The motivation for specification languages during software development

Dr. Constantinos Constantinides, P.Eng.
Computer Science and Software Engineering
Concordia University

Introduction to requirements

- Requirements is a collective term that describes the behavior of a system must have, as well as quality attributes or constraints that the system must satisfy in order to be accepted by stakeholders.
- Functional requirements (FRs): Observable behavior.
- Non-functional requirements (NFRs): Quality attributes and constraints.

Non-functional Requirements: Quality

- The degree to which a system, component, or process meets specified requirements (customer or user needs or expectations). (IEEE)

- The characteristics of an entity (component, system) that affect its ability to satisfy stated and implied needs. (ISO)

The ISO/IEC 25010 standard

- The ISO/IEC JTC is a joint technical committee of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC).
- Its purpose is to develop, maintain and promote standards in the fields of information technology (IT) and Information and Communications Technology (ICT).

source: iso25000.com

Functional Requirements are captured by Use cases

- **Actor**: Can be a person, or an external computer system. Actors interact with the system.

- **Scenario**: Specific sequence of actions and interactions between actors and the system under discussion, e.g. the scenario of using a bank machine to successfully depositing money into an account.

- **Use case**: A collection of related success and failure scenarios that describe actors using a system to support a goal.

Use Cases

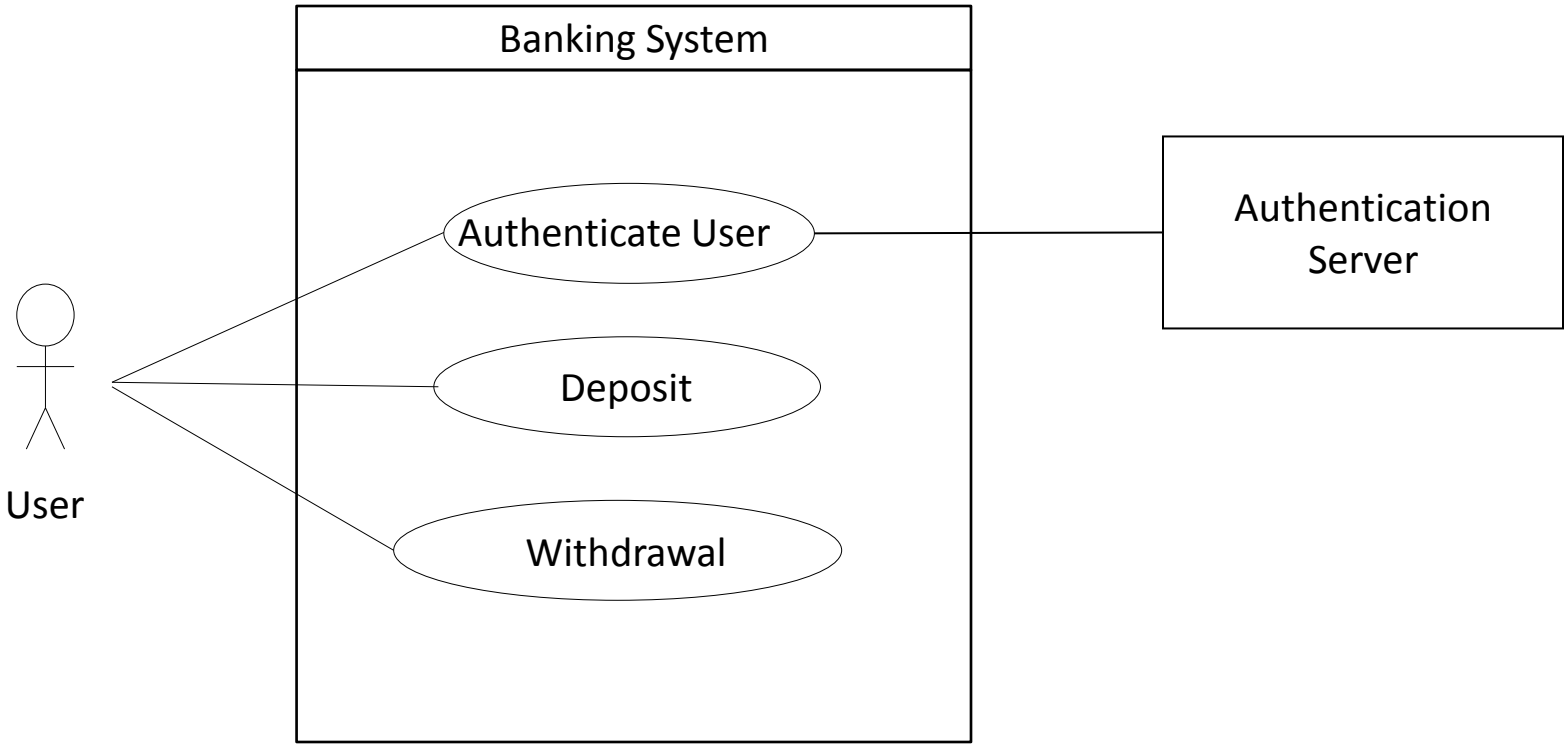
- Each Use Case will be expressed by a narrative description (in natural language).
- Normally a Use Case would contain a set of assertions: Precondition and postcondition.
- A use case contains a success scenario and one or possibly more failure scenarios.
- Example: Use Case Deposit

  Client enters the amount to deposit. The system responds with a confirmation.

  Precondition: Client is authenticated.

  Postcondition: Account state is updated. Transaction is recorded.

