DIS501 Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων

Lecture 6: Use Case Modelling: Διαγράμματα Περιπτώσεων χρήσης με UML

BY: ΔΡ ΑΥΓΟΥΣΤΑ ΚΥΡΙΑΚΙΔΟΥ ΖΑΧΑΡΟΥΔΙΟΥ

LEARNING OBJECTIVES

- Μοντελοποίηση με τη χρήση της UML
- Ορισμός μοντελοποίησης περιπτώσεων χρήσης (use case), και σεναρίων
- Εξηγήστε τη σημασία της μοντελοποίησης περιπτώσεων χρήσης
- Δημιουργία διαγραμμάτων περιπτώσεων χρήσης μέσω UML (use case diagrams)
- Δημιουργία σεναρίων περιπτώσεων χρήσης
- Παροχή κατευθυντήριων γραμμών για την αποτελεσματική μοντελοποίηση περιπτώσεων χρήσης

ΑΝΑΛΥΣΗ ΑΠΑΙΤΉΣΕΩΝ - ΜΠΟΡΟΥΜΕ ΤΩΡΑ ΝΑ ΞΕΚΙΝΗΣΟΥΜΕ ΤΗΝ ΚΑΤΑΣΚΕΥΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ;

Source: Dilbert.com

ΑΝΑΛΥΣΗ ΑΠΑΙΤΗΣΕΩΝ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Functional Requirements (Λειτουργικές απαιτήσεις)

Τι κάνει το σύστημα, τι λειτουργίες υποστηρίζει

Inputs, Outputs, Processes

Use-Cases primarily model Functional Requirements

Non-Functional Requirements (Μη λειτουργικές απαιτήσεις)

Περιορισμοί στο σύστημα

Performance, Speed, Security (Απόδοση, ταχύτητα, ασφάλεια κ.λπ.)

Απαιτήσεις ευχρηστίας: Αποτελεσματικότητα του χρήστη, φιλικότητα προς τον χρήστη, κ.λπ.

UML USE-CASE MODELLING

- A **user-centred** approach to systems analysis (Μια προσέγγιση της ανάλυσης συστημάτων με επίκεντρο τον χρήστη.)
- Αναλύει τις **απαιτήσεις** διερευνώντας τις **αλληλεπιδράσεις των χρηστών** με το σύστημα.
- Προσδιορίζει και περιγράφει τις λειτουργίες του συστήματος από τη σκοπιά των **εξωτερικών χρηστών**
- Use cases are derived from requirements and satisfy the requirements. Οι περιπτώσεις χρήσης (Use cases) προκύπτουν από τις απαιτήσεις και ικανοποιούν τις απαιτήσεις.
- **Μοντελοποιεί τις εμφανείς *λειτουργικές απαιτήσεις (Functional Requirements)***
  - "System will print receipt when payment is processed"
  - "Student needs to register in order to view their grades"
  - "Customer can search for books"
  - "Customer can write reviews"

USE CASE MODELLING: **ΔΙΑΔΙΚΑΣΙΑ**

Η διαδικασία περιλαμβάνει την τεκμηρίωση

Who/what **initiates** an interaction with the system (ποιος/τι αλληλοεπιδρά με το σύστημα)

What **information goes into** the system (τι πληροφορίες εισέρχονται στο σύστημα)

What information **comes out** and (τι πληροφ[illegible]προκύπτουν από το σύστημα)

What the **m**[illegible] e. what they get out of it – ***the f***[illegible] λειτουργία που του παρέχεται)

Use-Case	What is the functional goal?
Actor	Who wants it?
Scenario	What is needed to accomplish goal? – *business tasks*.

USE CASE MODELLING: ΔΙΑΔΙΚΑΣΙΑ

- Προσδιορίζουμε τους **Actors**
- Προσδιορίζουμε τα **Use-Cases**
- Προσδιορίζουμε το **System Boundaries**
- Παρουσιάζουμε τις περιπτώσεις χρήσης (use-cases) και τις σχέσεις μεταηύ αυτών και των χρηστών (actors) με την δημιουργία ενώς διαγράμματος περιπτώσεων χρήσης (**Use-Case Diagram)**
- Δημιουργούμε ένα **Use-Case scenario** για κάθε περίπτωση χρήσης(**use-case**)
- Βελτιώνουμε το μοτέλο σε συνεργασία με τον πελάτη

UML ACTORS

Αντιπροσωπεύουν **οποιονδήποτε/οτιδήποτε (anyone/anything)** το οποίο είναι εκτός του συστήματος και χρειάζεται να **αλληλοεπιδράσει με αυτό.**

- Οι πιο εμφανείς actors είναι οι **άνθρωποι που θα χρησιμοποιήσουν** το σύστημα
- **Άλλα συστήματα** (databases, servers maintained by other people, legacy systems, bank systems)
  - Τα οποία πρέπει να αλληλεπιδράσουν με το σύστημα μας - need to interact with our system

Initiate system activity, e.g a **use-case**, for the purpose of completing some system functionality

Ο actor αντιπροσωπεύει ένα ρόλο τον οποίο παίζει ο χρήστης όταν αλληλοεπιδρά με το σύστημα.

eg. DreamVideo case study

- Sales Staff
- Customer

Roles, not individual users or people

- Smith 
- SalesStaff 

ΓΙΑ ΝΑ ΠΡΟΣΔΙΟΡΙΣΕΤΕ ΤΟΥΣ ACTORS: ΚΑΝΕΤΕ ΑΥΤΕΣ ΤΙΣ ΕΡΩΤΗΣΕΙΣ

Ποιος θα χρησιμοποιήσει το σύστημα?

Ποιος θα εισάγει, θα χρησιμοποιεί, θα ενημερώνει ή θα διαγράφει πληροφορίες από το σύστημα;

Ποιος ενδιαφέρεται για μια συγκεκριμένη λειτουργία ή υπηρεσία που παρέχεται από το σύστημα;

Ποιος θα υποστηρίζει και θα συντηρεί το σύστημα;

Ποιοι είναι οι εξωτερικοί πόροι του συστήματος;

Ποια άλλα συστήματα θα πρέπει να αλληλοεπιδρούν με το υπό ανάπτυξη σύστημα για να μπορεί να ολοκληρώσει τις λειτουργίες του;

Review the list of stakeholders

Δεν είναι όλοι οι stakeholders actors. Αλλά αυτή είναι μια καλή αρχή για να αναγνωρίσουμε τους πιθανούς actors.

UML ACTORS: EXAMPLE

Online bookstore case study example Actors:

Customer - can order books online, browse catalogue, write reviews etc.

Sales Staff - can place orders on behalf of customers, write reviews, reply to enquiries etc.

Directors - can print sales reports

Warehouse System - needs to interact with the system to maintain audit, check availability

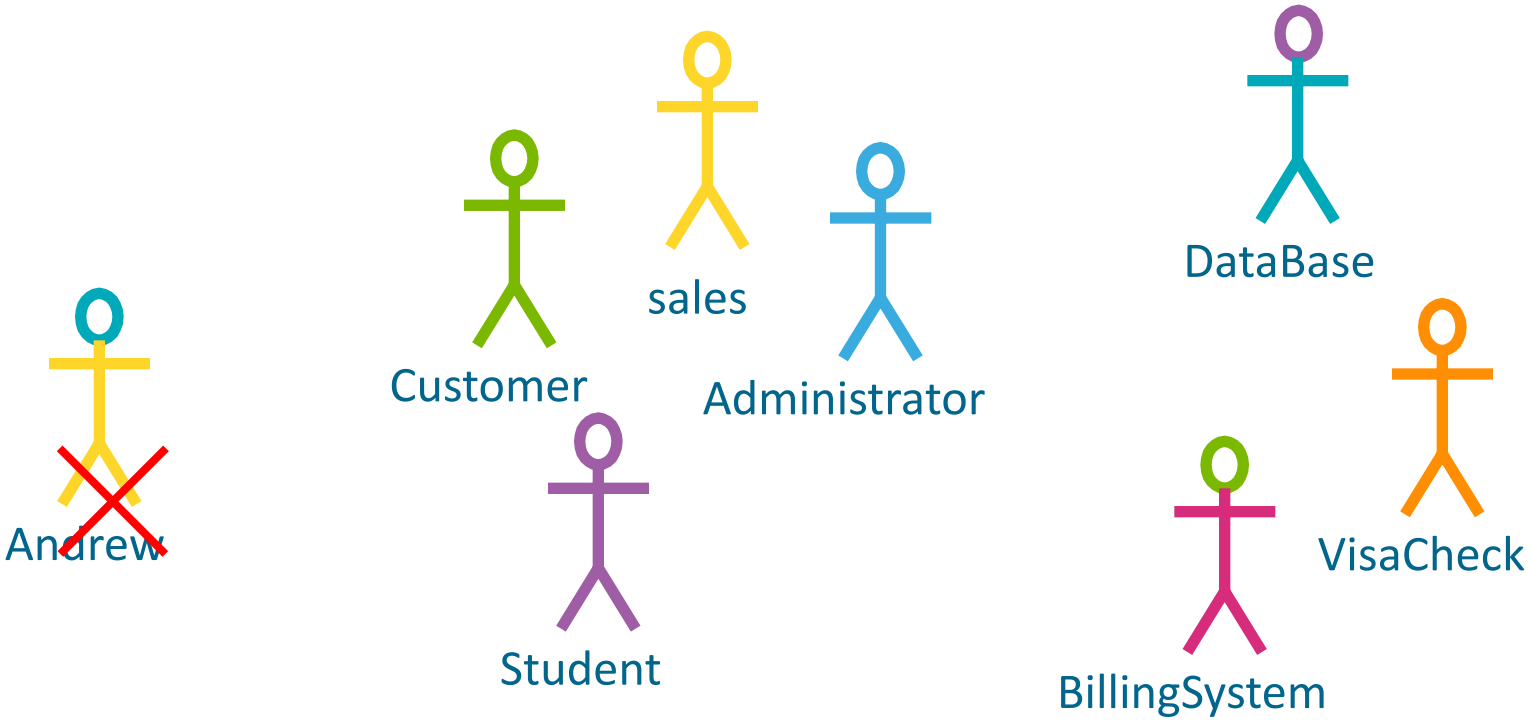
VisaCheck - needs to interact with system to verify credit card details

Human Actors

Να θυμάστε οι actors είναι οποιοσδήποτε η οτιδήποτε ενεργοποιεί μια λειτουργία του συστήματος ή πρέπει να αλληλεπιδράσει με το υπό ανάπτυξη σύστημα για την ανταλλαγή πληροφοριών

Machine Actors

UML ACTOR NOTATION



IDENTIFYING ACTORS

Ενδιαφέρων παραδείγματα:

- Το billing system για μια εταιρία δημιουργεί αυτόματα τα bills την Πέμπτη μέρα του μήνα (billing date)
- Μια τράπεζα κανονίζει να γίνονται οι συναλλαγές που πραγματοποιούνται με επιταγές κάθε μέρα στις 5:00pm
- Κάθε βράδυ δημιουργείται αυτόματα ένα report το οποίο καθορίζει ποια μαθήματα έχουν κλείσει για εγγραφή και ποια είναι ακόμη ανοικτά.

Ποιος/Τι είναι ο UML ACTOR??

TIME…Γιατί?

– Αυτά είναι παραδείγματα χρονικών γεγονότων (temporal events). Ένα χρονικό συμβάν (**temporal event)** είναι ένα συμβάν του συστήματος που ενεργοποιείται από το **ΧΡΟΝΟ.**

– Όλα τα παραπάνω παραδείγματα συμβάντων εκτελούνταν (ή ενεργοποιούνταν) αυτόματα όταν γινόταν μια συγκεκριμένη ημερομηνία ή ώρα. Συνεπώς ο actor αυτών των temporal events είναι ο χρόνος/**time.**

ΤΥΠΟΙ ACTORS
ΠΡΩΤΕΥΩΝ (PRIMARY) ACTORS

Primary Actors

- **Αλληλεπιδρούν άμεσα με το σύστημα**
- **Απαιτούν τη βοήθεια του συστήματος, και έτσι κάνουν Initiate ξεκινούν μια λειτουργία του συστήματος**
- Είναι αυτοί για τους οποίους σχεδιάζεται το μελλοντικό σύστημα
- Είναι αυτοί που εκπληρώνουν τους στόχους τους με τη χρήση του συστήματος
- E.g: **Borrower** in a Library System, **Student or Lecturer** in a University system
  Customer in an online shopping system.

ΤΥΠΟΙ ACTORS

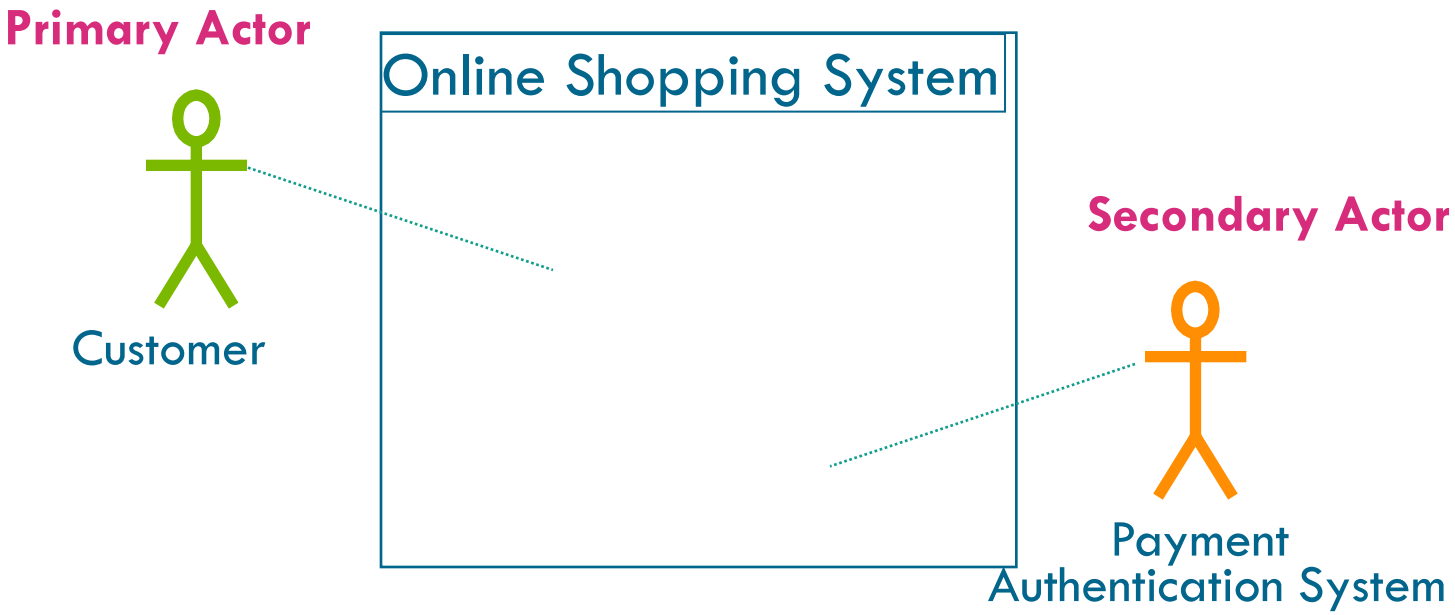
ΔΕΥΤΕΡΕΥΩΝ (SECONDARY) ACTORS

Secondary Actors

- **Αλληλεπιδρούν έμμεσα με το σύστημα**
- **Το σύστημα χρειάζεται την βοήθεια τους**
- Παρέχουν μια υπηρεσία στο μελλοντικό σύστημα, π.χ. μια ***βοηθητική υπηρεσία***
- Εκπληρώνουν έμμεσα τους στόχους του χρήστη παρέχοντας στο σύστημα βοηθητικές υπηρεσίες που θα ολοκληρώσουν αυτούς τους στόχους

E.g: **Database Administrator** in a Registry System,
VisaCheck system in a payment system,
Warehouse system in an e-commerce system

ΤΥΠΟΙ ACTORS: UML REPRESENTATION



Οι Actors τοποθετούνται εκτός του συστήματός. Όλοι οι actors σε ένα διάγραμμα πρέπει να σχετίζονται με τουλάχιστον μία περίπτωση χρήσης (use-case).

