COMP6901
Modelling with UML

Kyle E DeFreitas
Department of Computing and Information Technology
University of the West Indies

Why Use Models

- Representation easier to understand than large amounts of text
- Gives overview of information quicker and communicate ideas faster
- However note that models are not a replacement for details.

Model Perspectives

- An external perspective
  - Model of the context or environment of the system.
- An interaction perspective
  - Model of the interactions between the components of a system.
- A structural perspective
  - Model of the organization of a system or the structure of the data that is processed by the system.
- A behavioral perspective
  - Model of the dynamic behavior of the system and how it responds to events.

Model Use Context

- Models of the **existing system** are used during **requirements engineering**.
  - They help clarify what the existing system does
  - Models can be used as a basis for identifying the existing system's strengths and weaknesses.
  - Aim in identifying requirements for the new system.

Model Use Context

- Models of the **new system** are used during **requirements engineering** and **design stages**
  - Explain the proposed requirements to other system stakeholders.
  - Enable discussions and sharing of design proposals
  - Provide documentation for the implementation of the system.

Common Diagram Types

- Because the purpose of diagrams is to communicate ideas quickly and efficiently it is important to use standards.
- The standard diagram model is Unified Modeling Language (UML)
- UML is primarily used for Object Oriented Modeling
  - (i.e. the use of representing the conceptual ideas and components of the project as software artifacts)

UML - Unified Modeling Language

The Unified Modelling Language is a standard visual modelling language intended to be used for:

- Modelling business and similar processes
- Analysis, design and implementation of software-based systems

Source: http://www.uml-diagrams.org/

UML - Unified Modeling Language

UML can be applied to diverse application domains and can be utilized with all major object and component software development methods.

UML is intentionally process independent, however most suitable for use case driven, iterative and incremental development processes. (RUP, Scrum)

Common Diagram Types

- Many diagrams within UML however the five most commonly used types are:
  - Activity diagrams
    - show the activities involved in a process or in data processing.
  - Use case diagrams
    - show the interactions between a system and its environment.
  - Sequence diagrams
    - show interactions between actors and the system and between system components.
  - Class diagrams
    - show the object classes in the system and the associations between these classes.
  - State diagrams
    - show how the system reacts to internal and external events.

Common Diagram Types

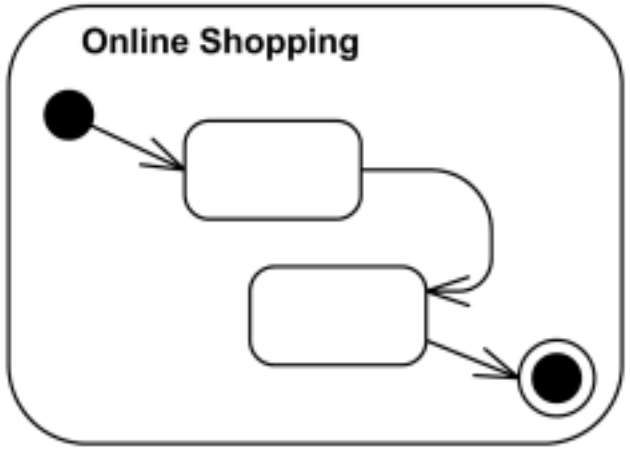
- Diagrams are divided into groups based on their purpose and information that they communicate:

Type	Diagrams
Structural	Class Diagrams
Behavioral	Activity Diagram Sequence Diagram Use Case Diagram State Diagram

Activity Diagrams

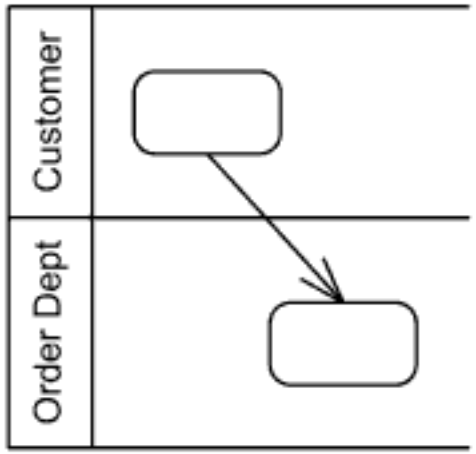
- Activity diagrams illustrate the dynamic nature of a system by modeling the flow of control from activity to activity.
  - An activity represents an operation on some class in the system that results in a change in the state of the system. (OR)
  - Activity is a parameterized behavior represented as coordinated flow of actions.
- Activity diagrams are used to model workflow or business processes and internal operations

Activity Diagram - Activity



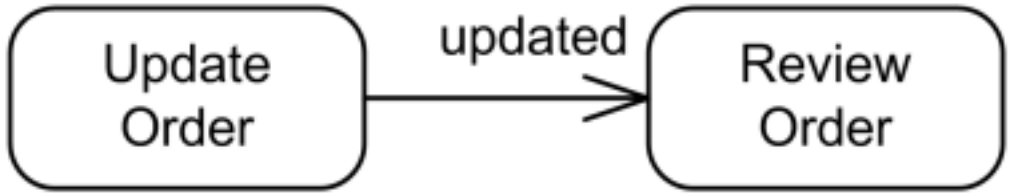
Online Shopping activity

Activity Diagram - Partition



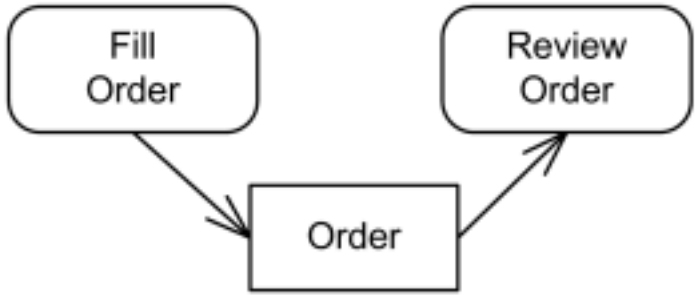
Activity partitions Customer and Order Dept

