BBM-382 – SOFTWARE ENGINEERING
SPRING 2021
Lecture 4

Assoc.Prof.Dr. Ayça KOLUKISA TARHAN
Dr. Tuğba ERDOĞAN

Before we start

- What is requirement?
  - Requirement, functional
    - A statement of some <u>function</u> or <u>feature</u> that should be implemented in a system
  - Requirement, non functional
    - A statement of a <u>constraint</u> that applies to a system

Functional (A) of non-functional (B) ?

The system sends an email to the customer when she places a new order.

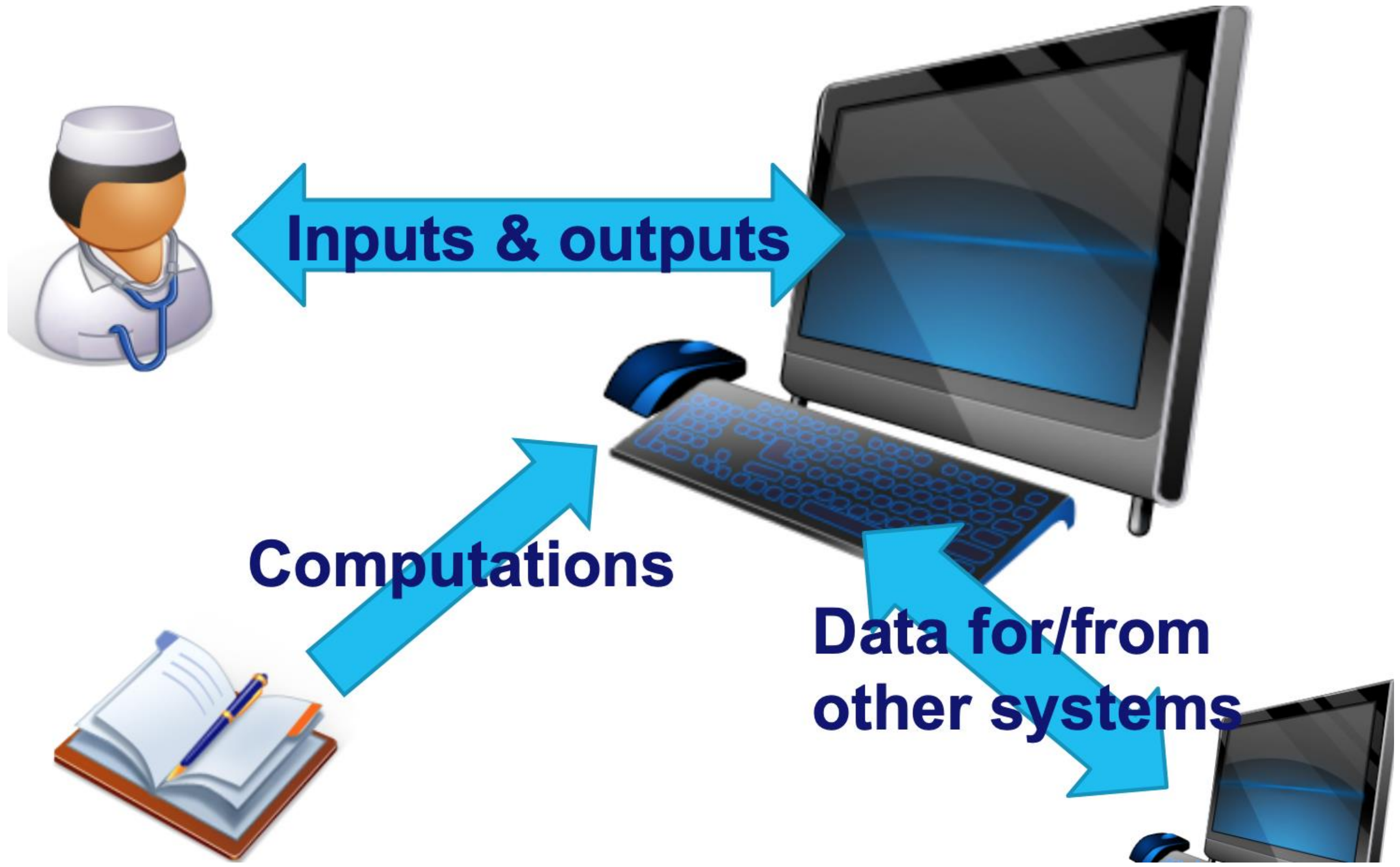
functional

Functional (A) of non-functional (B)?

The mail should be send not later than 12 hours after the order has been placed.

non-functional

Functional requirements frequently describe



Non-functional requirements

- Non-functional requirement relates to quality attributes: e.g., **performance, learnability, availability**
- functional requirement: "when the user presses the green button the Options dialog appears”:
  - **performance**: how quickly the dialog appears;
  - **availability**: how often this function may fail, and how quickly it should be repaired;
  - **learnability**: how easy it is to learn this function.

Exercise: Functional or Non-functional

- The system shall be able to process at least 40 executing jobs at a time.

  non-functional, performance

- The system shall provide the means for the resource provider to see on which project his resource is working.

  functional

- The system shall provide the means for the system admin to perform his actions on a computer with Windows XP, Mac OS X or Linux.

  non-functional, portability

- If one of the resource disappears while it was performing a job, the system should requeue the job.

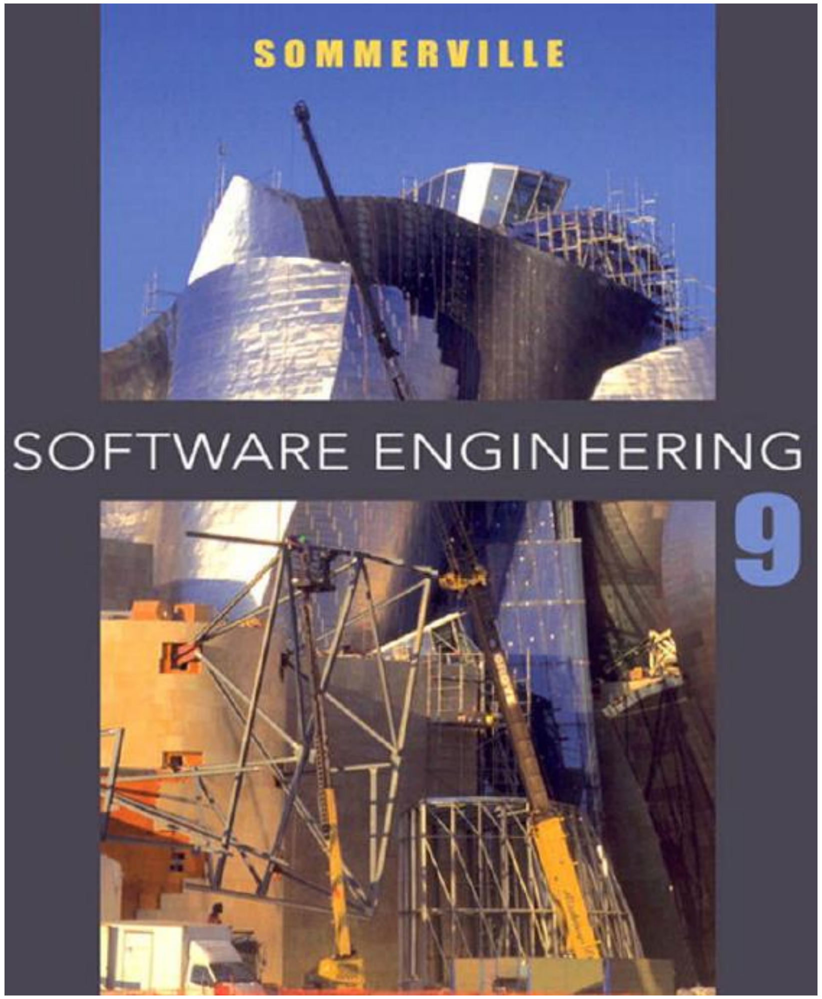
  functional

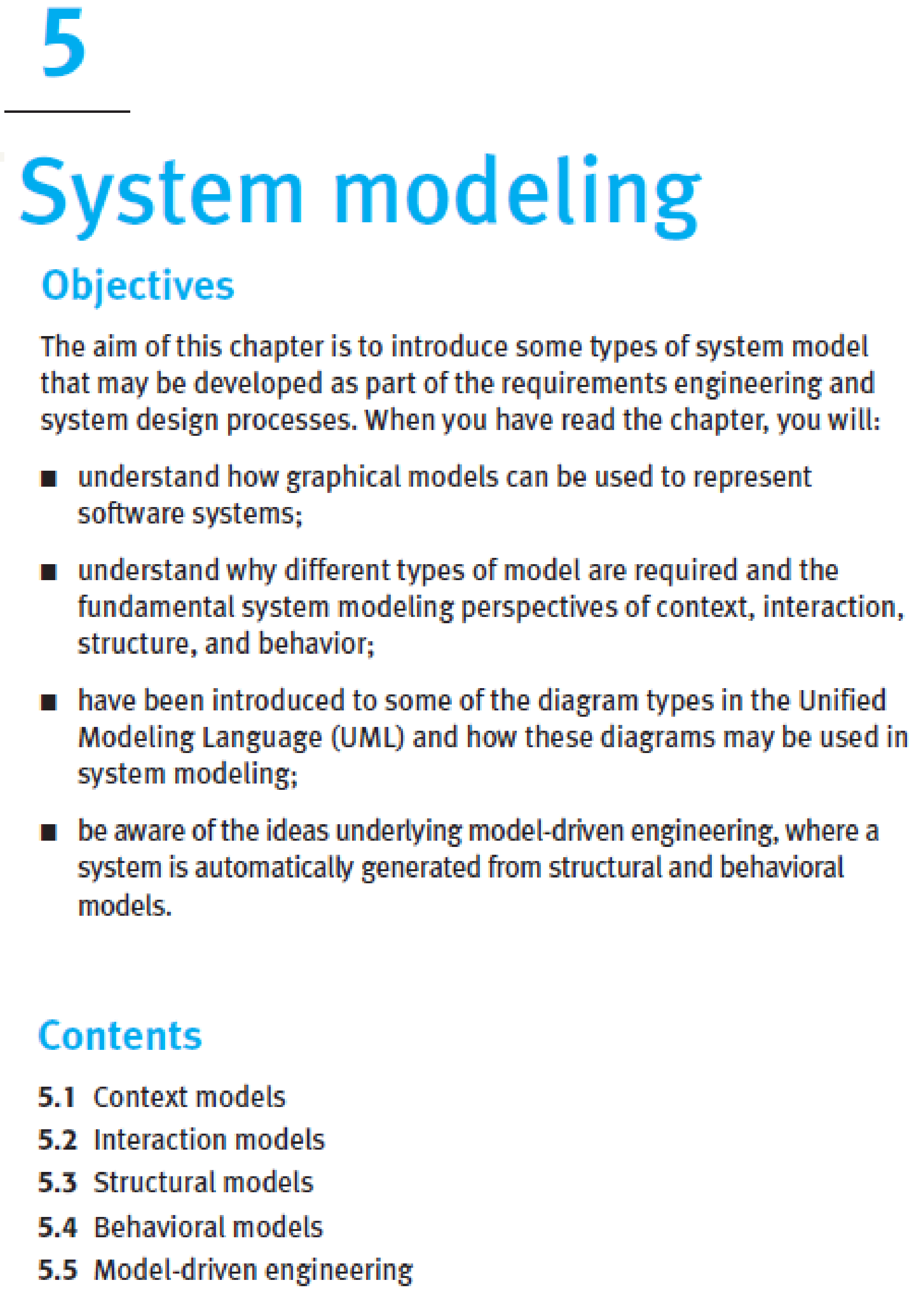
- The (de-)installation of the software needed by resource providers should not require a computer expert.

  non-functional, installability

From the book

- Read it online!





5

System modeling

Objectives

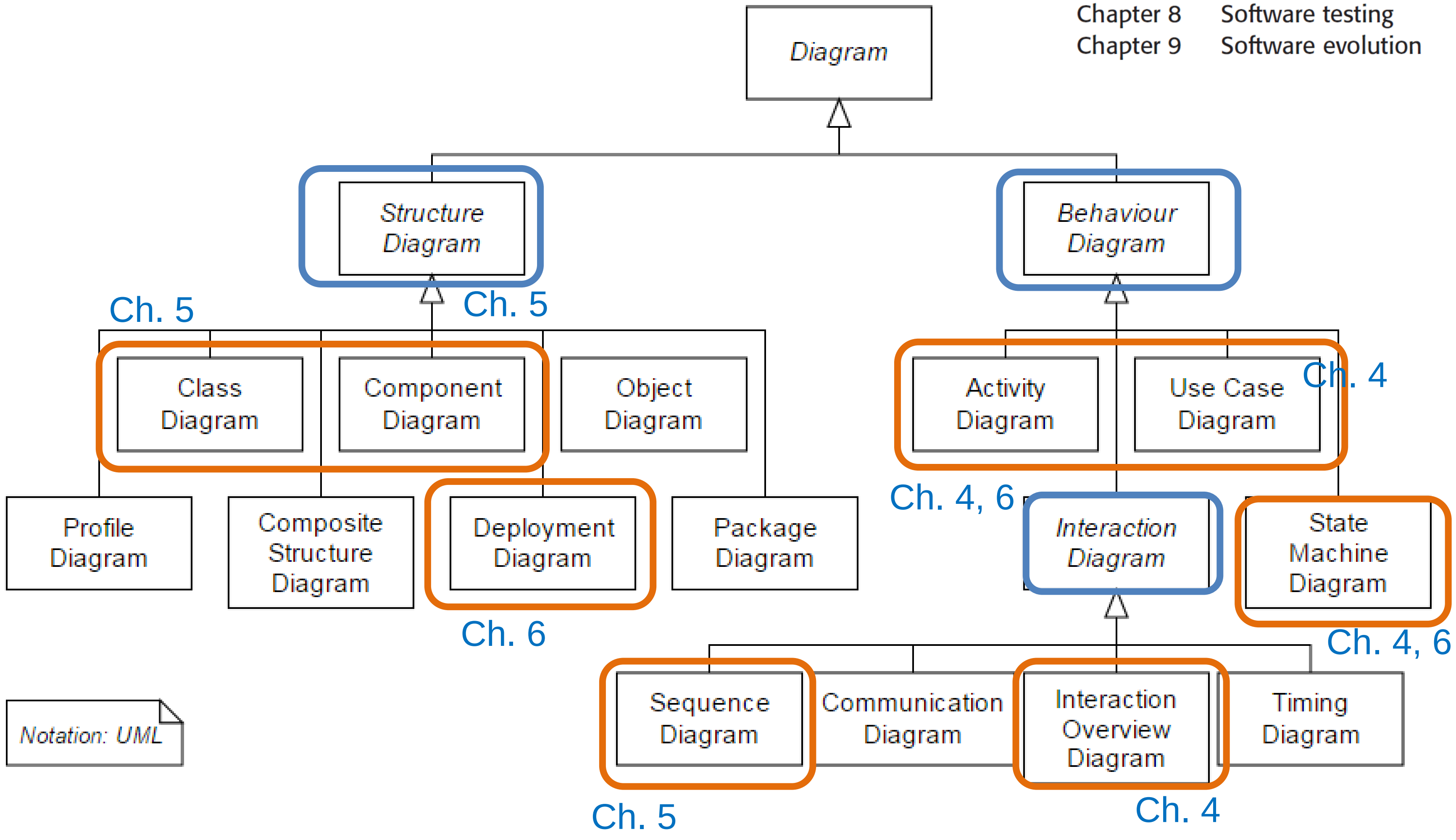
The aim of this chapter is to introduce some types of system model that may be developed as part of the requirements engineering and system design processes. When you have read the chapter, you will:

- understand how graphical models can be used to represent software systems;
- understand why different types of model are required and the fundamental system modeling perspectives of context, interaction, structure, and behavior;
- have been introduced to some of the diagram types in the Unified Modeling Language (UML) and how these diagrams may be used in system modeling;
- be aware of the ideas underlying model-driven engineering, where a system is automatically generated from structural and behavioral models.