ΜΟΝΤΕΛΟ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ (USE CASE MODELLING): ΟΡΙΣΜΟΣ ΠΕΡΙΠΤΩΣΗΣ ΧΡΗΣΗΣ (USE CASE)

- **Οι λειτουργίες που υποστηρίζονται από το σύστημα.**
- Περιγράφουν τις λειτουργικές απαιτήσεις (**functional requirements) που πρέπει το σύστημα να υποστηρίζει**
  - E.g **Place Order** **or** **Write Review** **or** **Search for products**
- **Τεχνική βασισμένη σε σενάρια (Scenario-based) στη UML**
  - Κάθε περίπτωση χρήσης περιγράφεται ως μια σειρά βημάτων του τρόπου με τον οποίο ένα συγκεκριμένο σύστημα μπορεί να χρησιμοποιηθεί από τους actors (τελικούς χρήστες ή άλλα συστήματα)
- Μια περίπτωση χρήσης (use case) ενεργοποιείται από τους actors (συνήθως τους Primary Actors)
  - Ένας actor μπορεί να συσχετίζεται με περισσότερα από ένα use-cases

USE CASES: UML NOTATION

Όνομα use case: Συνδυασμός ρήματος-ουσιαστικού – για να υποδηλώνει μια ενέργεια, κάτι που κάνει το σύστημα.

- **Κάθε use case** αντιπροσωπεύει μια **λειτουργία** την οποία ενεργοποιεί ο actor. Είναι μια περίπτωση χρήσης του συστήματος

- **Use-Cases αντιμετωπίζουν το σύστημα ως μαύρο κουτί**

  - Καταγράφουν ***ποιος*** **(actor)** κάνει ***τι (πως αλληλεπιδρά με το σύστημα*** **(interaction)** , για ποιο ***σκοπό*** **(goal),**
    χωρίς να ασχολείται με τα εσωτερικά του συστήματος.

IDENTIFYING USE-CASES

Για κάθε **actor** αποφασίστε ποιες λειτουργίες (**functions**) χρειάζονται

University Online System Example:

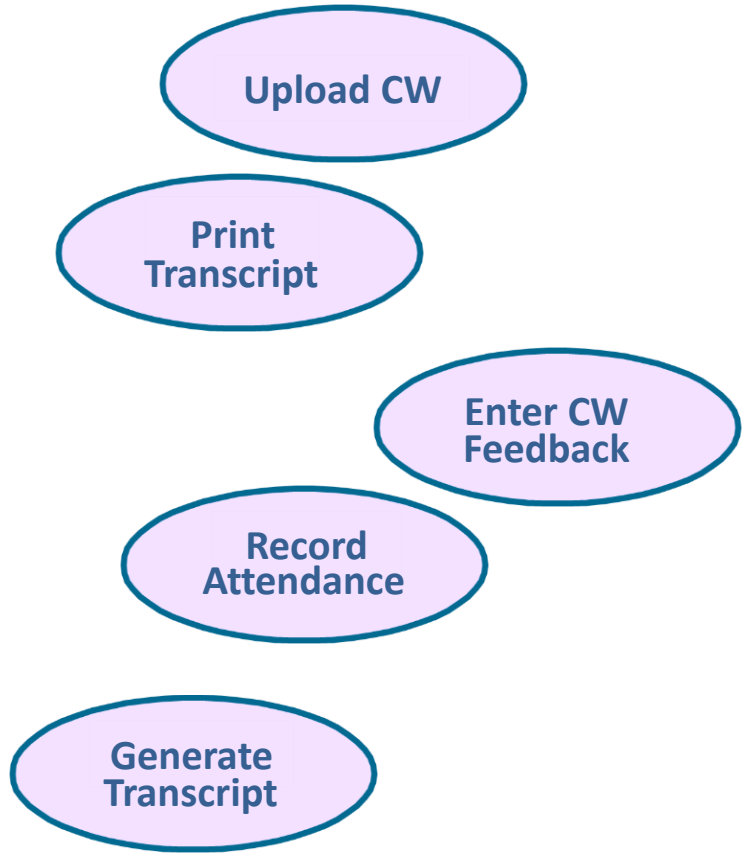
Student - upload coursework,

- print transcript

Lecturer - enter coursework feedback

- record attendance

StudentRecords - generate transcripts



IDENTIFYING USE-CASES

Χρήση των προδιαγραφών απαιτήσεων για τον προσδιορισμό των περιπτώσεων χρήσης από την οπτική γωνία ενός actor

- Ποιες είναι οι κύριες λειτουργίες (λειτουργικότητα που απαιτείται) για κάθε actor; (λειτουργικές απαιτήσεις)(functional requirements)
- Θα πρέπει ο actor να διαβάσει/γράψει/αλλάξει οποιαδήποτε πληροφορία του συστήματος;
- Θα πρέπει ο actor να ενημερώνει το σύστημα για εξωτερικές αλλαγές;
- Επιθυμεί ο actor να ενημερώνεται για απρόβλεπτες αλλαγές;

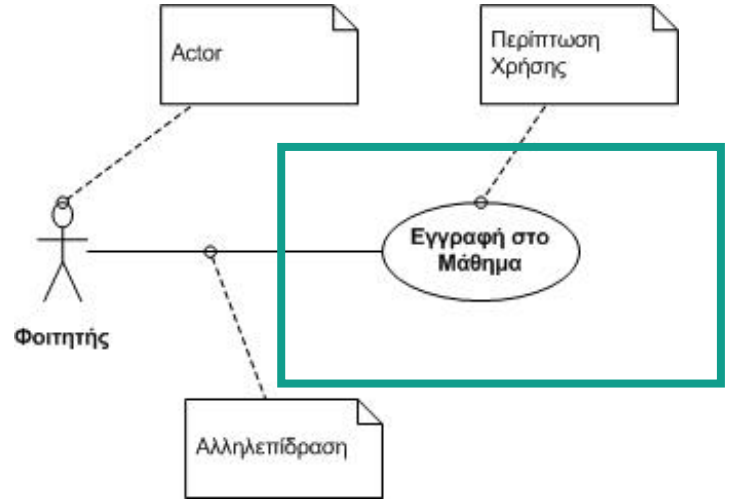
Θυμηθείτε ότι οι περιπτώσεις χρήσης αποτελούν το σημείο εκκίνησης για το δυναμικό μοντέλο (dynamic model).

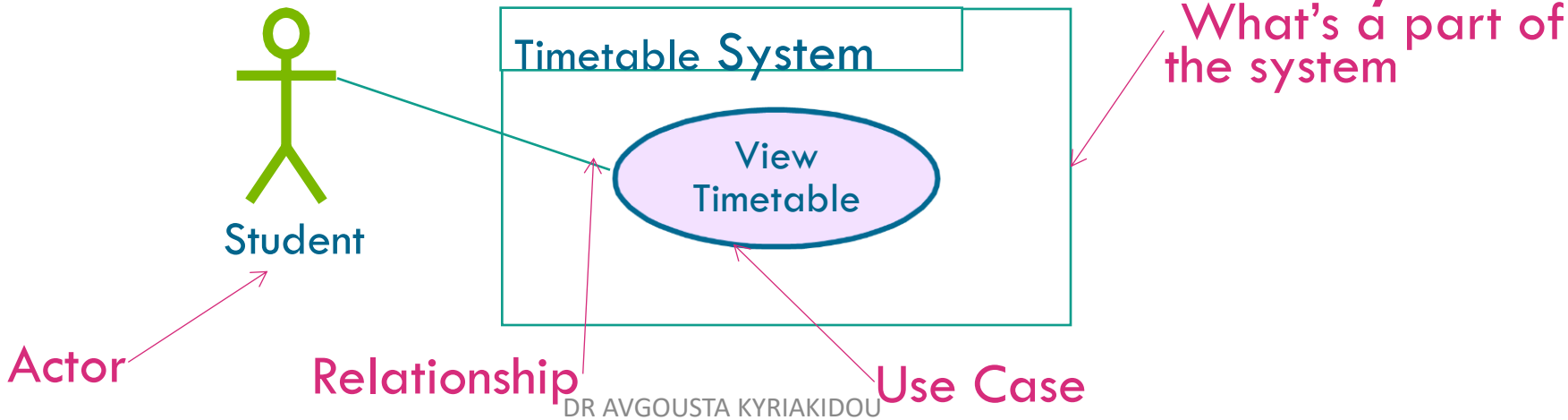
- Απεικονίζουν λειτουργία
- Αλλά δεν προσδιορίζουν τη συμπεριφορά ενός συστήματος η να δείχνουν την σειρά με την οποία συγκεκριμένες λειτουργίες ενεργοποιούνται. (Αυτό είναι κάτι που προσδιορίζεται στο object sequence model)
- Τα διαγράμματα περιπτώσεων χρήσης δεν δείχνουν πώς ανταλλάσσονται τα μηνύματα και δεν είναι διαγράμματα ροής

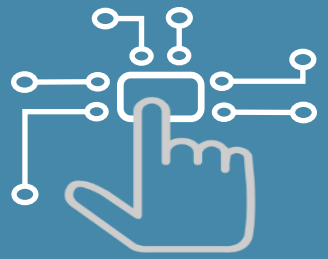
USE CASE DIAGRAMS

- **Τέσσερα στοιχεία:**
  - System – or part of system (καθορισμένα όρια)
  - Actors
  - Use-cases
  - Relationships

Ένα σύστημα μπορεί να απαιτεί πολλά διαγράμματα περιπτώσεων χρήσης και είναι σκόπιμο να διατηρούνται απλά και άμεσα.

