Projektowanie złożonych systemów telekomunikacyjnych

UML – Introduction

Łukasz Marchewka

28-02-2022

Agenda:

- What is the Modelling and UML
- Understanding the basics of UML
- UML diagrams
  - Structural diagrams
  - Behavioral diagrams
    - Interaction diagrams

- UML Modeling tools
  - OpenSource Tool - PlantUML

Modelling

Describing a system at a high level of abstraction

- A model of the system
- Used for requirements and specifications

Why do we model software systems?

- A model is a simplification, that helps in better understanding of the system

UML – Unified Modelling Language

- UML is a pictorial language for specifying, visualizing, constructing, and documenting the artifacts of software systems

- It is a simple modeling mechanism to model all possible practical systems

- UML is not a programming language but there are tools that can be used to generate code in various languages basing on UML diagrams

- UML is not dependent on any language or technology

Diagrams

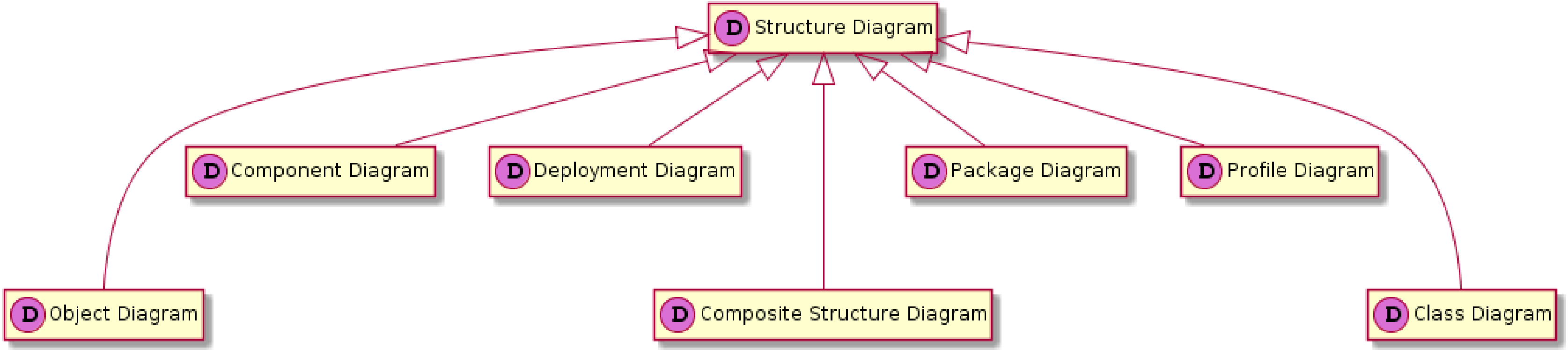
Structural Diagrams – describe static aspects of the software system using :

- Objects
- Attributes
- Operations and relationships

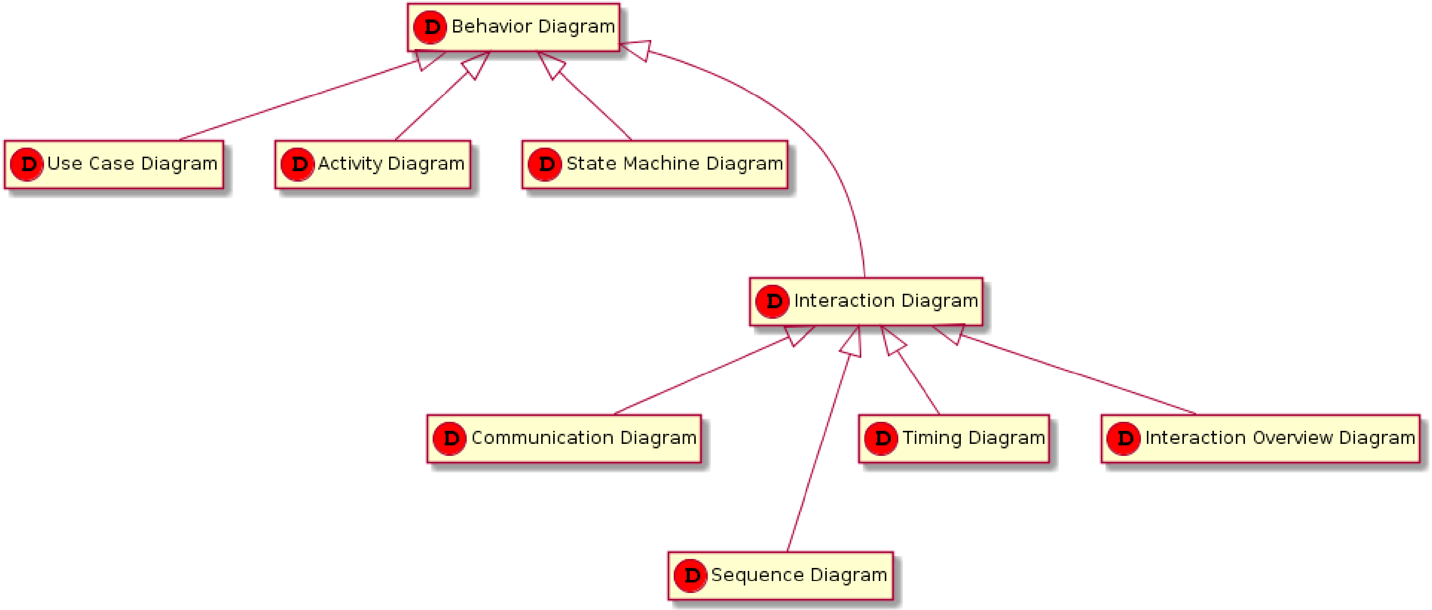
Behavioral (or Dynamic) Diagrams – describe dynamic aspects of the software system by showing collaboration among objects and changes to the internal states of objects:

