Making Use-Cases great again

A few observations about good-old use cases

האיגוד הישראלי להנדסת מערכות
INCOSE IL
Israeli Society For Systems Engineering

amir@amirtomer.com

Prof. Amir Tomer – short bio

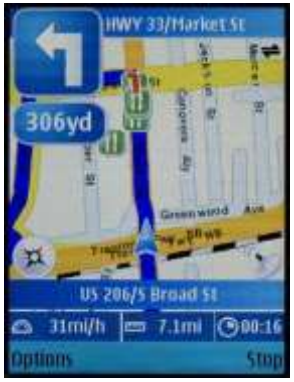
- Industrial experience
  - 1982-2009: Rafael, Advanced Defense System
    - 1999-2006: Director of Software and Systems Engineering Processes
- Academic experience
  - 2009-now: Head of Software Engineering Department, Kinneret College on the Sea of Galilee, Jordan Valley, Israel
  - 1994-now: Senior lecturer, Technion, Haifa, Israel
- Research interests
  - Software-intensive Systems Engineering
  - Model-Based systems/Software Engineering
  - Software reuse
- Professional Associations
  - The Israeli Chamber of Information Technology
    - Member of presidency, head of the academic liaisons and accreditation committee
  - The Israeli Society for Systems Engineering
    - "The Voice of the Systems" Journal Editor

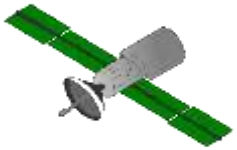
A question...

- What is your experience in writing Use Cases?

System in its Environment

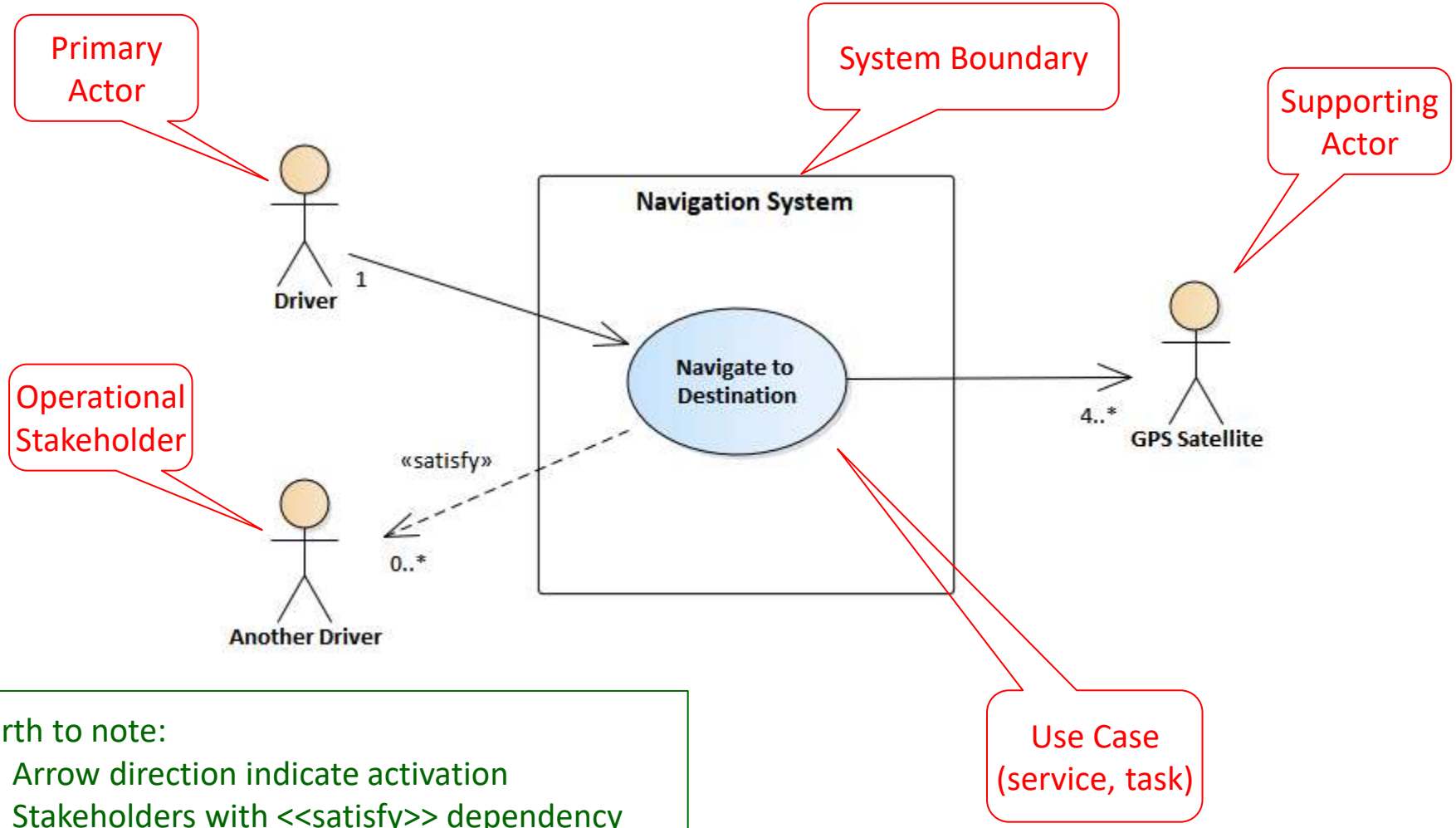
- The environment of a system [e.g. navigation system] is composed of 3 major types of entities
  - Users (aka primary actors) [e.g. car driver]
    - Entities who **interact** with the system to **achieve** something
  - Supporters (aka supporting actors) [e.g. GPS satellite]
    - Entities whom the system **interacts with** to **help** it do its tasks
  - Other operational stakeholders [e.g. other drivers]
    - Entities who **do not interact** with the system, but **get benefits** from its operation





