Content

Source: The Unified Modeling Language Reference Manual Second Edition by James Rumbaugh, Ivar Jacobson, Grady Booch

UML Introduction

Definition

The Unified Modeling Language (UML) is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system.

- It is intended for use with all development methods, lifecycle stages, application domains, and media

Unified

- Across the development lifecycle: UML is seamless from requirements to deployment. The same set of concepts and notation can be used in different stages of development and even mixed within a single model. It is unnecessary to translate from one stage to another. This seamlessness is critical for iterative, incremental development.
- Across implementation languages and platforms: The UML is intended to be usable for systems implemented in various implementation languages and plat- forms, including programming languages, databases, etc

Models

- Representation in a certain medium of something in the same or another medium
- Captures the important aspects of the thing being modeled from a certain point of view and simplifies or omits the rest
- Intended to be easier to use for certain purposes than the final system

Model Purpose

- Capture and precisely state requirements and domain knowledge so that all stakeholders may understand and agree on them
- To think about the design of a system: developers explore several architectures and design solutions easily before writing code.
- To organize, find, filter, retrieve, examine, and edit information about large systems: organizes information into several views
  - Each view is a projection of the information in the complete model as selected for a purpose

Model Contents

- Semantic information (semantics) and visual presentation (notation)
  - The semantic aspect captures the meaning of an application as a network of logical constructs, such as classes, associations, states, use cases, and messages. Semantic model elements carry the meaning of the model
  - The visual presentation shows semantic information in a form that can be seen, browsed, and edited by humans.
- Context: Models are themselves artifacts in a computer system, and they are used within a larger context that gives them their full meaning. This context includes the internal organization of the model, annotations about the use of each model in the overall development process

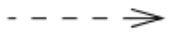
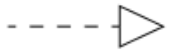
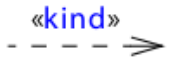
Terms

- A classifier is a modeling element that describes things containing values, e.g.,
  - Classes
  - Interfaces
  - Data types
- A package is a general-purpose organizational unit for owning and managing the contents of a model
  - Every element is owned by some package
- An object is a discrete unit out from which the modeler understands and constructs a system
  - It is an instance of a class