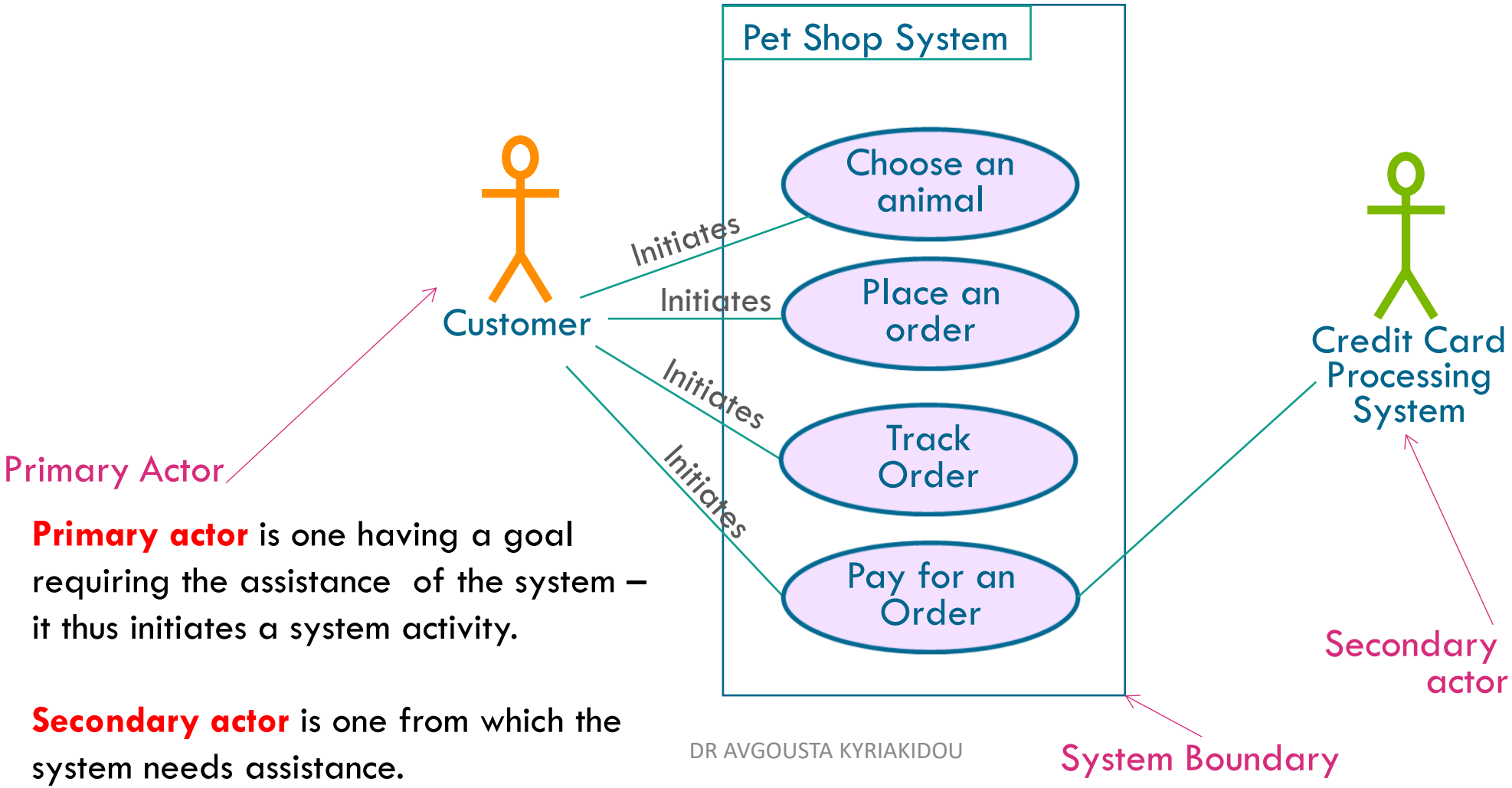






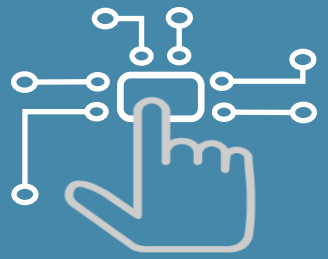
Use-Case Diagram

Example – Online Pet Shop

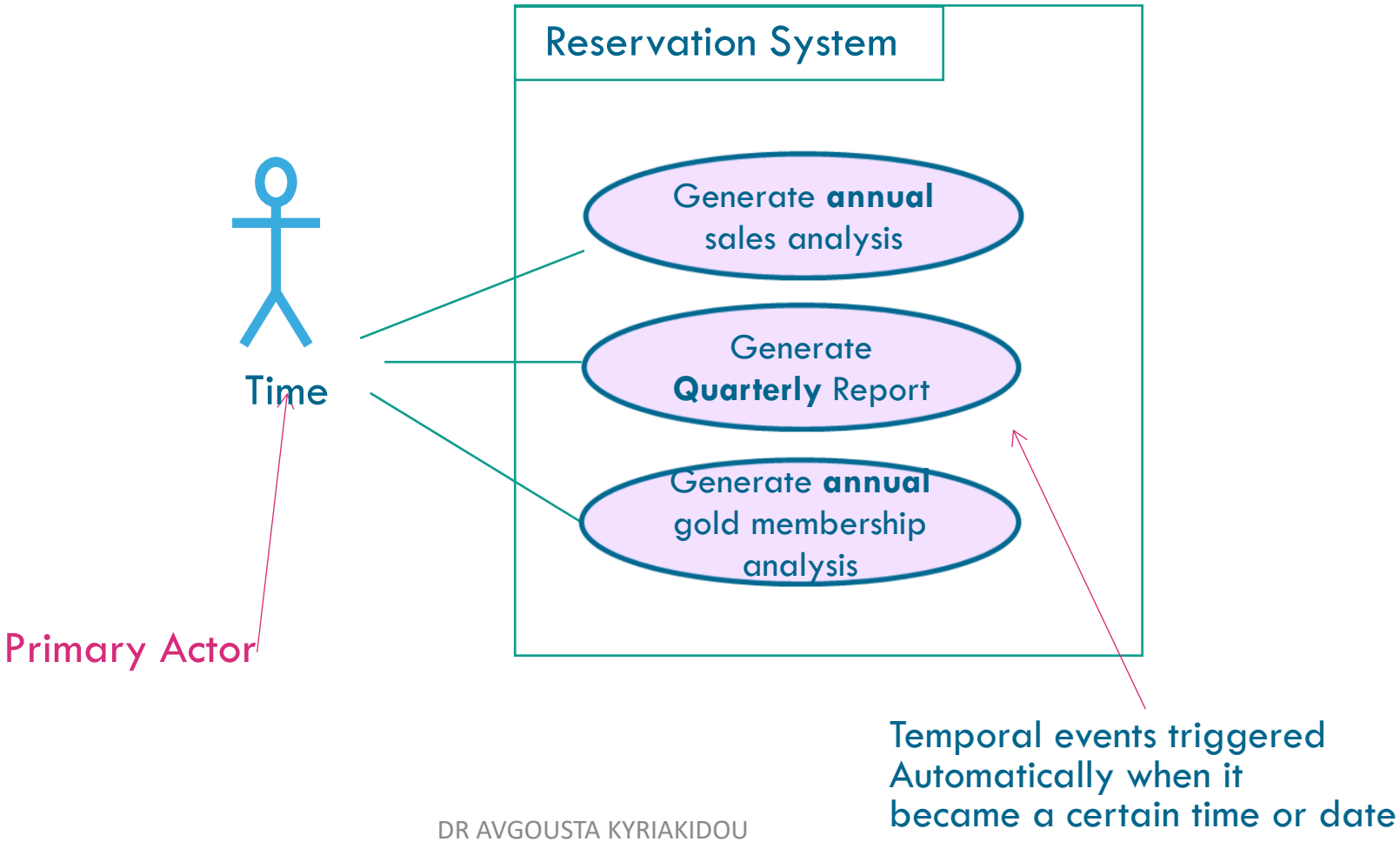


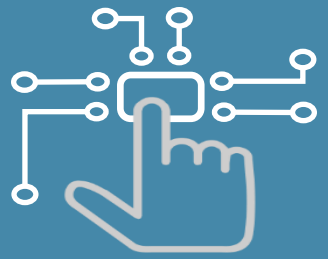
Primary actor is one having a goal requiring the assistance of the system – it thus initiates a system activity.

Secondary actor is one from which the system needs assistance.

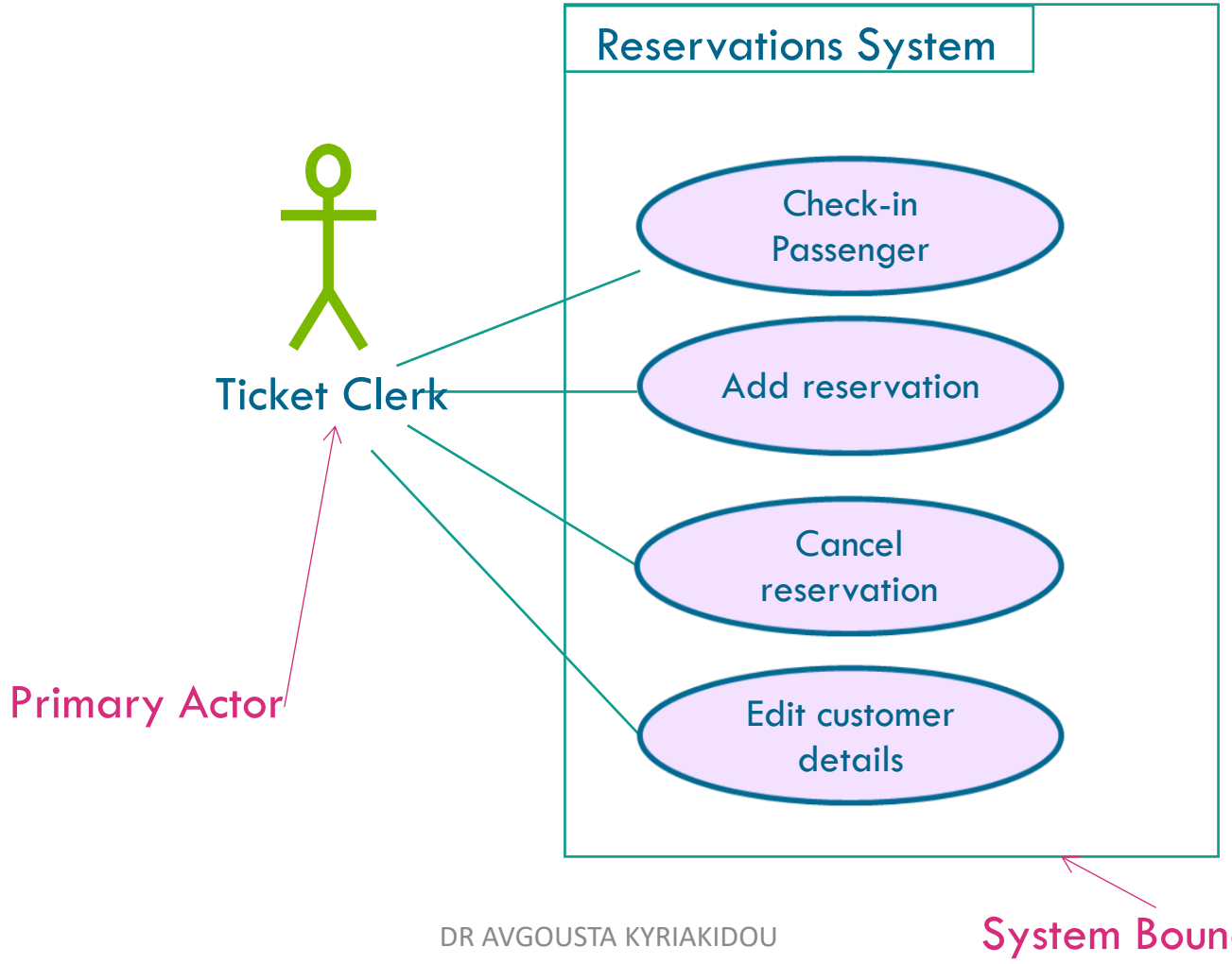


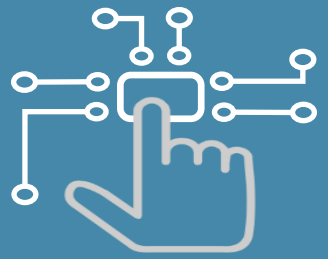
Use-Case Diagrams
Example – Reservations System 1



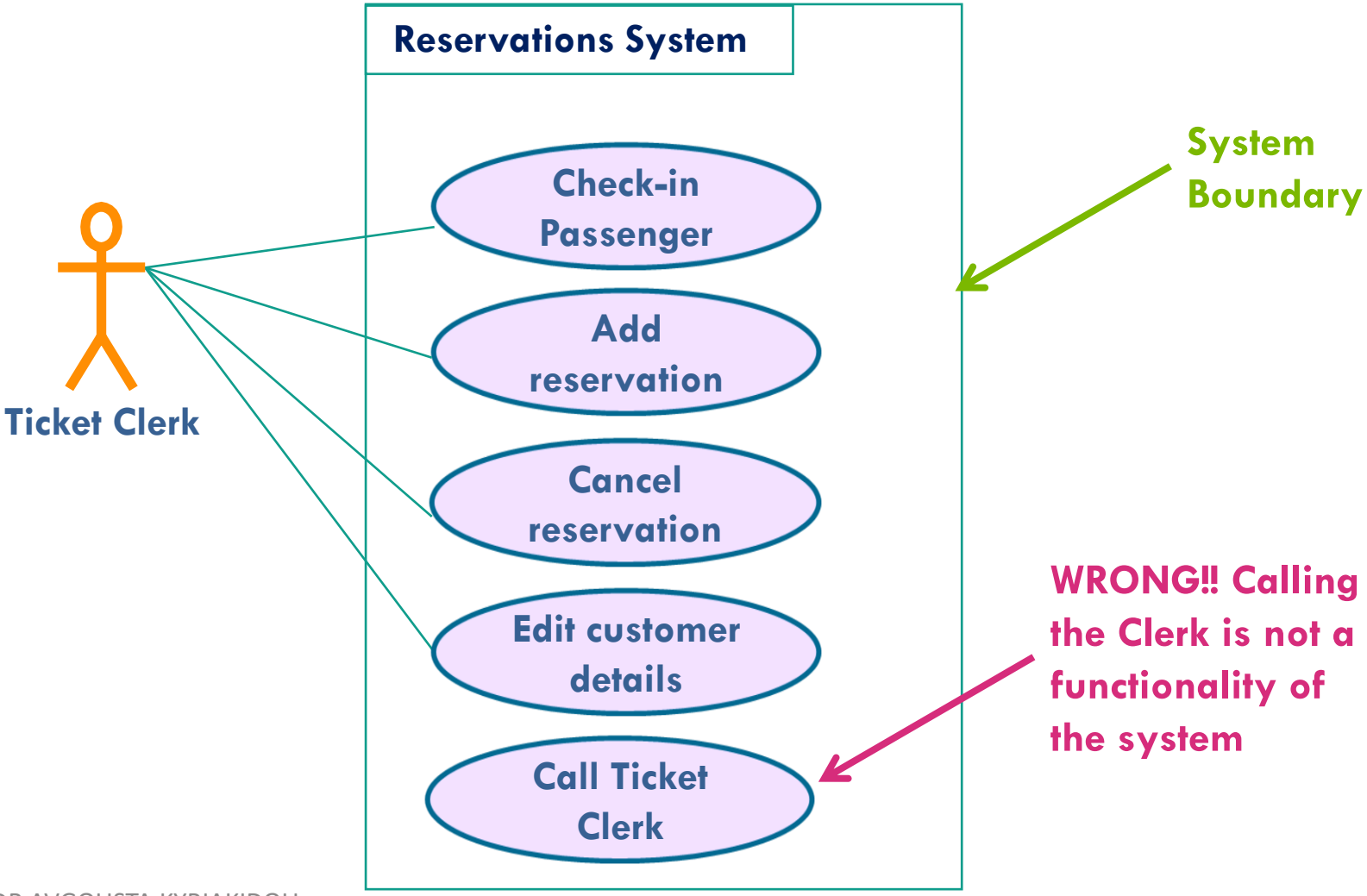


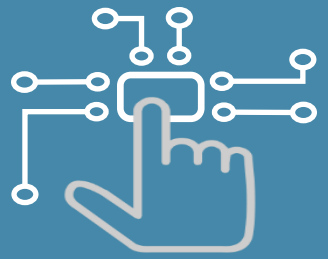
Use-Case Diagrams
Example – Reservations System 2





