Software Engineering

A Faculty of Engineering Course: CSEN 406

Unified Modeling Language (UML) System Design

4

Dr. Iman Awaad

iman.awaad@giu-uni.de

GIU GERMAN INTERNATIONAL UNIVERSITY الجامعة الألمانية الدولية

Acknowledgments

The slides are **heavily** based on the **slides** by **Prof. Dr. John Zaki**.

They are also **heavily** based on the slides and textbook by **Ian Somerville**.

Their contribution is gratefully acknowledged.

Any additional sources are referenced.

UML

- UML diagrams
  - Structural diagrams
    - Class
    - Component
  - Behavioural diagrams
    - Use case
    - State
    - Activity
  - Interaction diagrams
    - Sequence
  - …

What are the standards and best practices for visualising a system, in the planning, or the development or the maintenance phase?

What is UML?

…is a **general-purpose**, **object-oriented**, **visual modelling language** that provides a way to visualise the architecture and design of a system; like a blueprint…

Latest version: UML 2.5.1 from 2015; Managed by the Object Management Group (OMG)

What is UML?

So what?

…is a **general-purpose**, **object-oriented**, **visual modelling language** that provides a way to visualise the architecture and design of a system; like a blueprint…

Latest version: UML 2.5.1 from 2015; Managed by the Object Management Group (OMG)

Why use UML?

Standardisation

Communication

Visualisation

Documentation

Analysis and Design

https://www.theknowledgeacademy.com/blog/unified-modeling-language/

Different models for different uses…

Structural

Behavioural

Interaction

External

Different models for different uses…

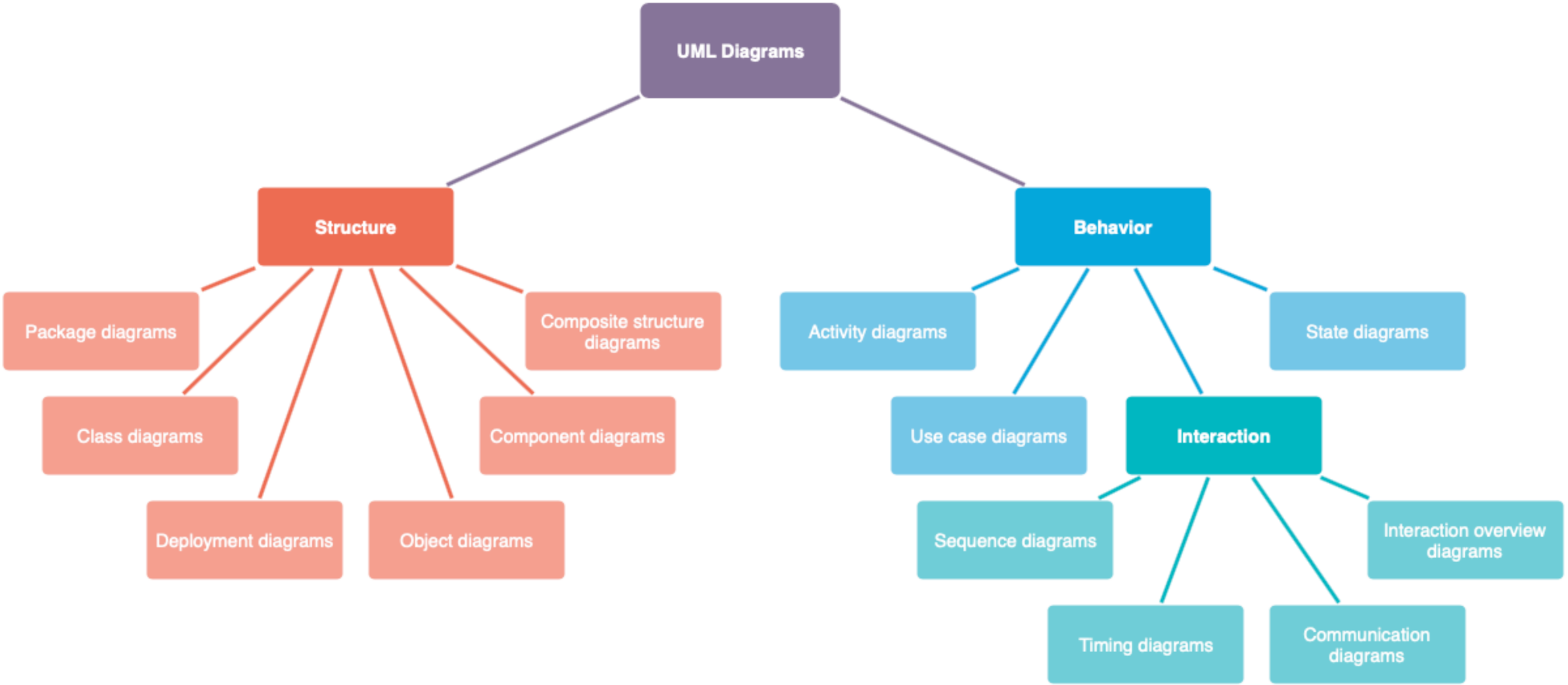
Structural … models the organisation of the system of the structure of the data

Behavioural …models the context of the system environment

Interaction …interaction between the system and the environment or between the system component

External …models the context of the system environment

Different models for different uses…



https://drawio-app.com/blog/uml-diagrams/

Here, focus is on these diagrams…

Use case	Sequence	Class	State	Activity
Interactions between a **system** and its **environment**	Interactions between **actors** and **system** and between **system components**	Object classes in the system and **associations between those classes**	How the system reacts to **internal** and **external events**	**Activities** involved in a **process** or in **data processing**

https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

Use case

Interactions between a **system** and its **environment**

When: created when looking at requirements of your system.

Represent: functions or **features**, the **actors** and how these **relate** to each other (their relationships)

https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

specific things your system can do

i.e. ...represents the functional requirements!

Use case

Interactions between a **system** and its **environment**

When: created when looking at requirements of your system.

Represent: functions or **features**, the **actors** and how these **relate** to each other (their relationships)

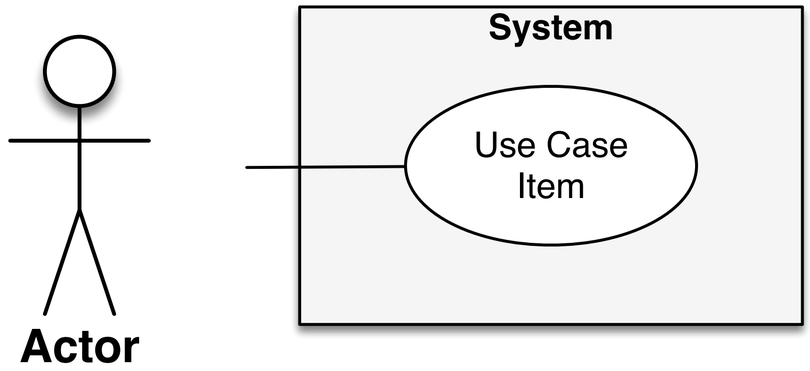
...Shows the interactions between the system and its external entities (actors) and outlines the use cases or functional requirements.

