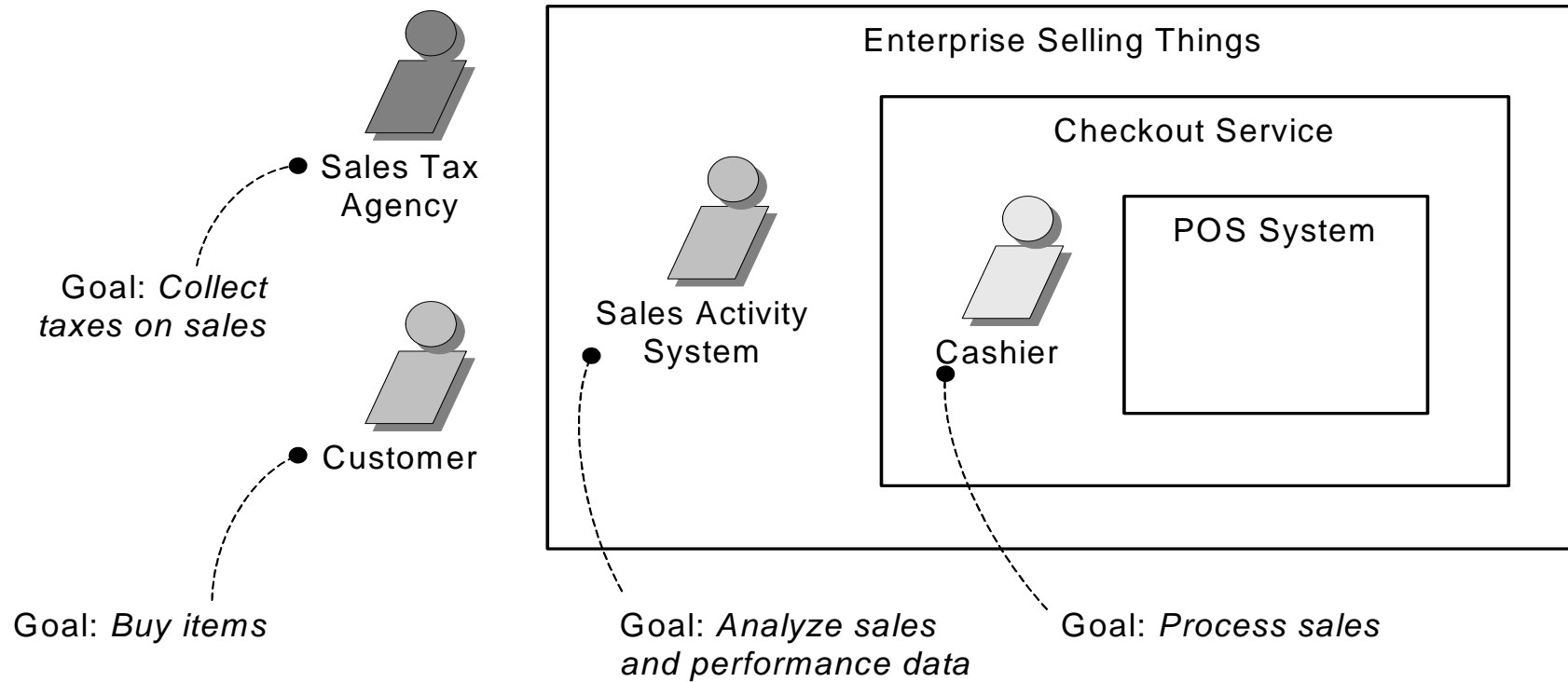
# Common UC Issues

- Granularity
  - Use elementary business process as guide
    - EBP: A task performed by 1 user in 1 place at 1 time in response to a business event, that adds measurable value to the business and leaves data in a consistent state.
- Base scenario's first step
  - Writing steps that precede the functionality to be implemented clutters the UC; may confuse developers
  - Start UC at the boundary of the system