Use-Case Diagram

- A use-case diagram is a UML Representation of a system in its environment



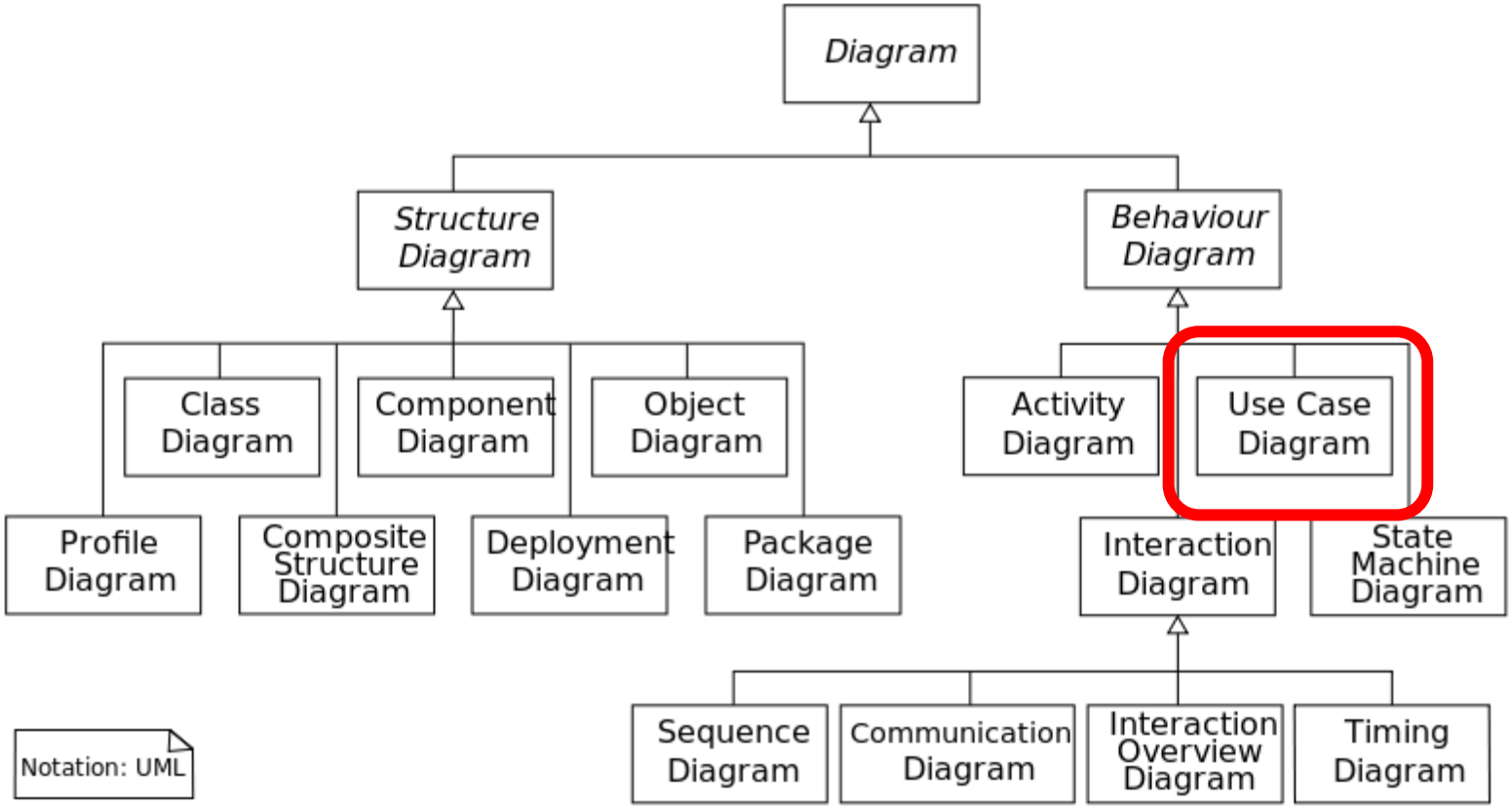
Worth to note:

1. Arrow direction indicate activation
2. Stakeholders with <<satisfy>> dependency

A question...