Contents

UML

- 14 diagrams in total



Unified Modeling Language - UML

- UML has become the de facto standard for modeling software applications and is growing in popularity in modeling other domains. Its roots go back to three distinct methods:
  - Booch Method -- Grady Booch
  - Object Modeling Technique -- James Rumbaugh
  - Objectory -- Ivar Jacobson
- Known as the Three Amigos, Booch, Rumbaugh, and Jacobson kicked off what became the first version of UML, in 1994.
  - In 1997, UML was accepted by the Object Management Group (OMG) & released as UML v1.1.
- Since then, UML has gone through several revisions and refinements leading up to the current 2.5.1 (2017) release.
  - https://www.omg.org/spec/UML/About-UML/
  - Specification is extensive in terms of page count (796 pages)

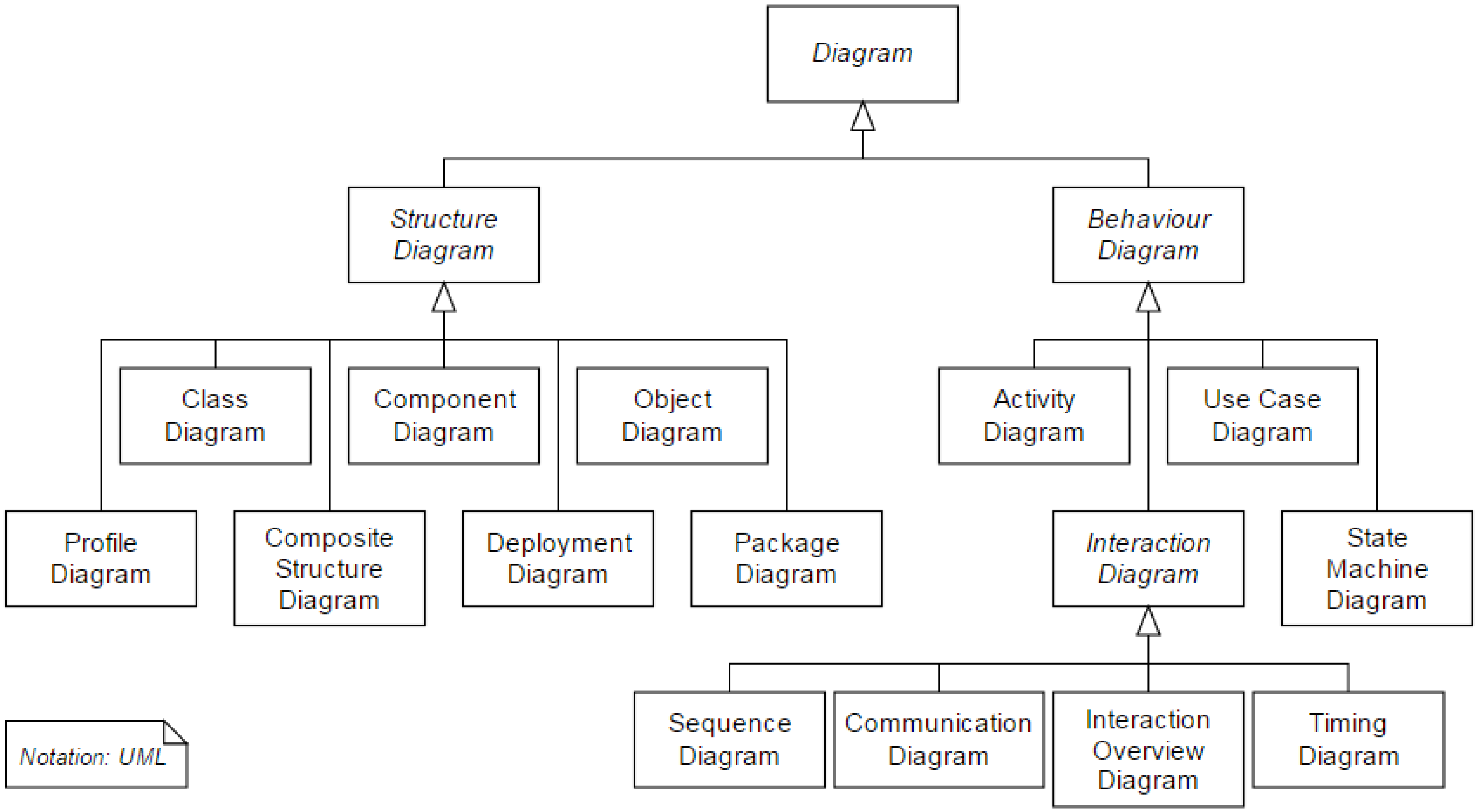
UML Basics – 1

- UML is a language, it has both syntax and semantics.
  - When you model a concept in UML, there are rules regarding how the elements can be put together and what it means when they are organized in a certain way.
  - UML is intended not only to be a pictorial representation of a concept, but also to tell you something about its context.

- You can apply UML in any number of ways; common uses include:
  - Designing software
  - Communicating software or business processes
  - Capturing details about a system for requirements or analysis
  - Documenting an existing system, process, or organization

UML Basics – 2

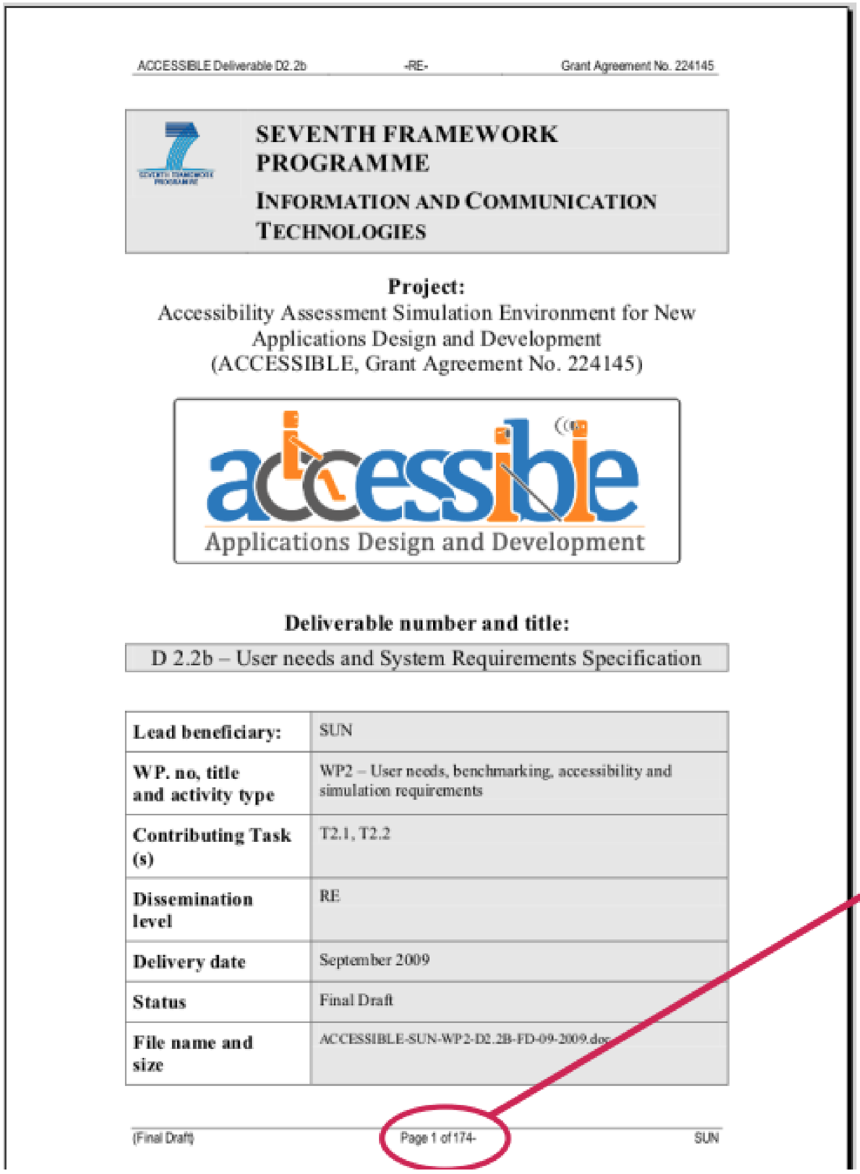
- The basic building block of UML is a *diagram*.
  - A diagram graphically represents things, and the relationships between these things.
    - These things can be representations of real-world objects, pure software constructs, or a description of the behavior of some other object.



UML Rules of Thumb

- Nearly everything in UML is optional
  - UML provides a language to capture information that varies greatly depending on the domain of the problem, and you don't need to use every part of UML in every model you create.
  - At times there is more than one way to convey the same information; use what is familiar to your audience.
- UML models are rarely complete
  - As a consequence of everything being optional, it is common for a UML model to be missing some details about a system. The trick is to not miss key details that could impact your system design.
- UML is designed to be open to interpretation
  - While the UML specification does a good job of laying down the groundwork for a modeling language, it is critical that within an organization or group of users you establish how and when to use a language feature.
    - E.g. an aggregation relationship to indicate a C++ pointer and a composition relationship to indicate a C++ reference.
  - It is a good practice to put together a document on modeling guidelines.

Requirements document in practice



ACCESSIBLE Deliverable D2.2b -RE- Grant Agreement No. 224145

SEVENTH FRAMEWORK PROGRAMME
INFORMATION AND COMMUNICATION TECHNOLOGIES

Project:
Accessibility Assessment Simulation Environment for New Applications Design and Development
(ACCESSIBLE, Grant Agreement No. 224145)

accessible
Applications Design and Development

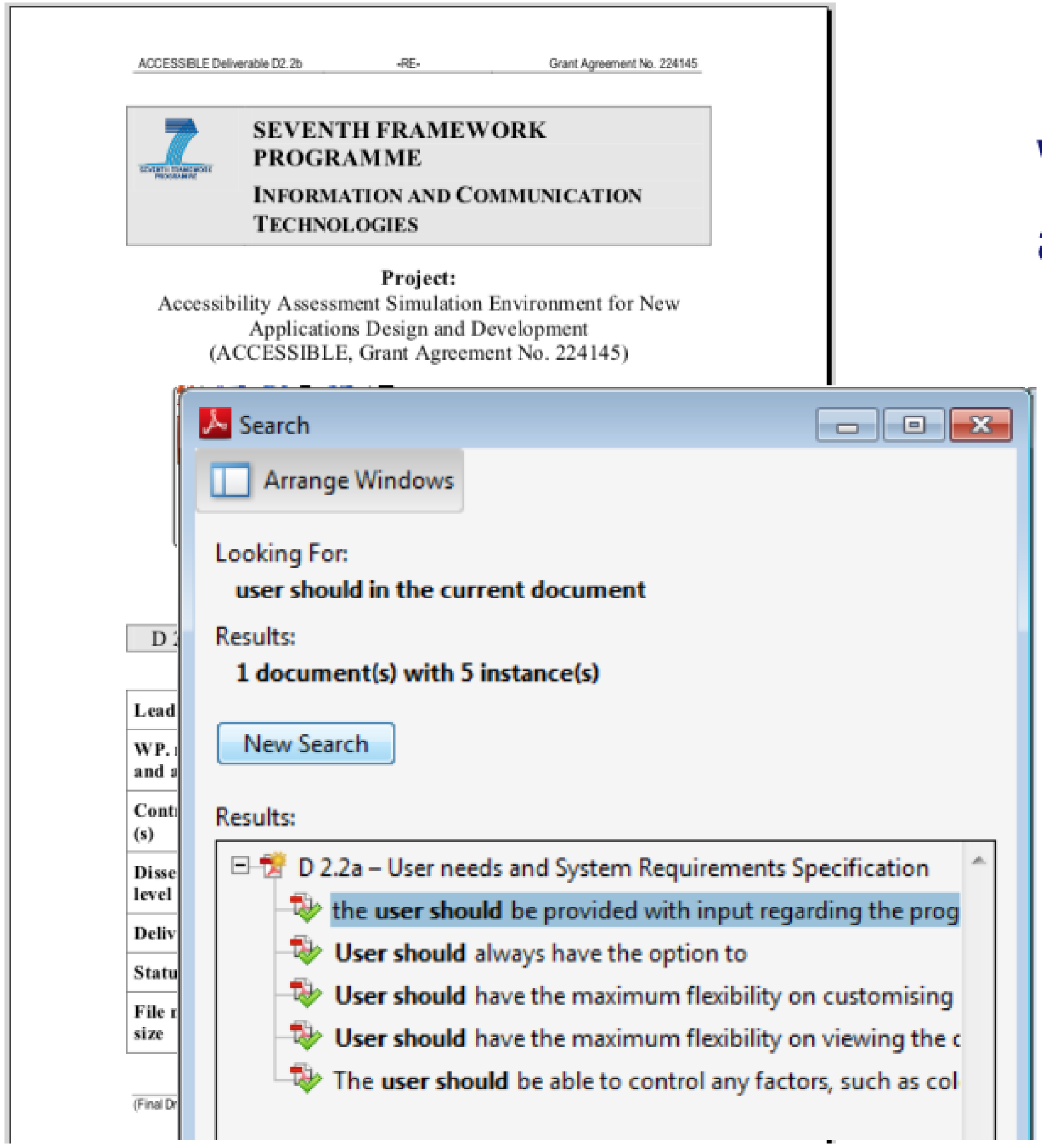
Deliverable number and title:
D 2.2b – User needs and System Requirements Specification