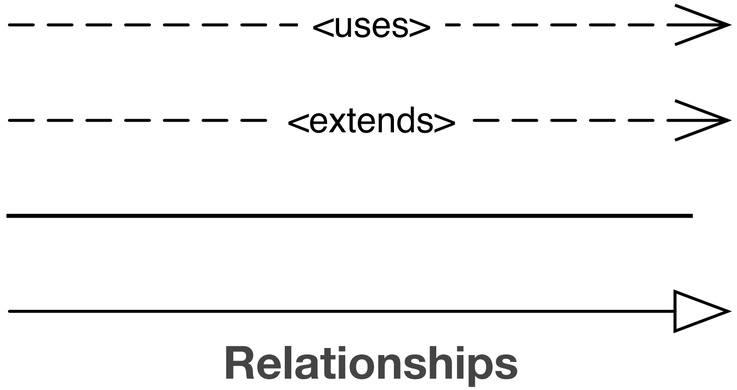
https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

Use case

Interactions between a **system** and its **environment**





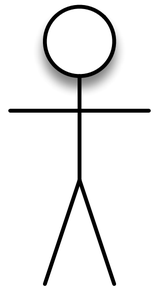
https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

Actors

Use case

Interactions between a **system** and its **environment**



e.g. a person, another system, a device, an organisation,…

Anything or **anyone** interacting with the system

Primary actor
Initiates the interaction (on the **left**)

Secondary actor
Reactive to the system (on the **right**)

Actors are
…placed **outside** of the system!
…**categorical** — not named!

https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

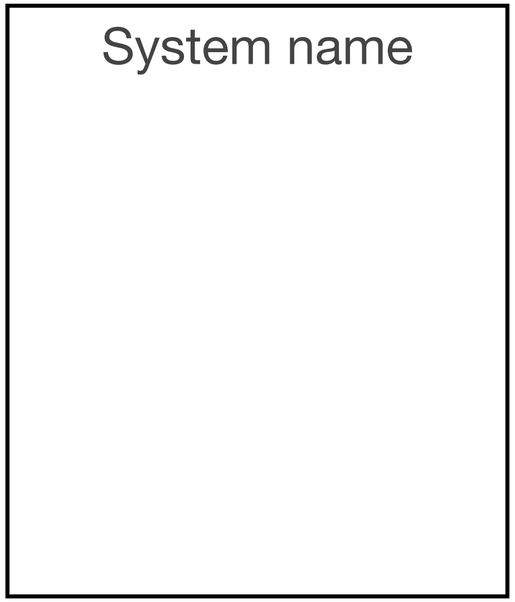
System

Use case

Interactions between a **system** and its **environment**

e.g. website, mobile app, business process, software component

System Boundary
Clearly outlines the boundaries of the system, indicating which components are internal to the system and which are external actors or entities interacting with the system



https://drawio-app.com/blog/uml-diagrams/

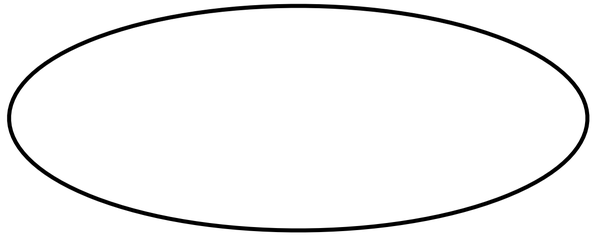
Use Case Diagrams

Use case

Use case

Interactions between a **system** and its **environment**

Use case
High-level view of the interaction between actors and the functionalities that the system enables; an action or scenario of a task (placed inside the system boundary)



https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

Relationships

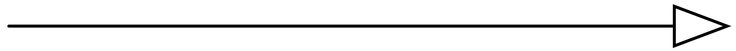
Use case

Interactions between a **system** and its **environment**

association
an actor participates in a use case

uses / **includes**
not initiated by an actor,
every time a base use case is executed,
the include use case **is also executed**

Generalisation/inheritance
parent-child use case
Each child will have some of the parent's characteristics but will also have its own characteristics



extend
not initiated by an actor,
when a base use case is executed,
the extend use case
can sometimes happen

https://drawio-app.com/blog/uml-diagrams/

Use Case Diagrams

Case Study: Instapay

Use case

Interactions between a **system** and its **environment**

You are required to design a **mobile application** similar to Instapay where the customer can carry out basic operations such as log in, check balance, make a transfer, or pay her bills

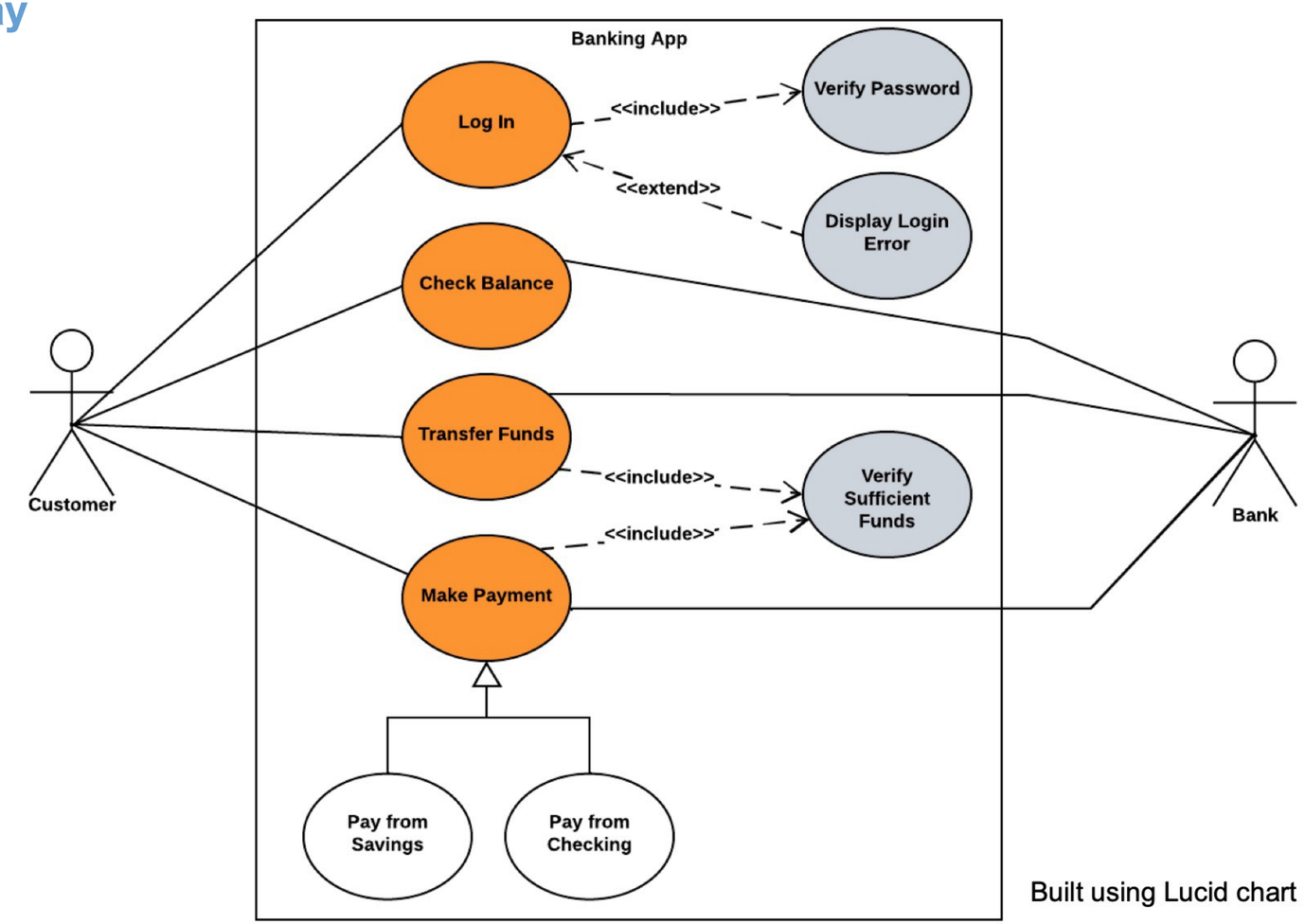
Use a use-case diagram to design the app flow with your development team. Present examples of the various types of relationships…

Use Case Diagrams

Case Study: Instapay

Use case

Interactions between a **system** and its **environment**



Built using Lucid chart

You are required to design a **mobile application** similar to Instapay where the customer can carry out basic operations such as log in, check balance, make a transfer, or pay her bills

Use a use-case diagram to design the app flow with your development team. Present examples of the various types of relationships…

What are extension points?