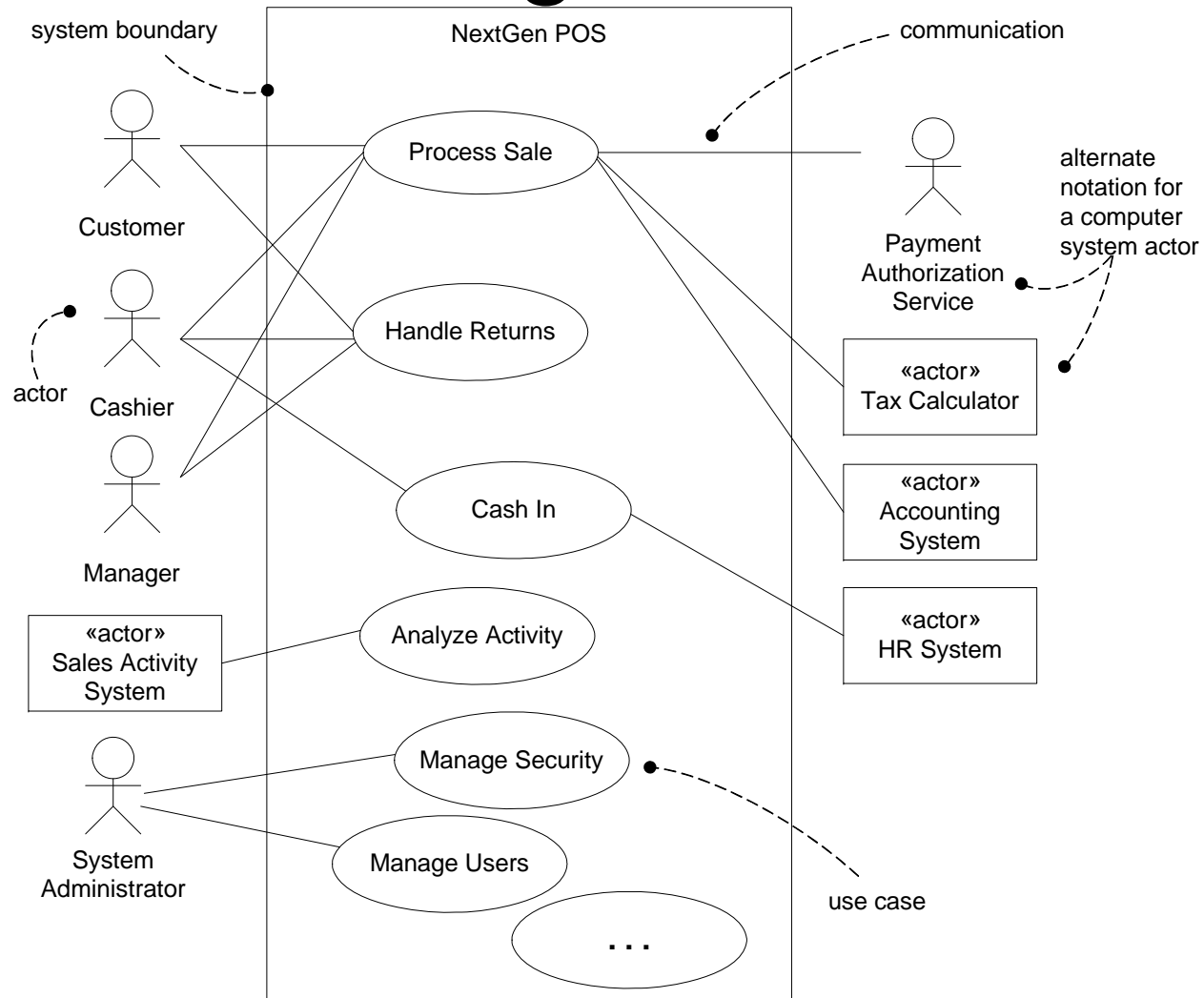
# Common UC Issues

- Write 'black box' UCs
  - Defer implementation details
  - Avoid reference to specific technologies
- Finding UCs
  - Define system boundary
  - Identify Actors and their goals
  - Construct simple Actor-Goal Matrix (p. 84)