Lead beneficiary:	SUN
WP. no, title and activity type	WP2 – User needs, benchmarking, accessibility and simulation requirements
Contributing Task (s)	T2.1, T2.2
Dissemination level	RE
Delivery date	September 2009
Status	Final Draft
File name and size	ACCESSIBLE-SUN-WP2-D2.2B-FD-09-2009.doc

(Final Draft) Page 1 of 174- SUN

Page 1 of 174-

Requirements document



ACCESSIBLE Deliverable D2.2b -RE- Grant Agreement No. 224145

SEVENTH FRAMEWORK PROGRAMME

INFORMATION AND COMMUNICATION TECHNOLOGIES

Project:
Accessibility Assessment Simulation Environment for New Applications Design and Development
(ACCESSIBLE, Grant Agreement No. 224145)

Search

Looking For:
user should in the current document

Results:
1 document(s) with 5 instance(s)

Results:

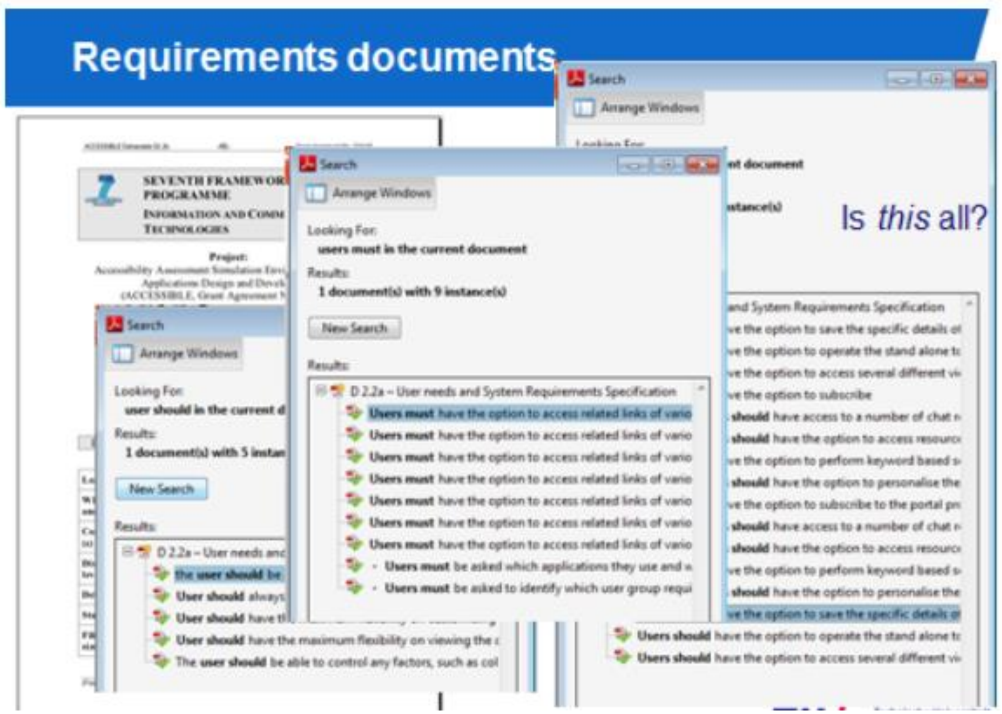
- D 2.2a – User needs and System Requirements Specification
  - the **user should** be provided with input regarding the prog
  - **User should** always have the option to
  - **User should** have the maximum flexibility on customising
  - **User should** have the maximum flexibility on viewing the c
  - The **user should** be able to control any factors, such as col

What should a user be able to do?

We need a better way

- **Goal:** specify how different parties (users, administrators, external systems, …) can interact with our system
  - Recall: unambiguous, realistic, verifiable and evolvable
- **Plain-text** does not work
  - A more **structured way**

  is needed



Looking For:
users must in the current document
Results:
1 document(s) with 9 instance(s)
Results:
D 2.2a – User needs and System Requirements Specification
Users must have the option to access related links of vario
Users must have the option to access related links of vario
Users must have the option to access related links of vario
Users must have the option to access related links of vario
Users must have the option to access related links of vario
Users must have the option to access related links of vario
Users must have the option to access related links of vario
- **Users must** be asked which applications they use and w
- **Users must** be asked to identify which user group requi

Looking For:
user should in the current d
Results:
1 document(s) with 5 instan
D 2.2a – User needs anc
the **user should** be
User should always
User should have tl
User should have the maximum flexibility on viewing the
The **user should** be able to control any factors, such as col

Users should have the option to operate the stand alone t
Users should have the option to access several different vi

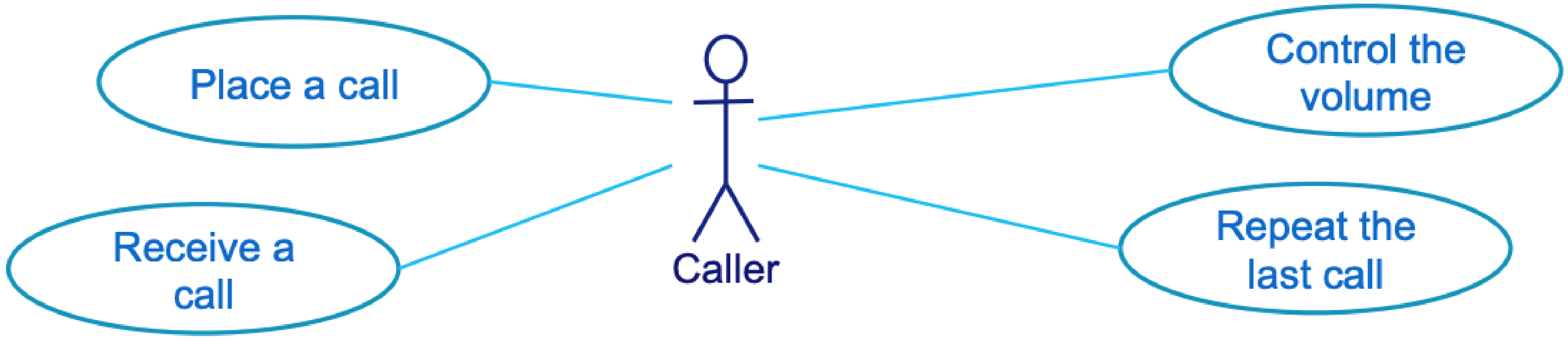
Simple example

- What can you do with a telephone?

http://globestaronline.com/panasonic-kx-ts500mx-corded-telephone/#.UrF1wPTuLy0

Simple example

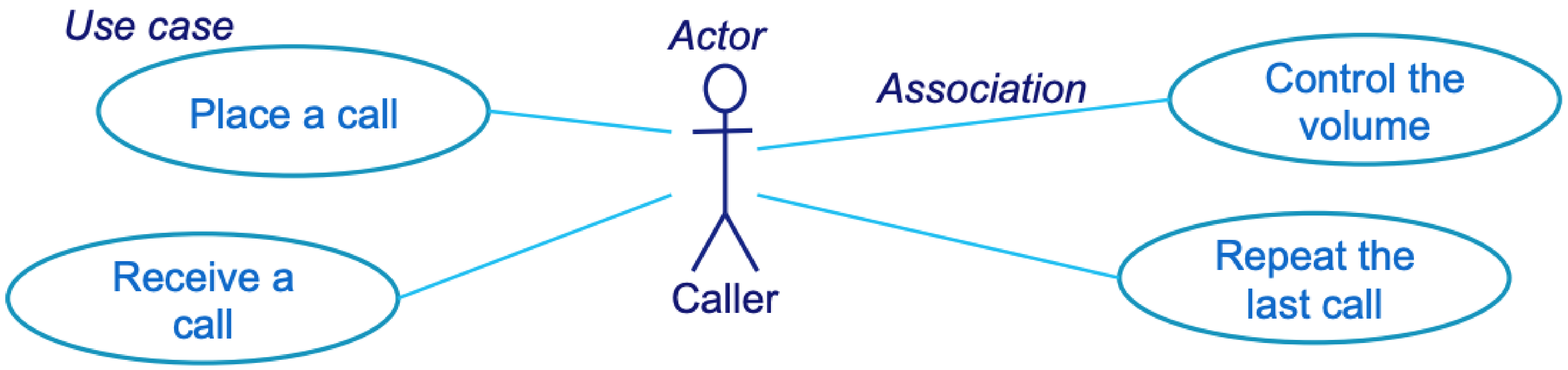
- What can you do with a telephone?



Basic use case diagram

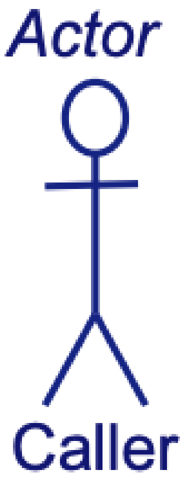
Simple example

- What can you do with a telephone?



Basic use case diagram

What is an *Actor*?



- **A class of entities** (human or computer), falling **beyond the system boundaries, interacting with the system**
  - Actor can be an **external software system**
  - One might have **multiple actors in a use case**: caller/callee
  - One user might be represented by **multiple actors**
    - doctor can be a patient

What is a *Use Case*?

Place a call

- **Use case:** a contract between the stakeholders of a system about its behavior
  - Stakeholder: a person, group or organization with an interest in a project
- Usually, a large **end-to-end** process comprising several (inter)actions
  - Internal operations are NOT use cases
  - Individual methods are NOT use cases
  - BUT it is one process: specification usually involves multiple use cases

Is placing a call large?

Place a call

- How do you make a phone call?

  a) The user picks up the telephone hook connected to the telephone line “A”.

  b) If the line is free, the user receives a dial tone sent by the line .

  c) The user dials number “B”.

  d) The call request is forwarded to the switch center.

  e) If line “B” is not busy, the call request is forwarded to “B” and a tone is sent to “A”.

  f) “B”’s telephone rings.

  g) If somebody at “B” picks up the hook, the ringing tone at “A” is stopped and a telephone connection will commence.

Main Scenario

What if the “if” conditions do not hold?

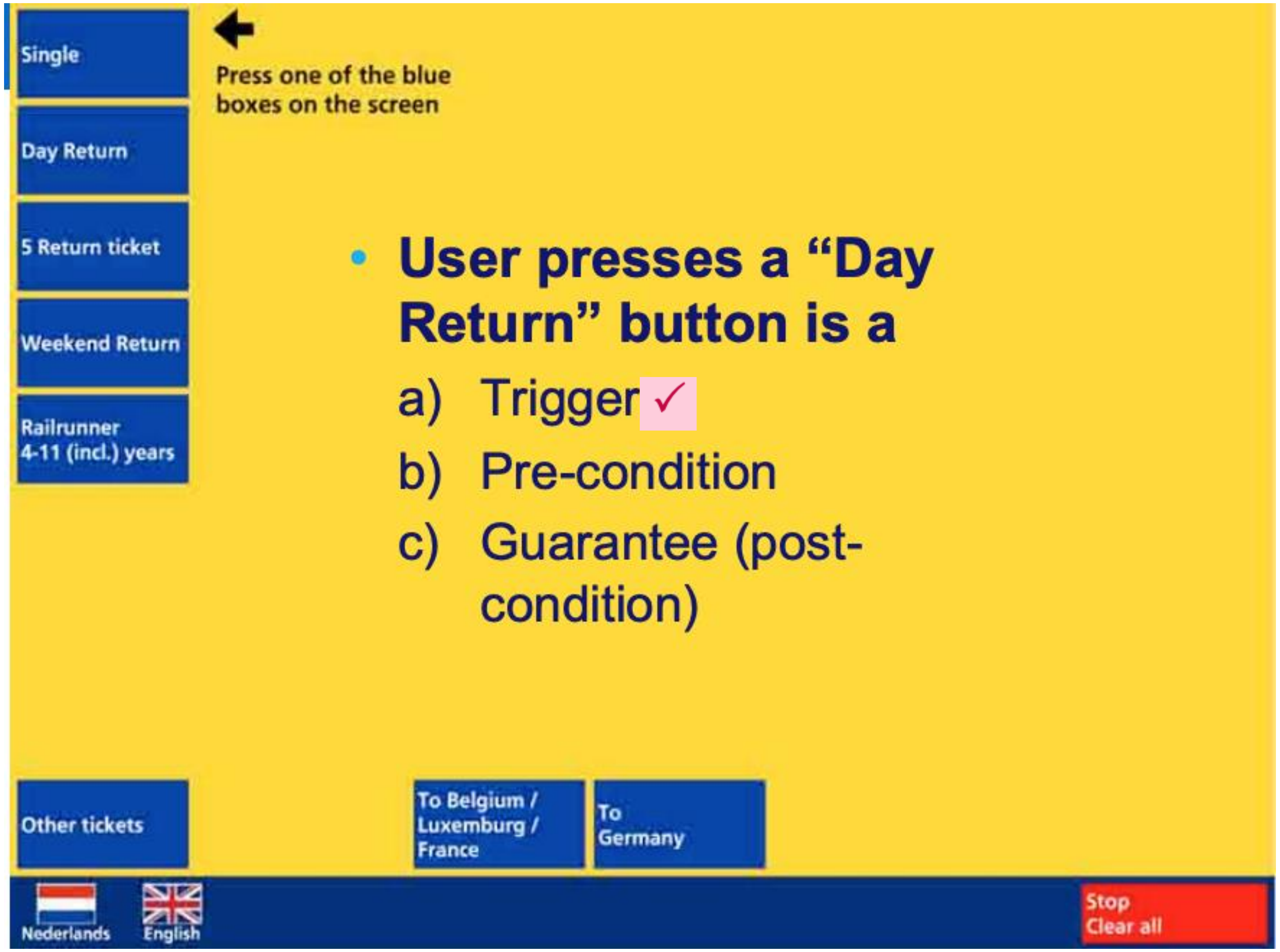
- If the line is free, the user receives a dial tone sent by the line.
  - b-1) If the telephone line is engaged in a conversation, the user will be connected to the same conversation.
  - b-2) If the user does not dial a number for a certain amount of time, a permanent tone is emitted by the switch center, no further call will be accepted and the user has to replace the hook.