- Is a Use-Case Diagram a behavioral model?

UML Classifies the Use Case Diagram as a behavioural model

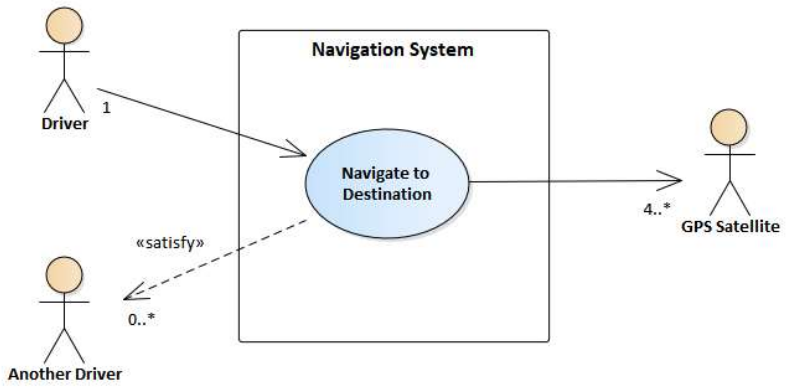


- However...

Observation # 1

The Use Case Diagram is NOT a behavioral Model

- The use-case diagram is a static model, and therefore it cannot represent (dynamic) behavior
  - The diagram does contain:
    - A set of use cases
    - A set of actors
    - A set of stakeholders
    - Relations between elements of those sets



- Conclusion
  - The behavior of the system throughout every use case should be modeled somehow
    - Textual description
    - Activity Diagram
    - Sequence Diagram

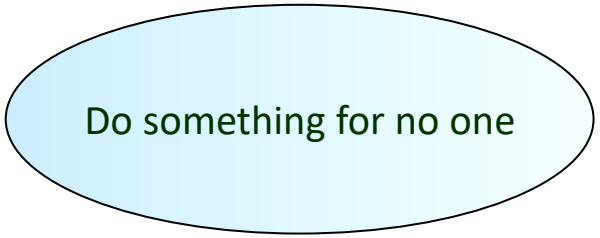
A question...

- Must every use-case have at least one primary actor?

Observation # 2

There could be use-cases WITHOUT primary actors

- A typical example: A use-case named “Perform a periodic BIT”
  - This use case is invoked spontaneously every TBD seconds/minutes by an internal event
- Such use cases are called “spontaneous” or “internal”
- However, a spontaneous use-case must have at least one stakeholder
  - Otherwise it is a “useless” case...



A question...

- Is a burglar a primary actor of an alarm system?

Observation # 3:

A burglar is NOT a primary actor of an alarm system

- A burglar activates the alarm system, so it might be considered a primary actor, however...

a) Other entities might as well activate the alarm:
   - A cat
   - A curtain
   - A lightning
   - A spot on the detector

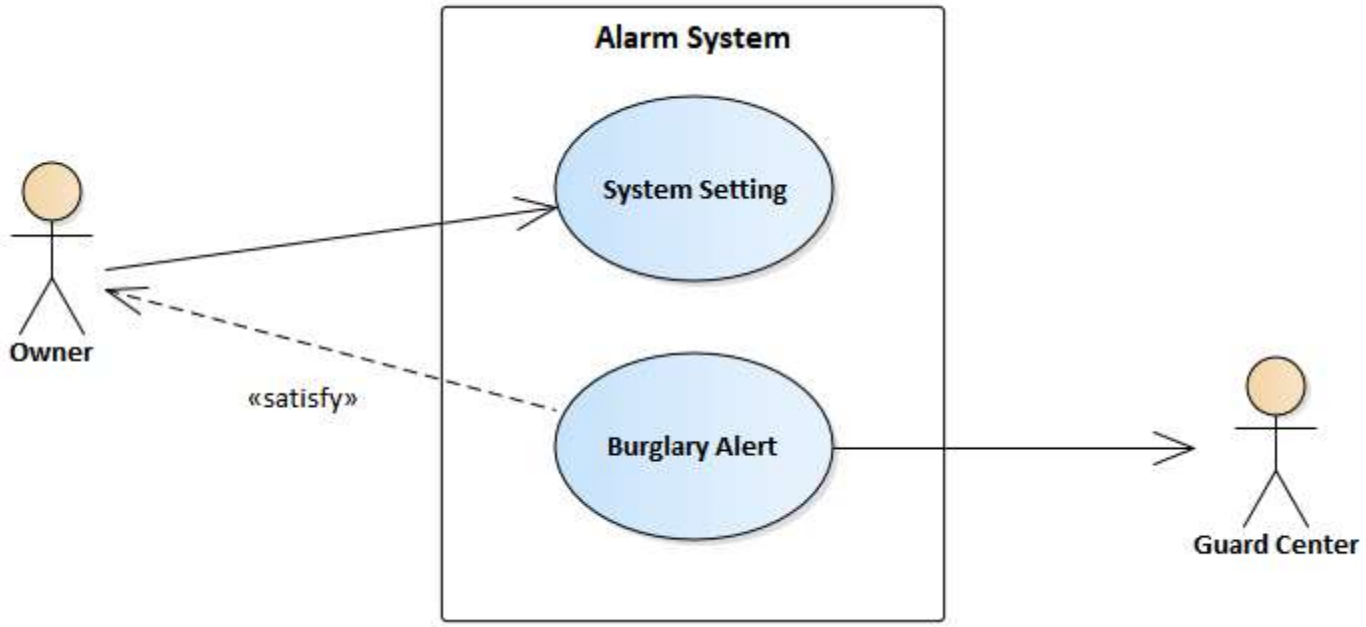
   Are all these primary actors?