Activity Diagram - Edge



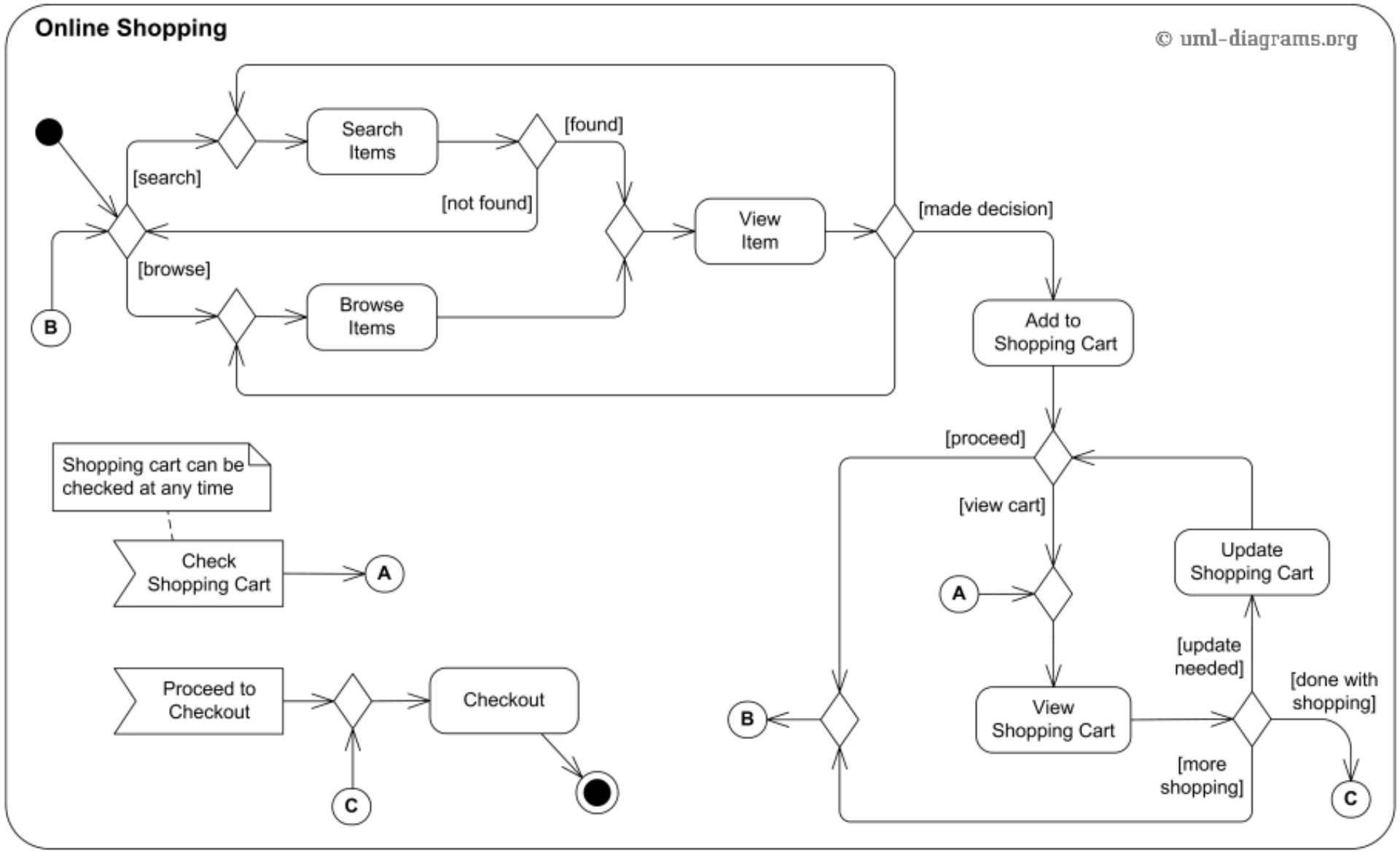
Activity edge *"updated" connects two nodes.*