Sequence Diagrams

Sequence

Interactions between **actors** and **system** and between **system components**

When: created during the design phase

Represent: describe the **sequence of interactions** (messages) between **actors** and **objects** (things like databases or external interfaces)

https://drawio-app.com/blog/uml-diagrams/

Sequence Diagrams

Sequence

Interactions between **actors** and **system** and between **system components**

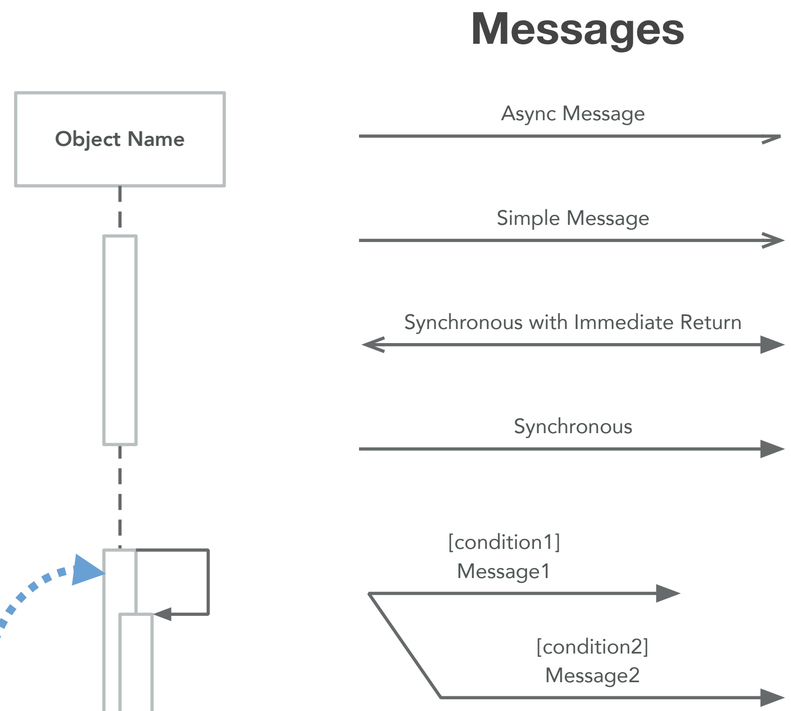
Use case:

- **Sequence of interactions**
- **Flow of events**
- **Message flow**

Sequence Diagrams

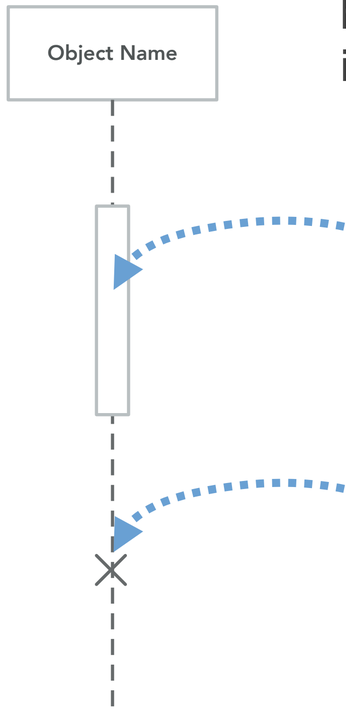
Sequence

Interactions between **actors** and **system** and between **system components**



Loop
… represent circumstances… if/then scenarios

Lifeline
…time as it passes



Object
…a class or object showing how the object will behave in the system

Activation Box
…time needed for an object to complete a task

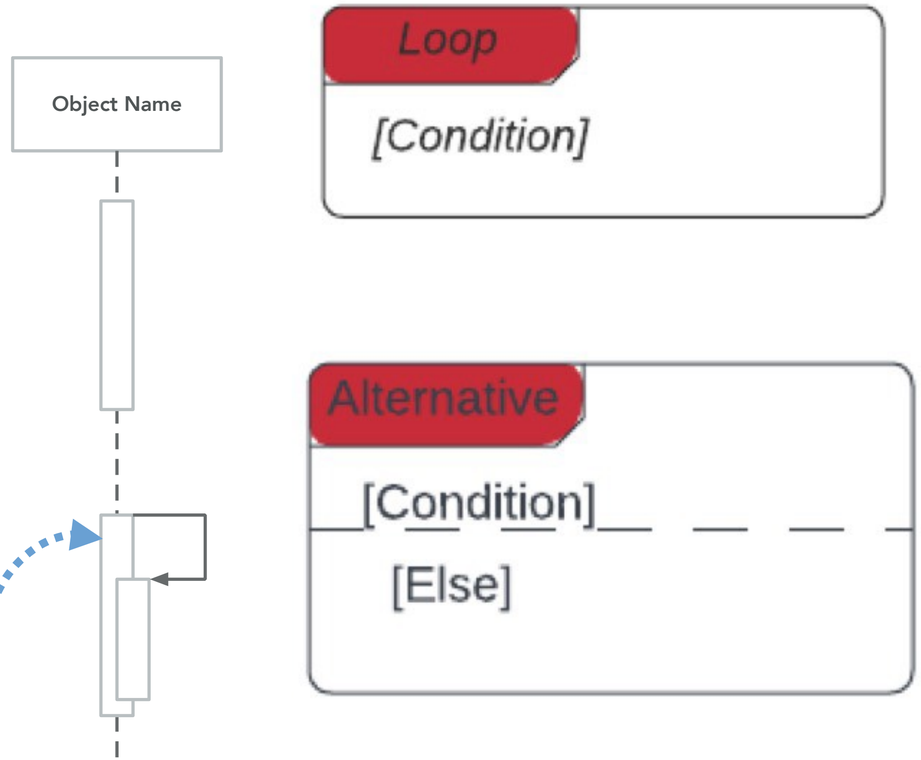
Delete/Destroy message
…destroy an object

Sequence Diagrams

Loops

Sequence

Interactions between **actors** and **system** and between **system components**



Alternative
…options between two or more message sequences (*aka* alternatives)

Loop
… represent circumstances… if/then scenarios

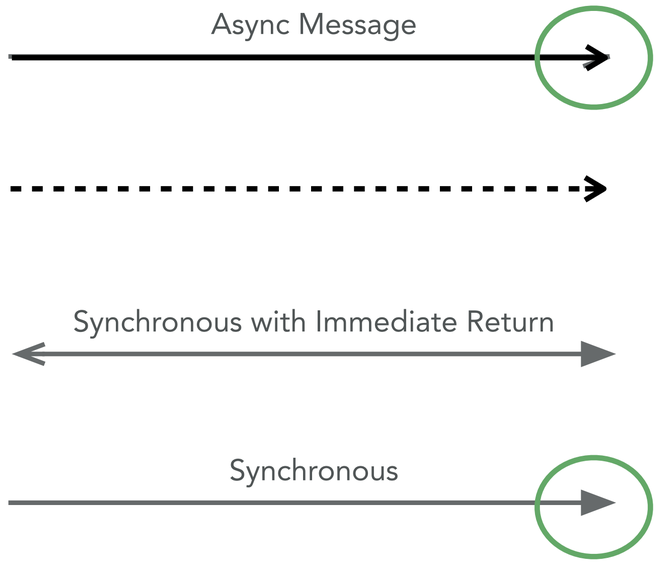
Lifeline
…time as it passes

Sequence Diagrams

Messages

Sequence

Interactions between **actors** and **system** and between **system components**



Asynchronous message …doesn't wait for a response before the sender continues. Only call should be included