System Boundary
Example



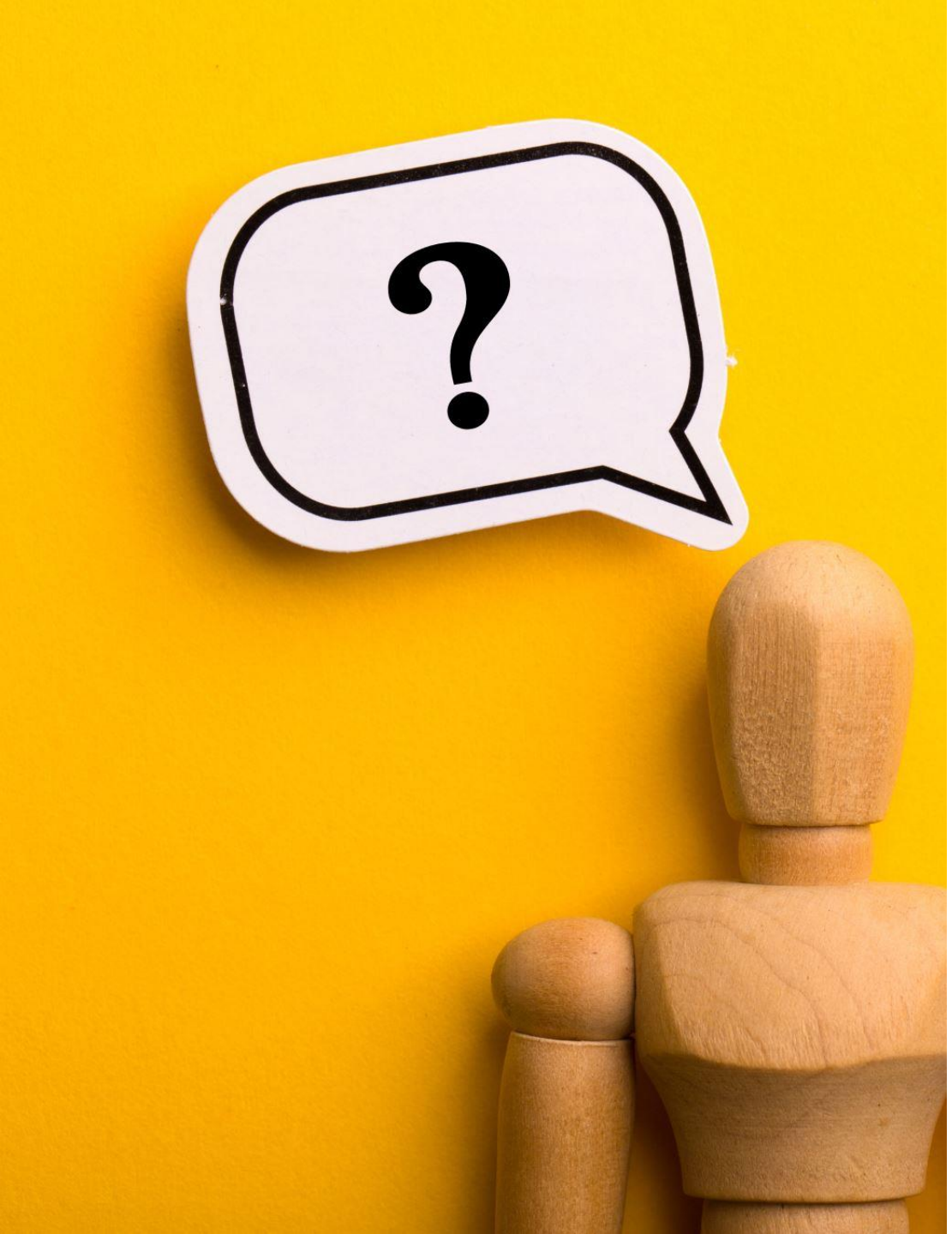


System Boundary

- Ο καθορισμός των ορίων μπορεί να είναι δύσκολος!

  (πού θα τραβήξετε τη γραμμή ή ποιες περιπτώσεις χρήσης θα πρέπει να είναι μέσα σε αυτή;)

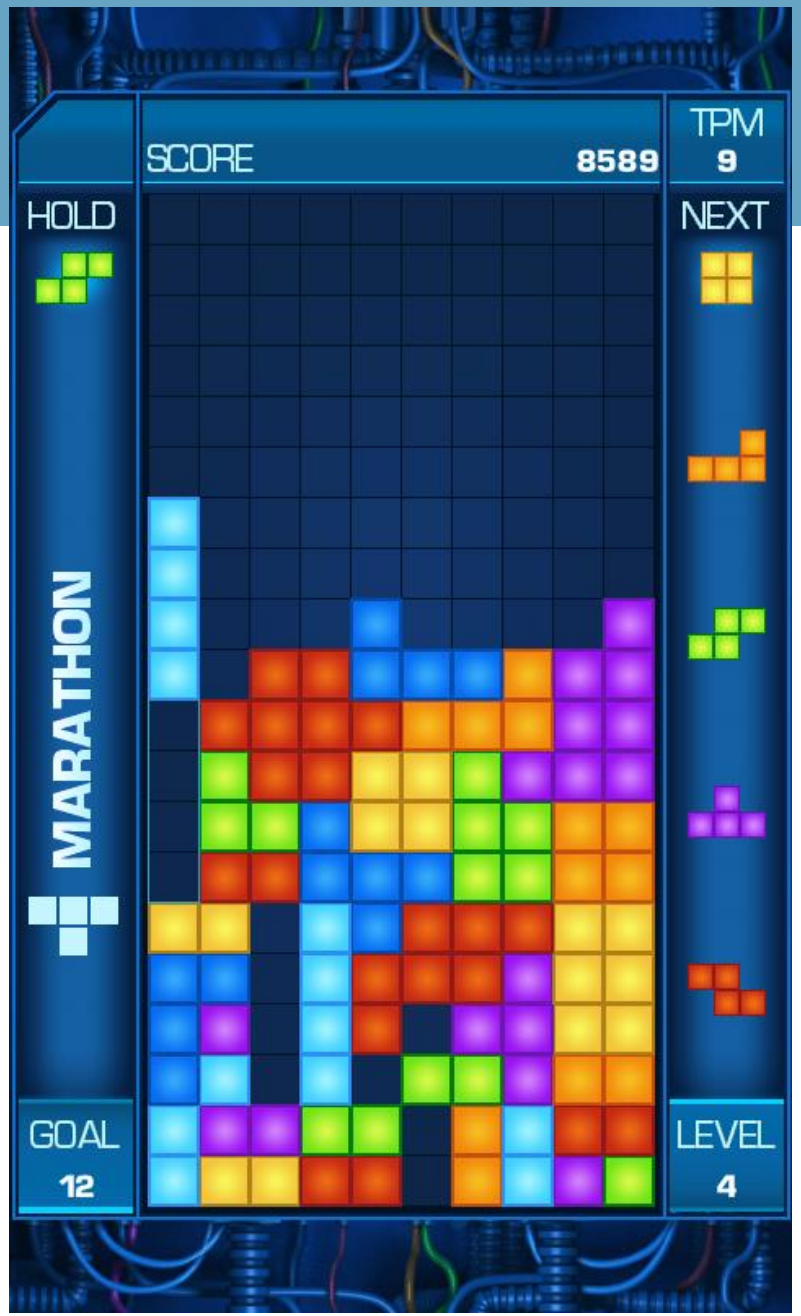
- Μια επαναληπτική διαδικασία.
  - Οι λειτουργικές απαιτήσεις (περιπτώσεις χρήσης) προέρχονται πάντα από τον πελάτη.
  - **Μην επινοείτε απαιτήσεις. Πάντα να ελέγχετε με τον πελάτη σας.**

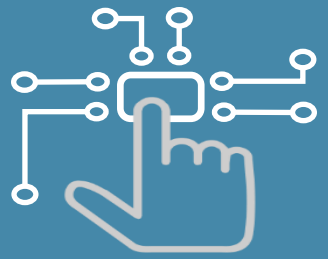


BREAKOUT SESSION

Tetris Game

- In groups of 3-4
- **Draw** a use-case diagram for the Tetris game





Tetris – sample diagram

IN GROUPS

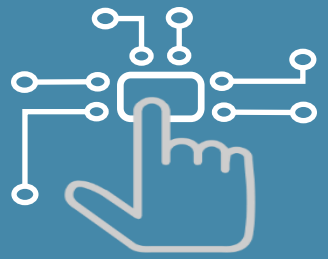
Αποφασίστε τις λειτουργικές απαιτησεις των πιο κάτω συστημάτων

- **telephone answering machine**
- **web-based online email system.**

1. Προσδιορίστε τις περιπτώσεις χρήσης, τους σχετικούς actors και ετοιμάστε ένα απλό διάγραμμα περιπτώσεων χρήσης για κάθε μία από αυτές τις εφαρμογές.

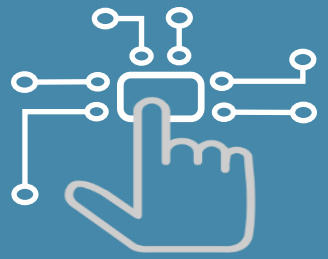
IN GROUPS

IN GROUPS



Use-Case Diagrams
adding more design detail

- Τα διαγράμματα περιπτώσεων χρήσης αναπαριστούν τη λειτουργικότητα από πάνω προς τα κάτω - in a top-down manner .
- Το αρχικό βήμα είναι να προσδιοριστεί όλη η συμπεριφορά του συστήματος που πρόκειται να δημιουργηθεί σε ανώτατο επίπεδο (top-level behaviour)
  - Αναγνωρίζουμε τις βασικές λειτουργίες και όλα τα πράγματα που πρέπει να μπορεί να κάνει το σύστημα.
- Συνεχίζουμε να προσθέτουμε λεπτομέρεια στο διάγραμμα μας με την αποσύνθεση των προσδιορισμένων περιπτώσεων χρήσης (use-cases) σε άλλες περιπτώσεις χρήσης (use-cases) οι οποίες **<<περιλαμβάνονται>> (χρησιμοποιούνται) (included)** ή **<<επεκτείνονται>> (extended)** από τις περιπτώσεις χρήσης ανωτάτου επιπέδου
- Με αυτό τον τρόπο μπορούμε να δείξουμε τη σχέση **γενίκευσης/ειδίκευσης (generalisation/specialisation)** μεταξύ των use cases



Adding More Detail to Use-Case Diagrams
Είδη σχέσεων

- **Μεταξύ του actor και των use-case**
  - Ο Actor χρησιμοποιεί/ενεργοποιεί ένα use-case , association relationship

- **Μεταξύ use-cases: Στερεότυπα**
  - Τρία στερεότυπα UML
    <<extend>> το οποίο προσδιορίζει σχέση επέκτασης η οποία είναι προαιρετική
    <<include>> το οποίο προσδιορίζει σχέση συμπερίληψης η οποία είναι υποχρεωτική

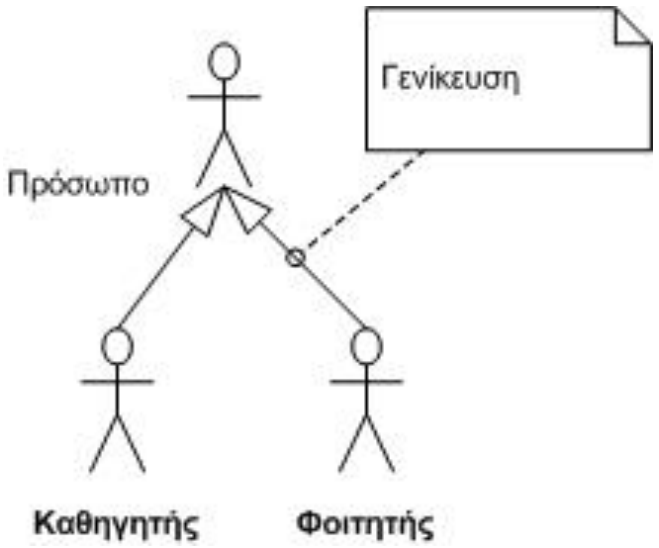
  Σχέση γενίκευσης μεταξύ use cases, η οποία δείχνει κληρονομικότητα (inheritance), την δείχνουμε με ένα άδειο (hollow) triangle

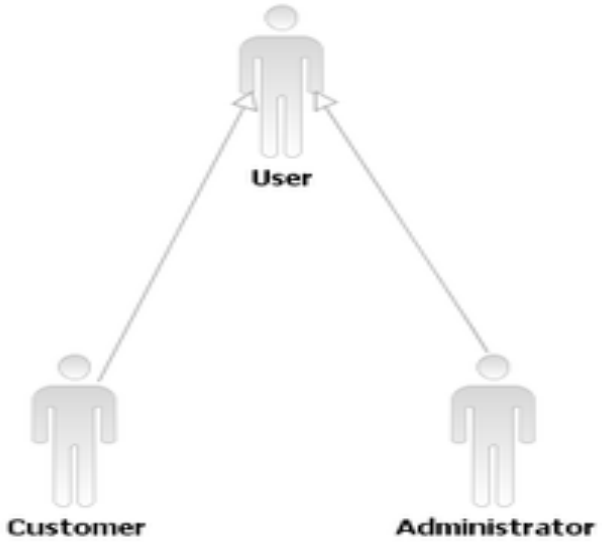
- **Μεταξύ actors: Γενίκευση (Generalisation) των actors (κληρονομικότητα- inheritance)**
  - Για να δείξουμε ότι μπορούμε να έχουμε πολλά είδη χρηστών

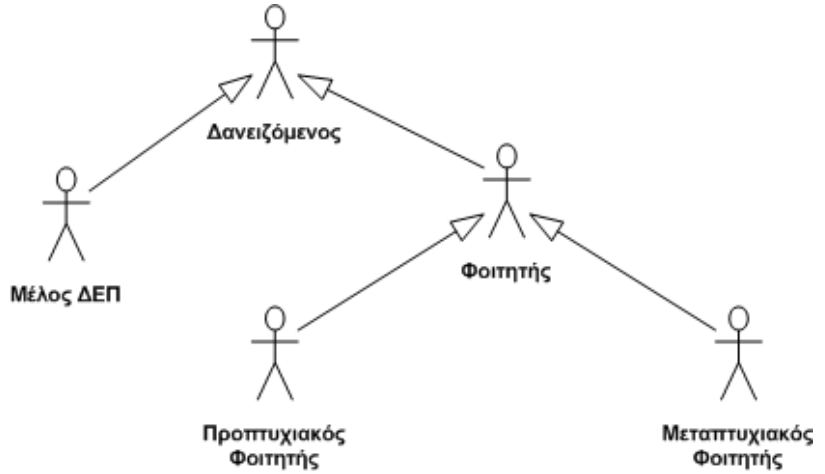
ΣΧΕΣΗ ΜΕΤΑΞΥ ACTORS - ΓΕΝΙΚΕΥΣΗ

- **Χρησιμοποιούμε τη γενίκευση των actor όταν θέλουμε να δείξουμε ομοιότητες μεταξύ των actors**
- **Σχέση Generalization/Specialization.**
- **Οι actors θα πρέπει να εμφανίζουν κοινή συμπεριφορά σε σχέση με το σύστημα**
- Ο συμβολισμός είναι μια συμπαγής γραμμή που καταλήγει σε ένα κενό τρίγωνο. Από τον εξιδεικευμένο στον γενικό actor.