- Use-case
- Interaction
- State Chart
- Activity

Structure Diagrams Hierarchy (Static)



Behavioral (Dynamic) Diagrams Hierarchy



Structure Diagrams

Class Diagram

Class Diagram describes static structure of a system using:

- Classes/Interfaces
- Attributes
- Operations (or methods)
- Relationships between classes

Class

Describes a set of objects having similar:

- Attributes (status)
- Operations (behavior)
- Relationships with other classes

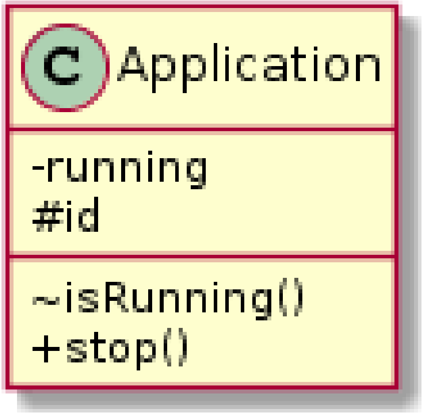
Attributes and operations may have their visibility marked:

"+" for public

"#" for protected

"−" for private

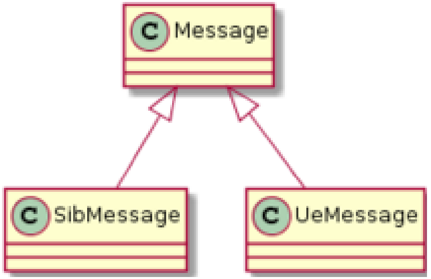
"~" for package



Class diagram – replationships between classes

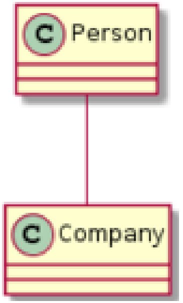
Generalization:

- Indicates that objects of the specialized class (subclass) are substitutable for objects of the generalized class (super-class)
- Generalization expresses a parent/child relationship among related classes
- Used for abstracting details in several layers



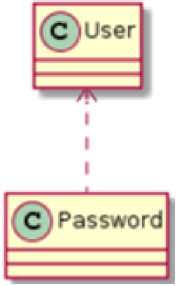
Association:

- Indicates that there is a relationship between objects, however the objects are independent



Dependency:

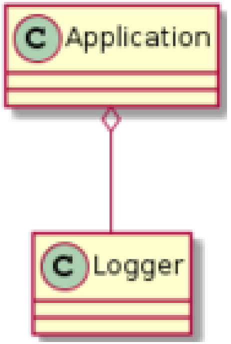
- An element, or set of elements, requires other model elements for their specification or implementation



Class diagram – replationships between classes

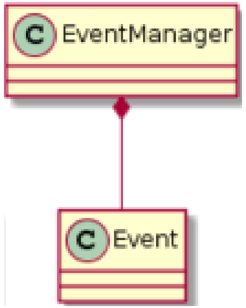
Aggregation:

- It can be treated as variant of the "has a" association relationship.
- It represents a part-whole or part-of relationship.