Alternatives

Use case description

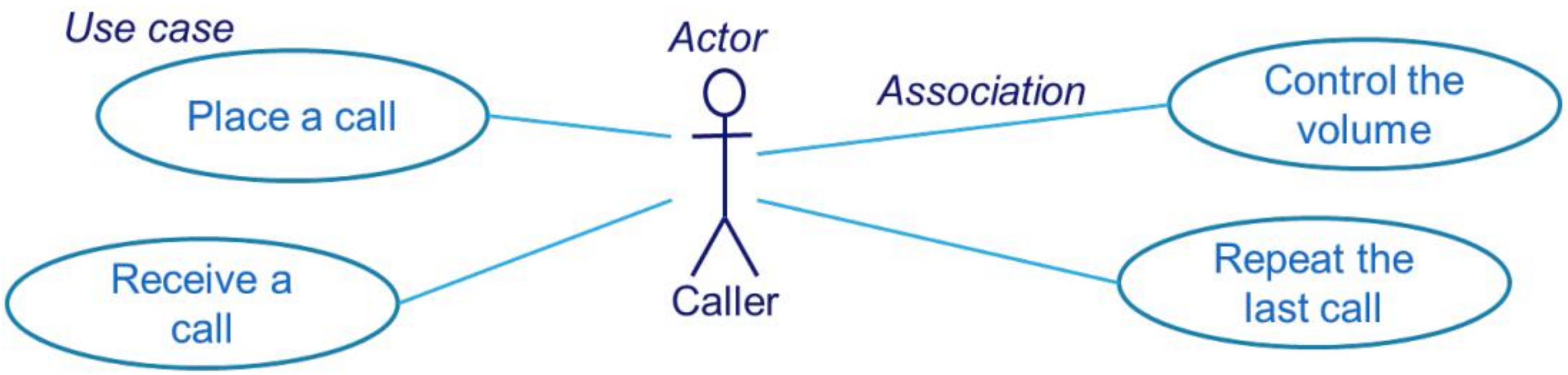
- **Pre-condition:** when a use case is available to its user
  - The telephone set is connected to the telephone line “A”, it is on-hook and there is no incoming call (it is not ringing).
- **Trigger**: action that initiates the use case
  - The user picks up the telephone hook connected of the telephone set (connected to line “A”) and dials number “B”.
- **Guarantee** (**post-condition**): what does the user achieve through the use case
  - A communication between “A” and “B” will commence
- **Main scenario**
- **Alternatives**

Exercise


Single
Day Return
5 Return ticket
Weekend Return
Railrunner 4-11 (incl.) years
Press one of the blue boxes on the screen
Other tickets
To Belgium / Luxemburg / France
To Germany
Nederlands
English
Stop Clear all

- **User presses a "Day Return" button is a**
  a) Trigger ✓
  b) Pre-condition
  c) Guarantee (post-condition)

Summary so far



What is an *actor*?

- A **class of entities** (human or computer), falling **beyond the system boundaries**, **interacting** with the system

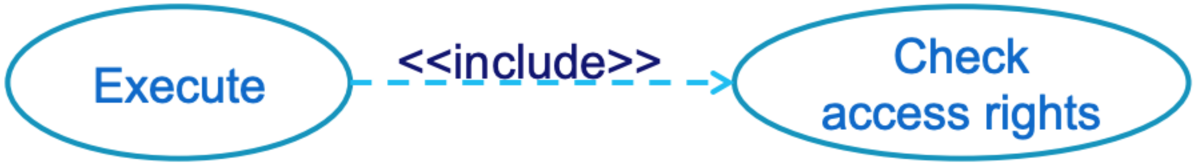
- NB:
  - Actor can be an **external software system**
  - One might have **multiple actors in a use case:** caller/callee
  - **One user** might be represented by **multiple actors**
    - doctor can be a patient

Use case description

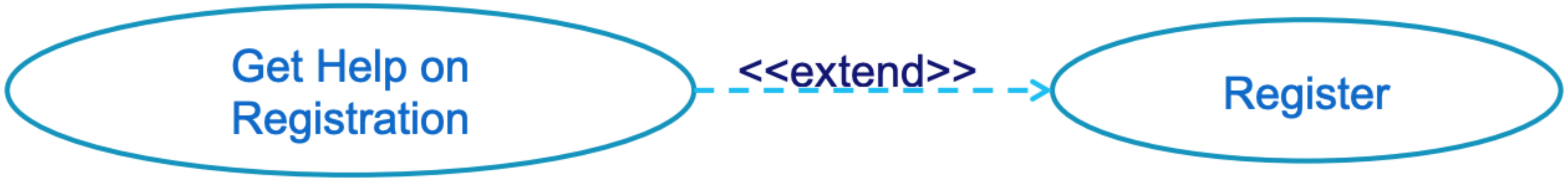
- **Pre-condition:** when a use case is available to its user
  - The telephone set is connected to the telephone line "A", it is on-hook and there is no incoming call (it is not ringing).
- **Trigger:** action that initiates the use case
  - The user picks up the telephone hook connected of the telephone set (connected to line "A") and dials number "B".
- **Guarantee (post-condition)**: what does the user achieve through the use case
  - A communication between "A" and "B" will commence
- **Main scenario**
- **Alternatives**

Formally

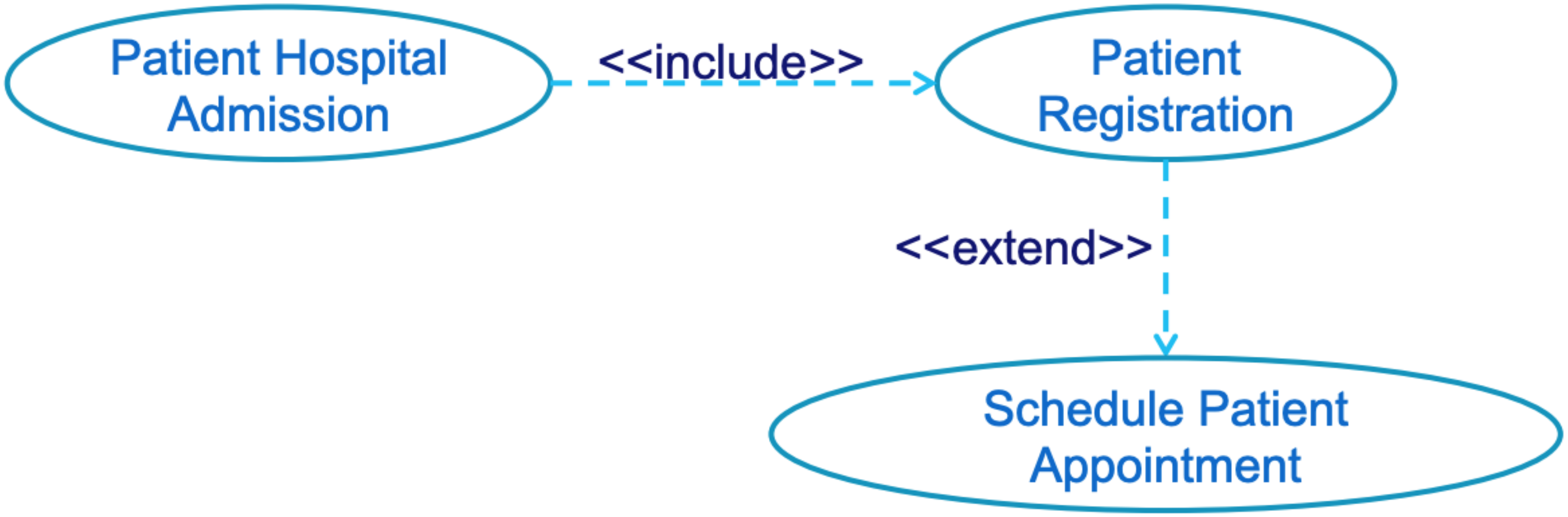
- Dependency: A use case depends on another use case for realizing its goal.
  - The target use case is always used by the source.



- What if the use case is used only sometimes?
  - Alternatives in the use case
  - Extension



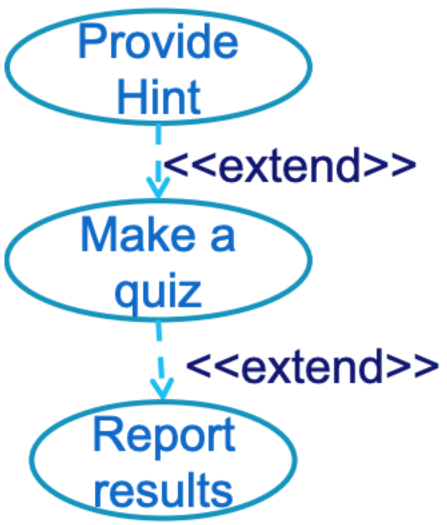
Example



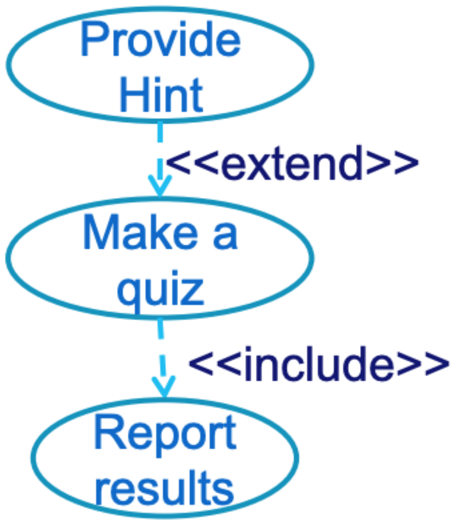
	<<include>> $A \dashrightarrow B$	<<extend>> $A \dashleftarrow B$
Read	A includes B	B extends A
A can operate without B	no	yes
B is aware of A	no	yes

Which model is correct?

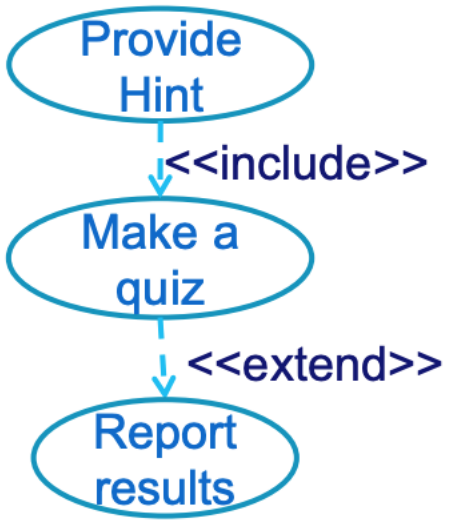
- Digital learning environment supports students’ learning through quizzes. Given your knowledge of this kind of systems which of the following models would be correct?



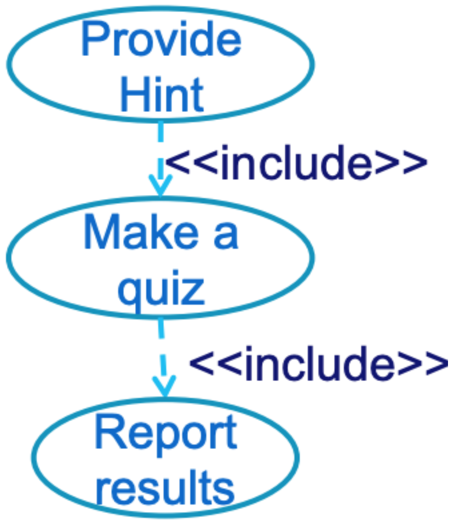
A



B ✓



C

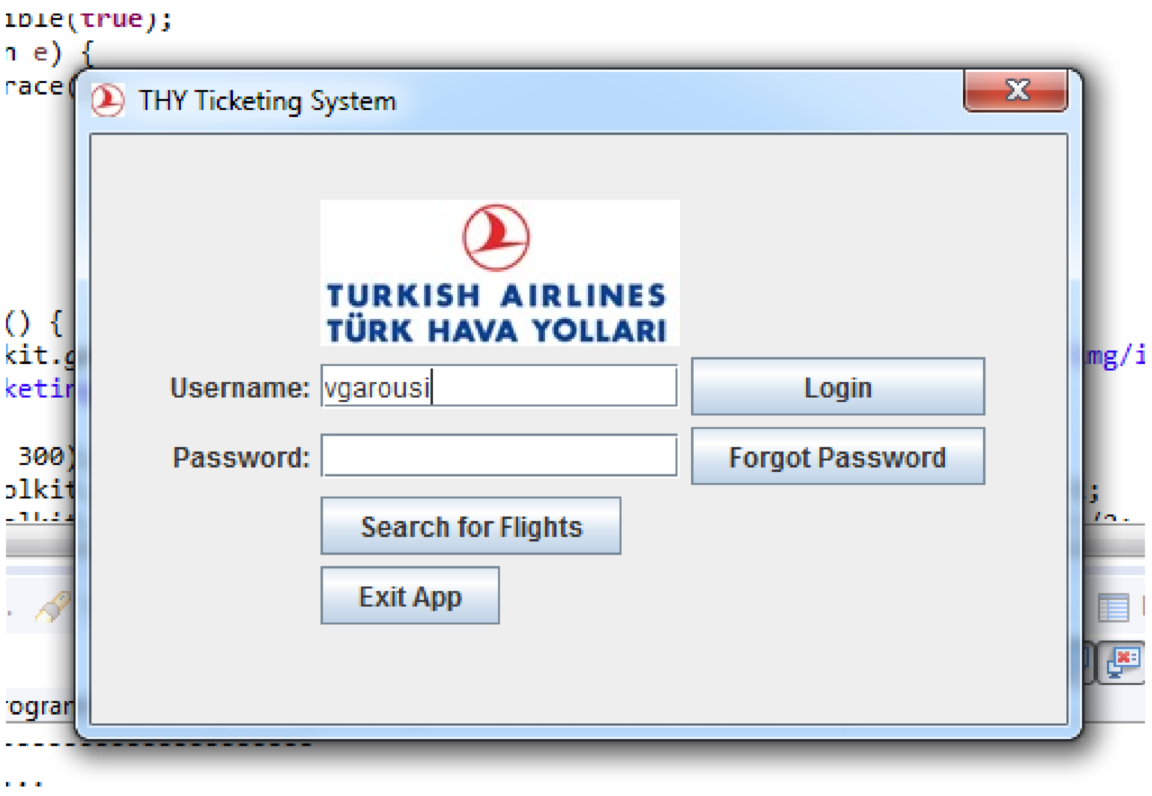


D

Flight Booking System (FBS)