b) A burglar does not **initiate** the alarm activation in order to **achieve** something
   - The alarm activation is a spontaneous event caused by the consolidation algorithm over its detectors

c) The "Burglary alert" use case is not for the burglar, but rather for the owner
   - The owner is a stakeholder, whose interest (benefit) is that a guard center is alerted when there is a **suspicion of burglary**

d) The owner may be a primary actor of other system use cases, e.g. "System setting"

A sample UC Diagram for an alarm system



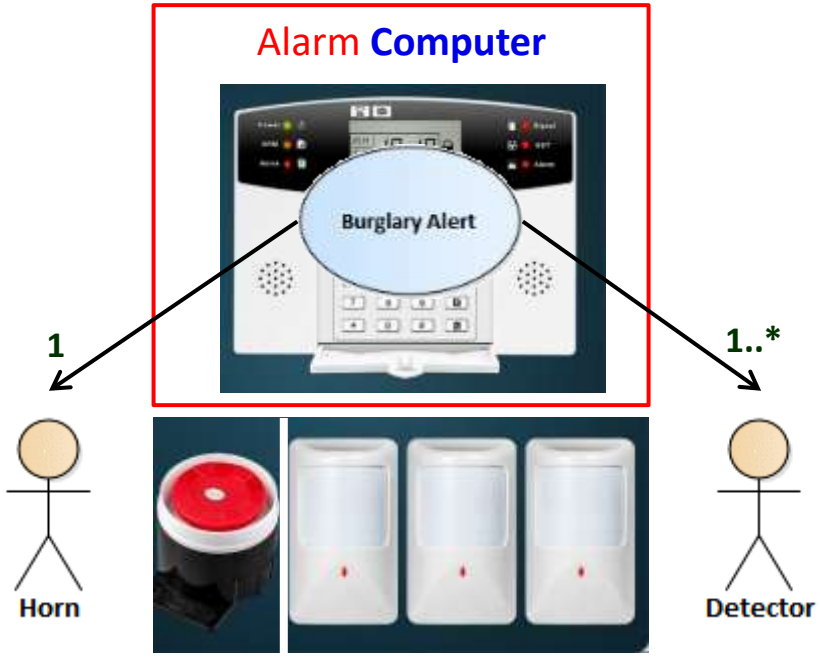
A question...

- Should the detectors and the horn be shown in the diagram?

Observation # 4:

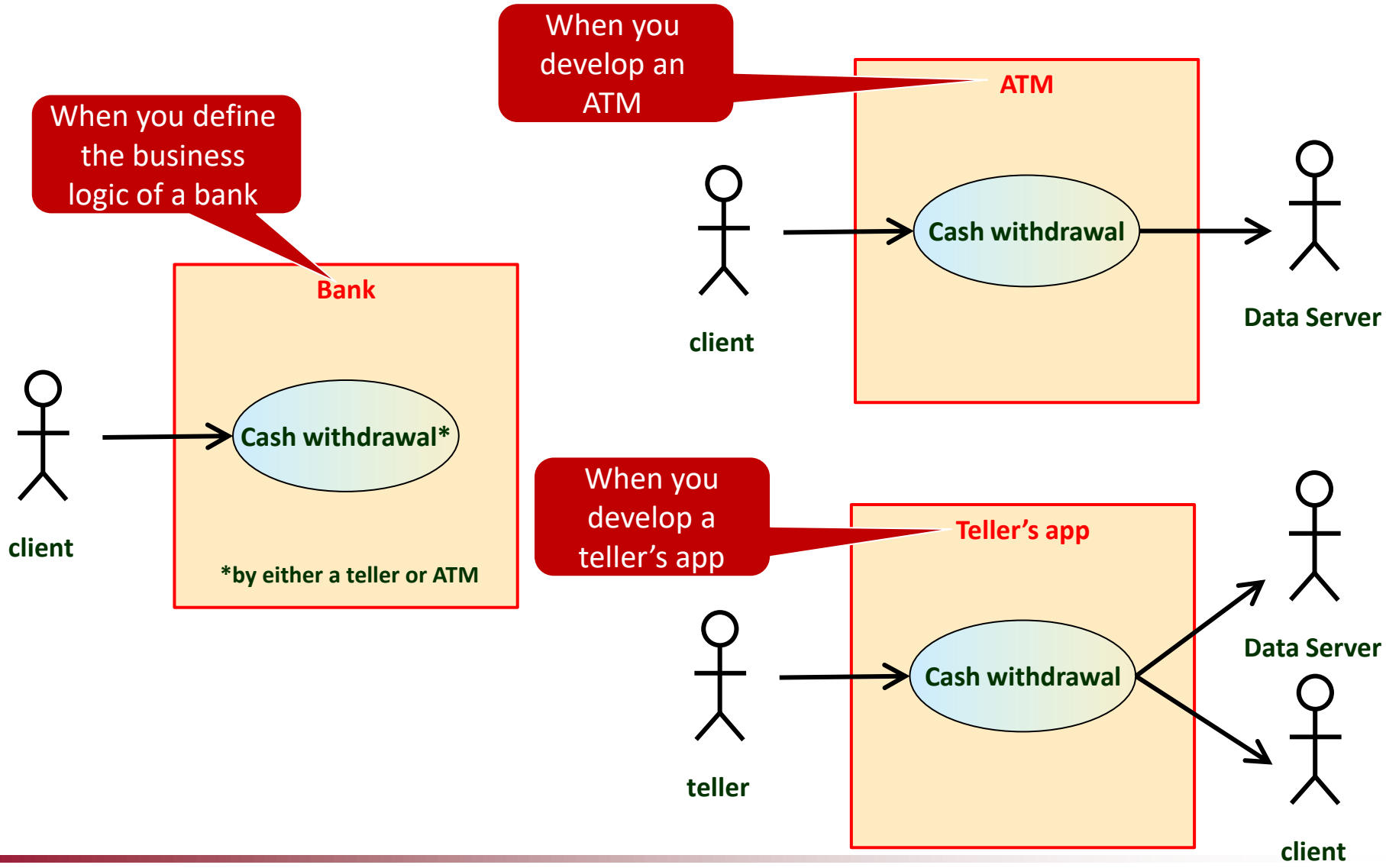
The presentation of entities in a UCD depends on the definition of "system"

- An alternative question:
  - Are the detectors and horn parts of the **system** or entities of the **environment**?
    - depends...
      - compare the two following representations



Picture source: ebay.com: Home Wired GSM Security Burglar Alarm System

Another example:
A “Cash withdrawal” use case in various contexts



Use Case Specification

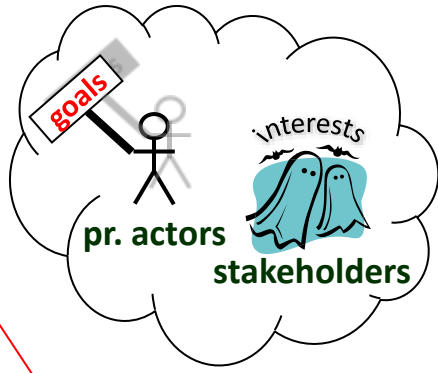
A use case is **successful** when:
1. The primary actors achieved their goals
2. The stakeholders' interests were fulfilled

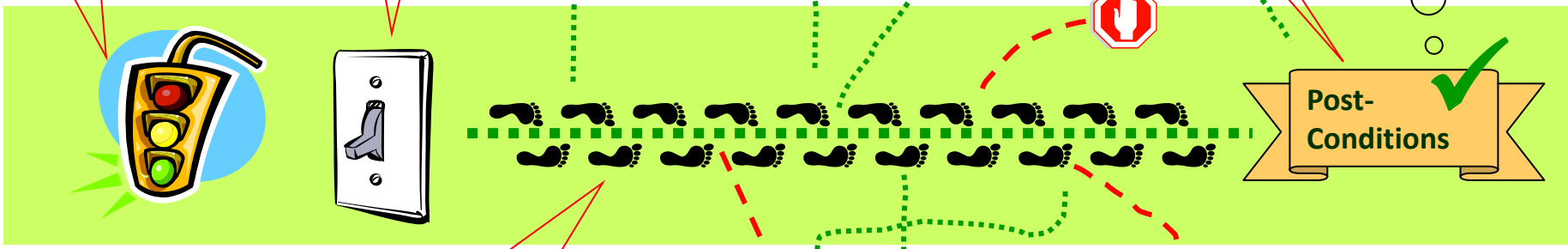
Pre-conditions
The circumstances under which the use case can start

Trigger
The initiative event of the use case

Alternative
A different scenario leading to "success"

Post-conditions
The results of "success"