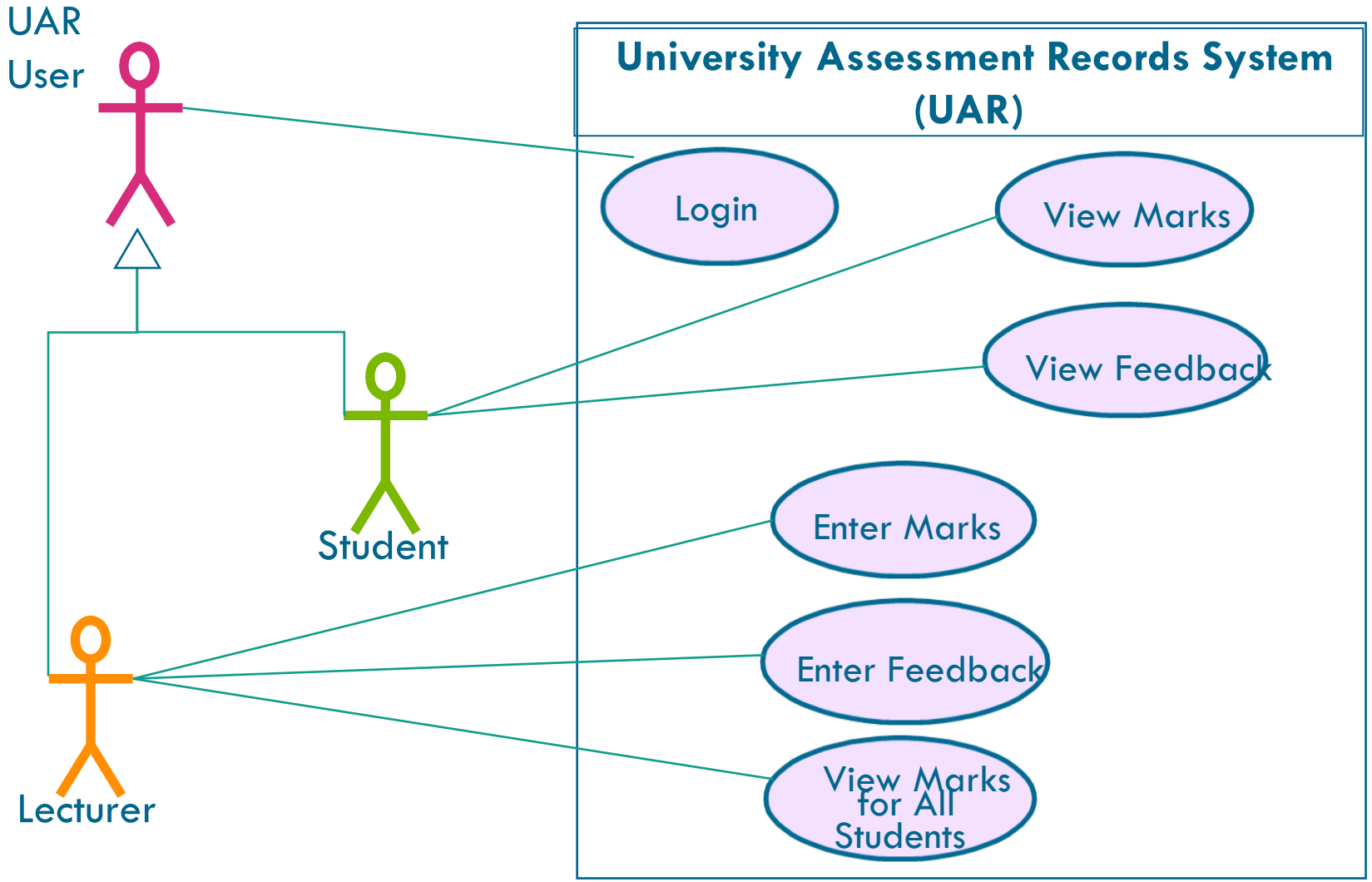
Inheritance

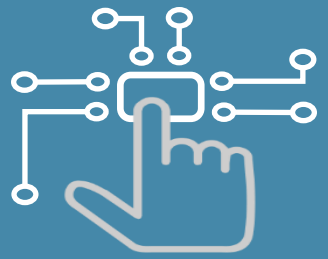






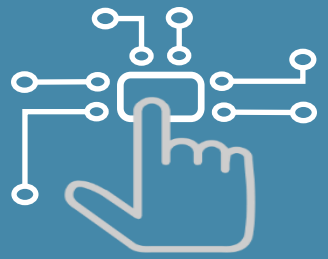
GENERALISATION OF ACTORS - EXAMPLE





Use-Case Relationship Types
Σχέση συμπερίληψης <<includes>>

- Σε αρκετές περιπτώσεις τα βήματα που εκτελούνται σε μία περίπτωση χρήσης μπορεί να επαναλαμβάνονται στην εκτέλεση βημάτων κάποιας άλλης περίπτωσης χρήσης.
- Προκειμένου να αποφύγουμε την επανάληψη των βημάτων μπορούμε να εισάγουμε τη σχέση της συμπερίληψης (includes) μεταξύ δύο περιπτώσεων χρήσης



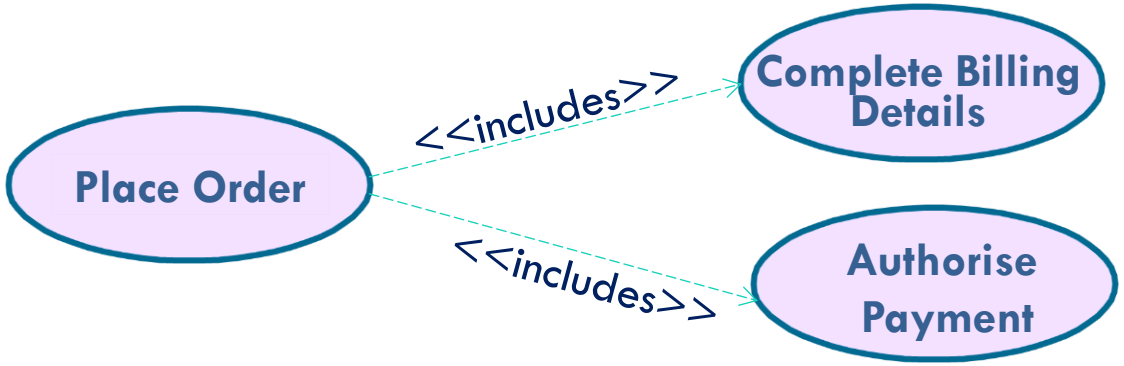
Use-Case Relationship Types
Σχέση συμπερίληψης <<includes>>

- **Στα βήματα της περίπτωσης χρήσης use-case (A) συμπεριλαμβάνονται τα βήματα της περίπτωσης χρήσης Use-Case (B)**

  - Η A αναφέρεται ως βασική και η B ως συμπεριλαμβανόμενη (included) περίπτωση χρήσης
  - Το τόξο αρχίζει από το use case A και πάει στο use case B για να δείξει ότι **"για να εκτελεστεί η λειτουργία (use case A) πρέπει η λειτουργία (use case B) να εκτελεστεί πρώτα τουλάχιστον μια φορά**."
  - **Είναι υποχρεωτική συμπερίληψη (mandatory inclusion)** – π.χ για να εκτελεστεί επιτυχώς η περίπτωση χρήσης A, η περίπτωση χρήσης B **ΠΡΕΠΕΙ ΝΑ ΟΛΟΚΛΗΡΩΘΕΙ ΕΠΙΤΥΧΩΣ τουλάχιστον μία φορά.**

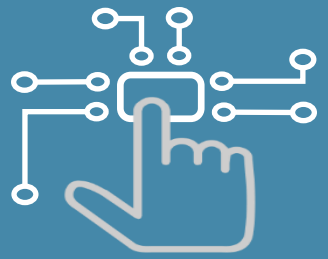
- Εάν μια περίπτωση χρήσης **<<περιλαμβάνει>>(<<includes>>)** αρκετές άλλες, όλες αυτές οι άλλες περιπτώσεις χρήσης **ΠΡΕΠΕΙ ΝΑ ΟΛΟΚΛΗΡΩΘΟΥΝ ΠΡΩΤΑ**, αν και στα διαγράμματα περιπτώσεων χρήσης δεν μας ενδιαφέρει η σειρά με την οποία ολοκληρώνονται.

Use-Case Relationship Types <<includes>>

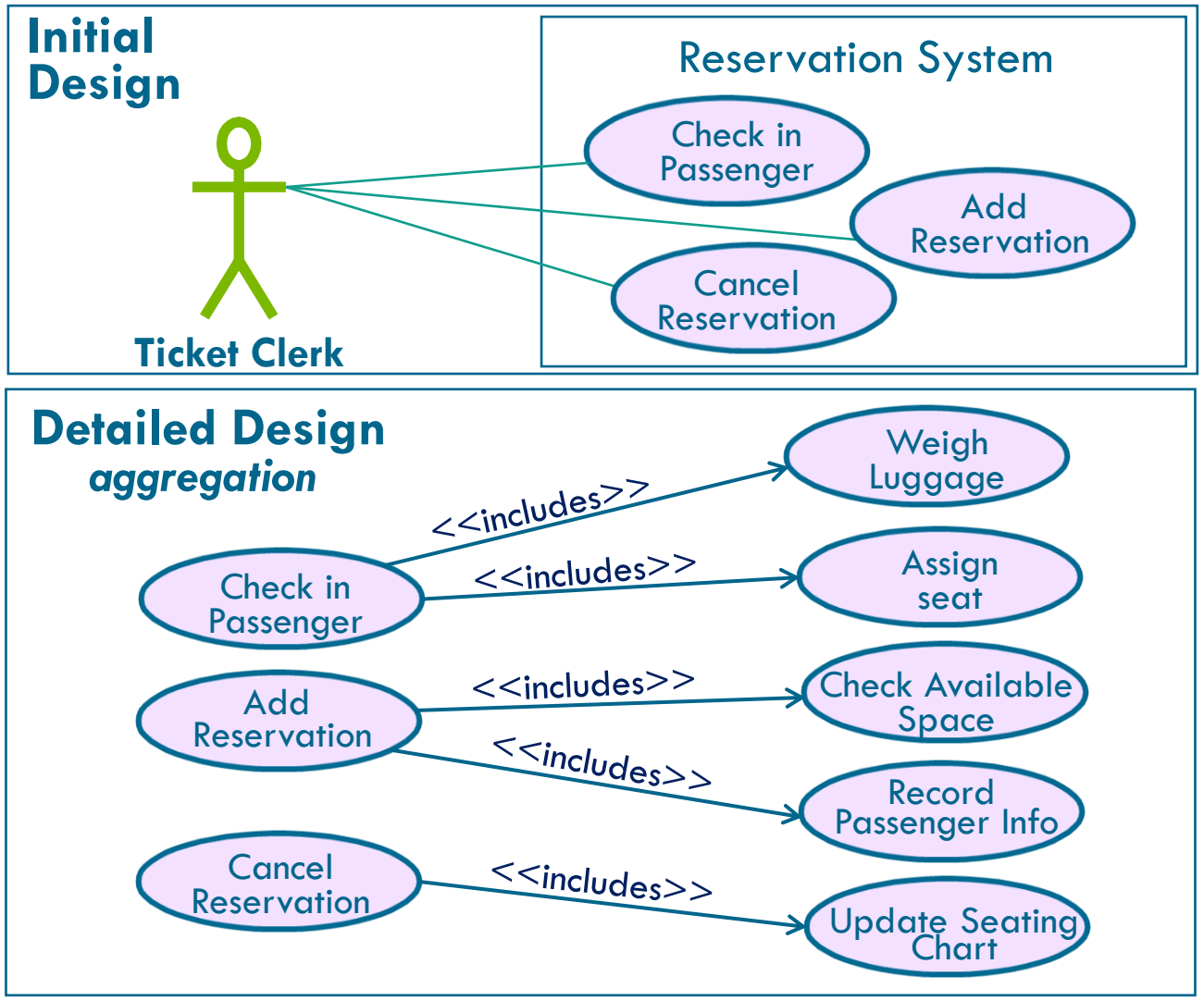


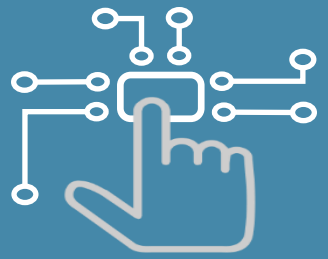
Reusability

- **<<includes>> is used when:**
  - **Abstracting common behaviour (Αφαιρούμε κοινή συμπεριφορά) -** συμπεριφορά που είναι παρόμοια σε πολλές περιπτώσεις χρήσης μπορεί να διαχωριστεί σε ξεχωριστές περιπτώσεις χρήσης.
  - **Decomposing complicated behaviour that is mandatory (Αποσύνθεση περίπλοκης συμπεριφοράς που είναι υποχρεωτική)-** Ξεχωρίστε την ως ξεχωριστή περίπτωση χρήσης και αφήστε τους άλλους να την "συμπεριλάβουν"