Activity Diagram – Object Flow Edge

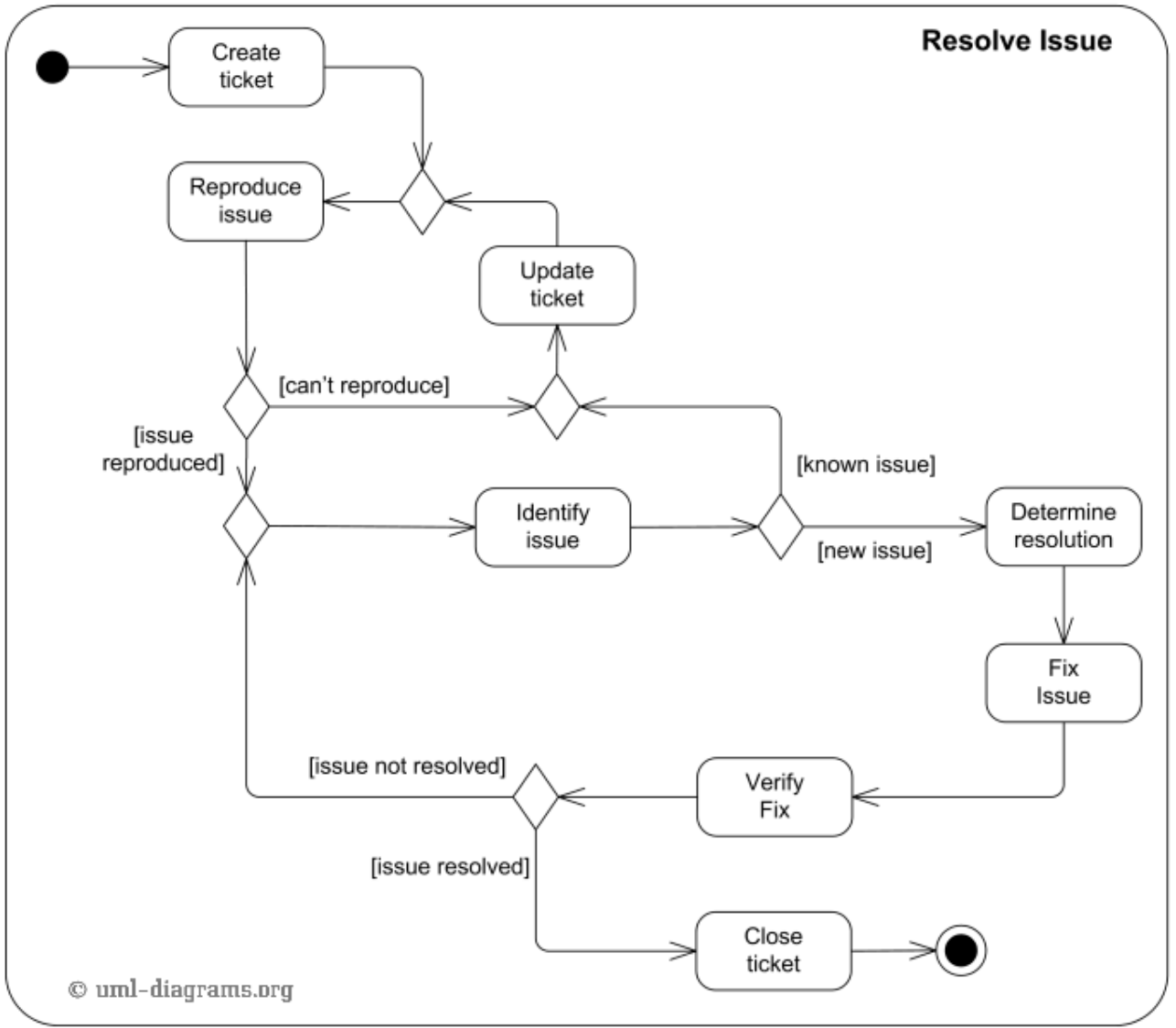


Object flow *of* ***Orders*** *between Fill Order and Review Order actions*

Activity Diagrams

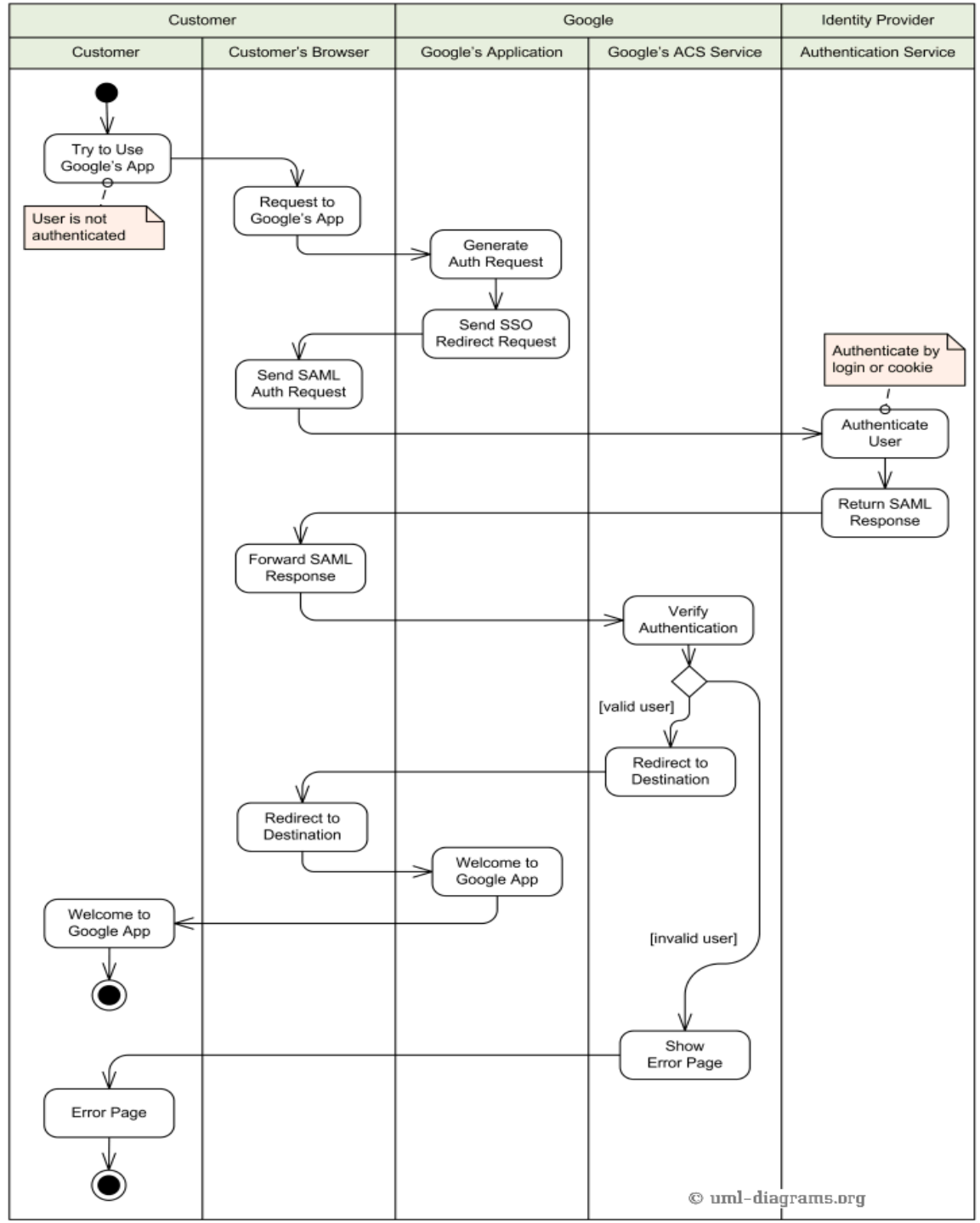


Activity Diagrams



Activity Diagrams

- While the diagrams before were easily understood. We are unsure about which actors within the system performs the identified tasks
- To make this more explicit we can partition the activity diagram to include the actors involved


Customer
Google
Identity Provider
Customer
Customer's Browser
Google's Application
Google's ACS Service
Authentication Service
Try to Use Google's App
User is not authenticated
Request to Google's App
Generate Auth Request
Send SSO Redirect Request
Send SAML Auth Request
Authenticate by login or cookie
Authenticate User
Return SAML Response
Forward SAML Response
Verify Authentication
[valid user]
Redirect to Destination
Redirect to Destination
Welcome to Google App
Welcome to Google App
[invalid user]
Show Error Page
Error Page
© uml-diagrams.org

Use Cases

- Use cases allow us to capture the requirements of systems under design or consideration, describe functionality provided by those systems, and determine the requirements the systems pose on their environment.

Use Cases

- UML specifications require that "*this functionality must always be completed for the UseCase to complete. It is deemed complete if, after its execution, the subject will be in a state in which no further inputs or actions are expected and the UseCase can be initiated again, or in an error state.*"

Use Cases

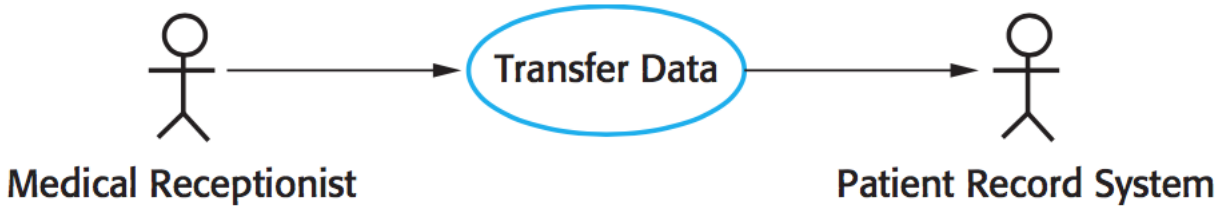
- Each use case represents a discrete task that involves external interaction with a system
- NB: Use case diagrams give a fairly simple overview of an interaction therefore more detail is required.