Main Success Scenario - MSS
The ideal and shortest scenario leading to success

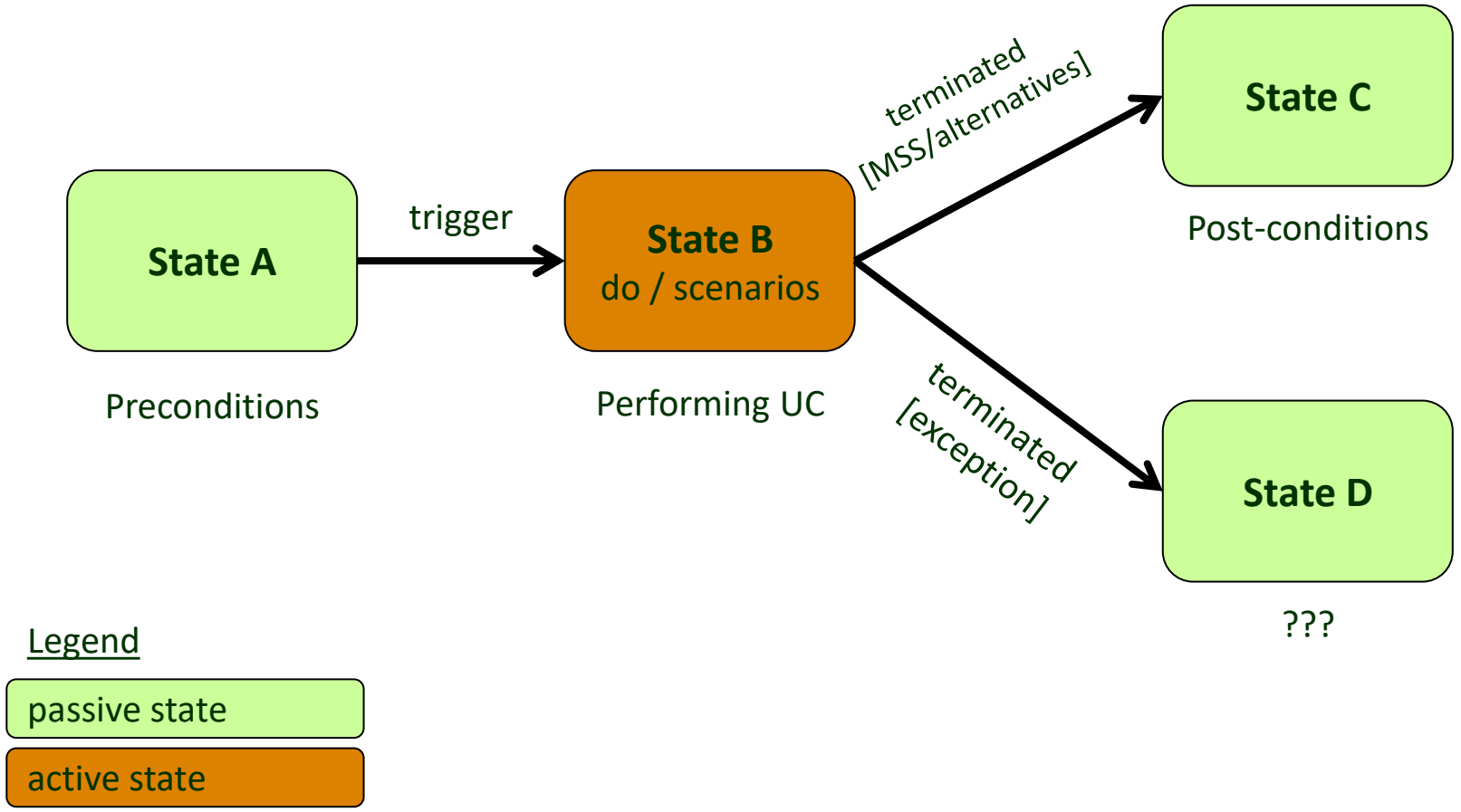
Exception
A scenario leading to failure

Textual representation of a use case (“Call Elevator”)

Actors and their goals	• User: Wants to get an available elevator
Other Stakeholders and their Interests	• Building owners: Ensure efficient service [reliability, availability]
preconditions	• User in front of an elevator’s door
Post-conditions	• An elevator is at the floor • The door is open
Trigger	**The user** presses one of the floor buttons
Main Success Scenario (MSS)	**1.** **The system** receives a call from the floor **2.** **The system** selects an elevator **3.** **The system** assigns the floor to the selected elevator **4.** **The system** puts on the pressed button **5.** **The assigned elevator*** arrives at the floor and opens its door **6.** **The system** puts off the pressed button
Branch A	*Alternative* at step 2 of MSS: An elevator is already assigned for this task 2A1. Skip to step 5
Branch B	*Exception* at step 3 of MSS: No available elevator 2B1. The scenario terminates

* **The assigned elevator** is a part of **the system**

A state-machine view



A question...

- What is the difference between a use case and a user story (e.g. in scrum)?

Observation #5:

User stories are NOT use cases

- A user story is of the form*

As a < type of user >, I want < some goal > so that < some reason >.

primary actor

actor's goal

post-conditions?
stakeholders' benefits?

- A user story does not answer questions like:
  - How to initiate the story? [trigger]
  - Under which circumstances? [preconditions]
  - What is the interaction between the user and the system during the story? [MSS]
  - Are there alternative ways to achieve the goal? [alternatives]
  - What happens if the goal is not achieved? [exceptions]

requirements or implementation?