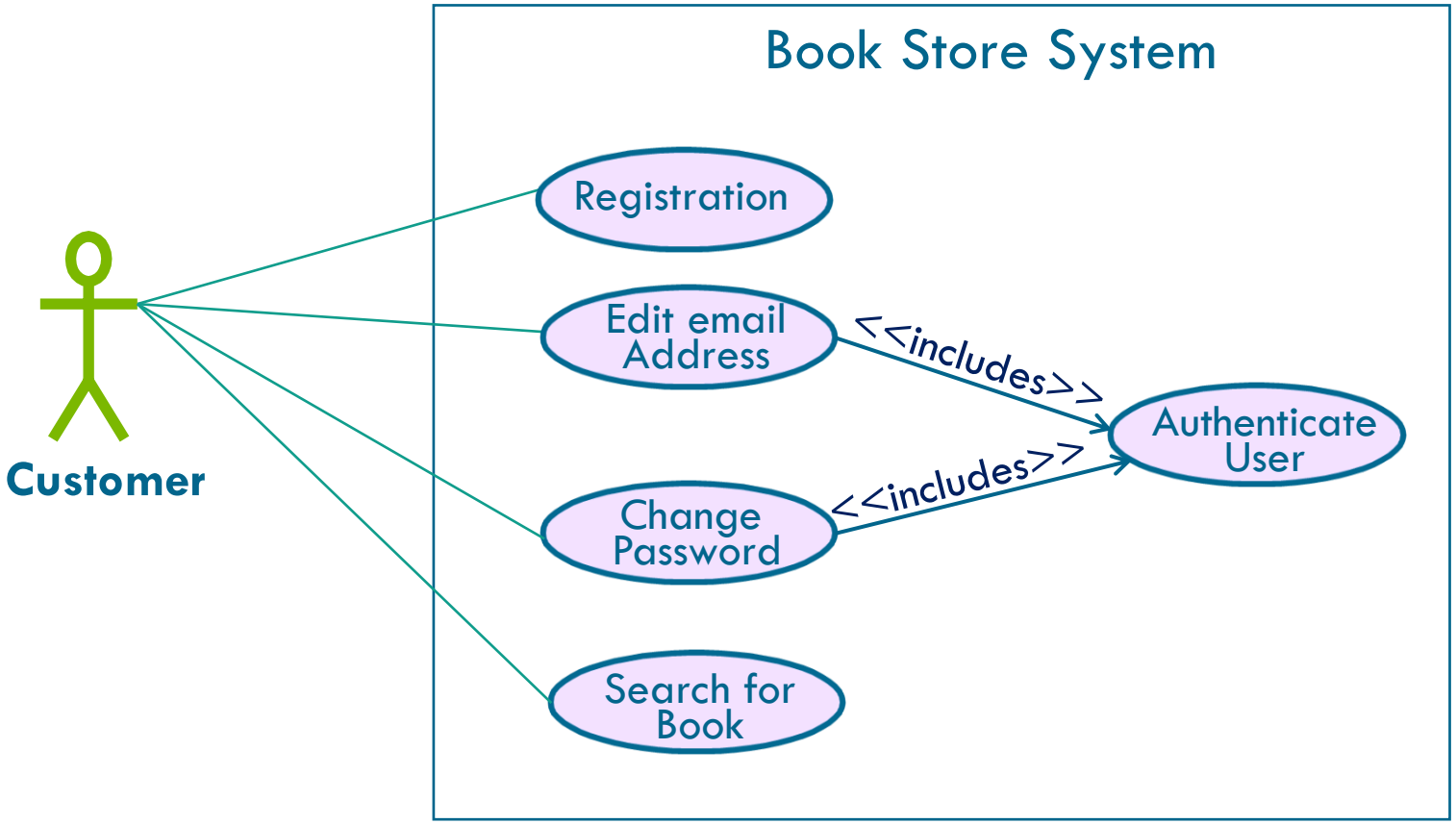
<<Includes>> example
reservations system





<<Includes>> example

book store system

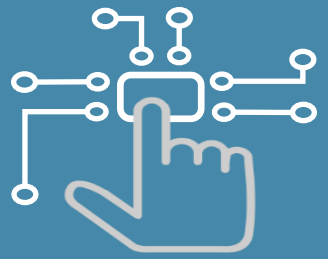


Use-Case Relationship Types
Σχέση επέκτασης <<extends>>

- Αυτή η σχέση υποδηλώνει ότι η περίπτωση χρήσης B επεκτείνει τη λειτουργικότητα της περίπτωσης χρήσης A χωρίς η A να το γνωρίζει

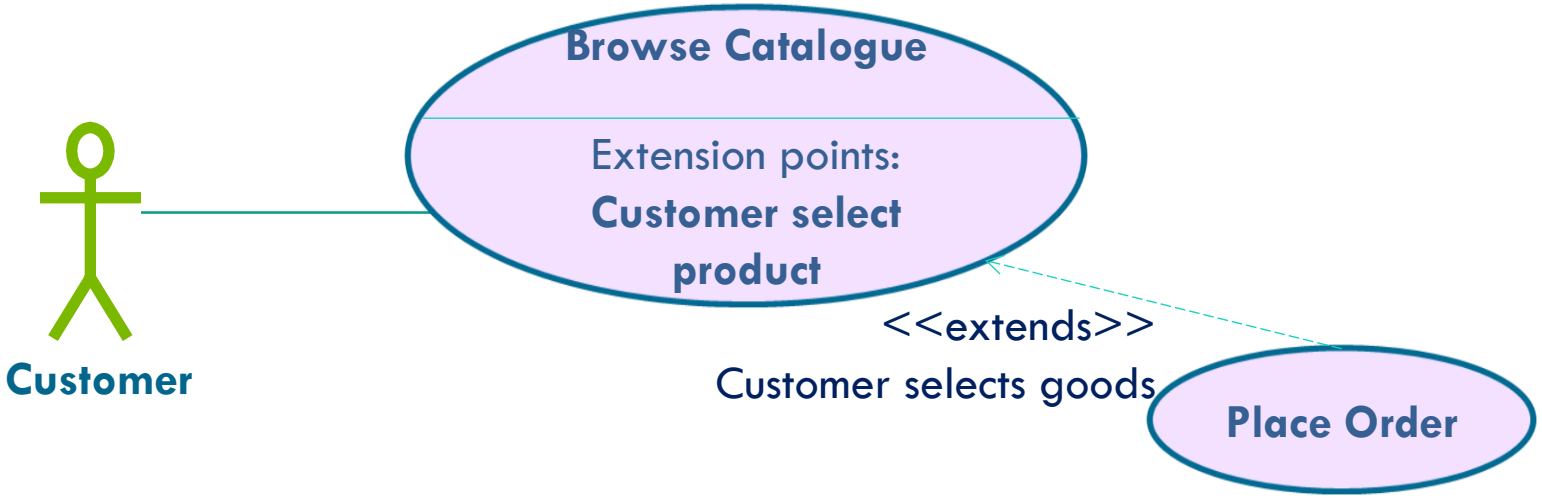
- **Το base use-case A μπορεί να ενεργοποίηση το use-case B όταν κάποια κριτήρια πληρωθούν. Αυτά τα κριτήρια ονομάζονται extension points**

1. **Υπονοεί προαιρετική επέκταση**– π.χ το use-case ***A*** μπορεί να εκτελεστεί επιτυχώς χωρίς το use case B. Το use case B επεκτείνει τη λειτουργικότητα του use case A (προσθέτει συμπεριφορά) μόνο όταν πληρούνται οι καθορισμένες συνθήκες.

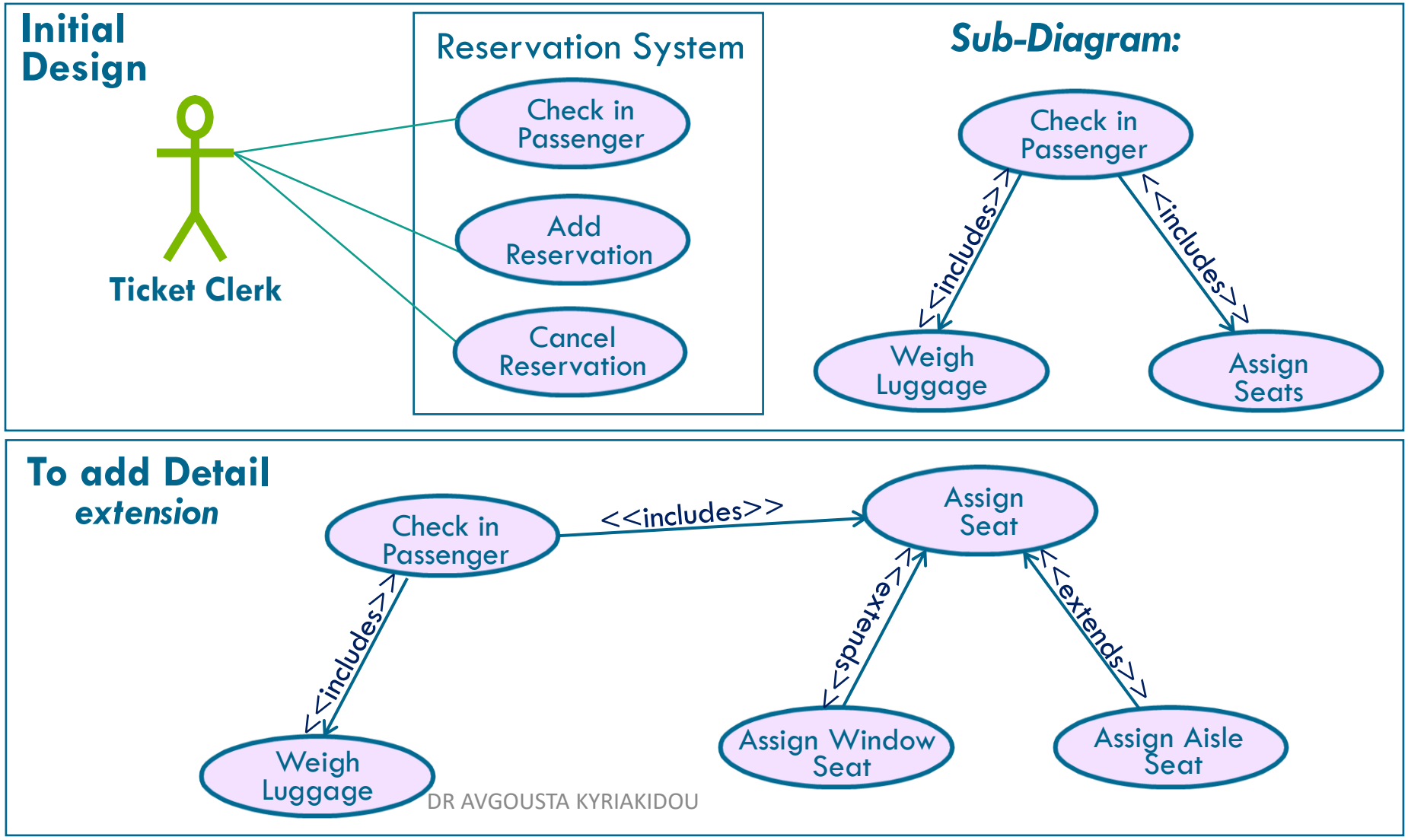


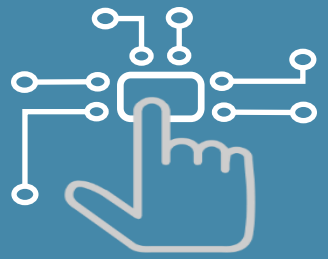
<<extends>>

Optional behaviour

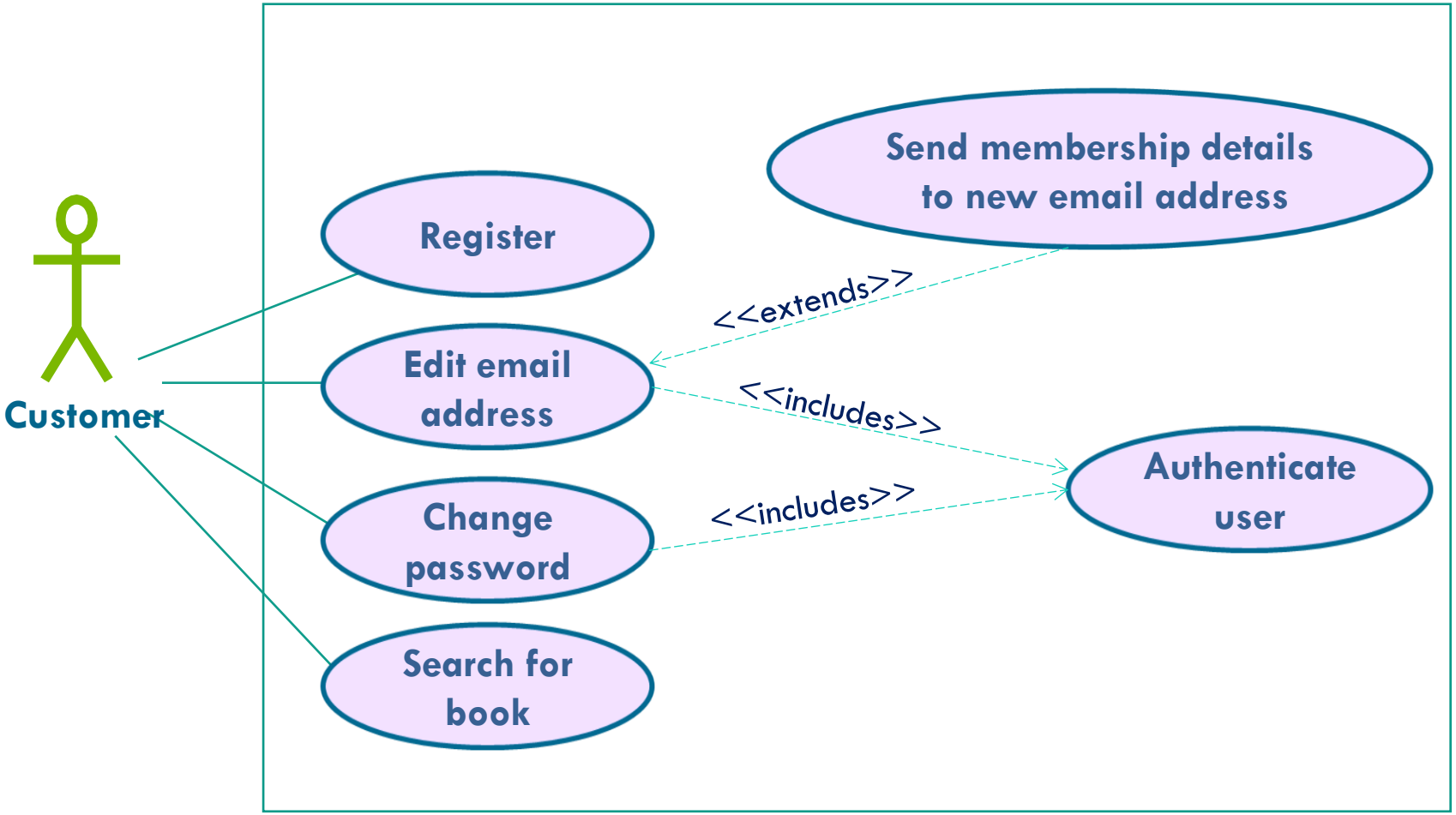


<<extends>> example
Reservations System



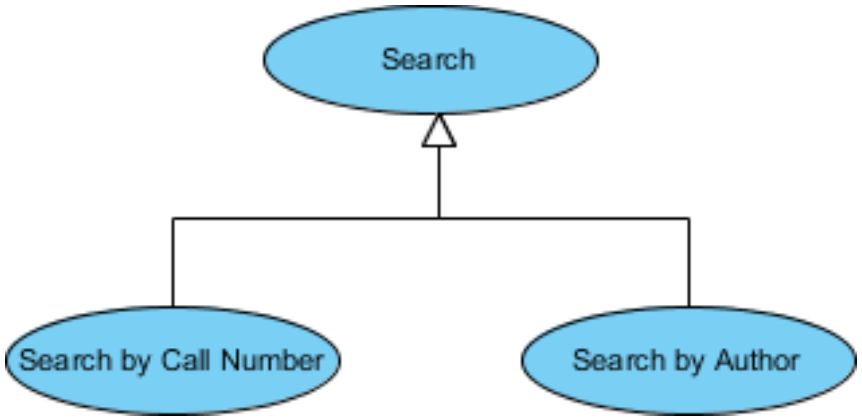


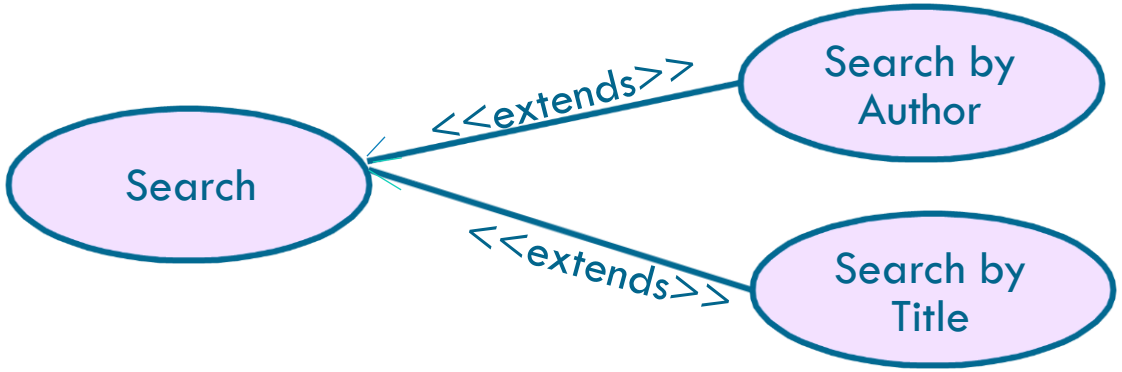
<<extends>> example
Book Store System



Note: The primary actor is only linked to the primary use case and not to the <<extends>> or <<includes>> use case