(see "FBS User Requirements and Use-case Modeling.pdf")

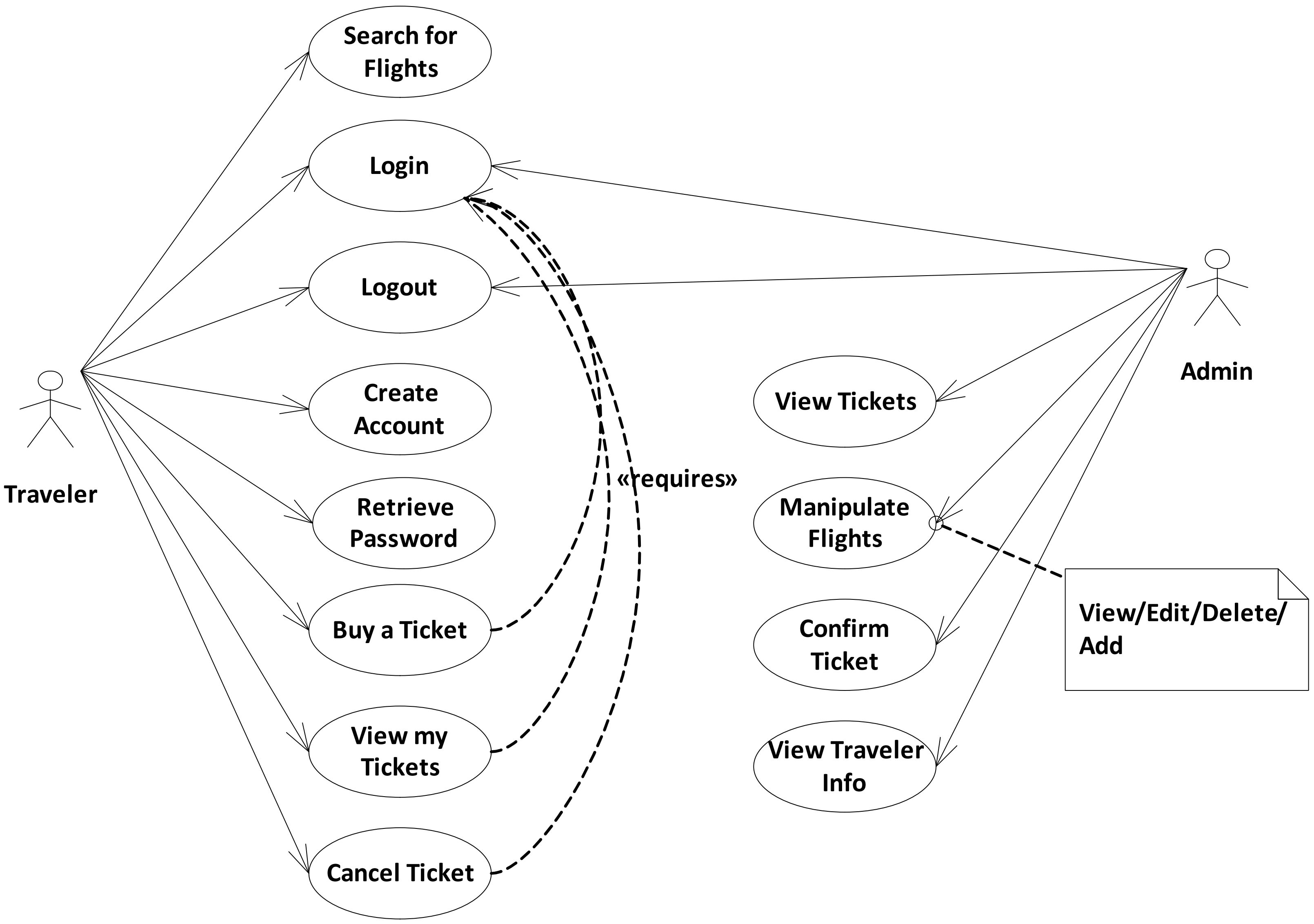
- A Flight Booking System (FBS) that serves different functionalities for reservation agents and reservation managers shall be designed and developed.



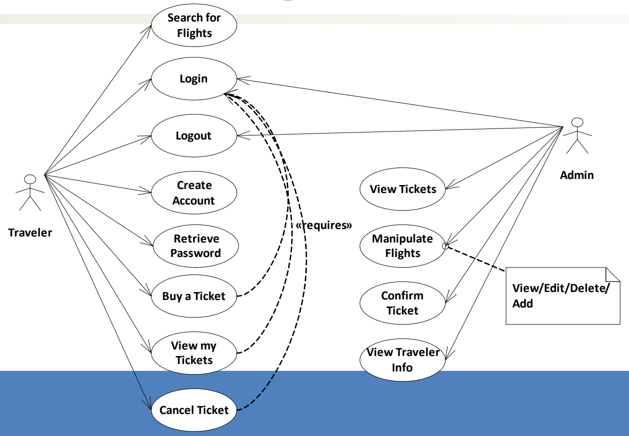
Username: vgarousi

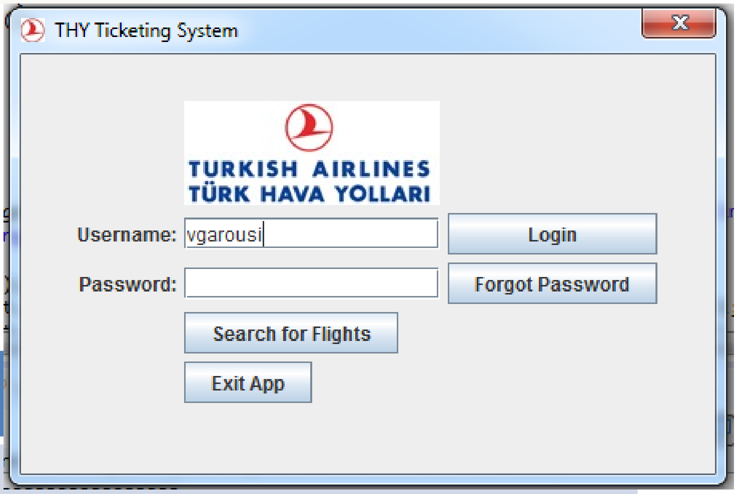
Password:

FBS – Use-Case Diagram



Tabular description of use-cases





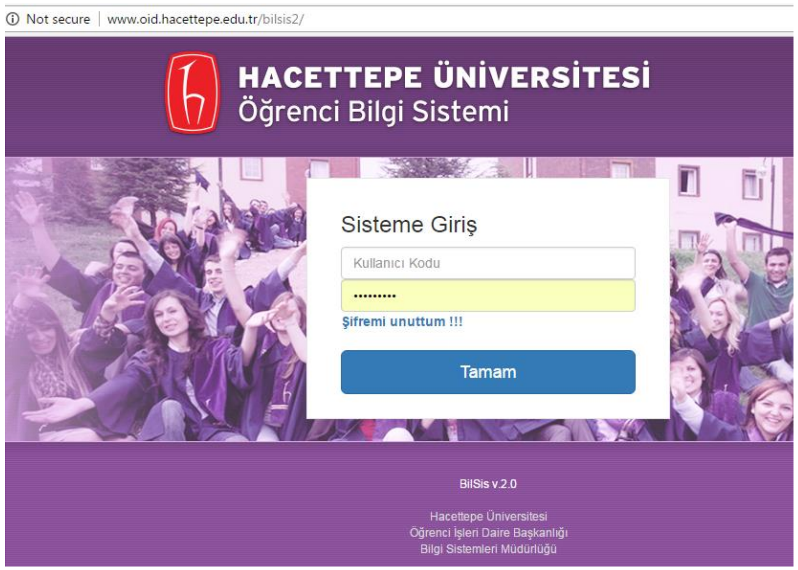
Username: vgarousi

Password:

Use case: Login	
Actors	Traveler
Precondition	Traveler is not logged in
Post-condition	Traveler is logged in
Main (happy) path:	1. User enters her/his username and password and presses on the Login button 2. System checks the username and password and shows the app's main window (page) 3. User will be able to use the feature afterwards
Alternative path:	1. If the username and password combination is invalid, the system will show a message indicating it 2. User will be able to enter another username and password

Exercise

- System: student information system (like BILBSIS2)
- Design:
  - The context diagram
  - The use-case diagram
  - Three tabular descriptions for three use-cases
  - Three activity diagrams for three <u>other </u>use-cases



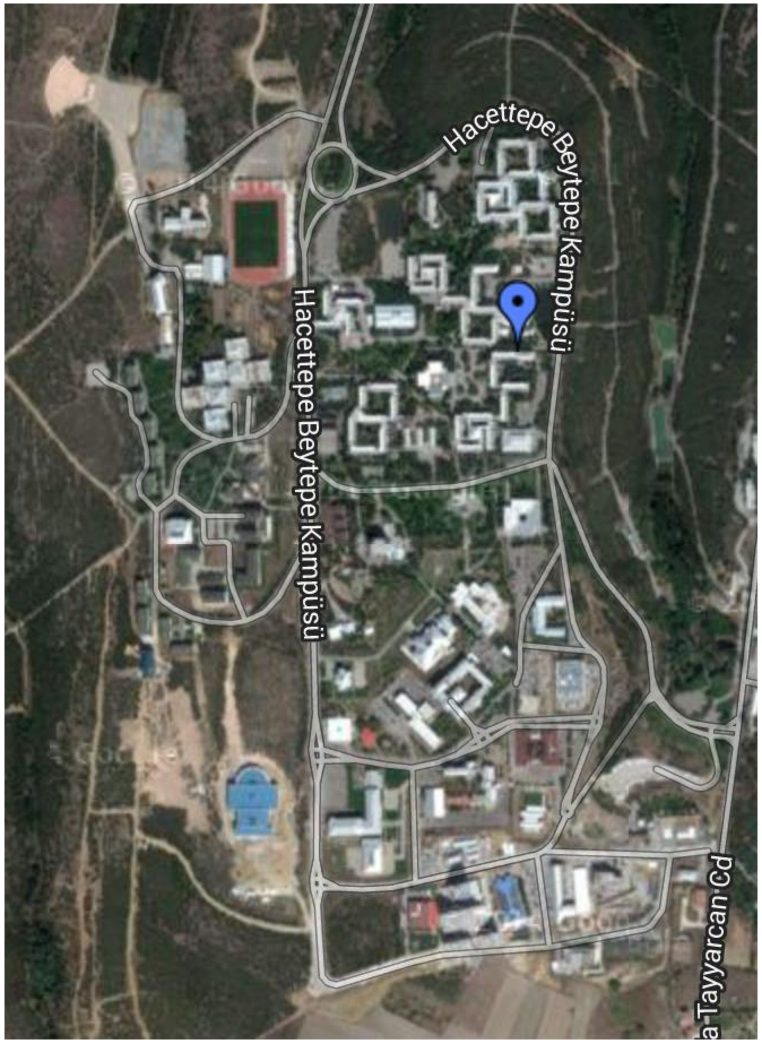
For student reading

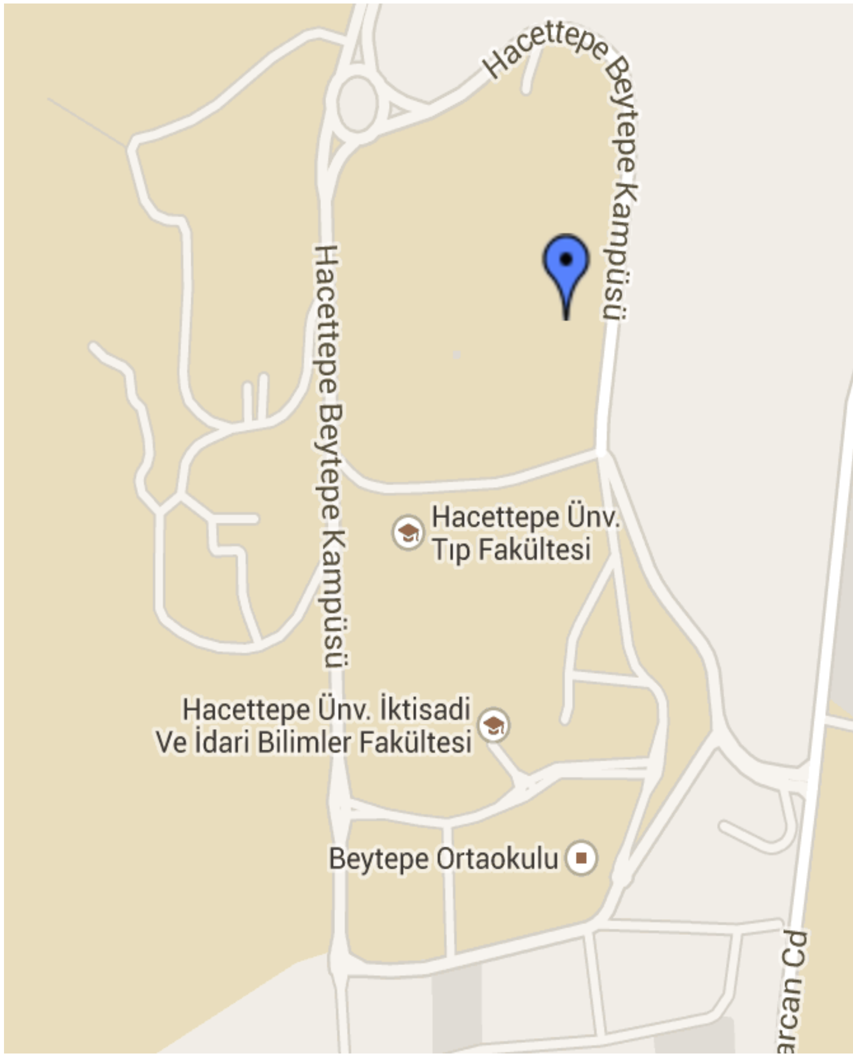
System modeling

- System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system.
  - System modeling has now come to mean representing a system using some kind of graphical notation, which is now almost always based on notations in the Unified Modeling Language (UML).
- System modelling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