Use Cases

- Further details of use cases can be provided by:
  - Textual description
  - Structured description in a table
  - Sequence Diagram
- Format use will depend on use case and the level of detail that you think is required in the model

Use Cases – Structured Detail

MHC-PMS: Transfer data	
Actors	Medical receptionist, patient records system (PRS)
Description	A receptionist may transfer data from the MHC-PMS to a general patient record database that is maintained by a health authority. The information transferred may either be updated personal information (address, phone number, etc.) or a summary of the patient's diagnosis and treatment.
Data	Patient's personal information, treatment summary
Stimulus	User command issued by medical receptionist
Response	Confirmation that PRS has been updated
Comments	The receptionist must have appropriate security permissions to access the patient information and the PRS.



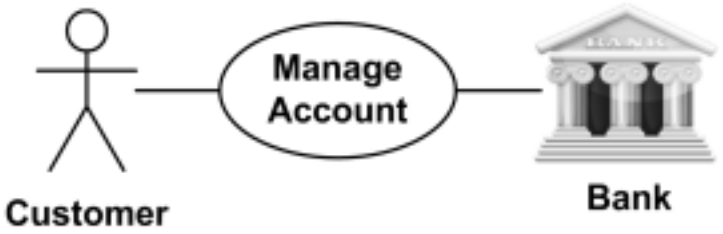
Use Case Diagrams

- Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).
- Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

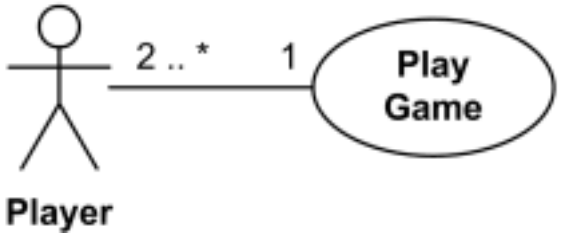
Use Case Diagrams

User Registration ***use case***

Student actor

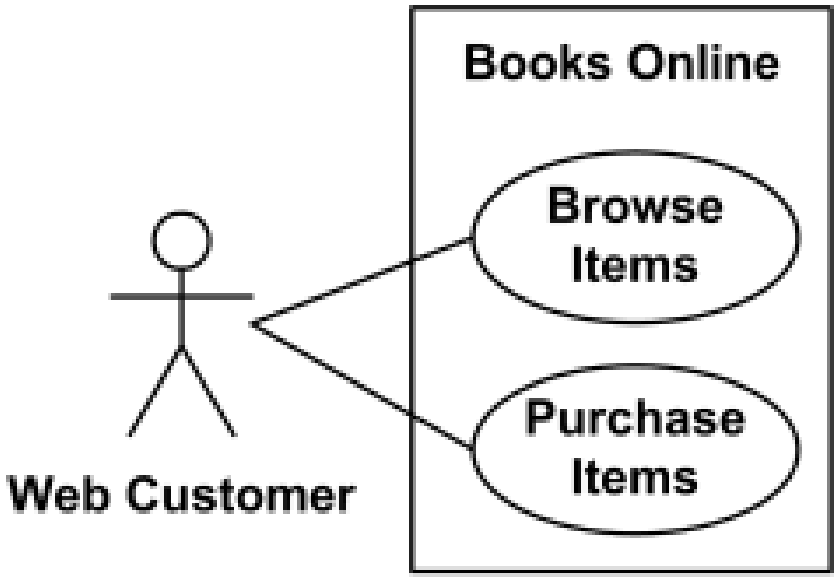


Manage Account use case is associated with Customer and Bank actors.



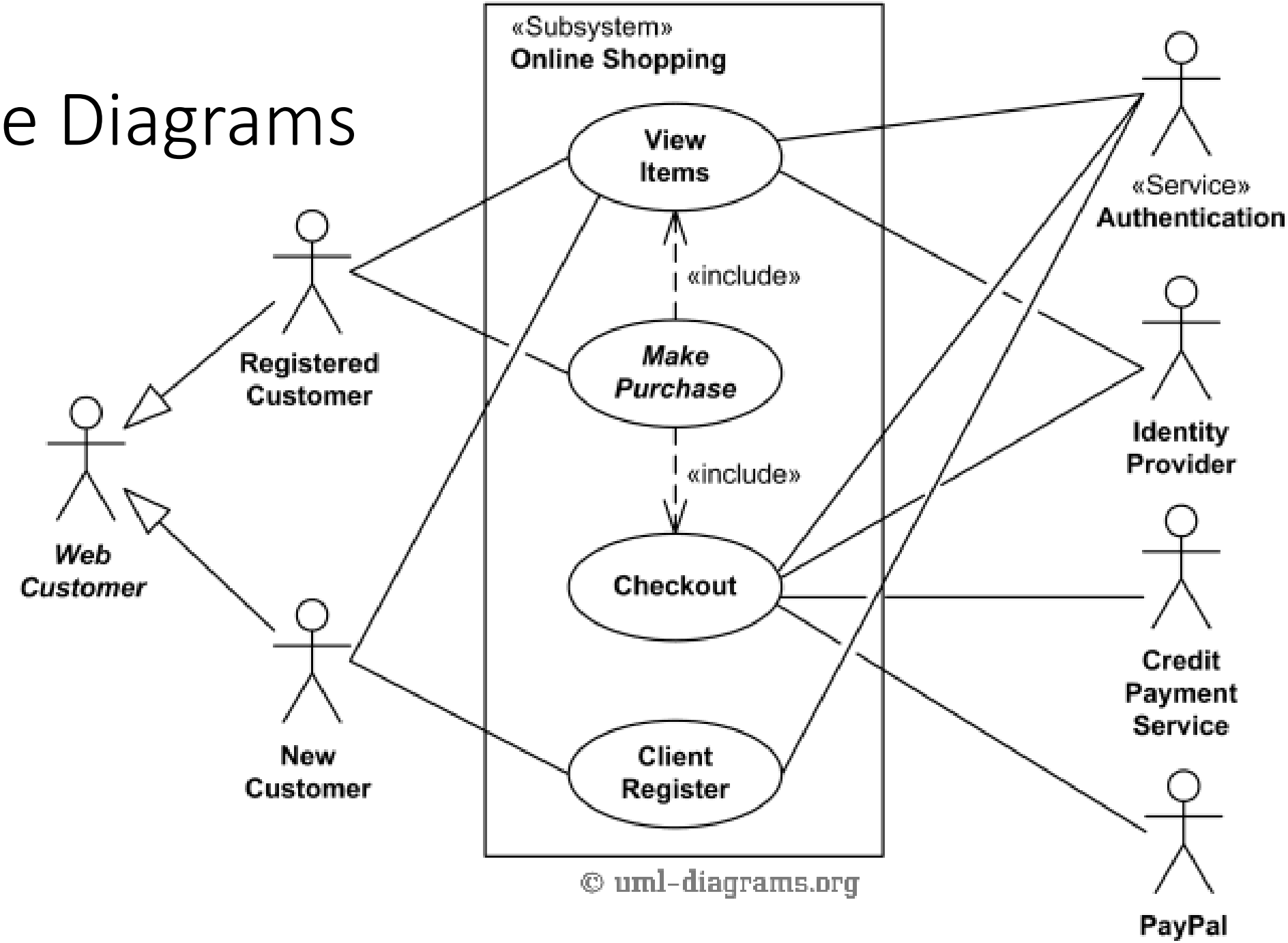
Two or more Player actors are involved in the Play Game use case.
Each Player participates in one Play Game.