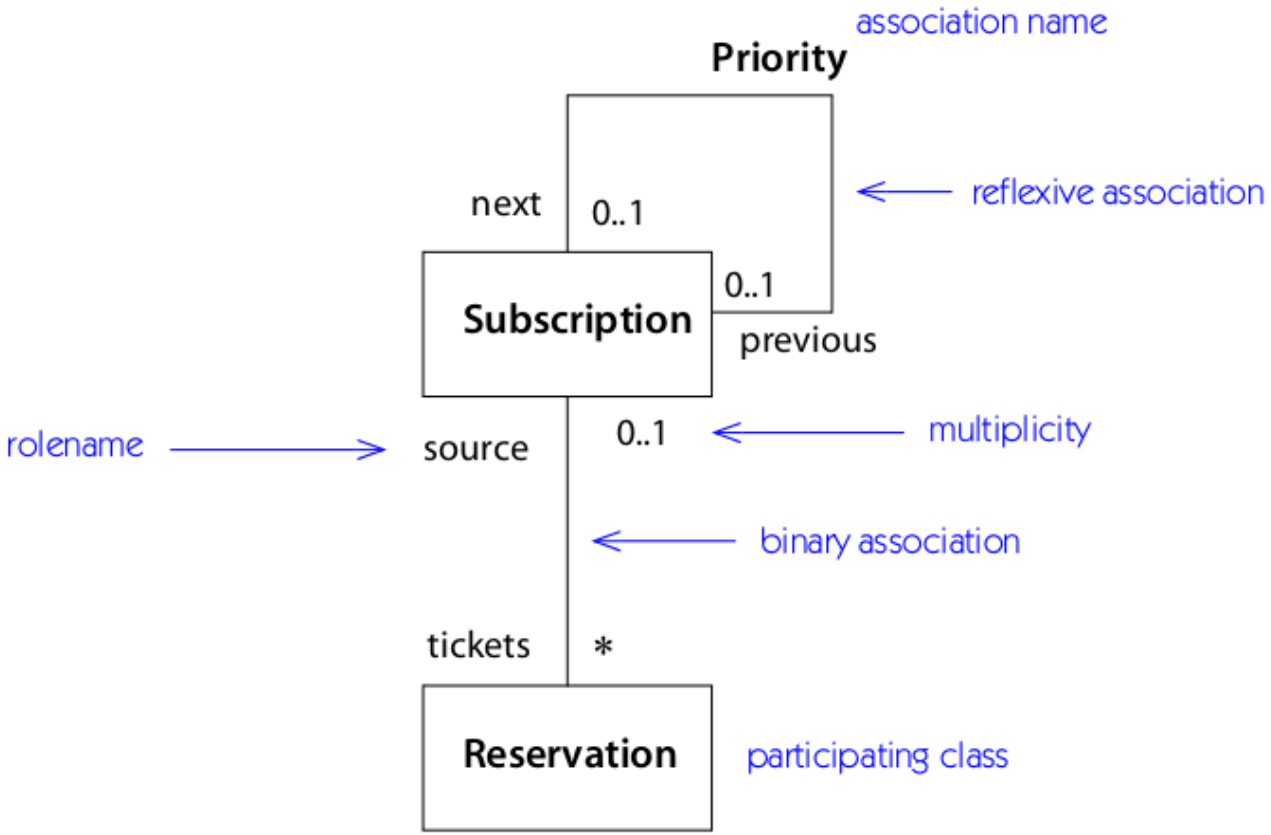
Terms

- Relationships:
  - Association: provide the connections with which objects of different classes can interact
  - Generalization: relates general descriptions of parent classifiers (superclasses) to more specialized child classifiers (subclasses), permit different classifiers to share the attributes, operations, and relationships that they have in common, without repetition.
  - Dependency: realization, usage

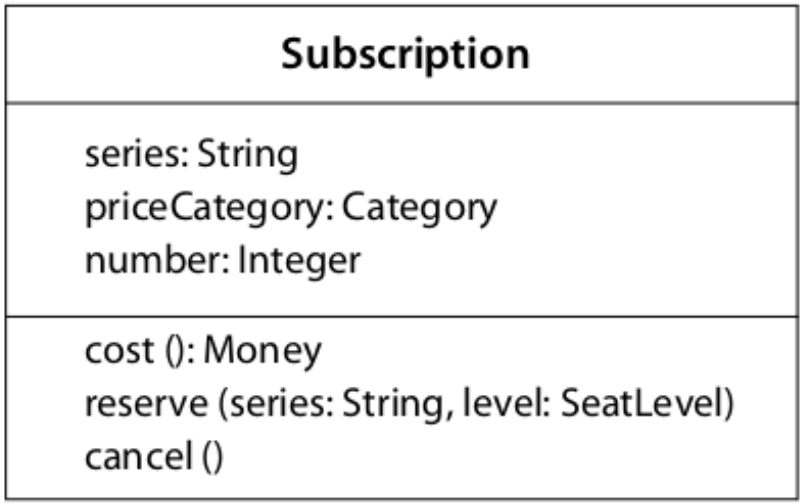
Types Of Relationships

Relationship	*Function*	*Notation*
association	A description of a connection among instances of classes	
dependency	A relationship between two model elements	
generalization	A relationship between a more specific and a more general description, used for inheritance and polymorphic type declarations	
realization	Relationship between a specification and its implementation	
usage	A situation in which one element requires another for its correct functioning	