---

*Mike Cohen: https://www.mountaingoatsoftware.com/agile/user-stories

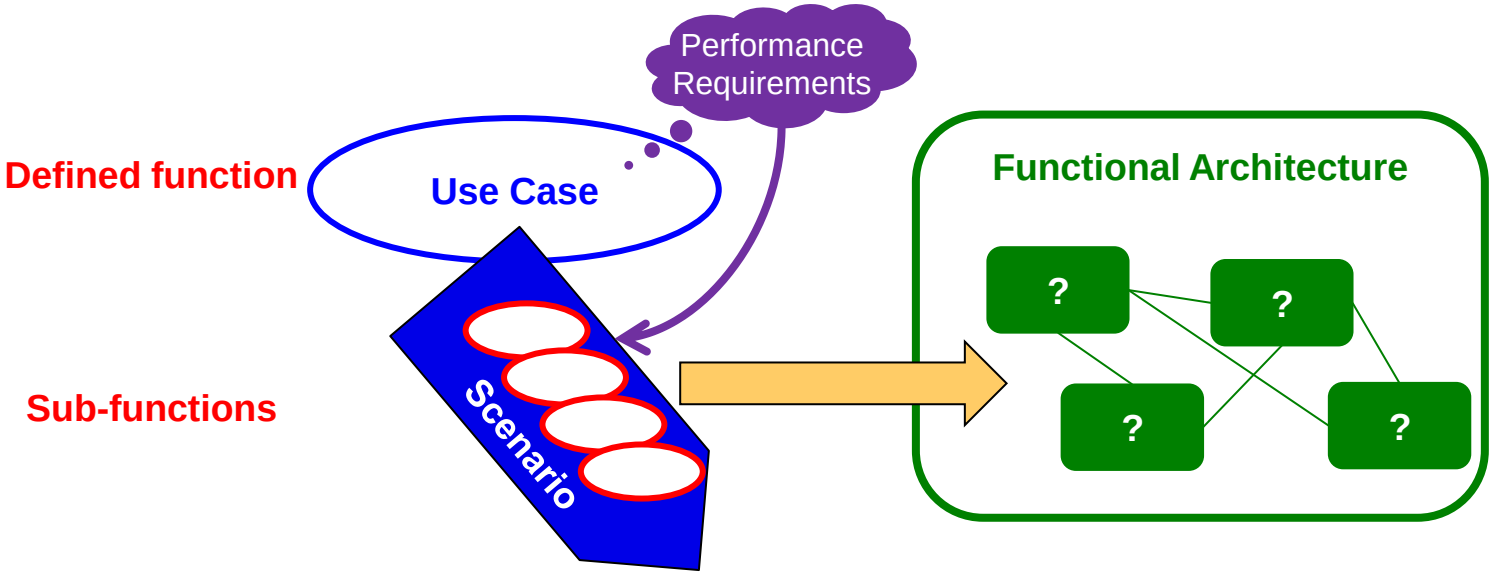
A question...

- What is the contribution of use cases to later life-cycle stages?

Observation #6:

Use cases are the basis for the functional architecture

- Functional Analysis*
  - Examination of a **defined function**
    - to identify all the **sub-functions** necessary to accomplish that function,
    - to identify **functional relationships and interfaces** (internal and external) and capture these in a **functional architecture**,
    - to flow down **upper-level performance** [and other non-functional] **requirements** and to assign these requirements to **lower-level sub-functions**



*ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary

Functional Analysis of the Call Elevator use case

- Proposed functional components
  - Central Command and Control (one for the entire system)
  - Elevator Command and Control (one for each elevator)
- The sub-functions are identified and each one is allocated to a functional component

Main Success Scenario (MSS)	**1. The system** receives a call from the floor **2. The system** selects an elevator **3. The system** assigns the floor to the selected elevator **4. The system** puts on the pressed button **5. The assigned elevator*** arrives at the floor and opens its door **6. The system** puts off the pressed button
Branch A	*Alternative* at step 2 of MSS: An elevator is already assigned for this task **2A1.** Skip to step 5
Branch B	*Exception* at step 3 of MSS: No available elevator **3B1**. The scenario terminates