Use Case Diagrams

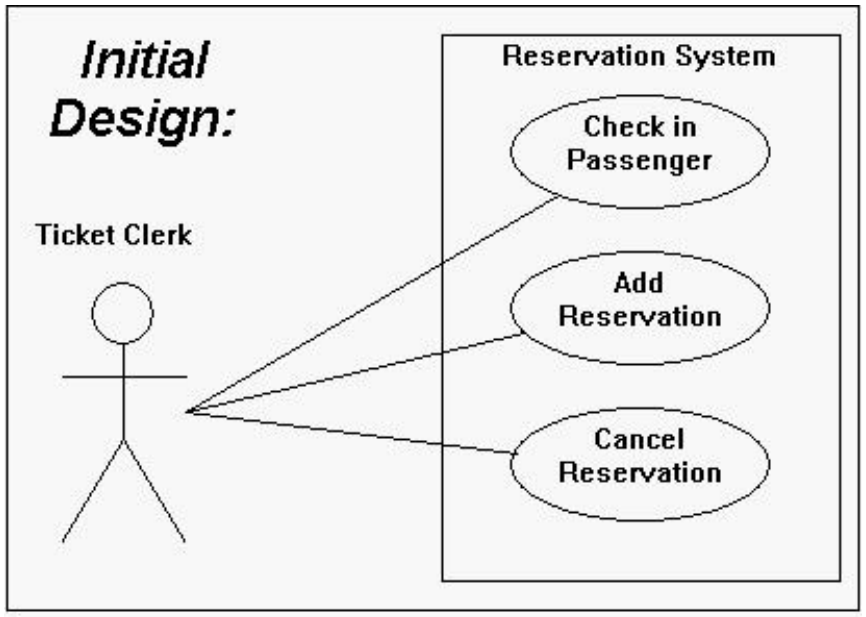


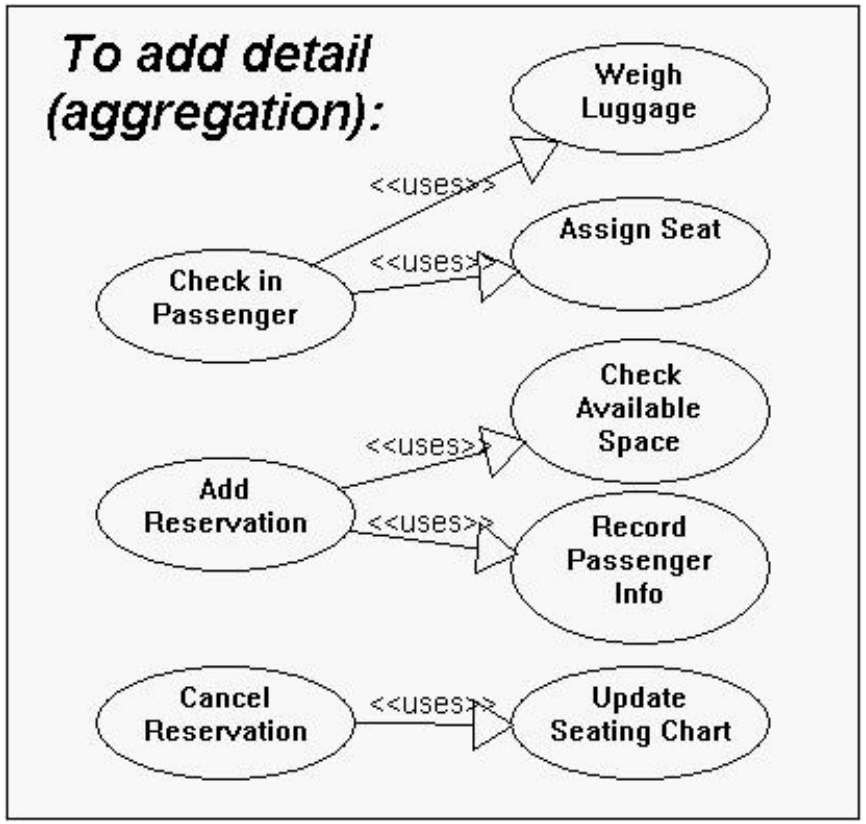
Books Online (subject) with applicable use cases and Web Customer actor.