Composition:

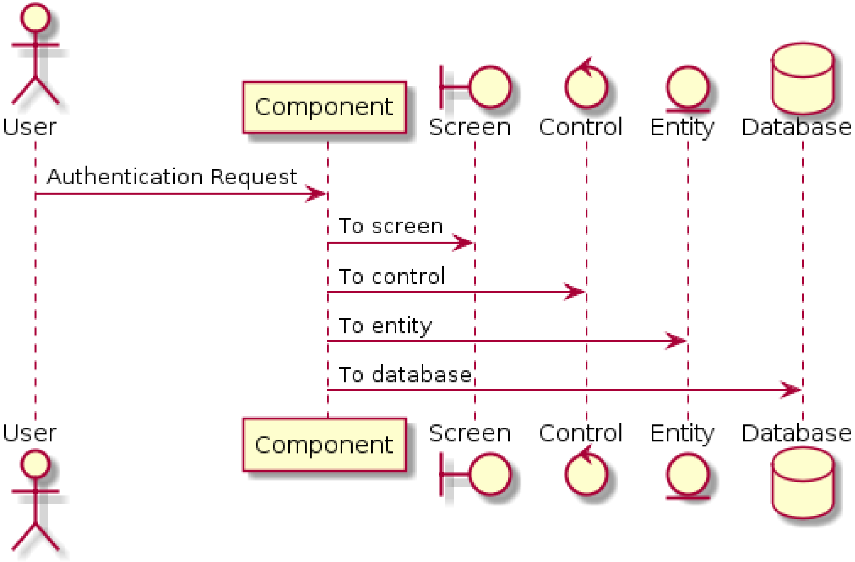
- A strong form of aggregation
- Whole/part relationship -> The whole is the sole owner of its part
- If a composite (whole) is deleted, all of its composite parts are also deleted



Behavioral Diagrams

Sequence Diagram

- Describes the interactions between objects in the sequential order
- The diagram conveys this information along the horizontal and vertical dimensions:
  - the vertical dimension shows, top down, the time sequence of messages/calls as they occur, and the horizontal dimension shows, left to right, the object instances that the messages are sent to.



Sequence Diagram

Lifeline:

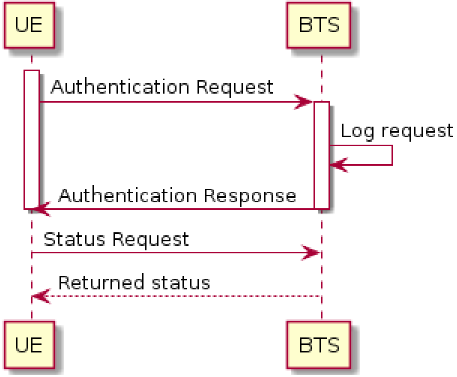
- A vertical line that represents the sequence of events that occur in a participant during an interaction, while time progresses down the line. This participant can be an instance of a class, component, or actor.

Messages:

- Messages are arrows that represent communication between objects. Half-arrowed lines are used to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

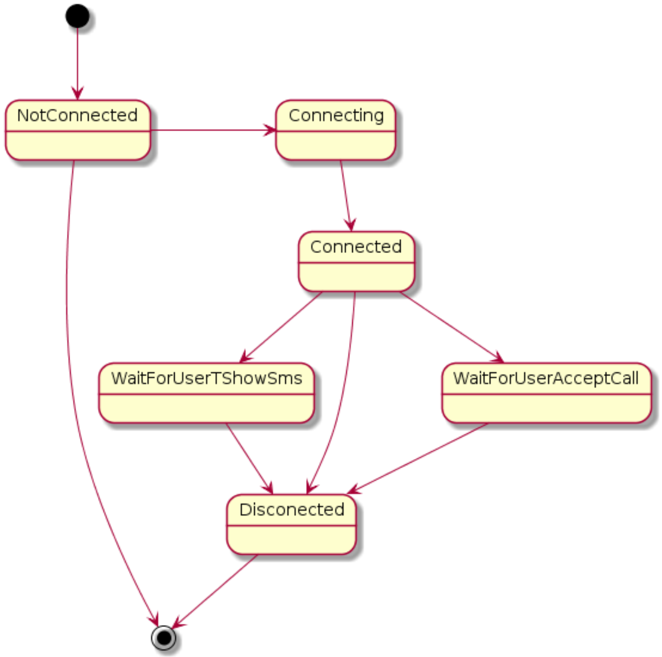
Activation boxes:

- Represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin rectangle placed vertically on



State (Machine) Diagram

- Describes the behavior of a single object and the flow of control from one state to another
- Specifies a sequence of events that an object goes through during its lifetime in response to events.