Asynchronous return message …the response (reply) to a message

Synchronous return message …the response (reply) to a message

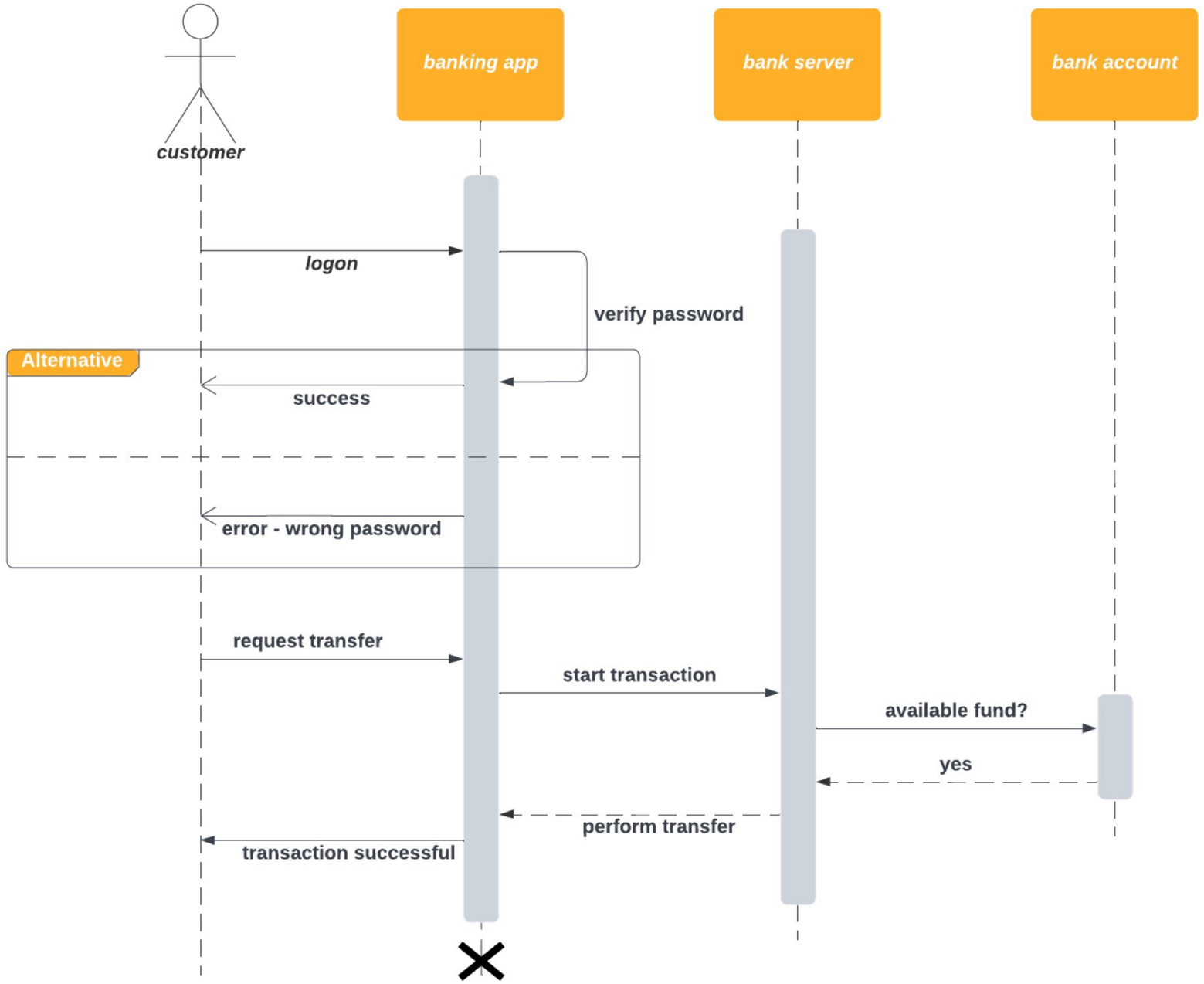
Synchronous message …when sender must wait for response to a message before it continues. Both call and reply are shown

Sequence

Case Study: Banking

Sequence

Interactions between **actors** and **system** and between **system components**



Class Diagrams

Class

Object classes in the system and **associations between those classes**

When: created in the design phase, but also used in the analysis, implementation and maintenance phases

Represent: classes their attributes (data) + their behaviour (member functions) and the relationships between these classes

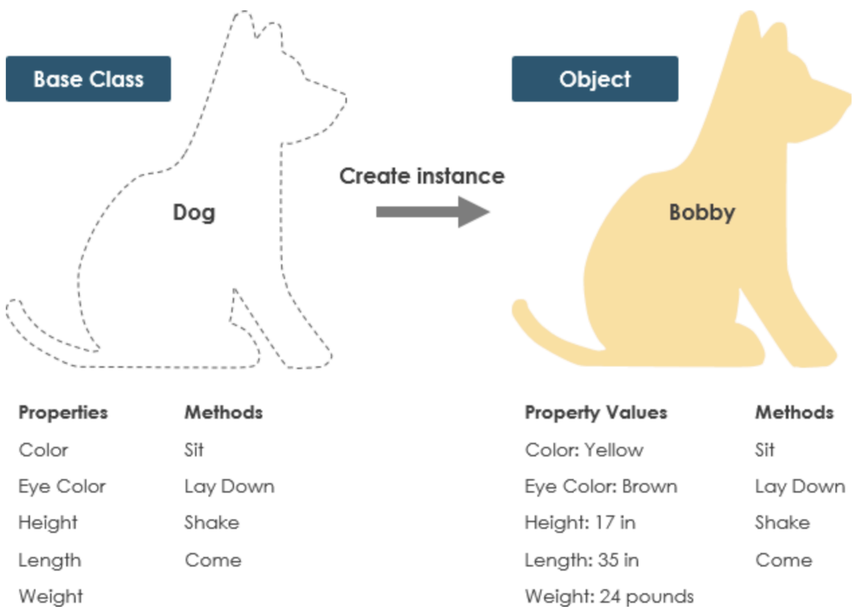
https://drawio-app.com/blog/uml-diagrams/

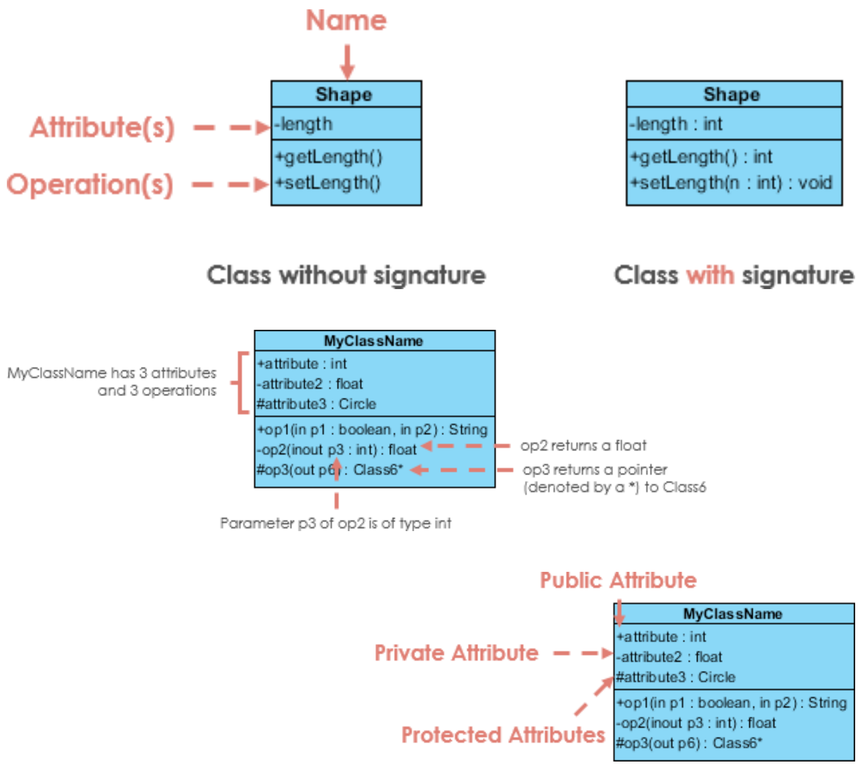
Class Diagrams

Classes

Class

Object classes in the system and **associations between those classes**





https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

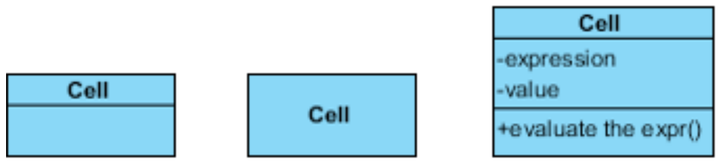
Class Diagrams

Perspectives

Class

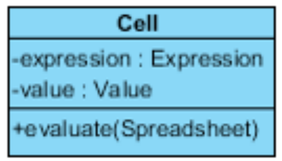
Object classes in the system and **associations between those classes**

represents the concepts in the domain



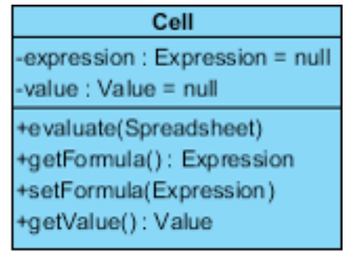
Conceptual

focus is on the interfaces of Abstract Data Type (ADTs) in the software



Specification

describes how classes will implement their interfaces



Implementation

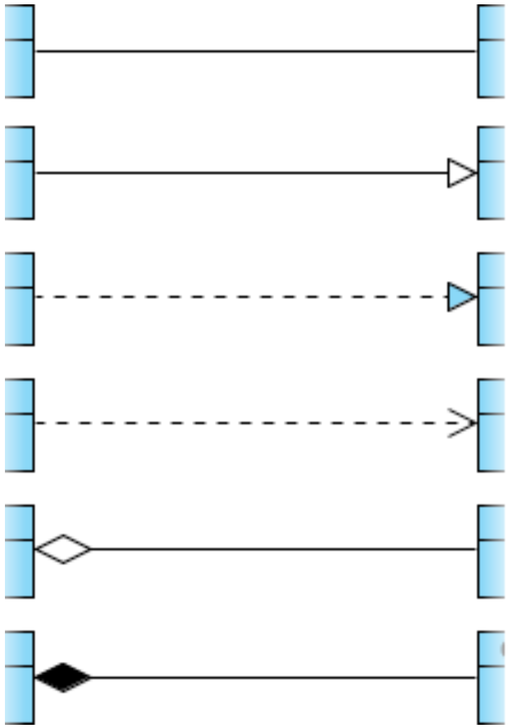
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