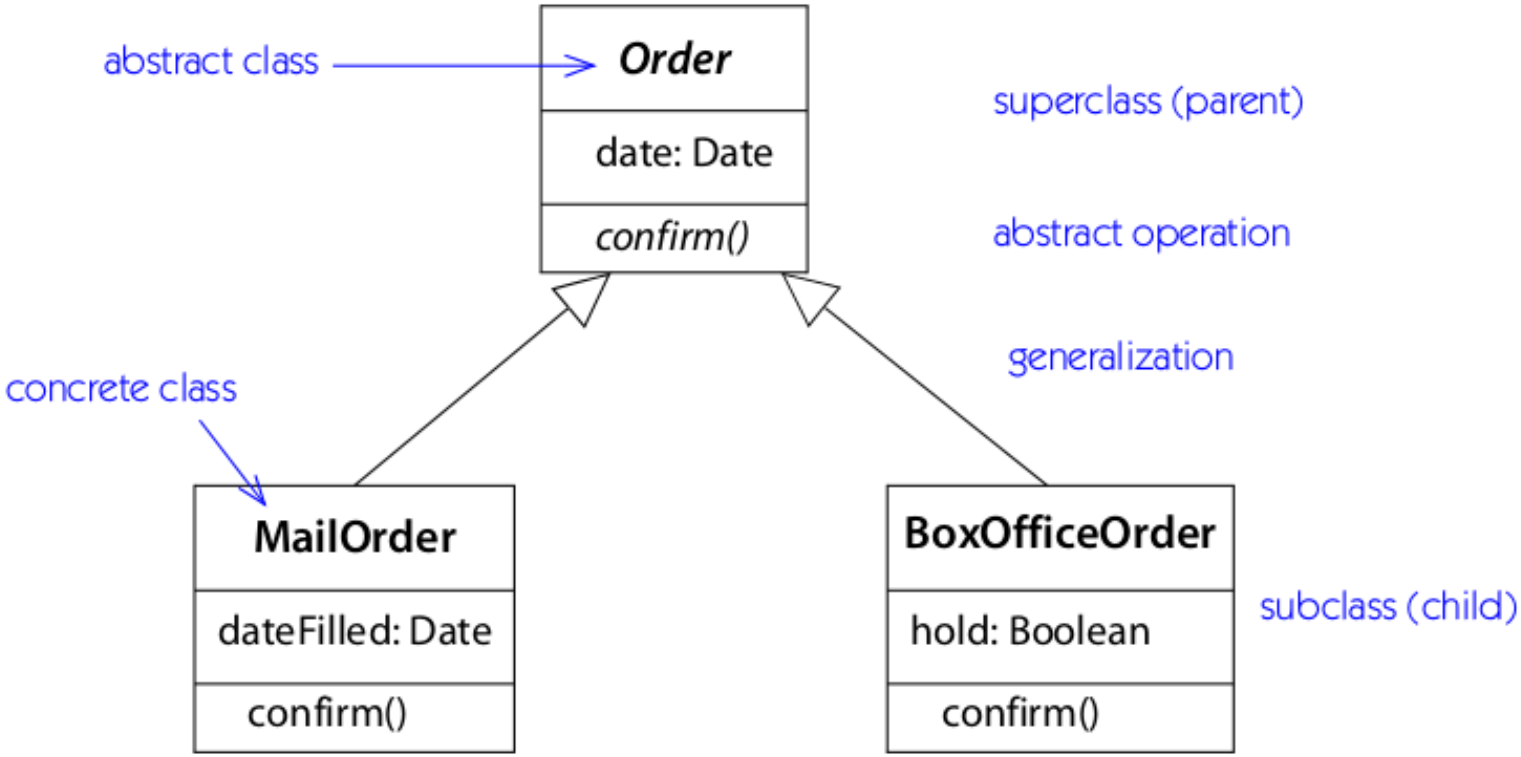
Association Notation



Class Notation



Generalization Notation



Views

- A view is simply a subset of UML modeling constructs that represents one aspect of a system
  - Static view
  - Design view: collaboration diagrams, component diagrams
  - Use case view
  - State machine view
  - Activity view