# Fig. 6.2



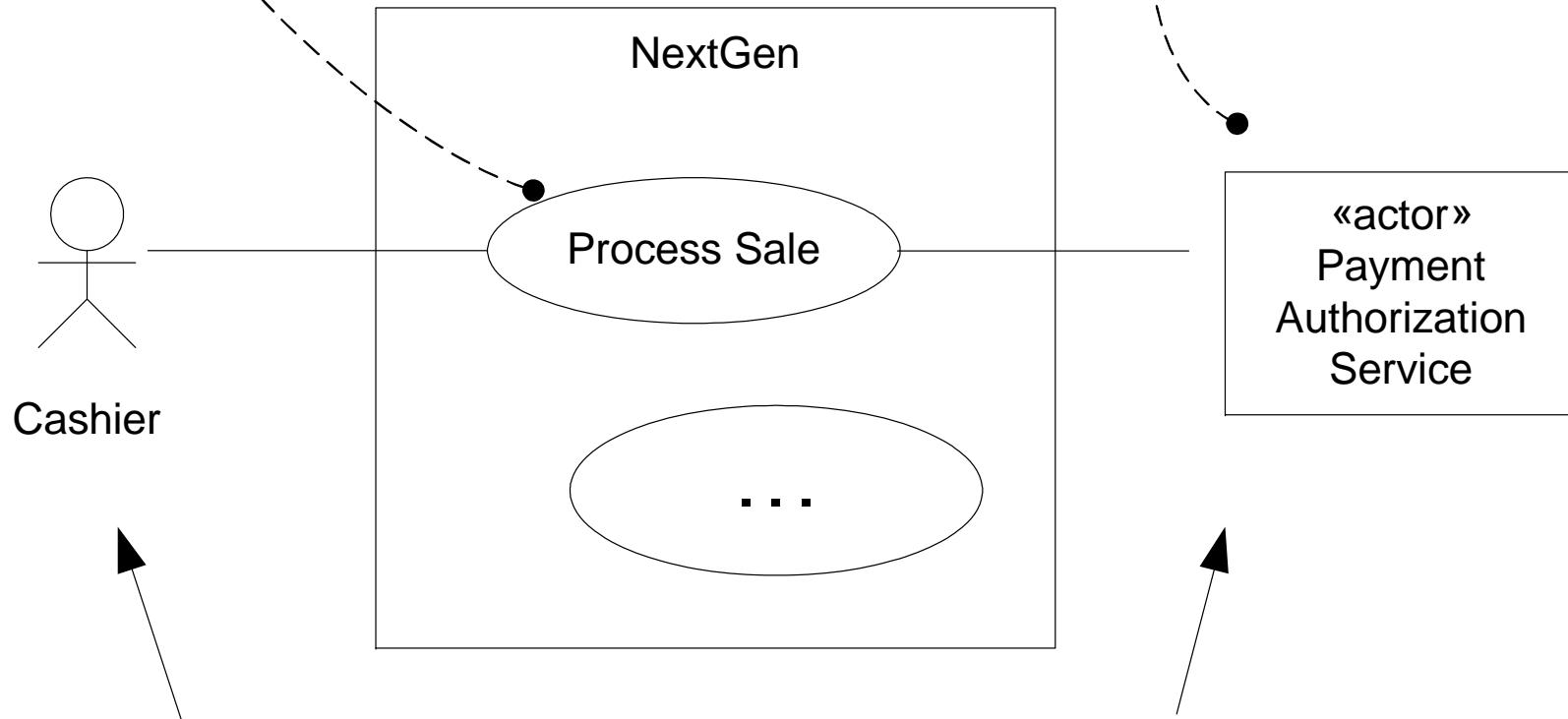
# Fig. 6.3



# Fig. 6.4

For a use case context diagram, limit the use cases to user-goal level use cases.

Show computer system actors with an alternate notation to human actors.