Relationships: Inheritance

Class

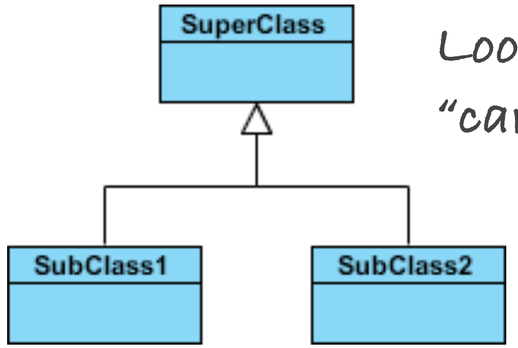
Object classes in the system and **associations between those classes**



Represents an "**is-a**" relationship.

An *abstract class name* is shown in italics

`SubClass1` and `SubClass2` are specialisation of `SuperClass`



Look up "cardinality"...

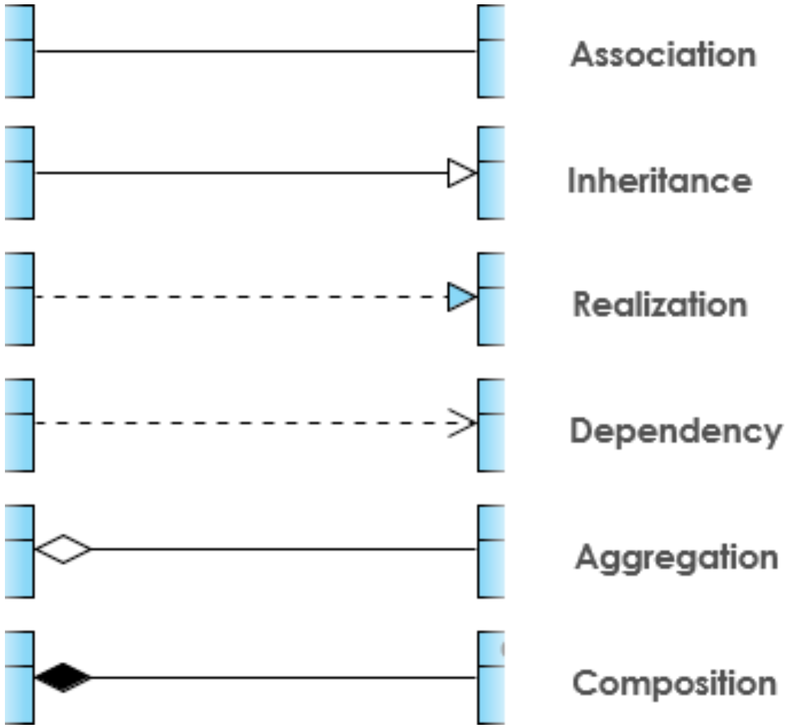
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

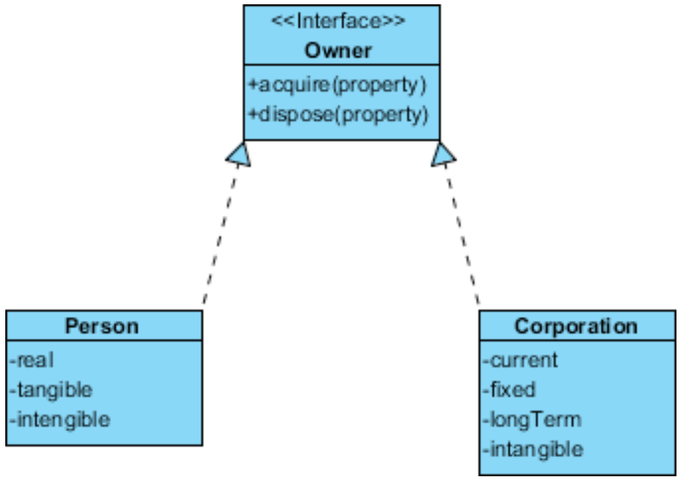
Relationships: Realisation

Class

Object classes in the system and **associations between those classes**



…relationship between the blueprint class and the object containing its respective implementation level details. This object is said to realise the blueprint class…



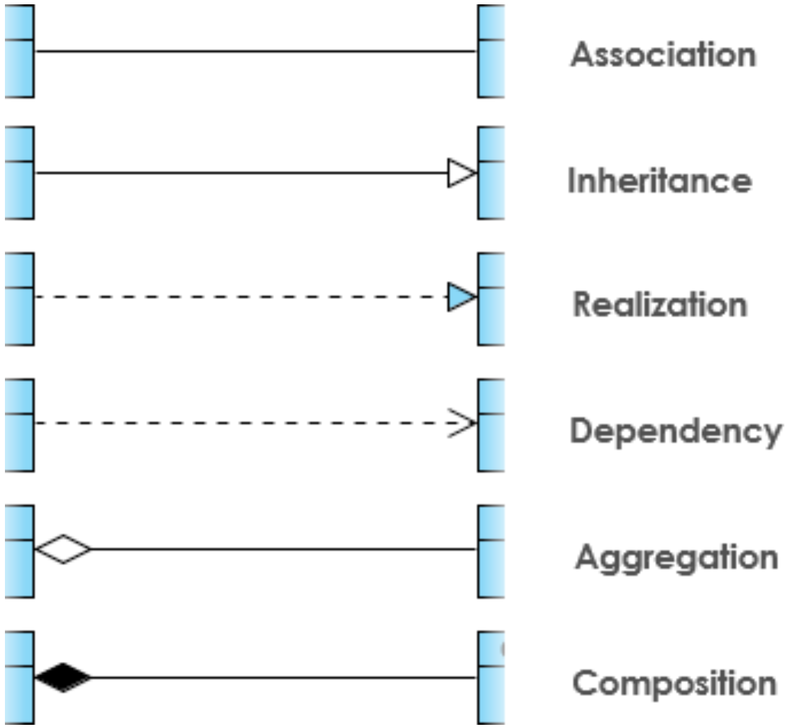
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

Relationships: Dependency

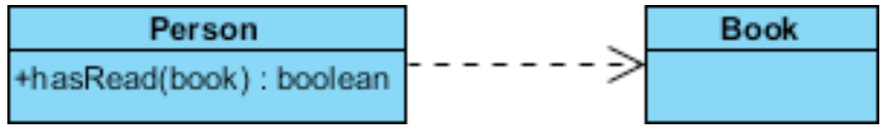
Class

Object classes in the system and **associations between those classes**



A special type of association that exists between two classes when changes to the definition of one may cause changes to the other (but not the other way around).