Use Case View

Definition

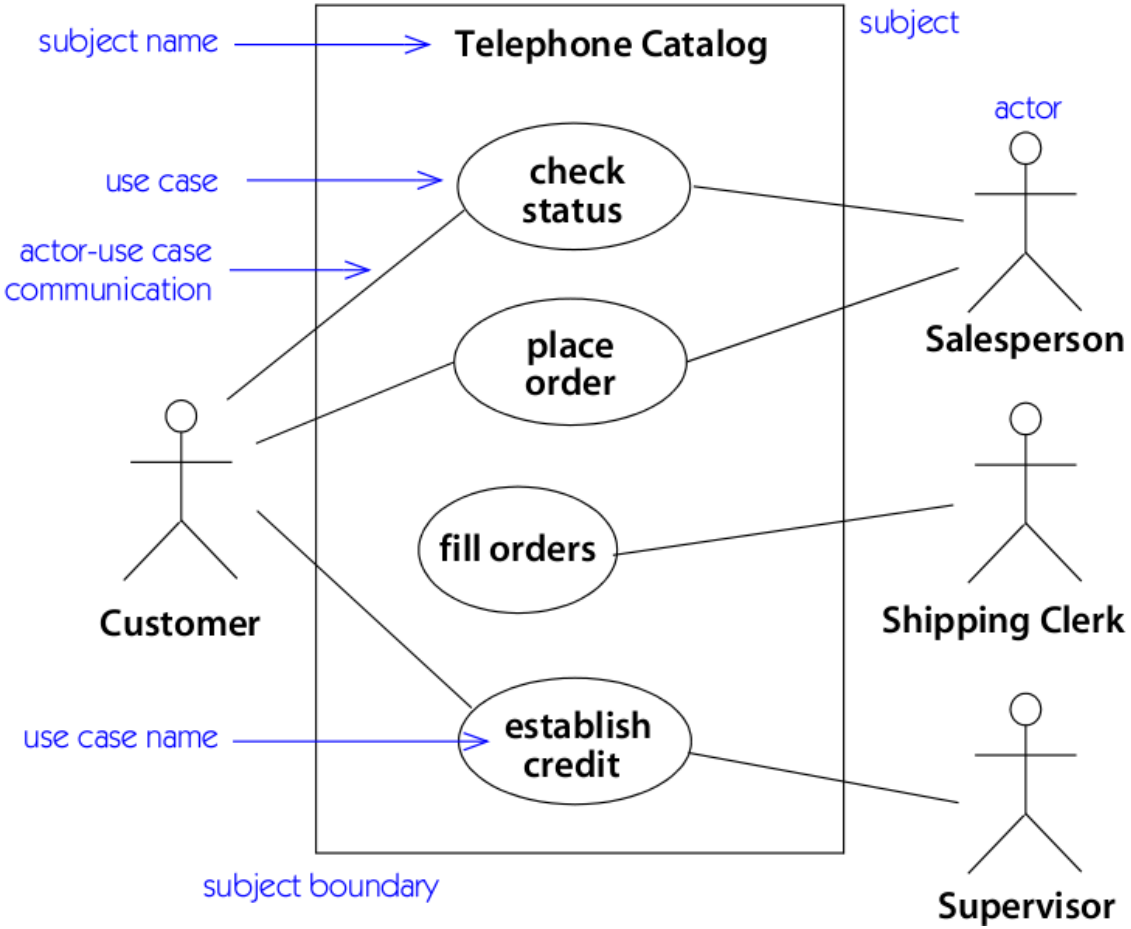
The use case view models the functionality of a subject (such as a system) as perceived by outside agents, called actors, that interact with the subject from a particular viewpoint.

- A use case is a unit of functionality expressed as a transaction among actors and the subject.
- It lists the actors and use cases and show which actors participate in each use case.
- The behavior of use cases is expressed using dynamic views, particularly the interaction view

Use Case Diagrams

- captures the behavior of a system, subsystem, class, or component as it appears to an outside user
- A use case describes an interaction with actors as a sequence of messages between the system and one or more actors.
- The term actor includes humans, as well as other computer systems and processes.

Example


subject name
Telephone Catalog
subject
actor
use case
check status
actor-use case communication
place order
Salesperson
Customer
fill orders
Shipping Clerk
use case name
establish credit
Supervisor
subject boundary

Actor

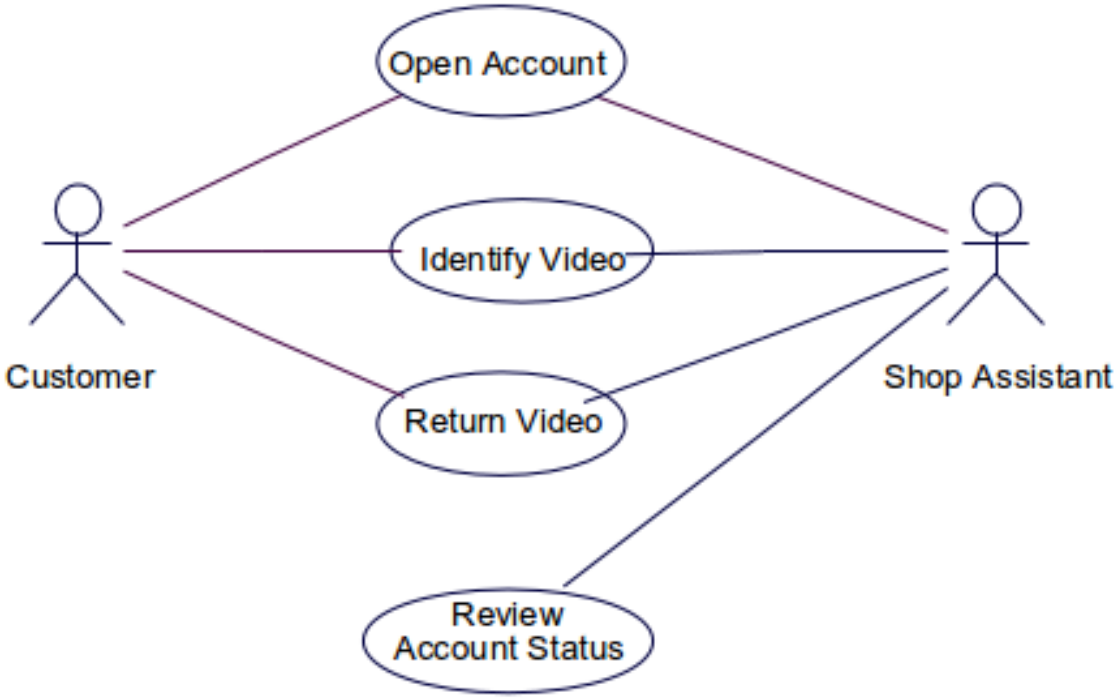
- An actor characterizes the interactions that a class of outside users may have with the system
- One physical user may be bound to multiple actors within the system e.g., one person may be a customer and a cashier of a store at different times.
- Different users may be bound to the same actor and therefore represent multiple instances of the same actor definition
- Each actor participates in one or more use cases