Real vs. Model – Beytepe Kampüsü

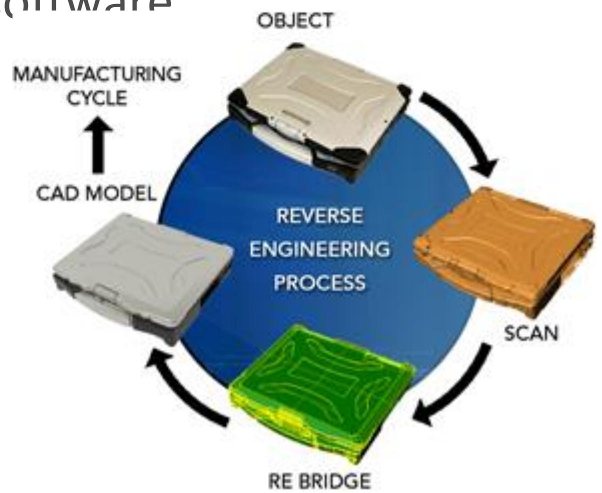
- Real – Satellite View
- Model – Vector Map




Hacettepe Beytepe Kampüsü
Hacettepe Beytepe Kampüsü
Hacettepe Ünv. Tıp Fakültesi
Hacettepe Ünv. İktisadi Ve İdari Bilimler Fakültesi
Beytepe Ortaokulu
arcan Cd

Existing and planned system models

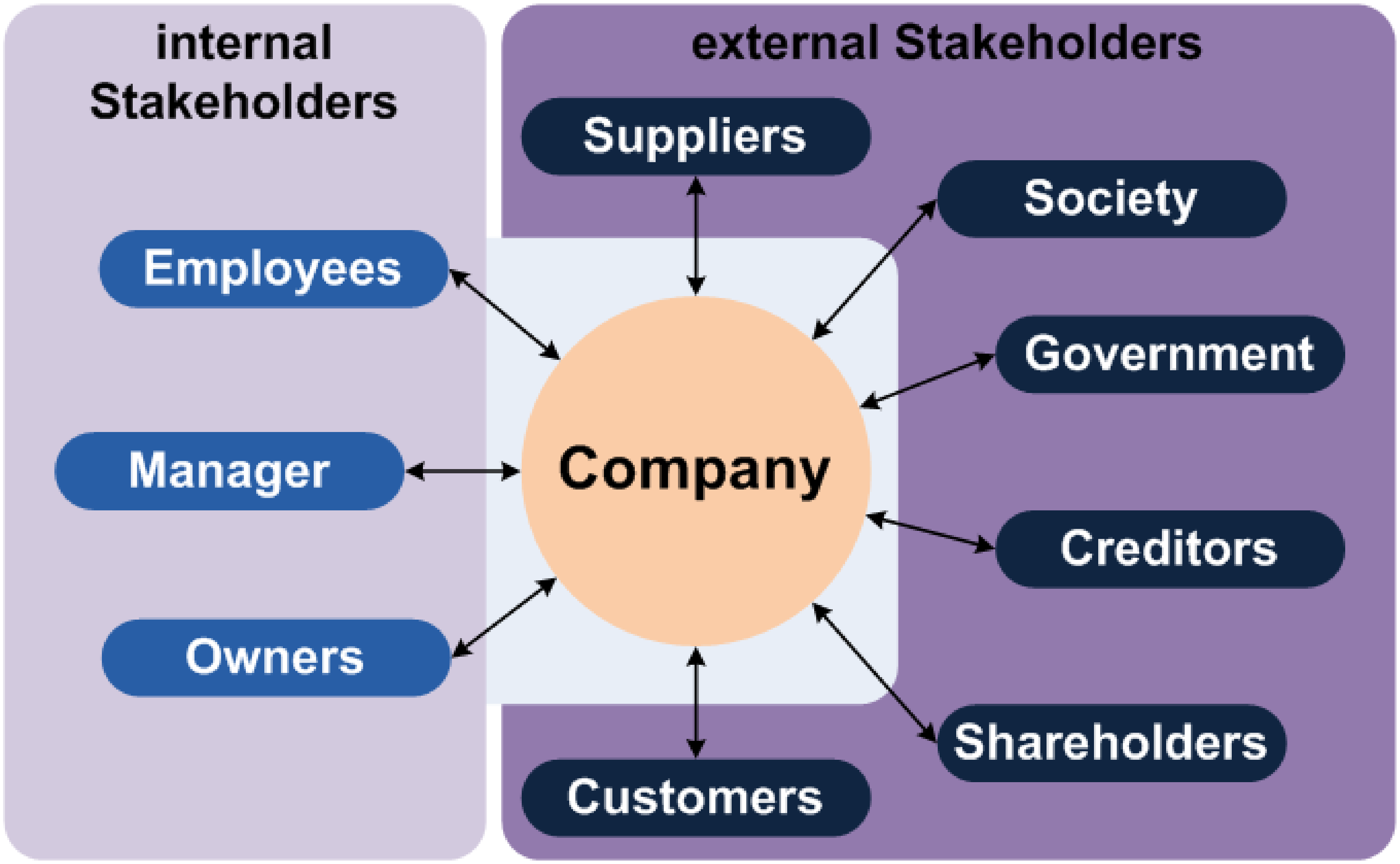
- Models of the existing system are used during requirements engineering, re-engineering and reverse-engineering .
  - They help analyst clarify what existing system does and can be used as a basis for discussing its strengths and weaknesses. These then lead to requirements for the new system.
  - Re-engineering: Systematic starting over and reinventing the way a firm, a software, or a business process, work (code refactoring in our context)
  - Reverse engineering: taking apart an object to see how it works in order to duplicate or enhance the object. The practice, taken from older industries, is now frequently used on computer hardware and software
  - Re-engineering ≠ Reverse engineering



Existing and planned system models

- Models of the new system are used during requirements engineering to help explain proposed requirements to other system stakeholders.
  - Engineers use these models to discuss design proposals and to document system for implementation.
  - In a model-driven engineering process, it is possible to generate a complete or partial system implementation from system model.

Stakeholders of a Company

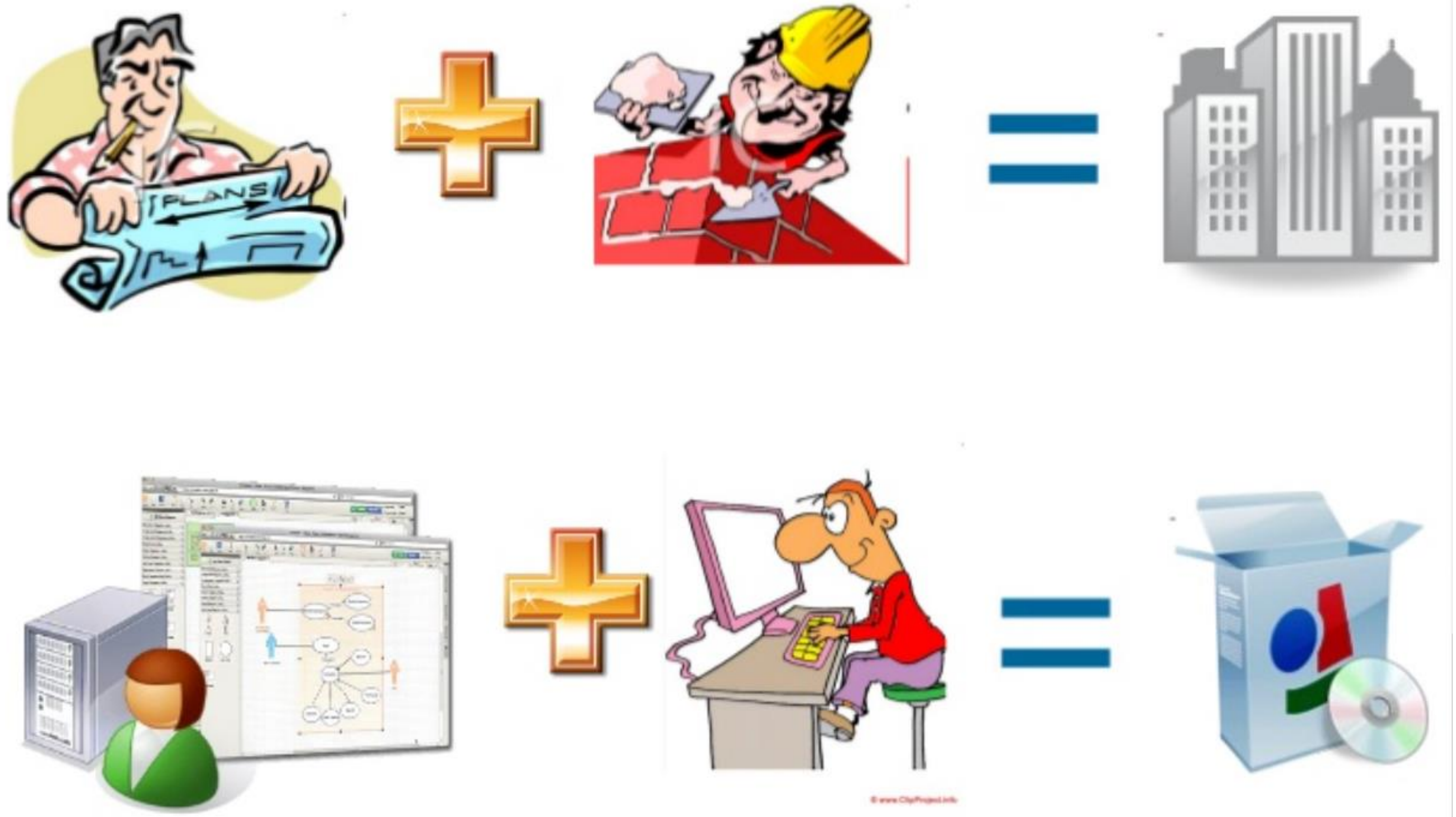


http://upload.wikimedia.org/wikipedia/commons/4/4c/Stakeholder_%28en%29.png

System perspectives

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

System modeling metaphore



http://www.slideshare.net/jcabot/mde-20-pragmatic-model-verification-and-other-stories-habilitation-public-lecture

FBS – User Requirements (READ LATER)

- The system shall be a simple, stand-alone application.
  - “Stand-alone” means that it shall not be developed as a web-based on-line application, but a one which runs as an application on a single computer.
- A reservation manager shall be able to use the system to create accounts for reservation agents. The system shall enable a reservation manager to log in with his/her username and password. The reservation manager shall be able to manipulate flights (enter new ones, update existing flight information, delete flights, etc.), and generate inventory reports of flights. When a flight is canceled by the reservation manager, all itineraries that are reserved or booked and include that flight shall be canceled as well. The reservation manager shall be able to view information of a reservation agent and confirm a ticket for him/her. The reservation manager shall be able to log out at any stage during his/her session.

FBS – User Requirements (..cont’d)

- A reservation agent (traveler) shall be able to buy a ticket by using the system. An agent shall be able to register himself/herself with the reservation site by entering personal profile information. The system shall enable a reservation agent to log in with his/her username and password. Once the agent is logged in, he/she shall be shown with a list of travel itineraries along with the status of each itinerary (reserved, booked, confirmed, or canceled). A travel itinerary is a travel arrangement with one or more flights, e.g., an example travel itinerary might consist of the following three flights: Calgary-Vancouver, Vancouver-Los Angeles and Los Angeles-New York. The reservation agent shall create a reservation and pay for it to book the flight. The agent shall cancel a previously created reservation in his/her itinerary list, but the cancellation of an itinerary cannot be rolled back.

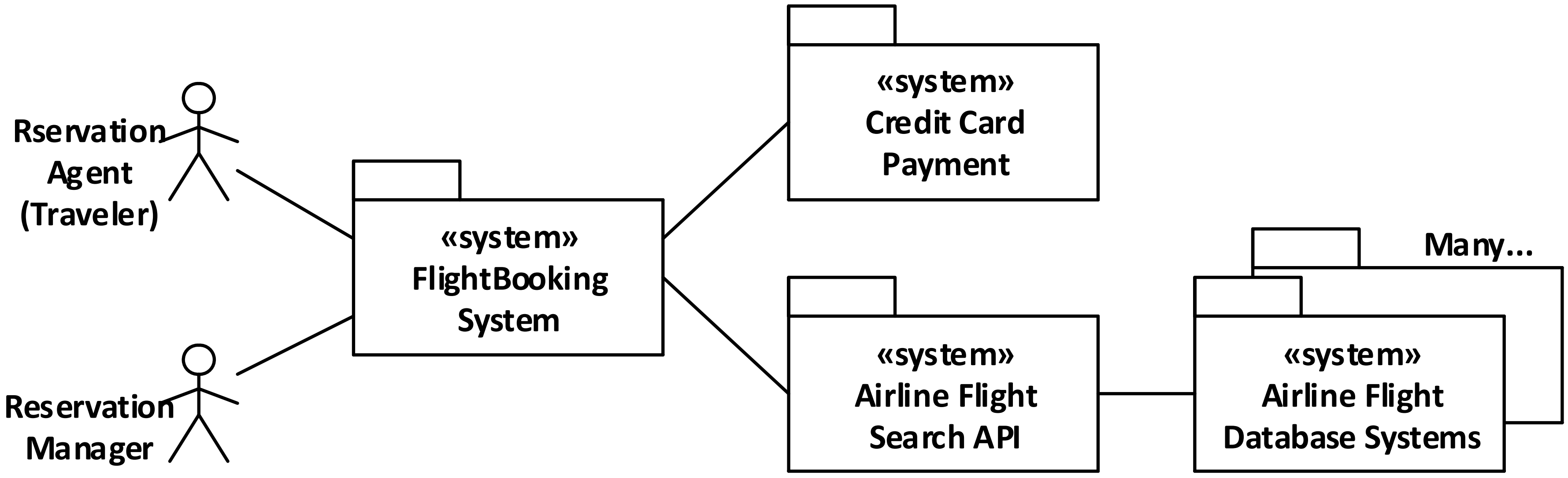
FBS – User Requirements (..cont’d)

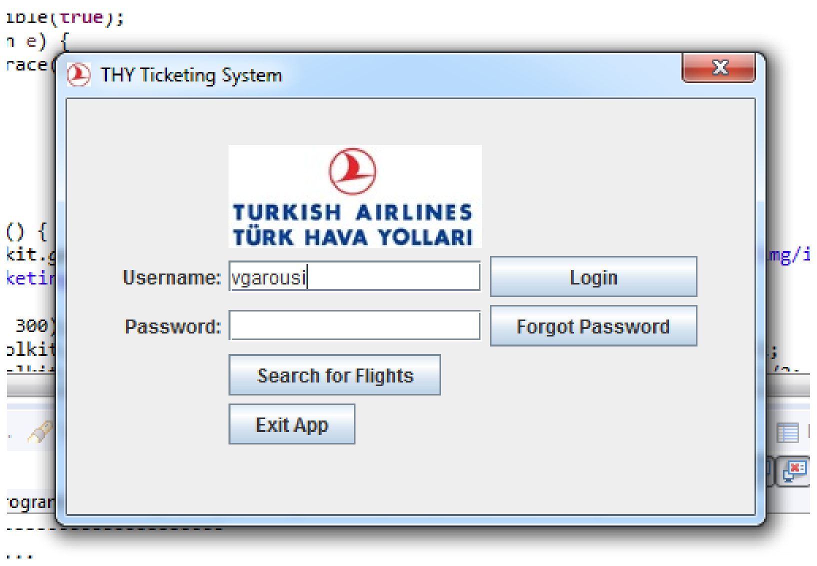
- When a reservation agent wants to create and book a travel itinerary, he/she first shall search for flight information. A list of available flight options shall be shown with departure/arrival and cost information. Once the agent selected an itinerary from the list, he/she shall have the option to reserve it. Once the reservation agent reserved an itinerary, he/she shall have the option to book it by providing payment information via credit card. If the credit card information is not on-file for the agent, he/she shall be prompted to enter the credit card information.