`Class1` depends on `Class2`



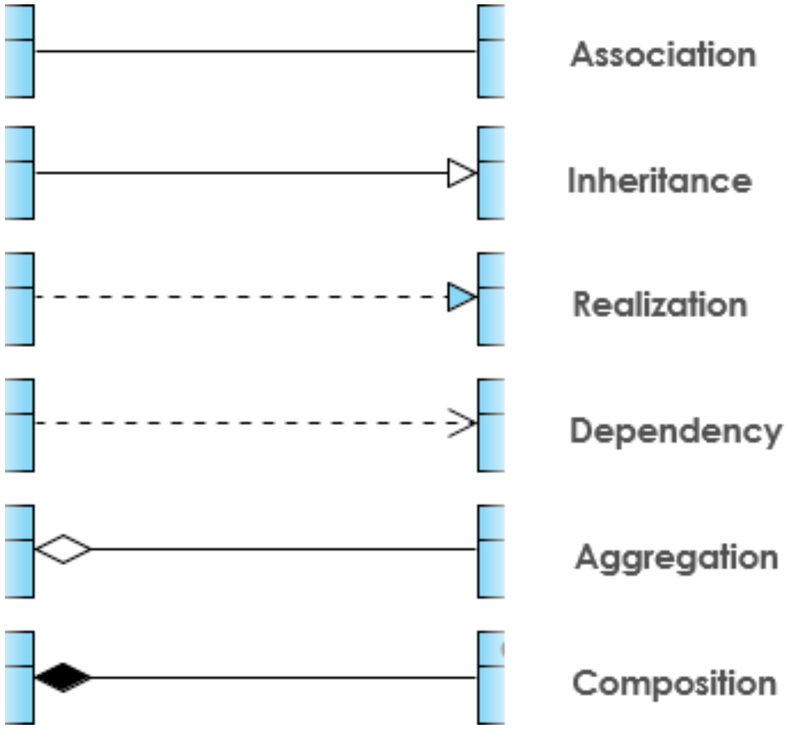
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

Relationships: Aggregation

Class

Object classes in the system and **associations between those classes**



It represents a “**part-of**" relationship.

`Class2` **is part of** `Class1`

Many instances (denoted by the *) of `Class2` can be associated with `Class1`

Objects of `Class1` and `Class2` have separate lifetimes

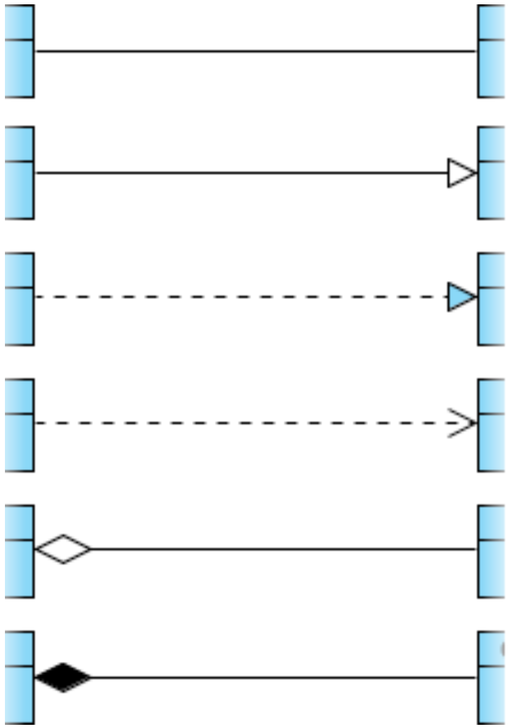
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

Relationships: Composition

Class

Object classes in the system and **associations between those classes**



A special type of aggregation where parts are destroyed when the whole is destroyed.

Objects of `Class2` live and die with `Class1`.

`Class2` cannot stand by itself.

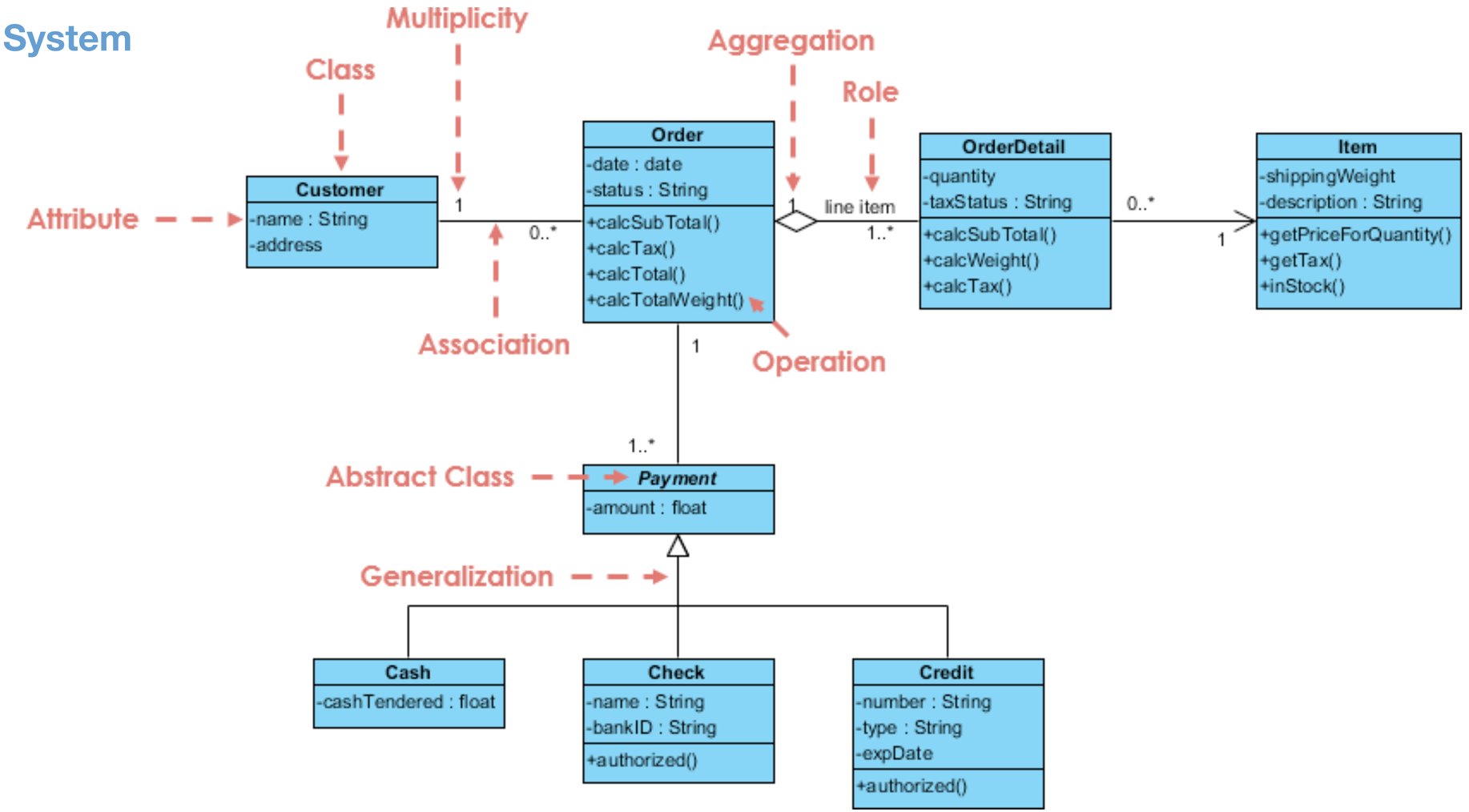
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

Class Diagrams

Case study: Order System

Class

Object classes in the system and **associations between those classes**



https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

State Diagrams

State

How the system reacts to **internal** and **external events**

When: created in the design phase

Represent: show **which states lead to each other**, and **what triggers a state change**

https://drawio-app.com/blog/uml-diagrams/

State Diagrams

State

How the system interacts to **internal** and **external events**

States: distinct conditions or phases an object or system can be in

Transitions: changes from one state to another, triggered by specific events

Events: specific inputs or occurrences that cause a transition from one state to another

Activities: actions performed when an object enters a state (entry activity), during its time in a state (do activity), or when it leaves a state (exit activity)