Interaction

- Actors interact with the use case (and therefore with the system or class that owns the use case) by exchanging messages
- An actor may be a human, a computer system, or some executable process
- An actor is drawn as a small stick person with the name below it

Example 2

Video rental store



Use Case

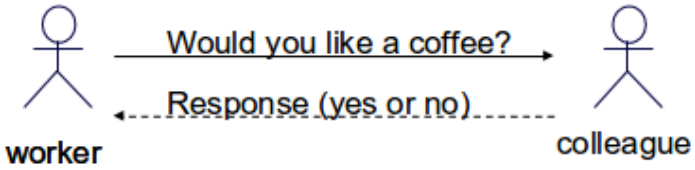
Definition

A use case is a coherent unit of externally visible functionality provided by a classifier (called the subject) and expressed by sequences of messages exchanged by the subject and one or more actors of the system unit.

- Purpose of a use case is to define a piece of coherent behavior without revealing the internal structure of the subject
- The definition of a use case includes all the behavior it entailsthe mainline sequences, different variations on normal behavior, and all the exceptional conditions that can occur with such behavior, together with the desired response.

Interaction View: Sequence Diagram

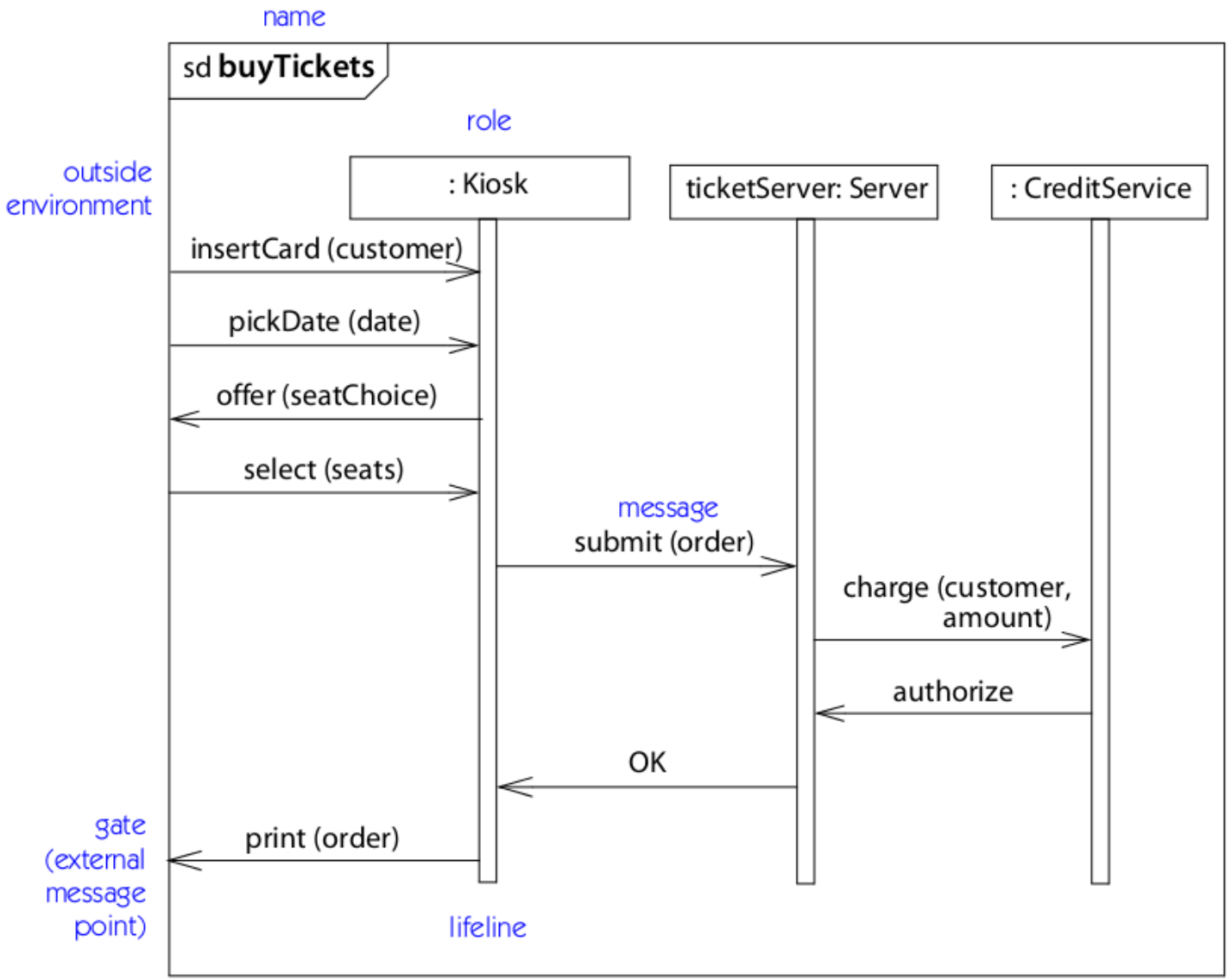
- Shows a set of messages arranged in time sequence
- Each role is shown as a lifelinethat is, a vertical line that represents the role over time through the entire interaction
- Messages are shown as arrows between lifelines
- Shows the behavior sequence of a use case (specifies the dynamics)



Sequence Diagram

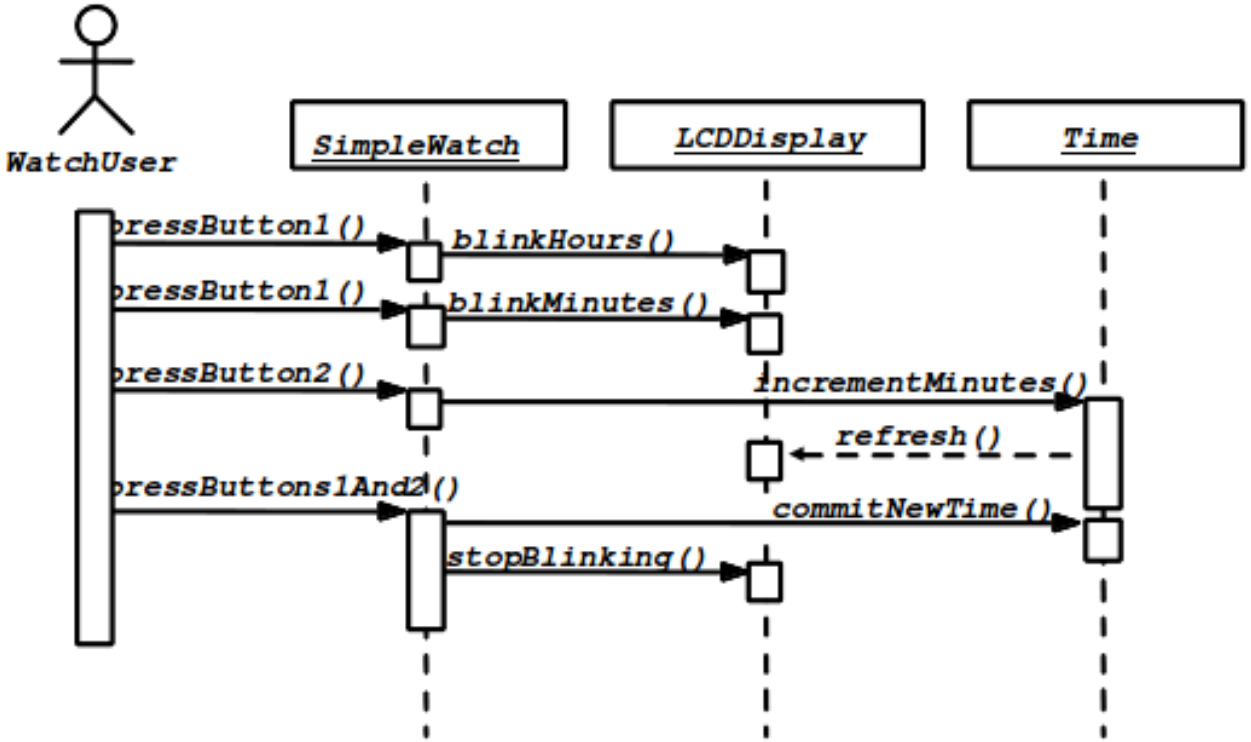
- Displays an interaction as a two-dimensional chart
- The vertical dimension is the time axis; time proceeds down the page
- The horizontal dimension shows the roles that represent individual objects in the collaboration
- Each role is represented by a vertical column containing a head symbol and a vertical line - a lifeline
- During the time an object exists, it is shown by a dashed line
- A message is shown as an arrow from the lifeline of one object to that of another. The arrows are arranged in time sequence down the diagram.