Use Case Diagram



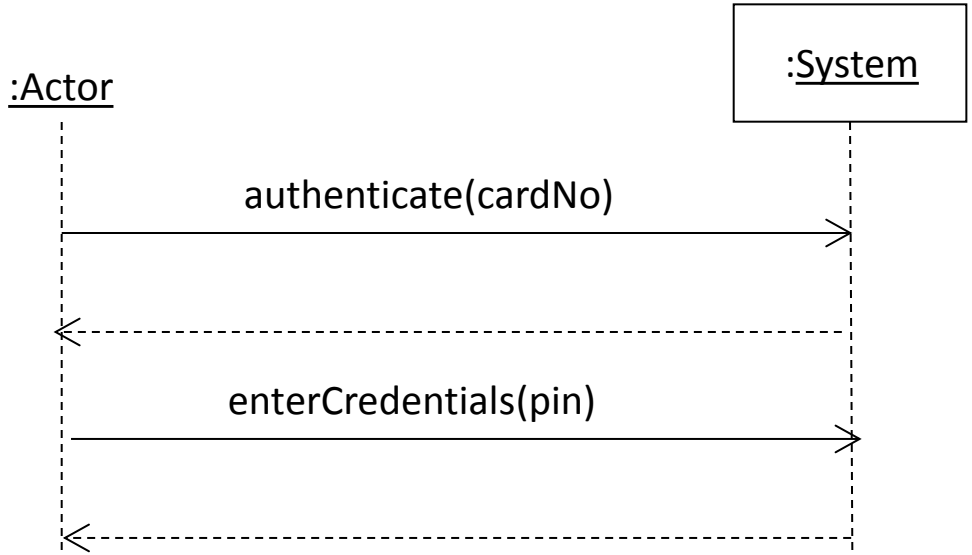
System behaviour

- It is useful to investigate and define the expected behaviour of the entire system as a black box.
- The system behaviour is a description of **what** the system does (without an explanation of how it does it).
- Use cases describe how external actors interact with the software system.
- During this interaction, an actor generates events.

From Use Cases to System Sequence Diagrams

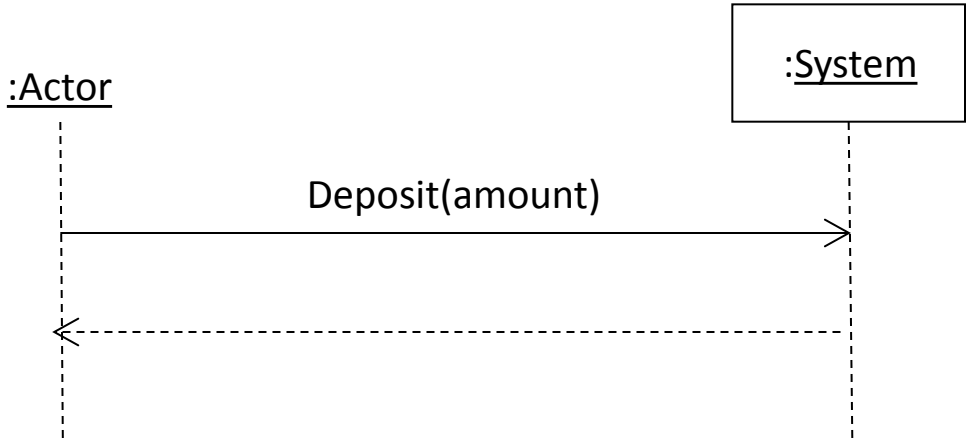
Authenticate User

Client enters their card. The system responds by requesting for the client's PIN….



Deposit

Client enters the amount to deposit.
The system responds with a confirmation…



All use cases map onto System Sequence Diagrams.

System behaviour and System Sequence Diagrams

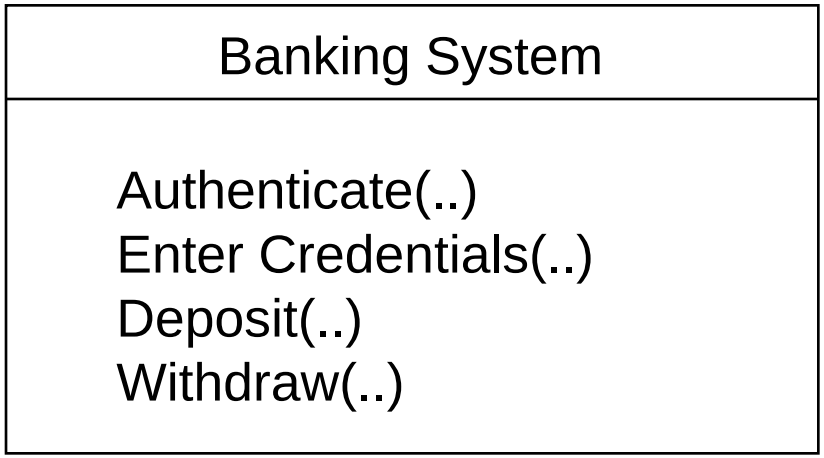
- A System Sequence Diagram shows, for a particular scenario of a use case, the events that external actors generate, their order, and possible inter-system events.

- All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems.

System behaviour

- A request event initiates an operation upon the system.
- The set of all required system operations forms the exposed interface of the system:
    - Authenticate
    - Enter Credentials
    - Deposit
    - Withdraw

System operations and the system interface



- We can use **Z** (or **Object Z**) to create a specification of the system as a whole.
- In the **Z Specification Language**, the system is viewed by its **State** and a collection of **Operations**.
- In the **Object-Z Specification Language**, the system is viewed as a **Class**, where State and Operations are encapsulated.

Software Requirements Specification

- In Z/Object-Z, operations describe in detail **what is expected**.
- Each operation is associated with a pair of assertions: Precondition and postcondition.
- In addition, in Object-Z, a class would normally be associated with an invariant.
- Essentially, the specification describes a **contract**.
- Together with the rest of the artifacts produced during Analysis, this contract is placed in a document called **Software Requirements Specification**.