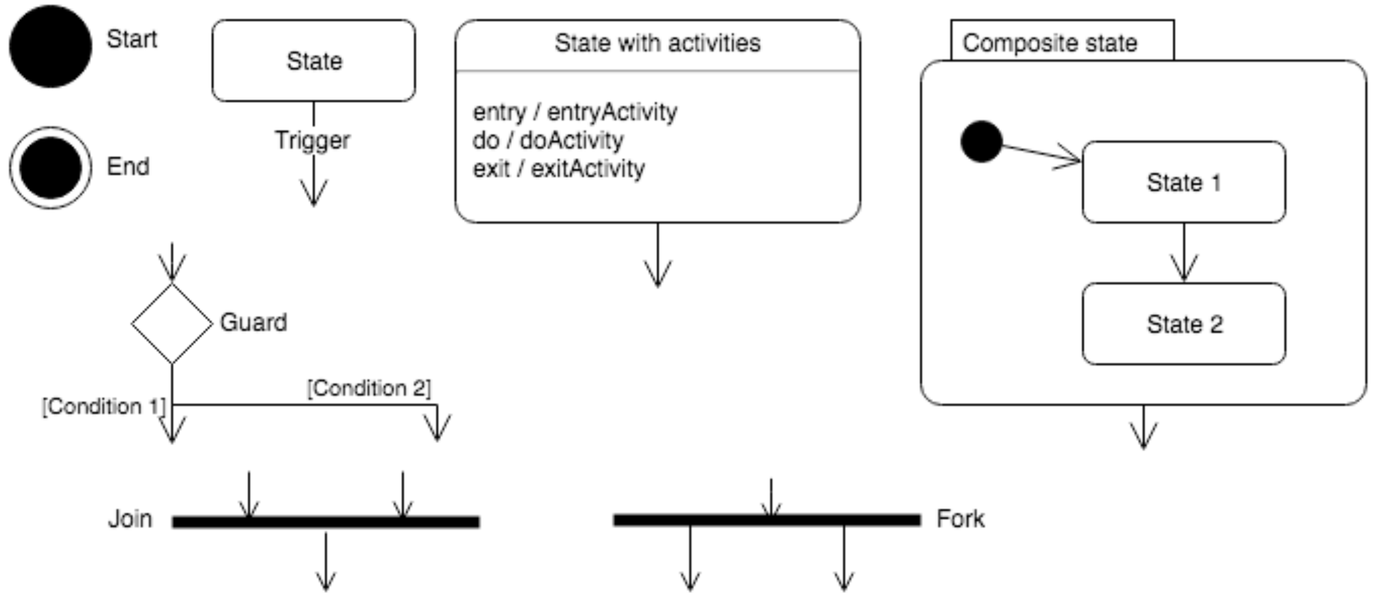
https://www.softwareideas.net/uml-state-machine

State Diagrams

State

How the system interacts to **internal** and **external events**

States
Transitions
Events
Activities



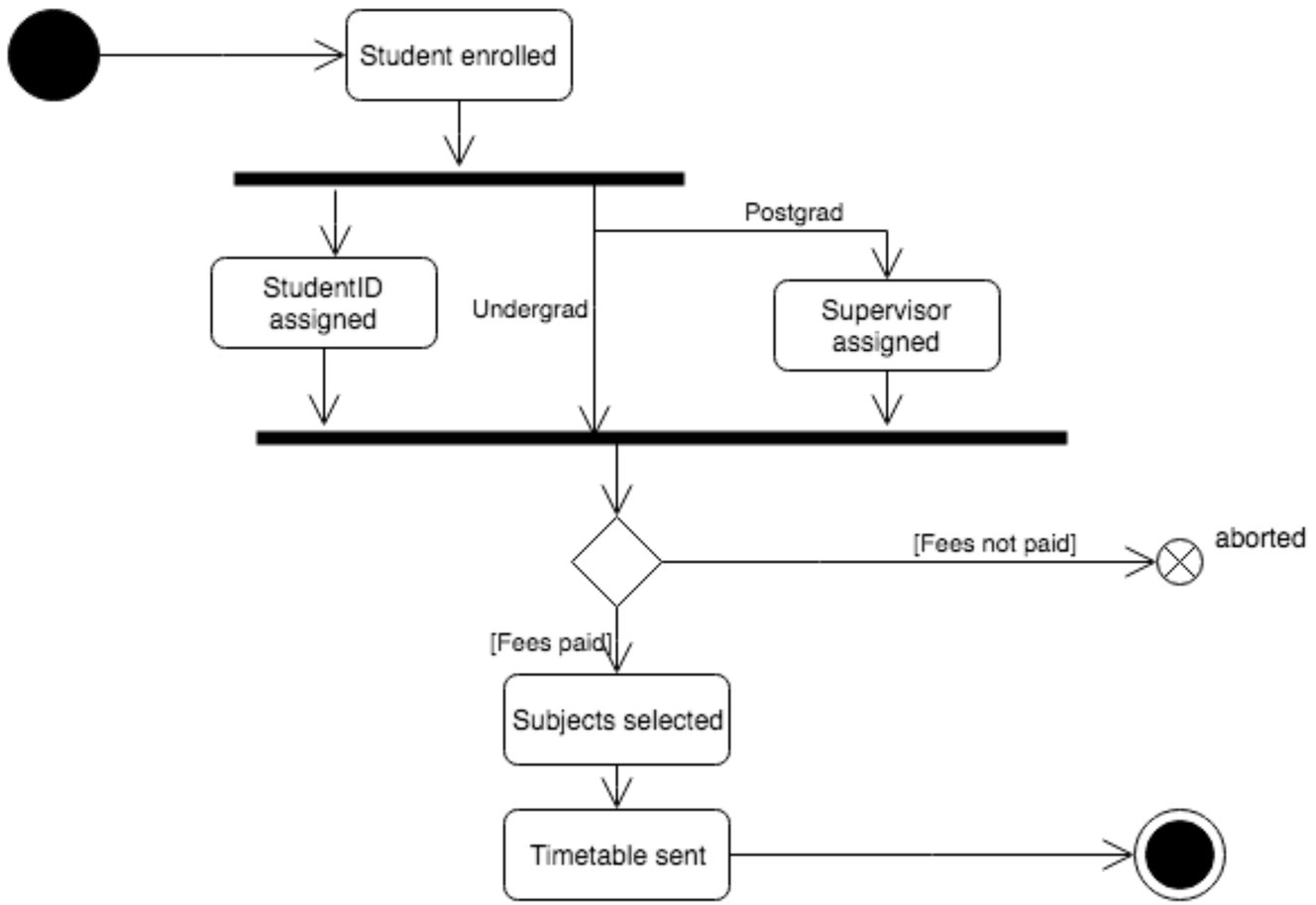
https://drawio-app.com/blog/uml-state-diagrams-with-draw-io/

State Diagrams

Case study: Student administration

State

How the system interacts to **internal** and **external events**



https://drawio-app.com/blog/uml-state-diagrams-with-draw-io/

Activity Diagrams

Activity

Activities involved in a **process** or in **data processing**

When: during requirements gathering, analysis, and design phases

Represent: behaviour of **users** and **systems** as they **follow a process**… a type of flow chart or workflow, but they use slightly different shapes

https://drawio-app.com/blog/uml-diagrams/

Activity Diagrams

Activity

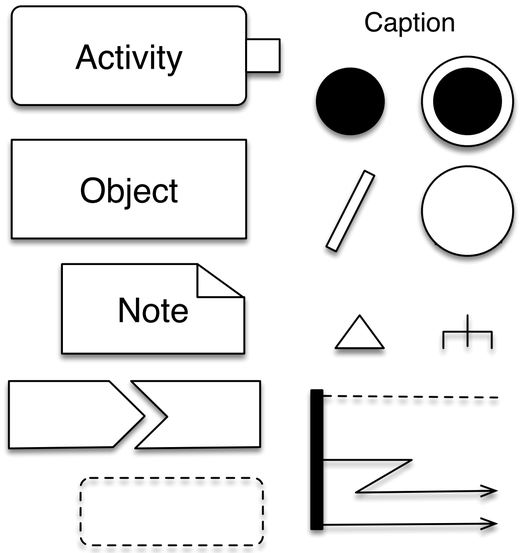
Activities involved in a **process** or in **data processing**

Start/End: black or solid circle: where the diagram **starts**; solid circle with a ring around it: the **end** of the process

Actions: rounded rectangles with the action name

Decisions: diamonds at choice points. Include decision as a question within the diamond, or indicate the decision outcome on the outgoing arrows (instead of simply using yes/no labels)

Split/Join: thick bar: where activities split or join



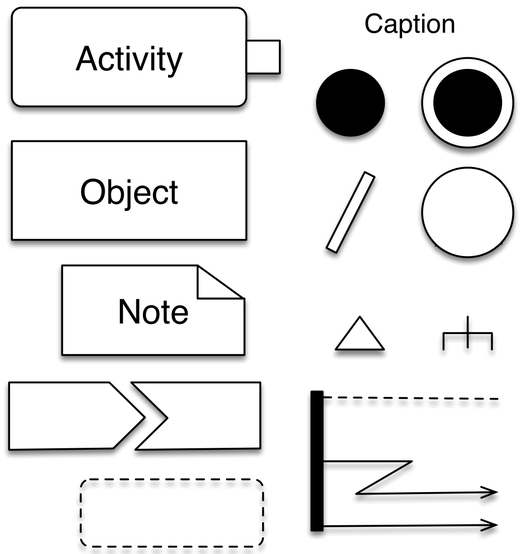
https://drawio-app.com/blog/uml-diagrams/