Example 1

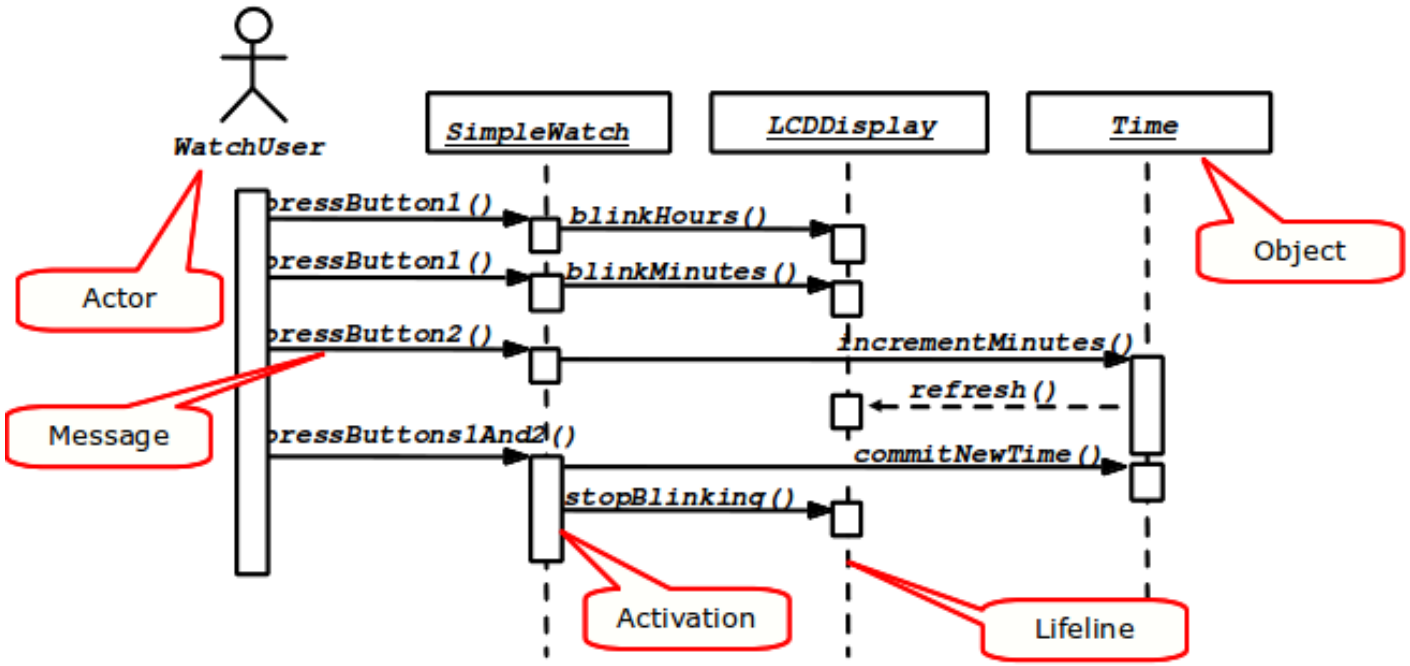

name
sd buyTickets
role
outside environment
: Kiosk
ticketServer: Server
: CreditService
insertCard (customer)
pickDate (date)
offer (seatChoice)
select (seats)
message
submit (order)
charge (customer, amount)
authorize
OK
print (order)
gate (external message point)
lifeline