FBS – User Requirements (..cont'd)

- Once the credit card is validated, the reservation agent shall be shown with the actual ticket information. The ticket number shall be unique for each agent in each flight. If the credit card number is invalid, an error message shall be displayed to the agent and the agent shall be asked to re-enter the credit card information or cancel the booking process. If a reserved itinerary is not paid for by 5 minutes from the time of reservation, it cannot be booked anymore and it shall be canceled by the system. Cancellation after payment shall not be permitted. The reservation agent shall be able to log out at any stage during his/her session. If the agent is logged out during the preparation of an itinerary, the next time when he/she is logged in again, the reservation agent shall be able to continue with the previous itinerary-preparation process.

FBS – Context Diagram

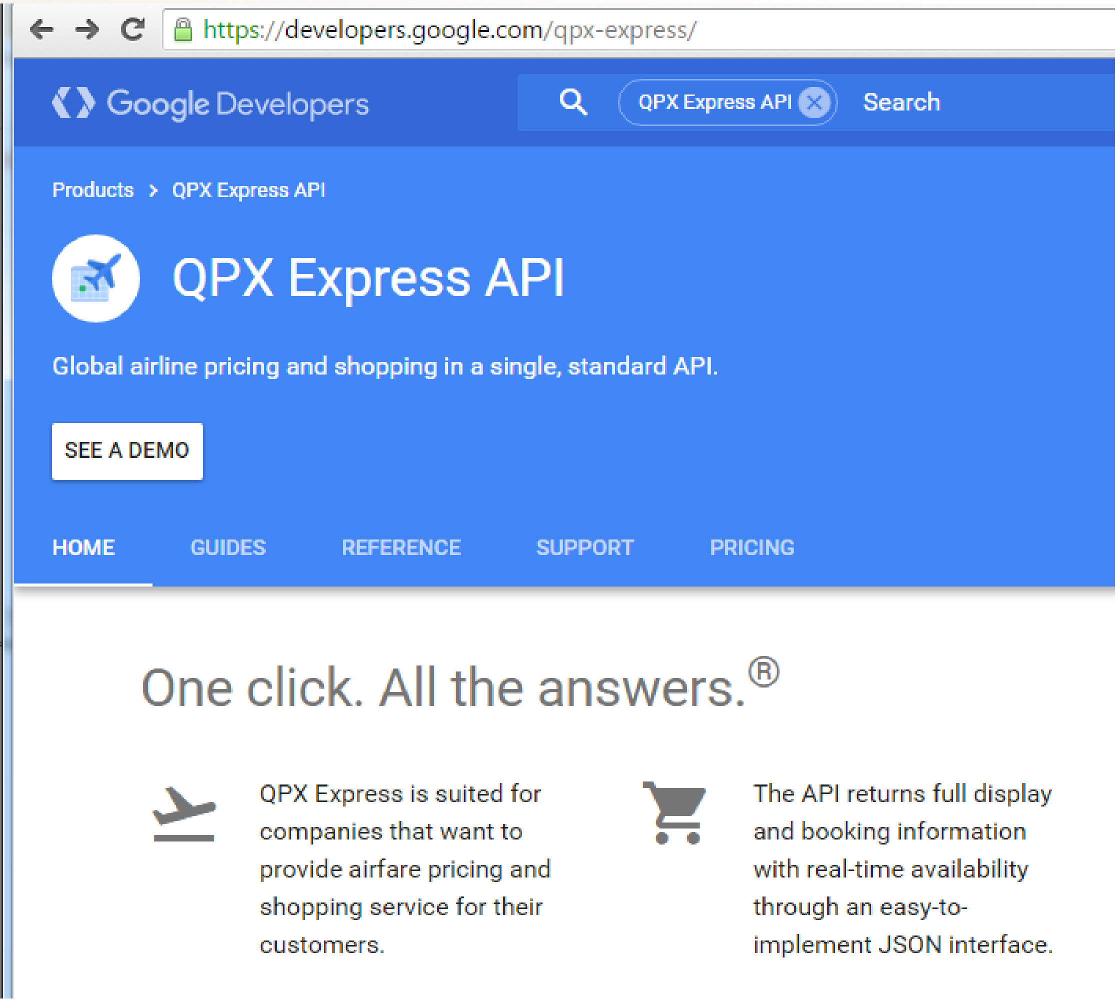

Rservation Agent (Traveler)
Reservation Manager
«system» FlightBooking System
«system» Credit Card Payment
«system» Airline Flight Search API
«system» Airline Flight Database Systems
Many…



Username: vgarousi
Password:

Context diagram of FlightBooking System

- Example of APIs…



https://developers.google.com/qpx-express/

Products > QPX Express API

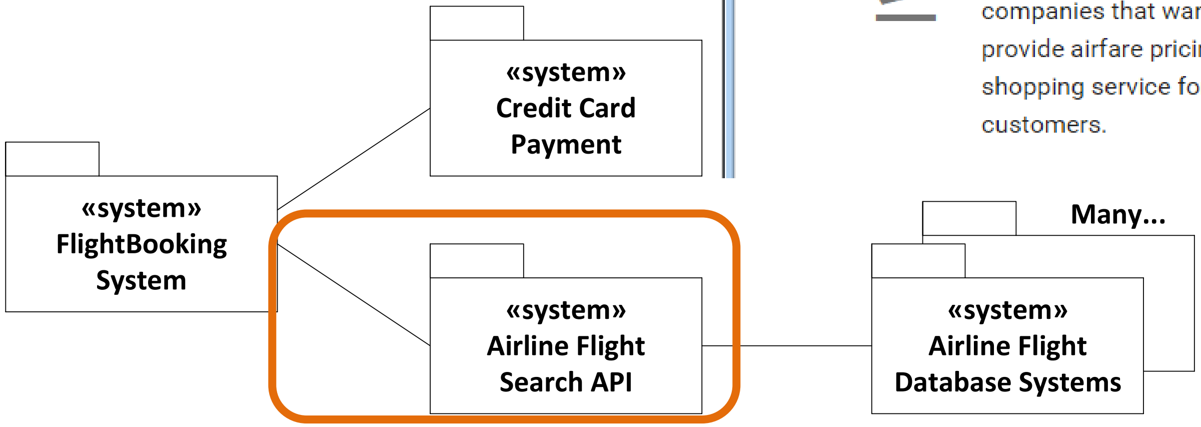
QPX Express API

Global airline pricing and shopping in a single, standard API.

One click. All the answers.®

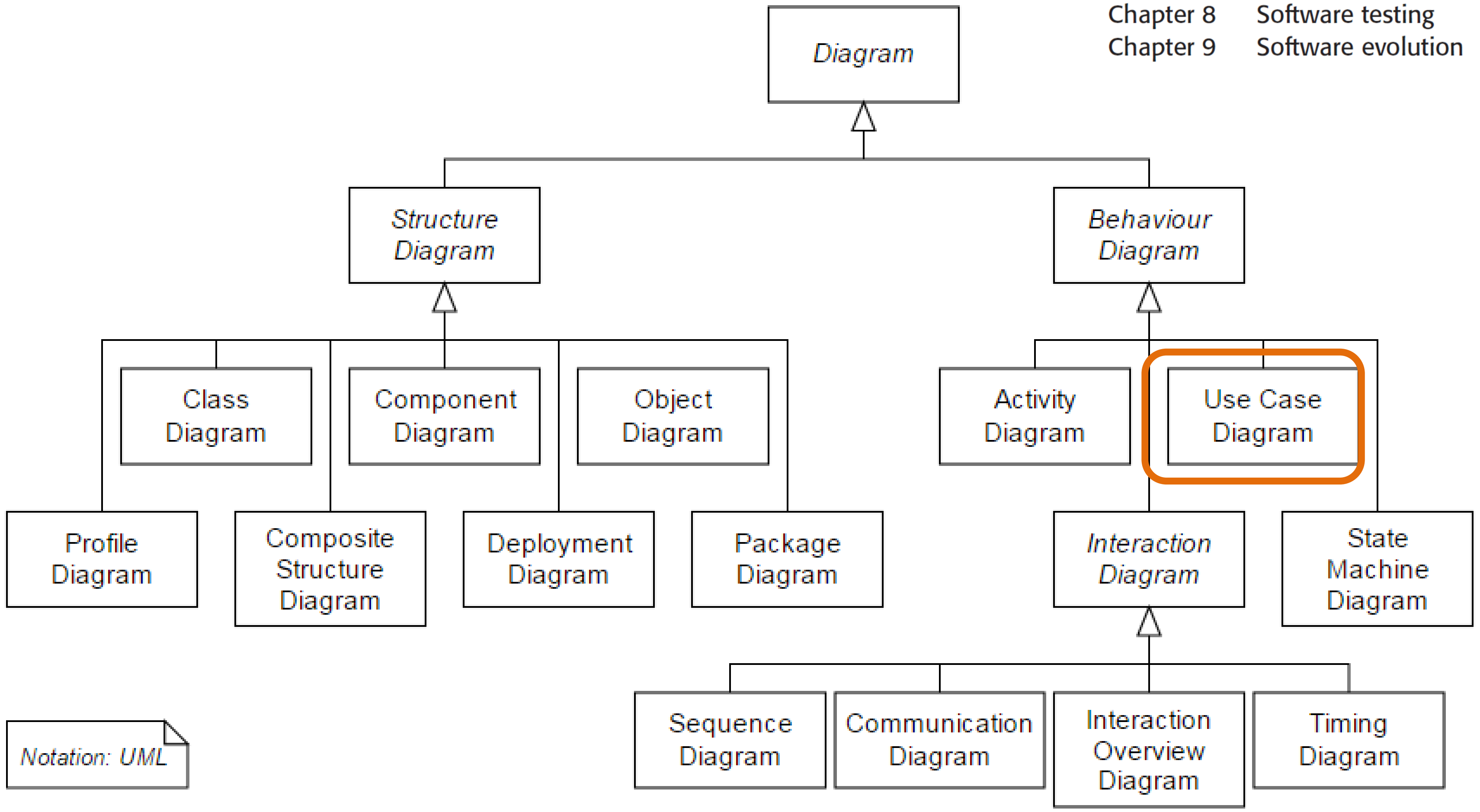
QPX Express is suited for companies that want to provide airfare pricing and shopping service for their customers.

The API returns full display and booking information with real-time availability through an easy-to-implement JSON interface.