Σχέση γενίκευσης
<<generalization relationship>>

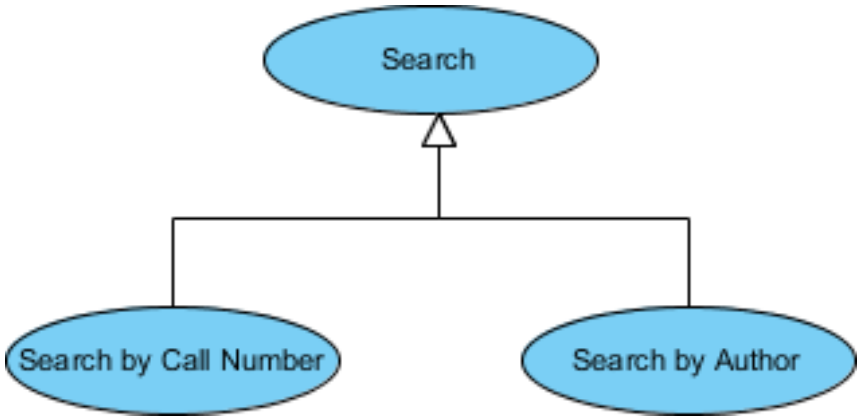


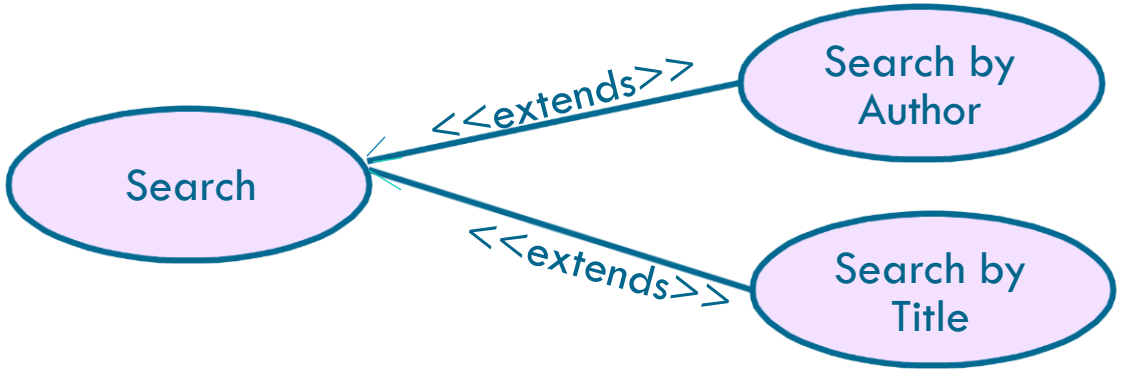


Inheritance polymorphism

- Η σχέση γενίκευσης χρησιμοποιείται όταν:
  - Μια περίπτωση χρήσης είναι παρόμοια με μια άλλη, αλλά κάνει λίγο περισσότερα.
  - Βάζουμε την κανονική/κοινή συμπεριφορά σε μία περίπτωση χρήσης
  - την ειδική συμπεριφορά σε μια ξεχωριστή περίπτωση χρήσης

Σχέση γενίκευσης
<<generalization relationship>>





Inheritance
polymorphism

Η σχέση επέκτασης <<extends>> μπορεί επίσης να χρησιμοποιηθεί για να δείξει και την σχέση γενίκευσης.

- Σχεδιάζεται από μια περίπτωση χρήσης B σε μια περίπτωση χρήσης A για να δείξει ότι η περίπτωση χρήσης B επεκτείνει τη λειτουργική συμπεριφορά της γενικότερης περίπτωσης χρήσης A.

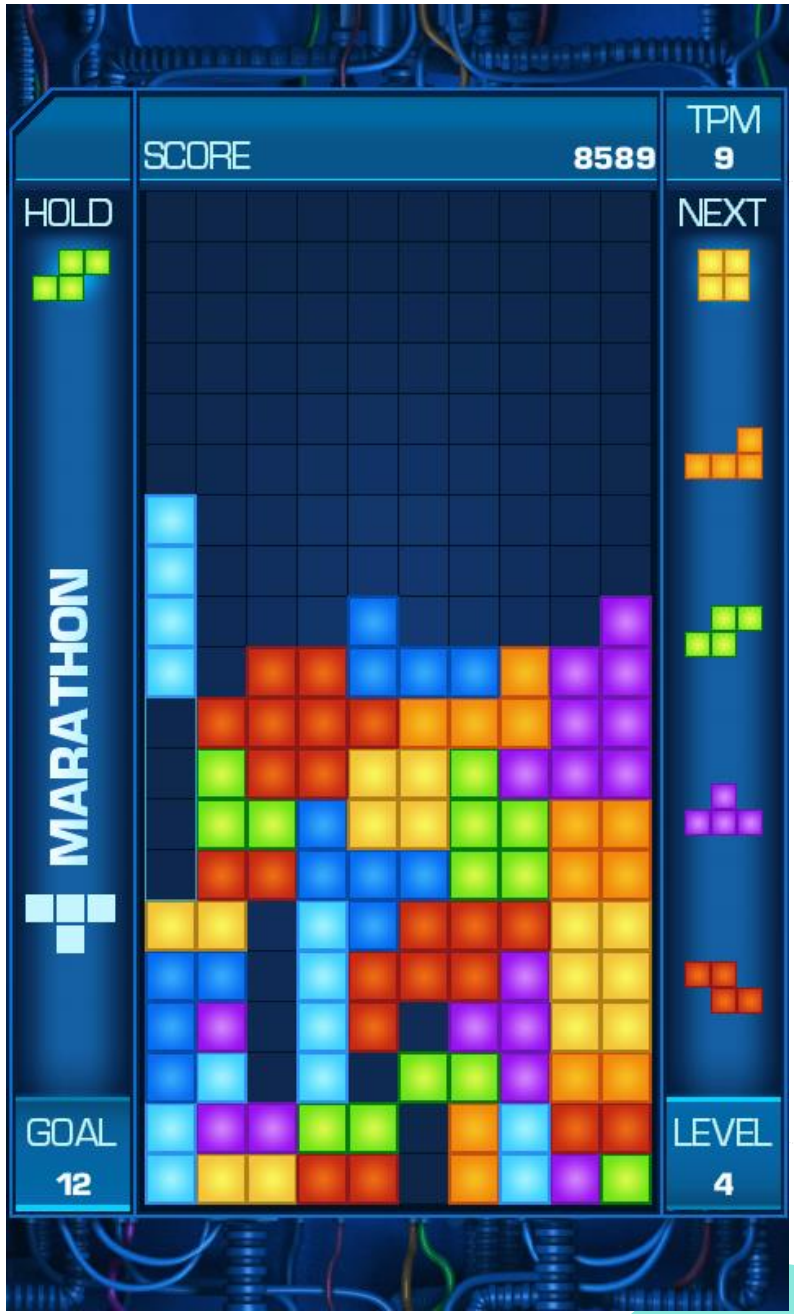
Ωστόσο, στις περισσότερες περιπτώσεις αυτή η συμπεριφορά θα μπορούσε επίσης να καταδειχθεί οπτικά με μια σημειογραφία κληρονομικότητας μεταξύ των περιπτώσεων χρήσης, το κενό τρίγωνο από τις γενικές στις εξειδικευμένες περιπτώσεις χρήσης.

BREAKOUT SESSION

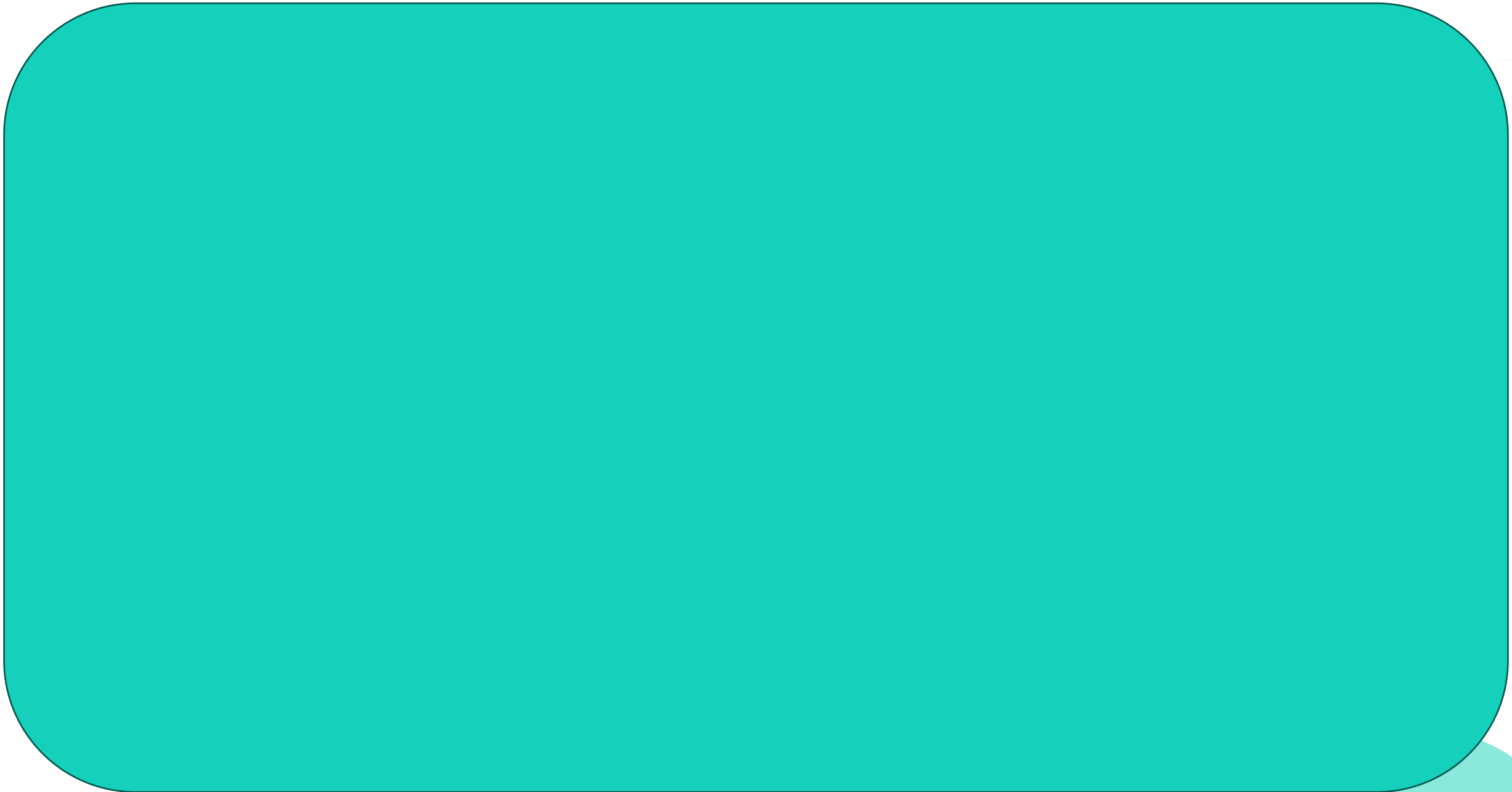
DR AVGOUSTA KYRIAKIDOU

TETRIS GAME

- In groups of 3-4
- **Draw** a use-case diagram for the Tetris game
- Add more design detail to your previous diagram with includes and extends



TETRIS GAME SAMPLE USE CASE DIAGRAM



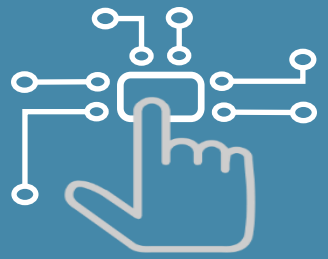
ΤΕΚΜΗΡΙΩΣΗ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ – ΣΕΝΑΡΙΑ (SCENARIOS)

Τα **σενάρια (use case scenarios)** είναι μια ακολουθία βημάτων για το πώς μπορεί να χρησιμοποιηθεί μια συγκεκριμένη λειτουργικότητα του συστήματος από τον τελικό χρήστη (ή, σε ορισμένες περιπτώσεις, από άλλο σύστημα λογισμικού).

- **Περιλαμβάνεται ως μέρος της περιγραφής περιπτώσεων χρήσης**
  - Κάθε περίπτωση χρήσης πρέπει να περιγράφεται σε ένα έγγραφο ροής γεγονότων ή σεναρίου.
  - Το scenario ορίζει τι πρέπει να κάνει το σύστημα όταν ο δράστης ξεκινά μια περίπτωση χρήσης.