Example 2

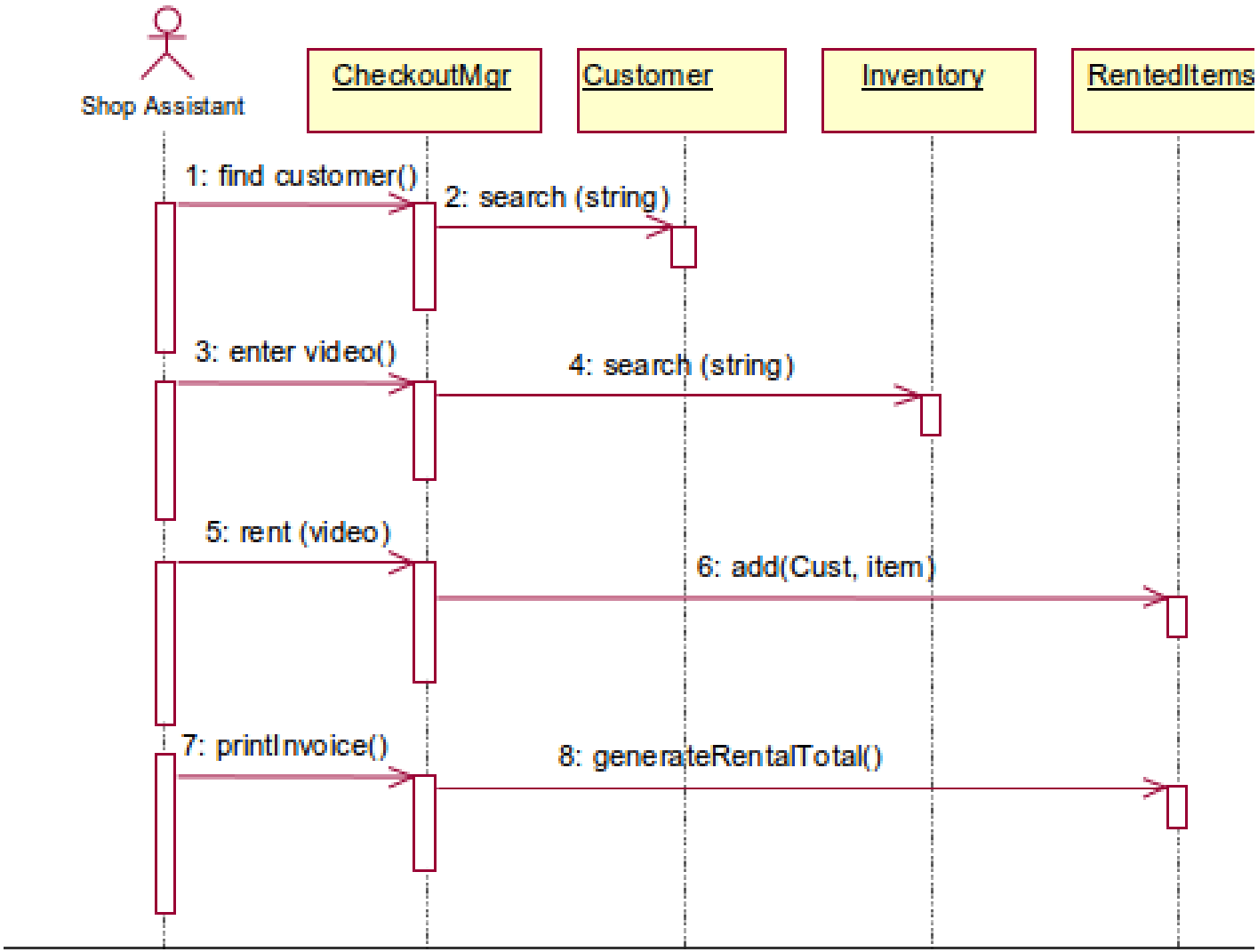
Time setting on a two button watch



Example 2 cont



Sequence Diagram for Rent Video



Exercise

Describe a scenario for a passenger lift (elevator) and model:

1. a use case
2. a sequence diagram