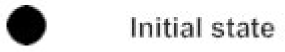


State (Machine) Diagram

State is defined as a condition in which an object exists and it changes when some event is triggered

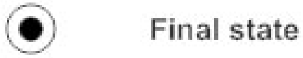
Initial State:

- A filled circle followed by an arrow represents the object's initial state



Final State:

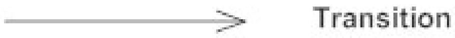
- An arrow pointing to a filled circle nested inside another circle represents the object's final state



Transition:

- A solid arrow represents the path between different states of an object. Label the transition with the event that triggered it and the action that results from it. A state can have a transition that points back to itself

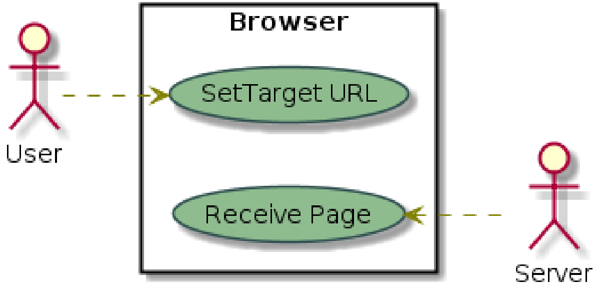
Decision points:



- A decision point models a choice that must be made within the sequence of states

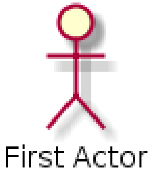
Use case diagram

- Ilustrates a unit of functionality provided by the system.
- Visualizes the different types of roles in a system and how those roles interact with the system.
- Identify functions and how roles interact with them
- High level view of the system
- Identify internal and external factors



Use Case Diagram objects

Use case diagrams consist of 4 objects:

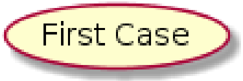


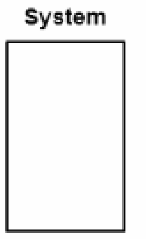
Actor:

- Any entity that performs a role in one given system. This could be a person, organization or an external system

Use case:

- Represents a function or an action within the system.



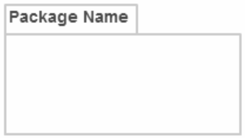


System (optional):

- Defines the scope of the use case, an optional element but useful when your visualizing large systems.

Package (optional):

- Used to group use cases together.



PlantUML

PlantUML is an Open Source project used to draw UML diagram, using a simple and human readable text description.

It's more a *drawing* tool than a *modeling* tool.

http://plantuml.com/

NOKIA

Copyright and confidentiality

The contents of this document are proprietary and confidential property of Nokia. This document is provided subject to confidentiality obligations of the applicable agreement(s).

This document is intended for use of Nokia's customers and collaborators only for the purpose for which this document is submitted by Nokia. No part of this document may be reproduced or made available to the public or to any third party in any form or means without the prior written permission of Nokia. This document is to be used by properly trained professional personnel. Any use of the contents in this document is limited strictly to the use(s) specifically created in the applicable agreement(s) under which the document is submitted. The user of this document may voluntarily provide suggestions, comments or other feedback to Nokia in respect of the contents of this document ("Feedback").

Such Feedback may be used in Nokia products and related specifications or other documentation. Accordingly, if the user of this document gives Nokia Feedback on the contents of this document, Nokia may freely use, disclose, reproduce, license, distribute and otherwise commercialize the feedback in any Nokia product, technology, service, specification or other documentation.

Nokia operates a policy of ongoing development. Nokia reserves the right to make changes and improvements to any of the products and/or services described in this document or withdraw this document at any time without prior notice.

The contents of this document are provided "as is". Except as required by applicable law, no warranties of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose, are made in relation to the accuracy, reliability or contents of this document. NOKIA SHALL NOT BE RESPONSIBLE IN ANY EVENT FOR ERRORS IN THIS DOCUMENT or for any loss of data or income or any special, incidental, consequential, indirect or direct damages howsoever caused, that might arise from the use of this document or any contents of this document.

This document and the product(s) it describes are protected by copyright according to the applicable laws.

Nokia is a registered trademark of Nokia Corporation. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Revision history and metadata

Please delete this slide if document is uncontrolled

Document ID: DXXXXXXXXX
Document Location:
Organization:

Version	Description of changes	Date	Author	Owner	Status	Reviewed by	Reviewed date	Approver	Approval date
		DD-MM-YYYY					DD-MM-YYYY		DD-MM-YYYY