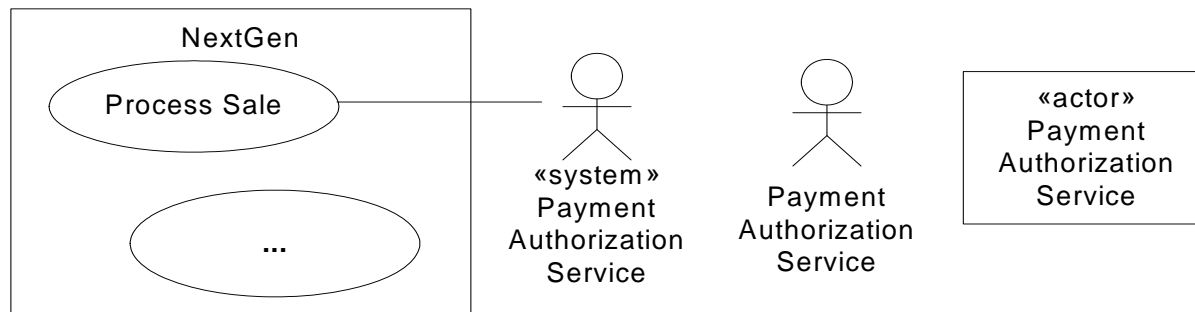


Cashier

primary actors on the left

supporting actors on the right

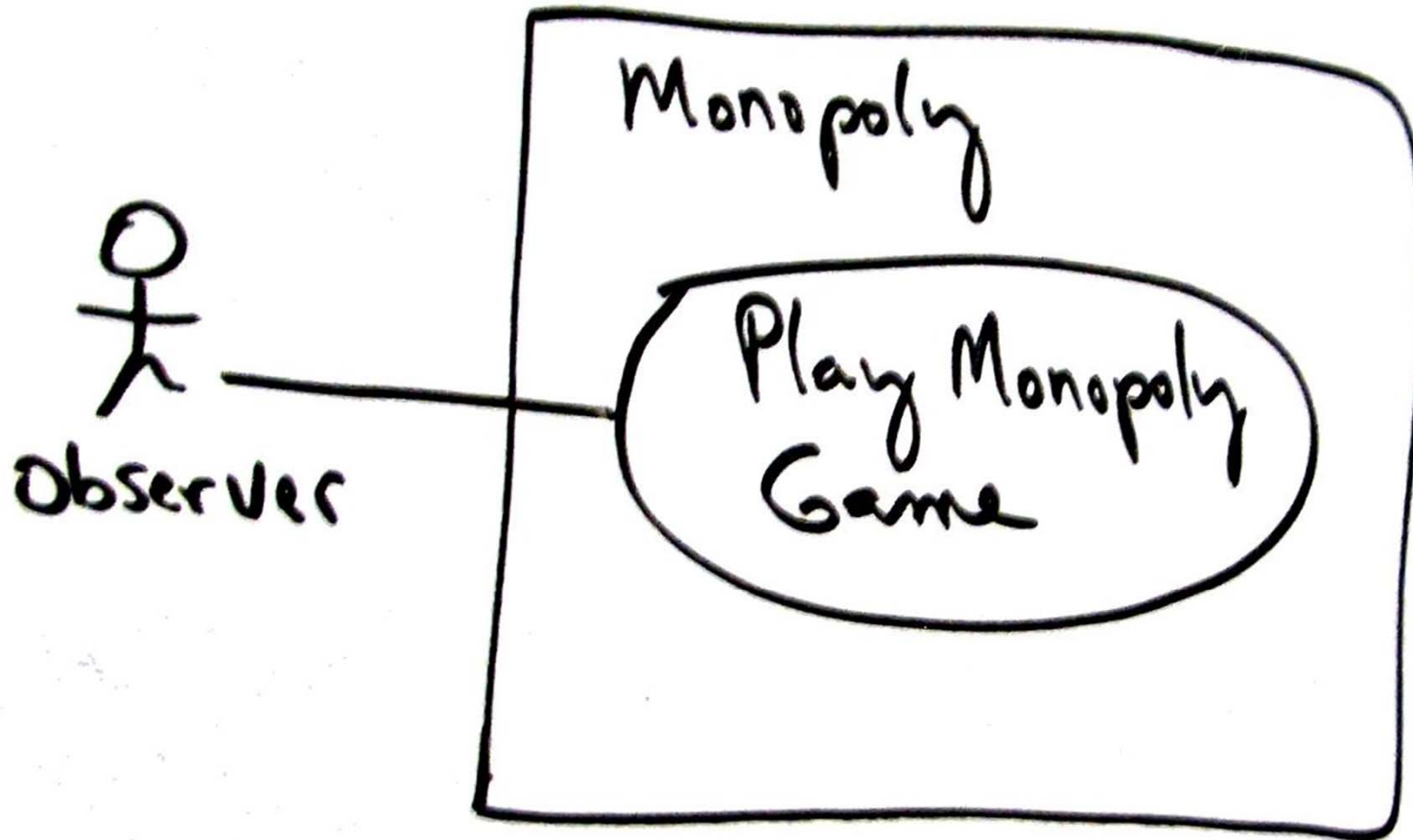
# Fig. 6.5



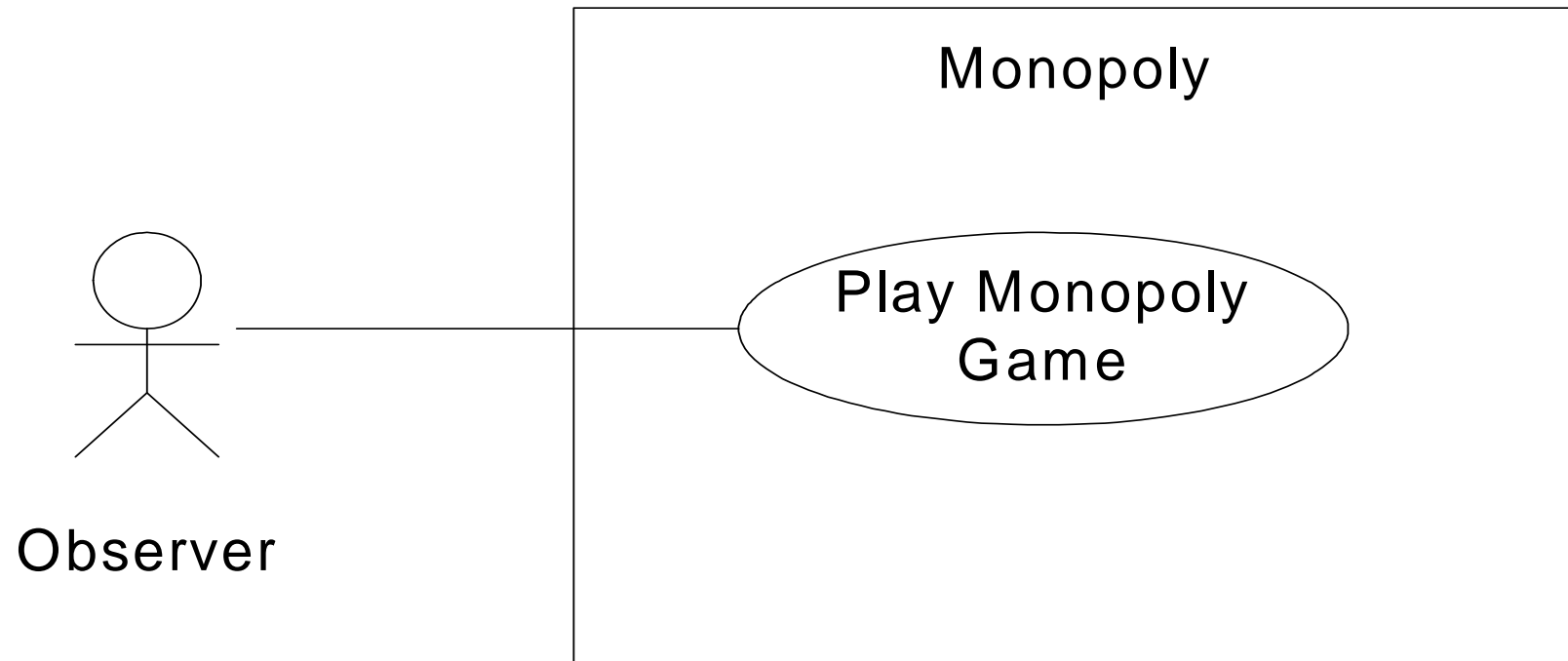
Some UML alternatives to illustrate external actors that are other computer systems.

The class box style can be used for any actor, computer or human. Using it for computer actors provides visual distinction.

# Fig. 6.6



# Fig. 6.6



# Fig. 6.7