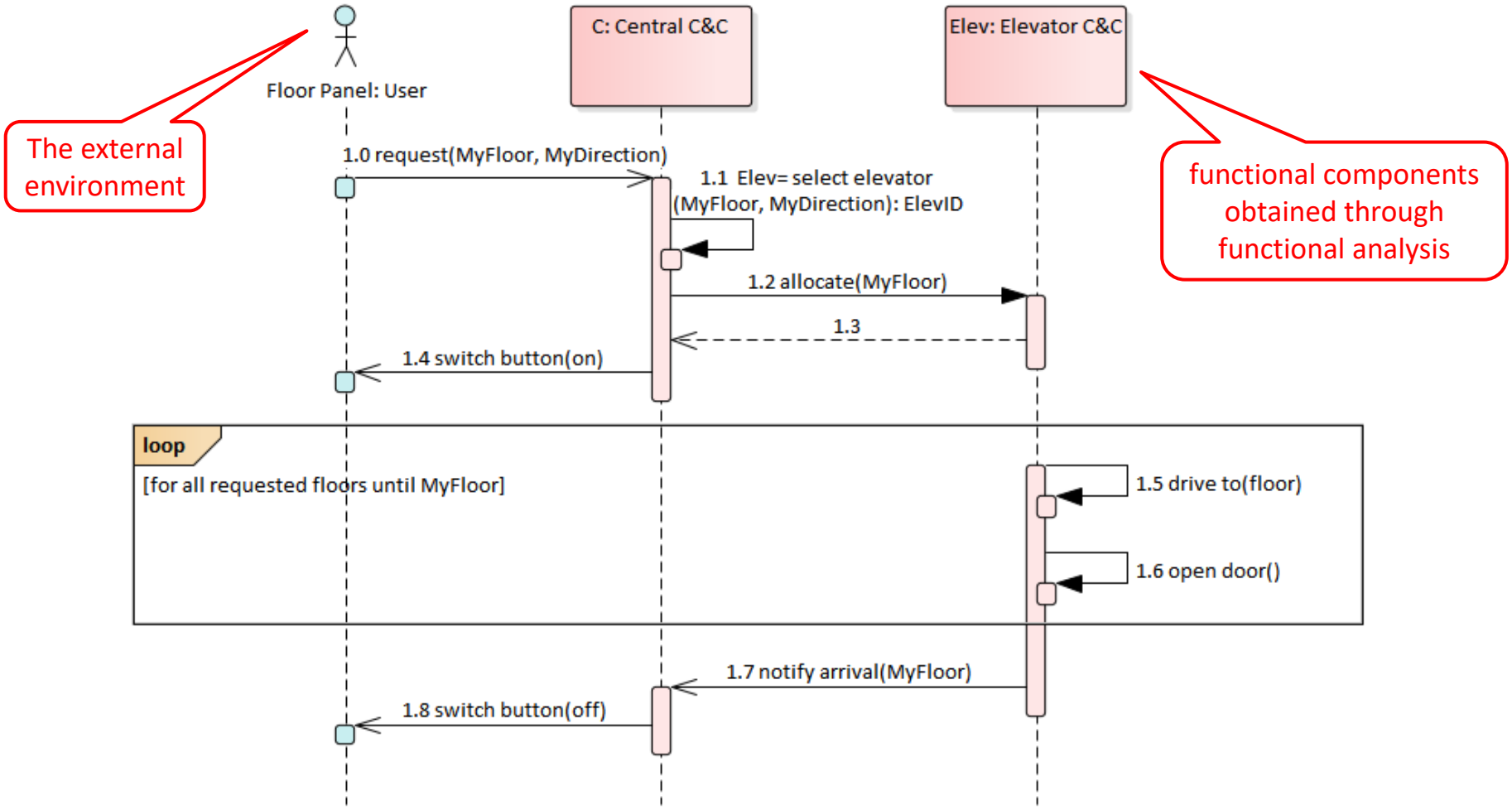
Central C&C

Elevator C&C

Observation #7:

Behavioral models of the system are derived from use cases

- Example: Calling an elevator – an interaction model (sequence diagram)



Observation #8:

Tests cases are derived from use cases

Test Case Example1 (simple test)

Test Case #: 2.2 **Test Case Name:** Change PIN **Page:** 1 of 1

System: ATM **Subsystem:** PIN

Designed by: ABC **Design Date:** 28/11/2004

Executed by: **Execution Date:**

Short Description: Test the ATM Change PIN service

Pre-conditions

The user has a valid ATM card - The user has accessed the ATM by placing his ATM card in the machine
The current PIN is 1234
The system displays the main menu

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Click the 'Change PIN' button	The system displays a message asking the user to enter the new PIN		
2	Enter '5555'	The system displays a message asking the user to confirm (re-enter) the new PIN		
3	Re-enter '5555'	The system displays a message of successful operation The system asks the user if he wants to perform other operations		
4	Click 'YES' button	The system displays the main menu		
5	**Check post-condition 1**			

Post-conditions

1. The new PIN '5555' is saved in the database

Conclusion

- What have we learnt?
  - Use cases specify the interaction of a system with its environment
  - Use cases capture the full functionality of the system
  - Use cases may be very useful if they are written correctly and precisely
  - Use cases are the baseline of developing a software –intensive system

- What is still missing?
  - Relations between use cases (include, extend, ...)
  - The full function of the system as a set of separate use cases
  - Relation to non-functional requirements
  - More example and hands-on experience

More to come...

- Other tutorials by this author
  - Webinars
    - The functionality behind the non-functional requirements (~1 hour)
    - Relay Race: The shared challenge of systems engineers and software architects (~1 hour)
  - Workshops
    - Software requirements and use cases (1 day)
    - Software engineering for systems engineers (2 days)
    - Software-intensive systems architecture (3 days)
- For details go to https://incoseil.org

Thank you for listening