Use Case Diagrams

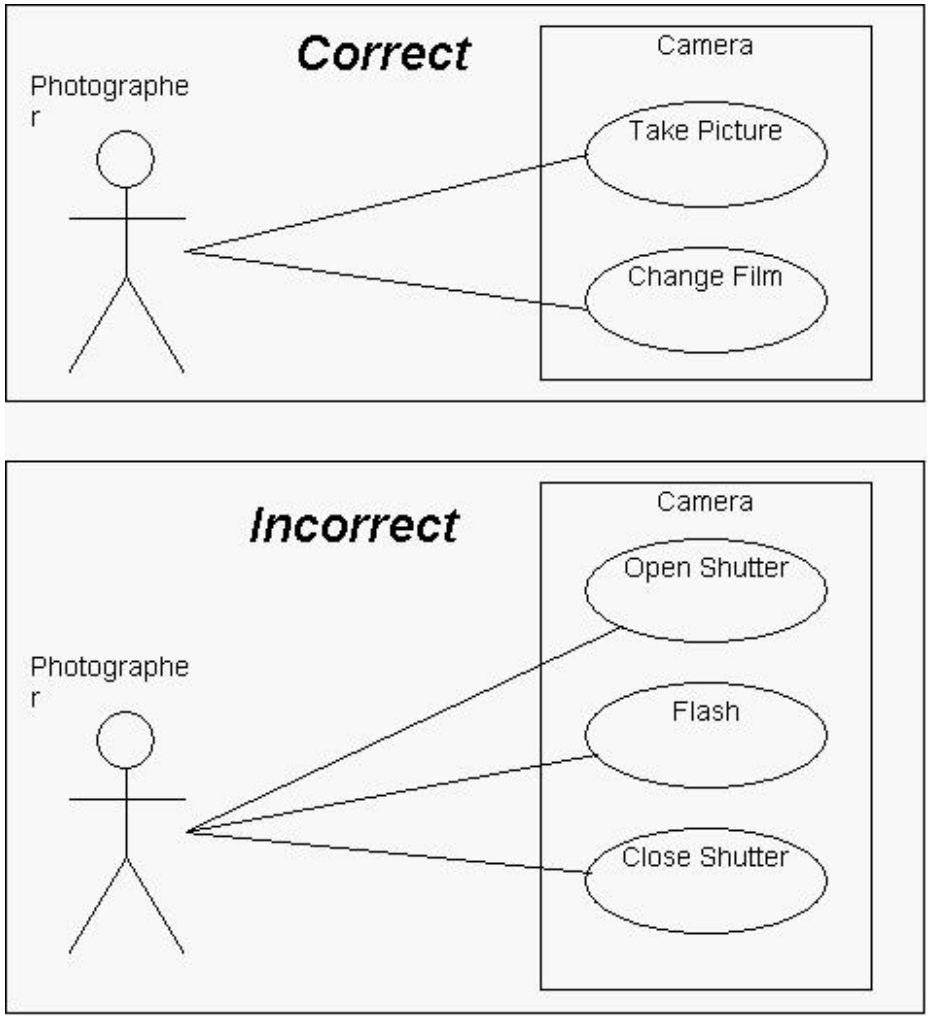


Use Case Diagrams





Use Case Diagrams



Sequence Diagrams

- Use to model the interactions between the actors and the objects in a system and interaction between the object themselves
- Shows a sequence of interactions that take place during a particular use case

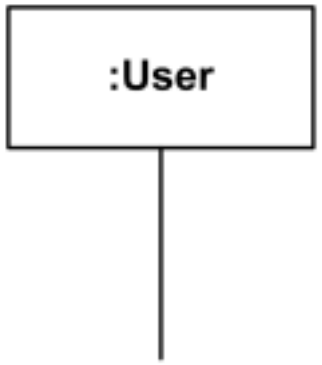
Sequence Diagrams

- Sequence diagrams are typically used to model:
  - Usage scenarios
    - Potential way a system is used
  - The logic of methods
    - The description of steps and interactions between complex operations, functions or procedures
  - The logic of services
    - Interaction between a variety of technologies/clients

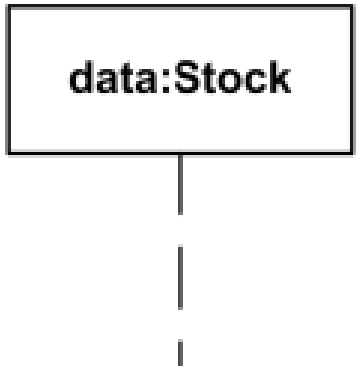
Sequence Diagrams

- Unless you are using sequence diagrams for code generation or detailed documentation, you don’t have to include every interaction in these diagrams.