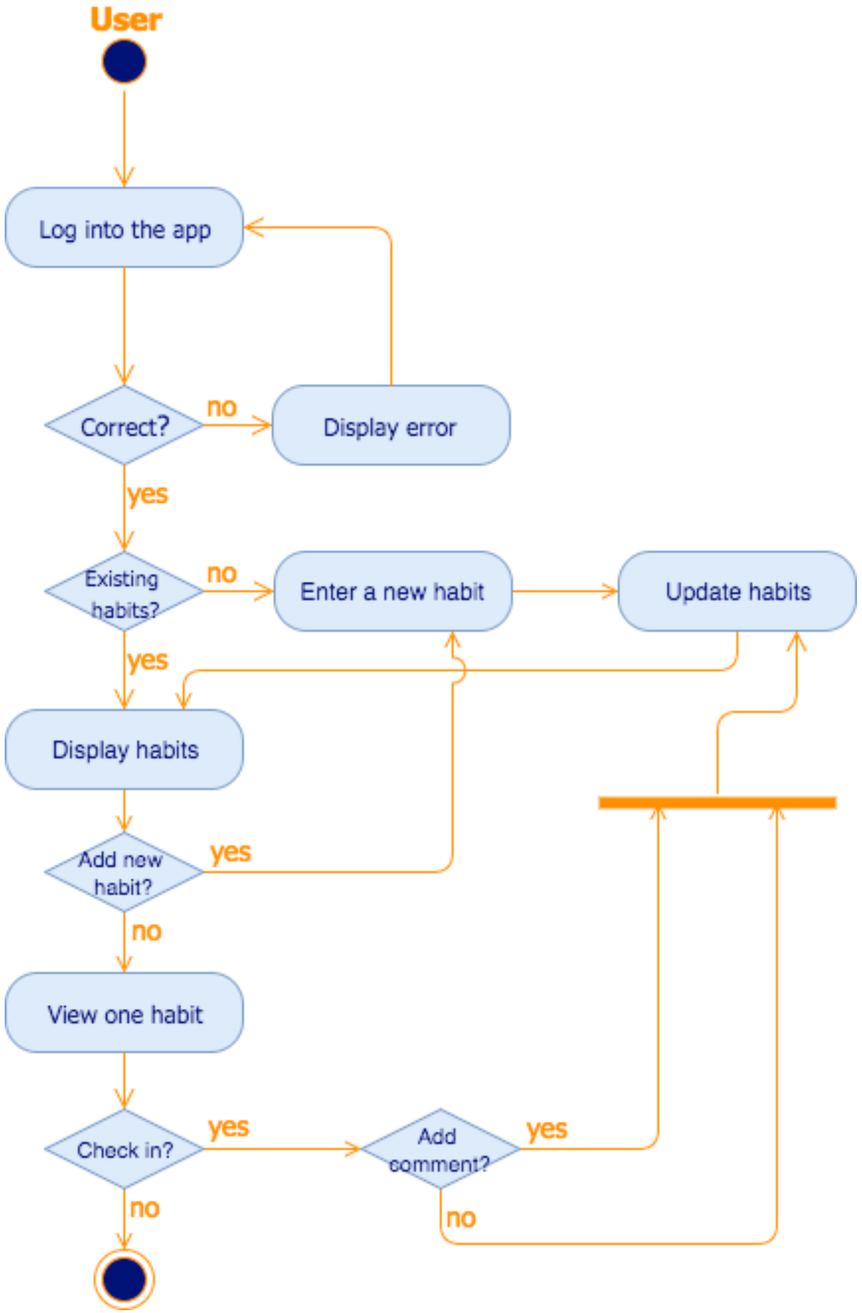
Activity Diagrams

Case study: Habit tracker

Activity

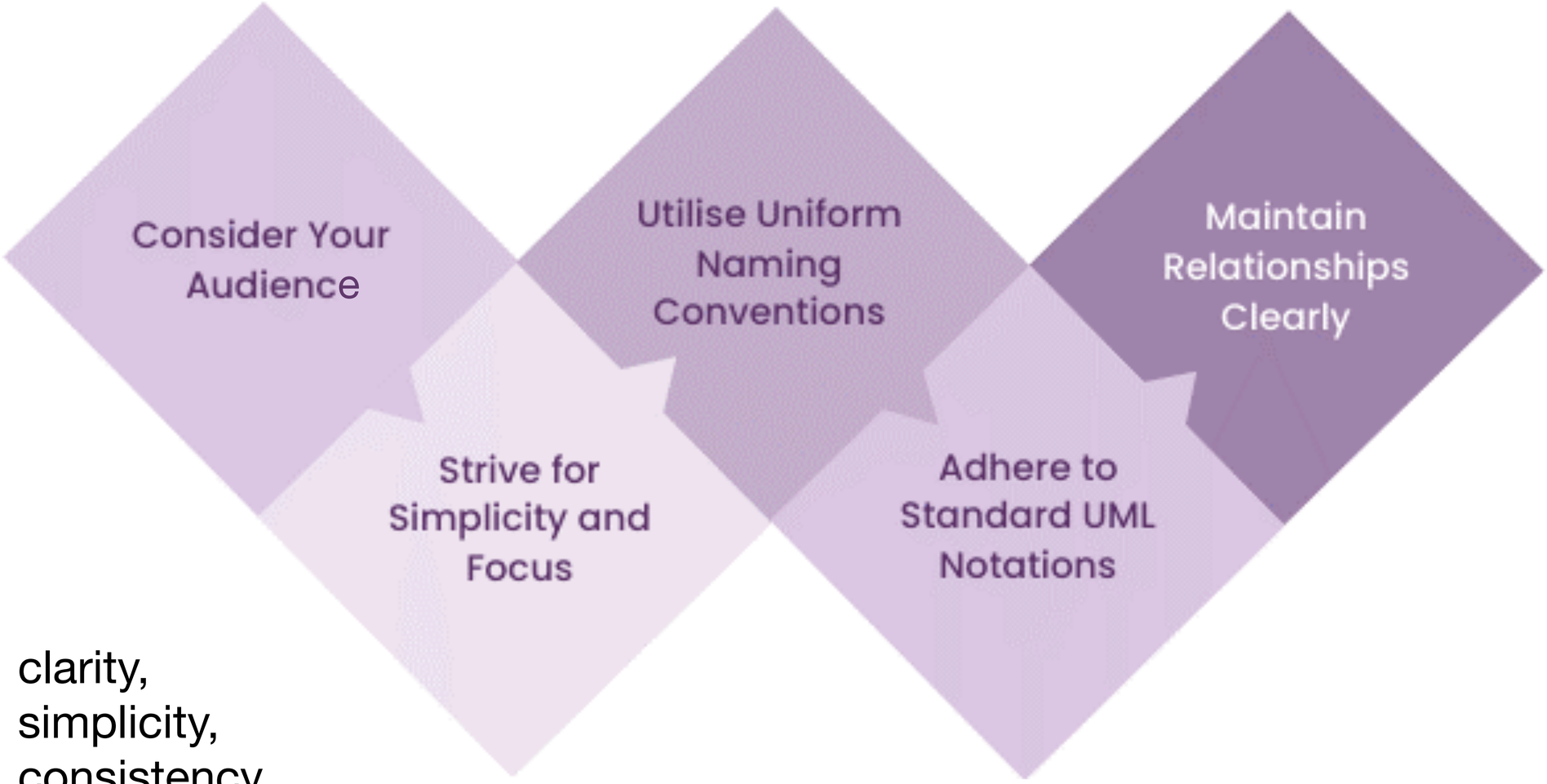
Activities involved in a **process** or in **data processing**





https://drawio-app.com/blog/uml-state-diagrams-with-draw-io/

Best practices for UML diagrams



clarity,
simplicity,
consistency

https://www.theknowledgeacademy.com/blog/unified-modeling-language/