«system» FlightBooking System
«system» Credit Card Payment
«system» Airline Flight Search API
«system» Airline Flight Database Systems
Many…

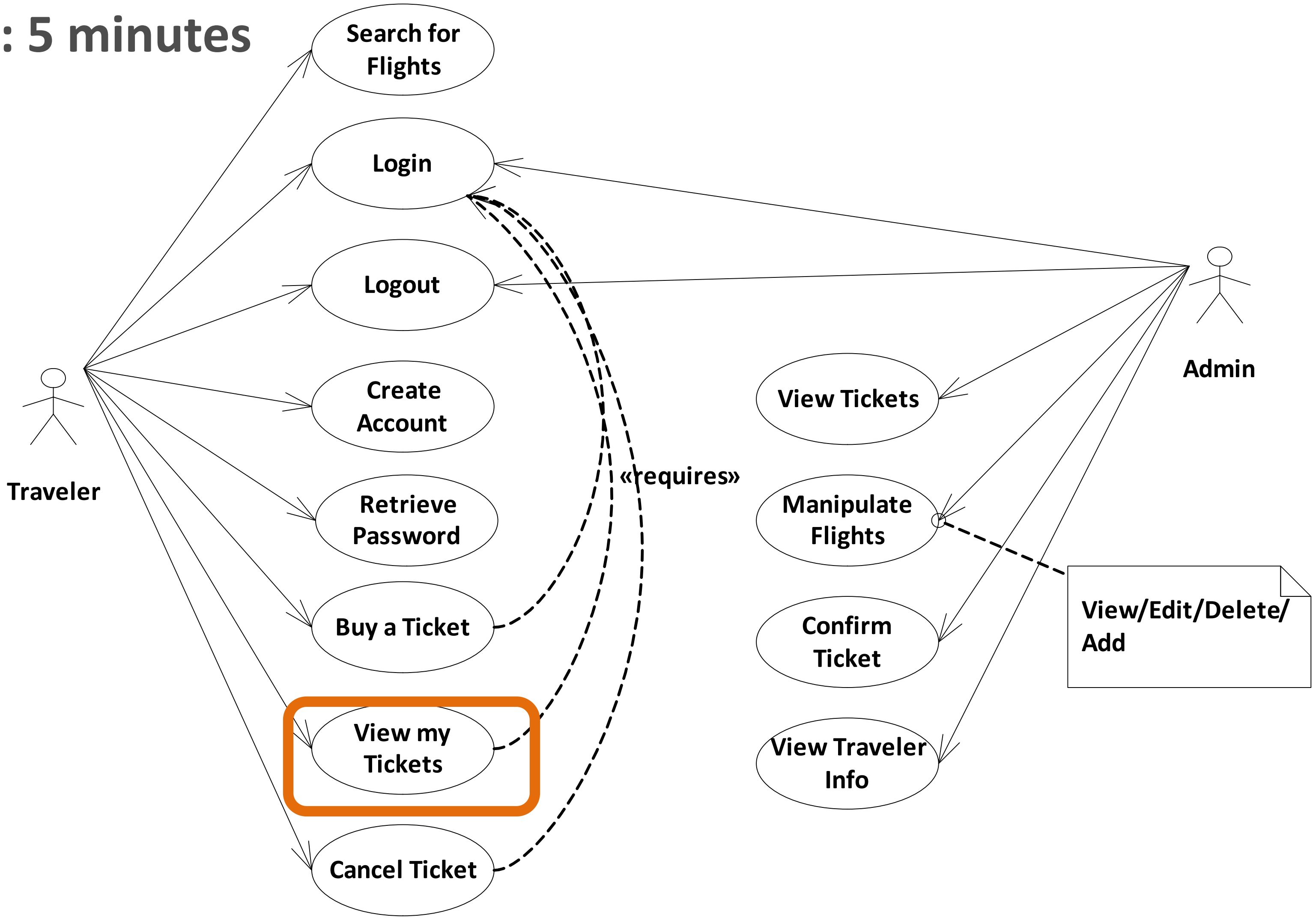
UML

- 14 diagrams in total

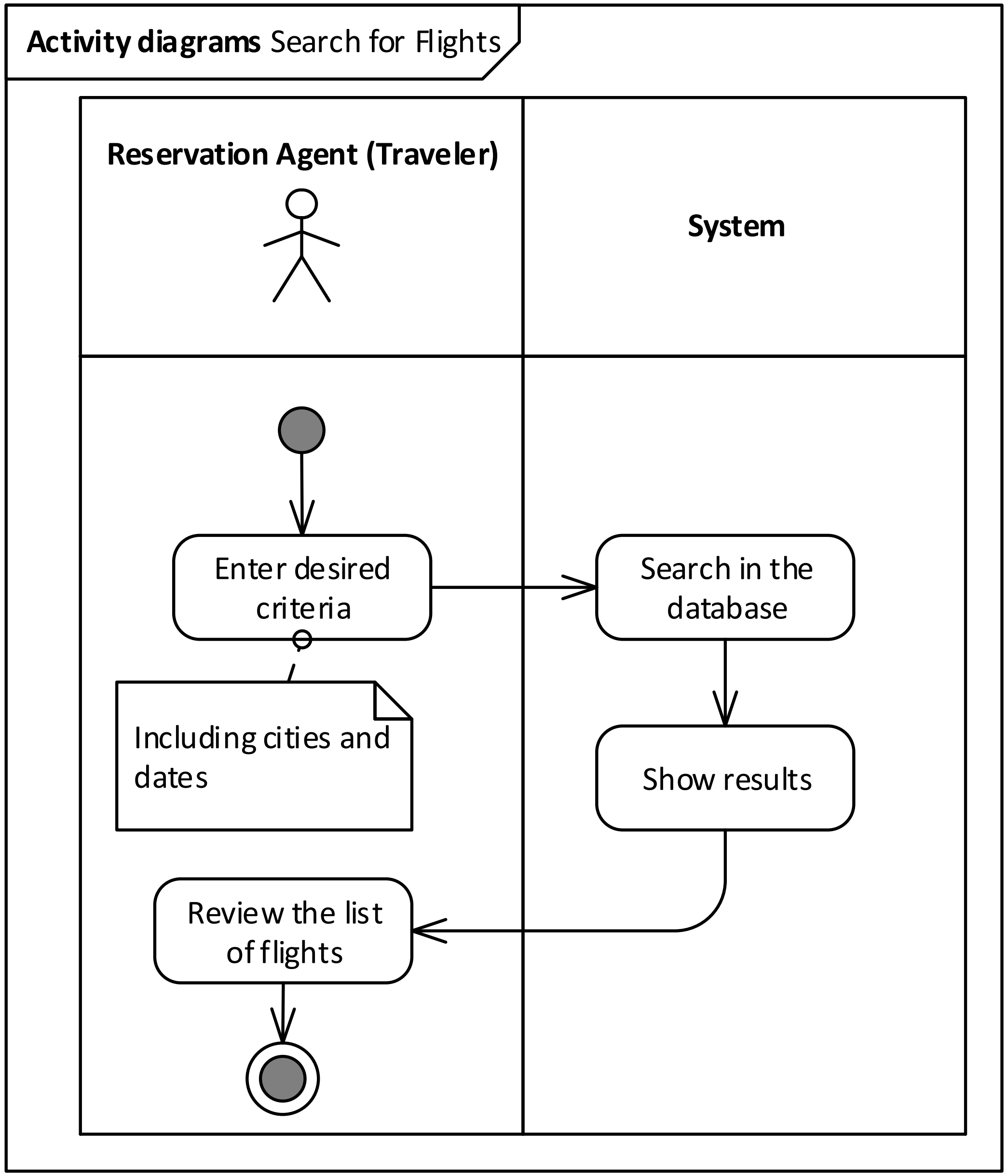


In-class exercise

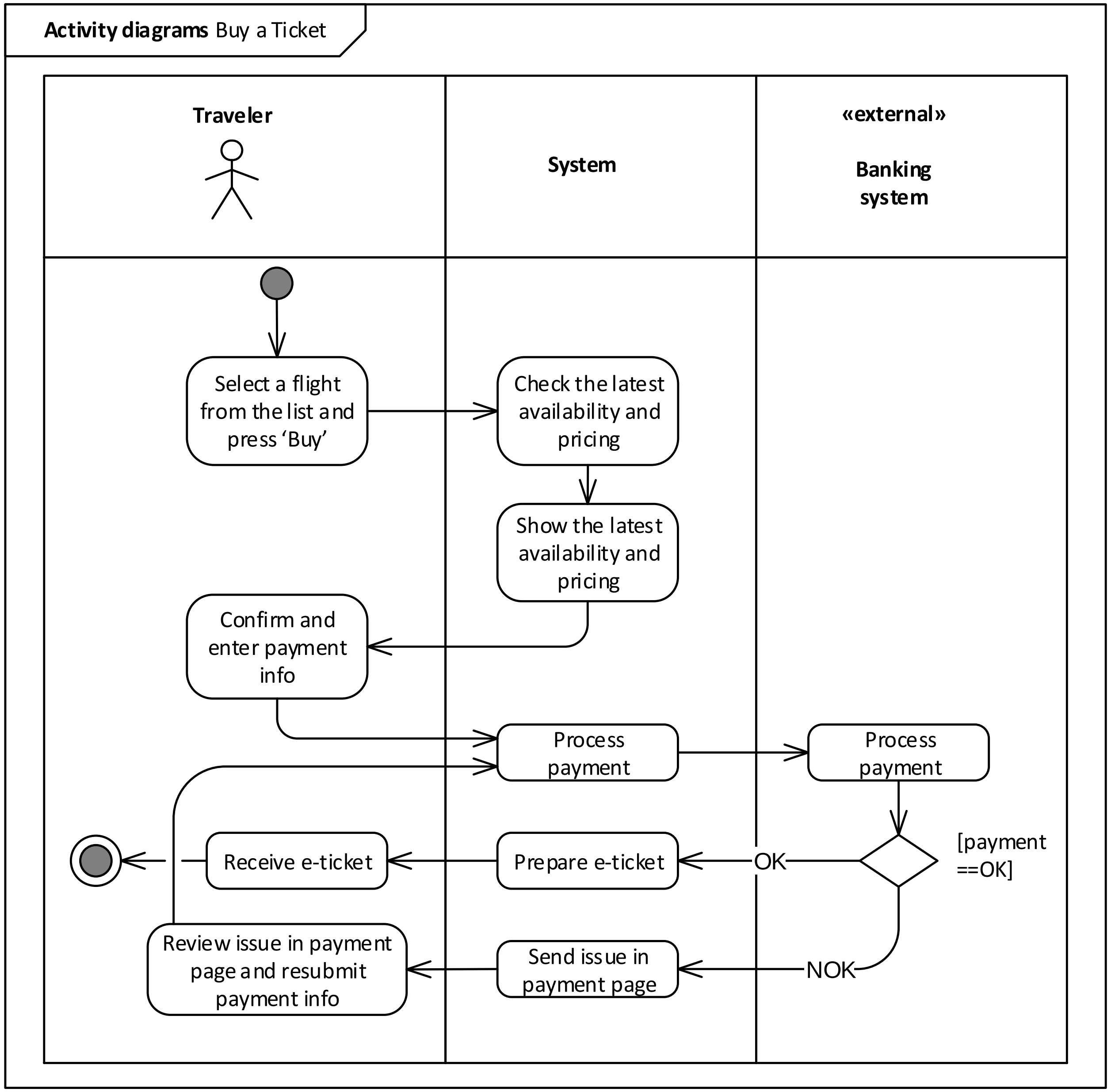
- Write the tabular description of use-case “View My Tickets”
- Time: 5 minutes



FBS – Details of "Search for Flights" use-case desc. by Activity Diagram


Activity diagrams Search for Flights
Reservation Agent (Traveler)
System
Enter desired criteria
Including cities and dates
Search in the database
Show results
Review the list of flights

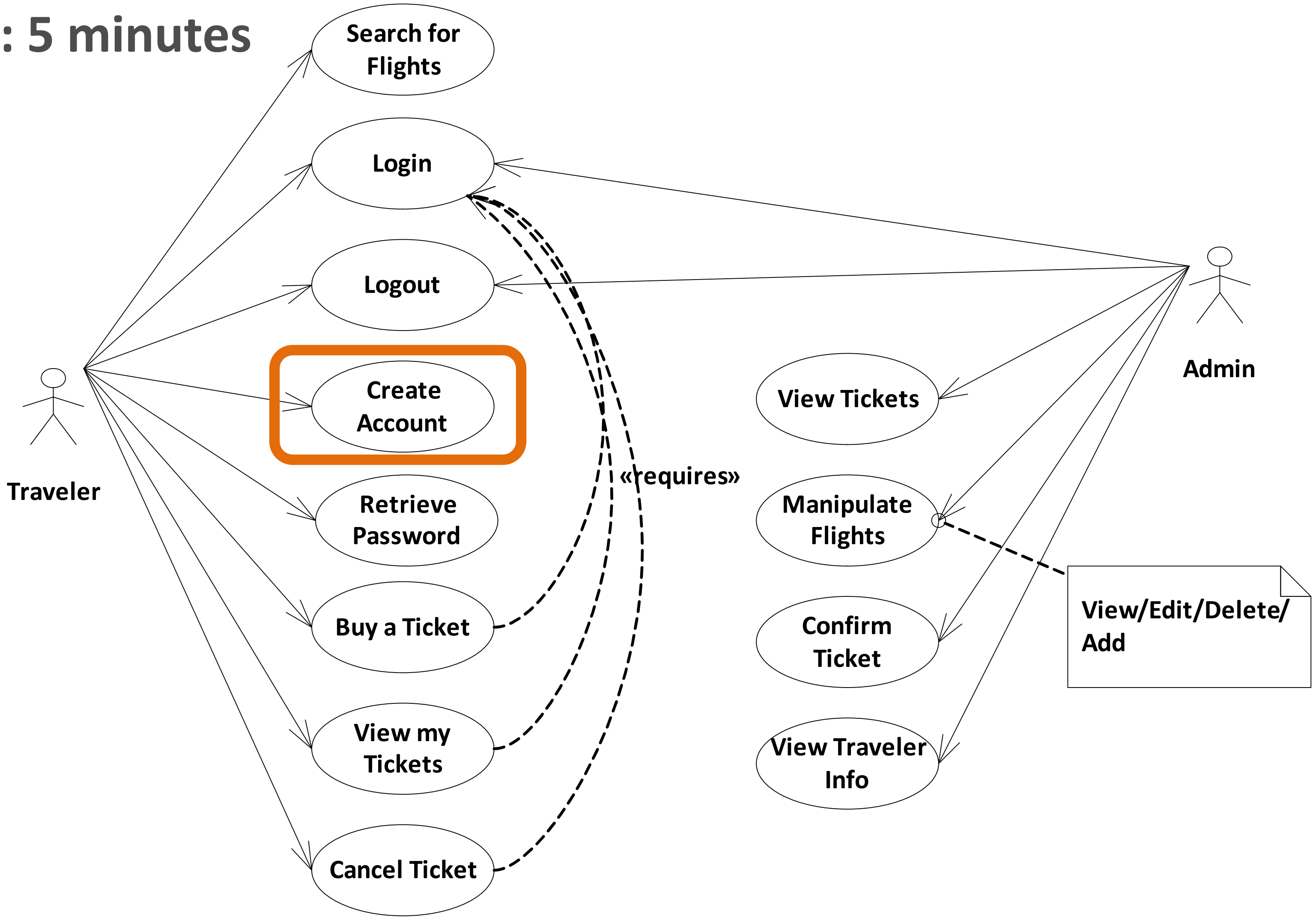
FBS – Details of "Buy a Ticket" use-case description by Activity Diagram



Activity diagrams Buy a Ticket
Traveler
System
«external»
Banking system
Select a flight from the list and press 'Buy'
Check the latest availability and pricing
Show the latest availability and pricing
Confirm and enter payment info
Process payment
Process payment
[payment ==OK]
OK
Prepare e-ticket
Receive e-ticket
NOK
Send issue in payment page
Review issue in payment page and resubmit payment info

In-class exercise

- Draw the Activity Diagram for use-case “Create Account”
- Time: 5 minutes


Search for Flights
Login
Logout
Create Account
Retrieve Password
Buy a Ticket
View my Tickets
Cancel Ticket
Traveler
«requires»
View Tickets
Manipulate Flights
Confirm Ticket
View Traveler Info
Admin
View/Edit/Delete/ Add