Sequence Diagram

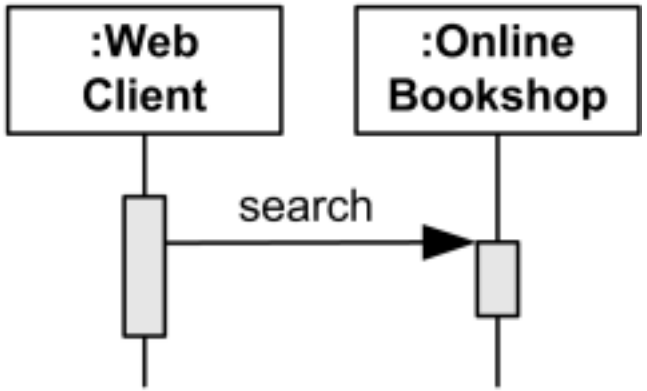


Lifeline of a class User

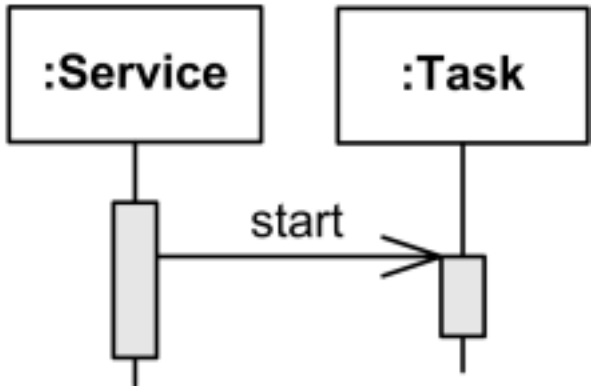


Lifeline “data” of a class Stock

Sequence Diagram

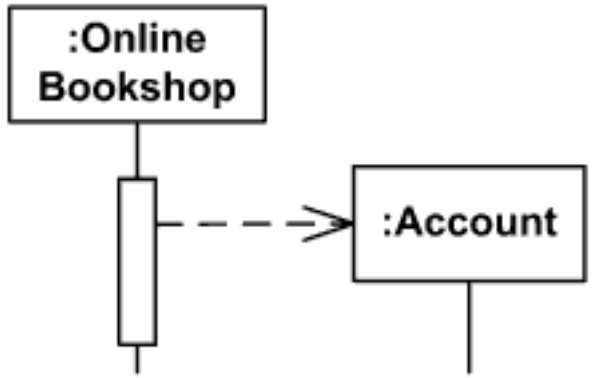


Synchronous

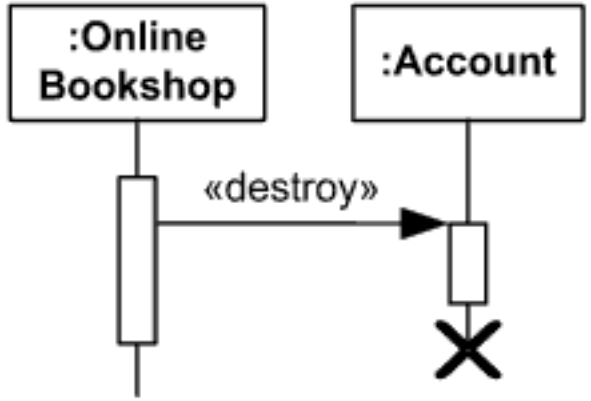


Asynchronous

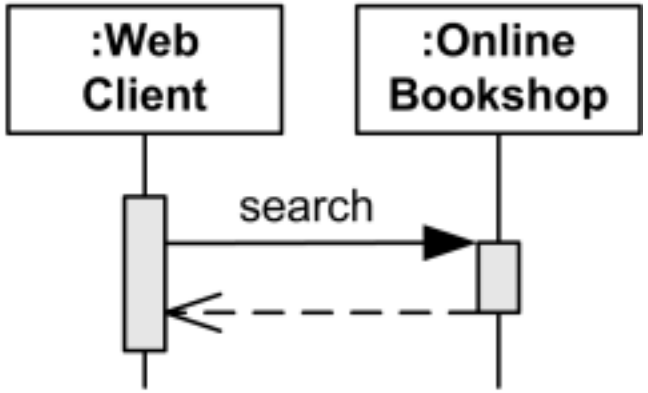
Sequence Diagram



Online Bookshop creates Account

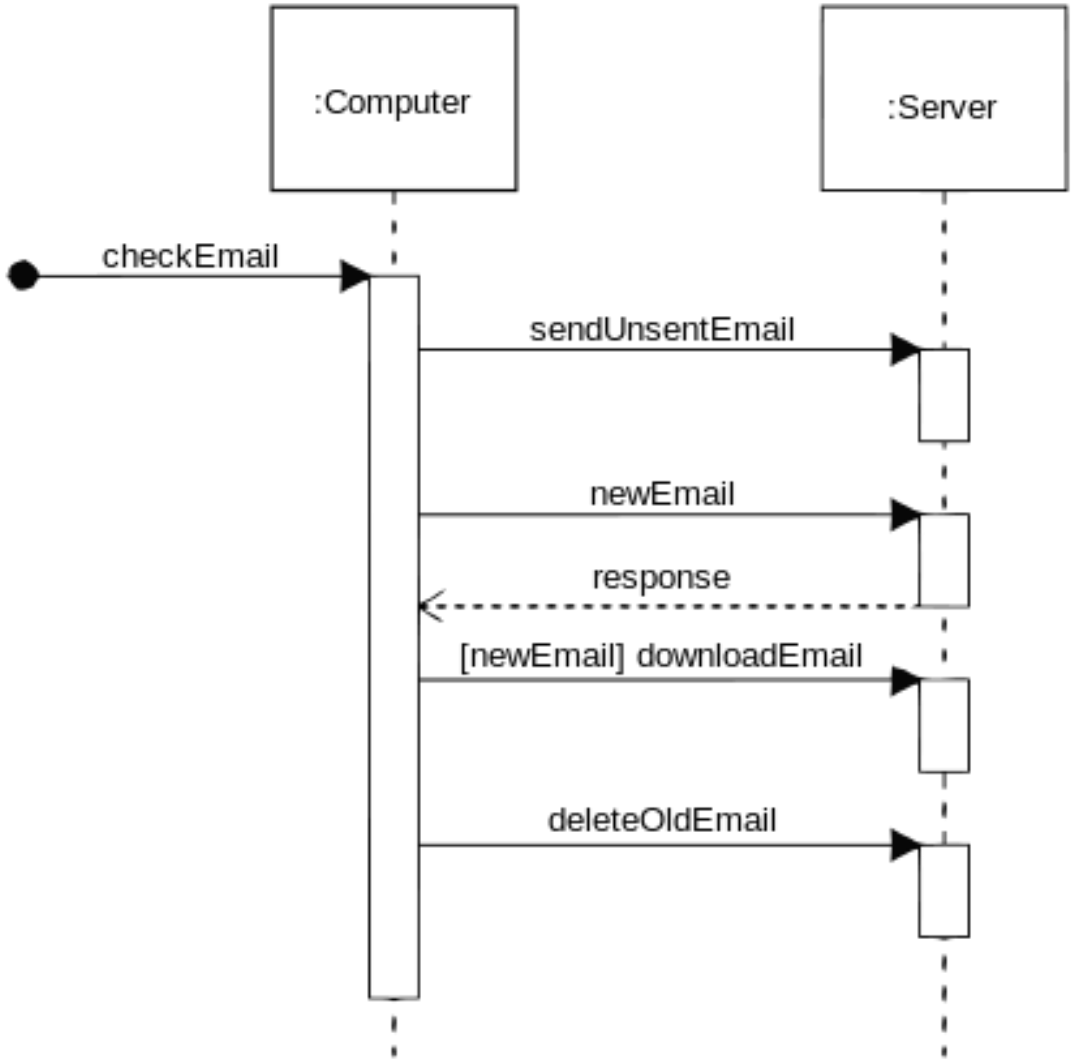


Online Bookshop terminates Account

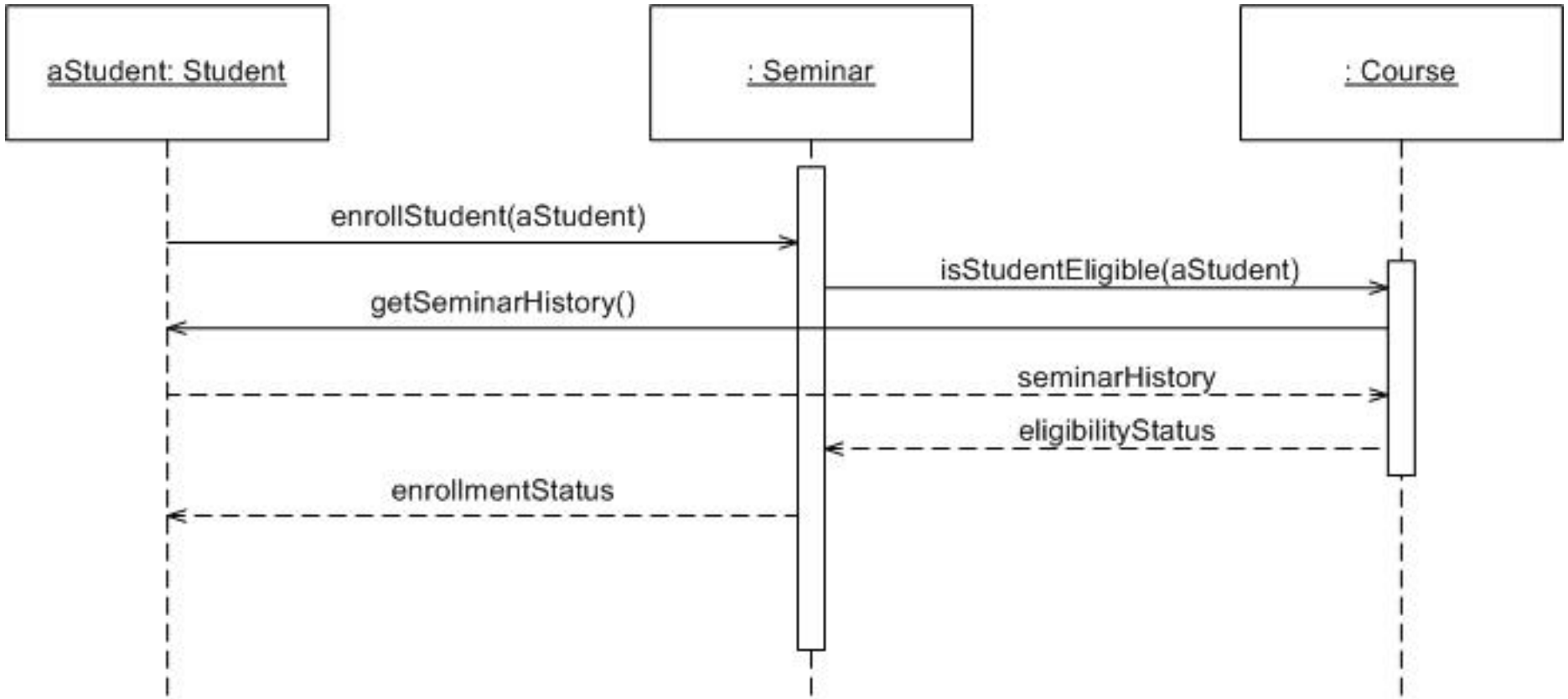


Web Client searches Online Bookshop and waits for results to be returned

Sequence Diagram



Sequence Diagram



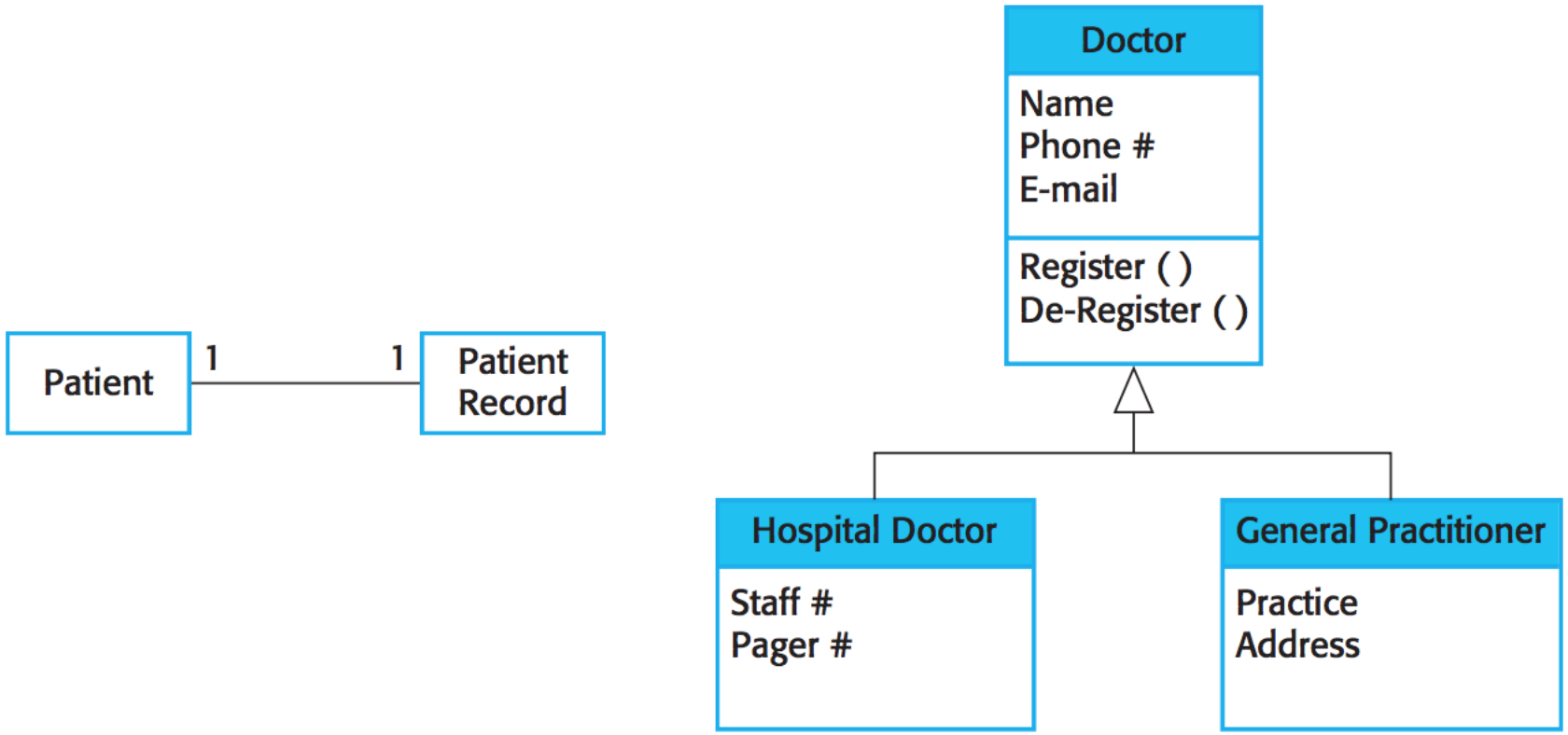
Class Diagrams

- Class diagrams are used when developing an object-oriented system model to show the classes in a system and the associations between these classes

Class Diagram

- Class diagrams in the UML can be expressed at different levels of detail. When you are developing a model, the first stage is usually to look at the world, identify the essential objects, and represent these as classes

Class Diagram



Class Diagram