ΠΕΡΙΕΧΟΜΕΝΑ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ

- Τα βήματα των περιπτώσεων χρήσης περιγράφονται με απλές καταφατικές και σύντομες προτάσεις.
- Διατυπώνουν με ακρίβεια για το τι κάνει το σύστημα και τι ο πρωτεύων actor.
- Δεν περιγράφεται το πώς δουλεύει το σύστημα αλλά μόνο το τι κάνει.
- Δεν περιγράφονται στοιχεία της διεπαφής χρήστη, όπως και άλλα στοιχεία που αφορούν τη σχεδίαση του λογισμικού.



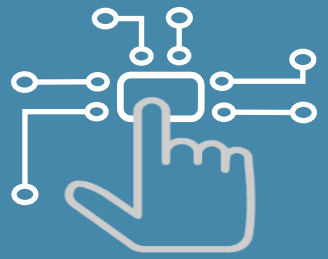
Documenting Use-Cases
Πως δημιουργούμε τα σενάρια

Για κάθε περίπτωση χρήσης, καταγράψτε την περιγραφή της. Αυτό περιλαμβάνει την παροχή των ακόλουθων πληροφοριών

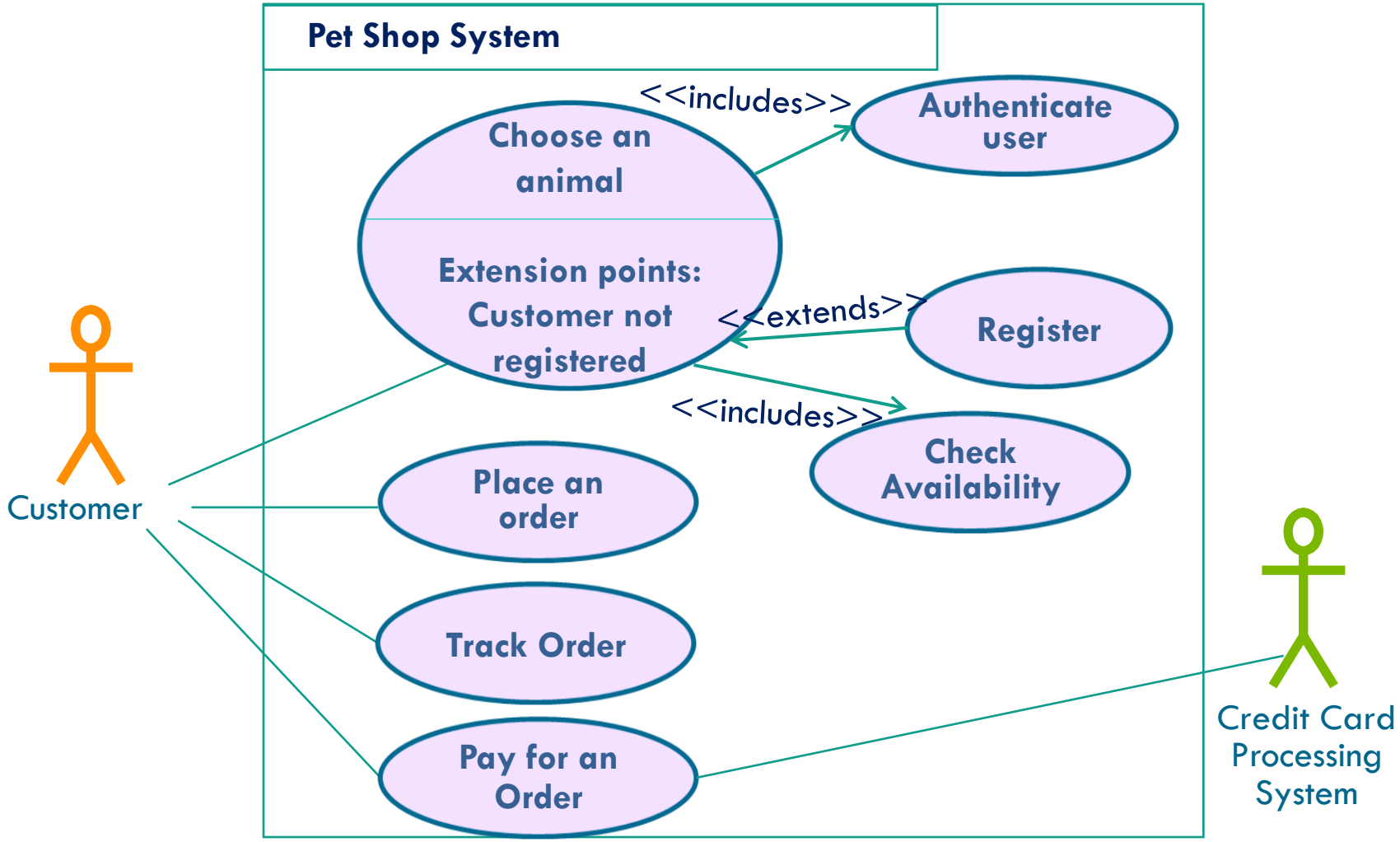
1. **Use-case name**
2. **Actors** involved (Primary actors, αυτούς που ενεργοποιούν/initiate το use case)
3. **Brief description (Σύντομη περιγραφή)**
4. **Typical course of events** περιλαμβάνει την κύρια ροή γεγονότων (δράση του actor και απόκριση του συστήματος) (actor action and system response)
5. **Alternative course of events** εναλλακτικές ροές που είναι εναλλακτικές επιτυχημένες ή αποτυχημένες ροές εκτέλεσης της περίπτωσης χρήσης.
6. **Preconditions** προϋποθέσεις που είναι απαραίτητες για την έναρξη της περίπτωσης χρήσης.
7. **Post conditions** μετα-συνθήκες που καθορίζουν την κατάσταση του συστήματος μετά το τέλος της περίπτωσης χρήσης.
8. **Assumptions** – οποιεσδήποτε άλλες εικασίες μπορεί να κάνουμε

DOCUMENTING USE CASES - WRITING THE SCENARIO

Use case Name:		
Actors:		
Description:		
Typical course of events:	Actor Action:	System Response:
Alternate course of events:		
Precondition:		
Postcondition:		
Assumptions		



Example
Pet Shop



Use case name	Choose an animal	
Actors	Customer	
Description	This describes how a customer can select a particular animal from the website	
Typical course of events	**Actor Action** **Step 1:**This use-case is initiated by a customer **Step 3:** Customer provides username/password **Step 5:** Customer selects an animal from the list **Step 7:** Customer receives confirmation	**System response** **Step 2:** Invoke <<includes>> use-case Authorise User. **Step 4:** If correct, Customer is presented with a list of animals **Step 6:** Invoke <<includes>> use case check availability
Alternate courses	Step 2: If the customer is not already registered, then Invoke <<extends>> use-case, Register. Step 4: If username/password incorrect, then inform the customer and start the process again Step 6: If the customer chooses an animal that is not available, then a notification is sent to the customer	
Precondition	This use-case is initiated by customers	
Post condition	confirmation	
Assumptions	None at this time	

ΠΩΣ ΝΑ ΚΑΤΑΣΚΕΥΑΣΕΤΕ ΈΝΑ ΔΙΑΓΡΑΜΜΑ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ

Step 1:

Προσδιόρισε ποιοι είναι οι actors.

Για κάθε actor, καθόρισε όλους τους σημαντικούς στόχους που έχουν και τους οποίους θα υποστηρίζει το σύστημα.

Step 2:

Δημιούργησε μια περίπτωση χρήσης για κάθε στόχο (λειτουργικές απαιτήσεις).

Step 3:

Δημιούργησε το διάγραμμα περίπτωσης χρήσης για να απεικονίσεις τι θα κάνει το σύστημα και ποιο είναι το όριο (system scope and boundaries).

HOW TO BUILT A USE-CASE DIAGRAM(CONT..)

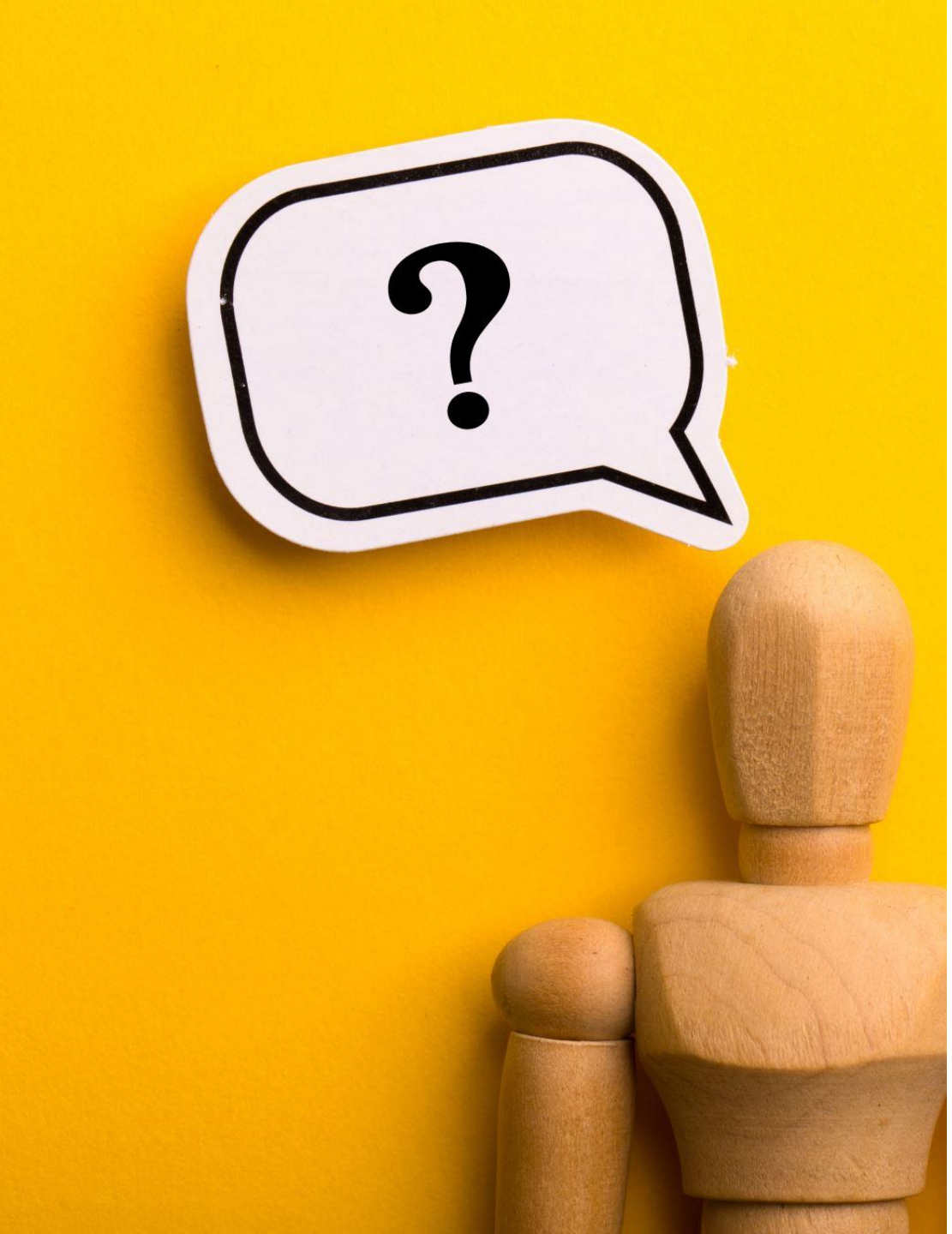
Step 4:

- Για κάθε περίπτωση χρήσης που εντοπίζεται, καταγράψτε το σενάριό της (τα γενικά βήματα - τυπική πορεία και εναλλακτικές πορείες δράσης).
  - Στόχος είναι η γρήγορη τεκμηρίωση όλων των περιπτώσεων χρήσης προκειμένου να καθοριστούν και να επικυρωθούν οι απαιτήσεις.
  - Αυτή η αρχική έκδοση των περίπτωσεων χρήσης ονομάζεται περίπτωση χρήσης απαιτήσεων **(requirements use case)**. Εδώ δημιουργείται το διάγραμμα περίπτωσης χρήσης ανώτατου επιπέδου (**Top level use case diagram)**

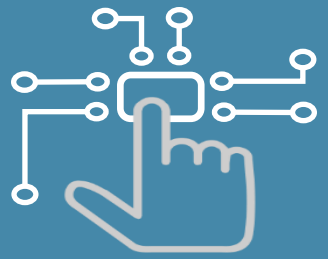
Step 5: Μόλις εγκριθούν όλες οι περιπτώσεις χρήσης απαιτήσεων

- Βελτιώστε κάθε περίπτωση χρήσης για να συμπεριλάβετε περισσότερες πληροφορίες προκειμένου να προσδιορίσετε τη λειτουργικότητα του συστήματος με μεγαλύτερη λεπτομέρεια.
- Χρησιμοποιήστε τις σχέσεις "extends" και "includes" για να προσθέσετε περισσότερες λεπτομέρειες σχεδιασμού
- Σε αυτό το βήμα ενδέχεται να χρειαστεί να δημιουργηθούν νέες περιπτώσεις χρήσης.

– Οι περιπτώσεις χρήσης που προκύπτουν ονομάζονται περιπτώσεις χρήσης ανάλυσης (**analysis use cases).**

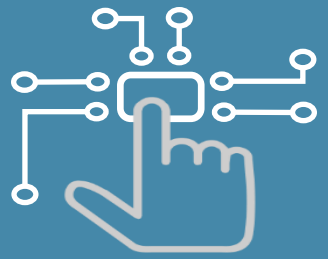


BREAKOUT SESSION



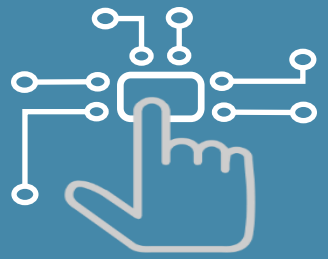
Exercise 1
University online Library system Case Study1

- Κάθε φοιτητής(student) μπορεί να δανειστεί έως και 3 βιβλία ταυτόχρονα.
- Οι φοιτητές μπορούν να περιηγηθούν σε έναν ηλεκτρονικό κατάλογο βιβλίων (browse an online catalogue for books).
- Οι δανεισμοί(Loans) πρέπει να καταγράφονται για συγκεκριμένα αντίτυπα βιβλίων και όχι απλώς για τους τίτλους τους, καθώς μπορεί να υπάρχουν πολλαπλά αντίτυπα του ίδιου βιβλίου.
- Εάν όλα τα αντίτυπα ενός βιβλίου (book copies) είναι δανεισμένα, ο φοιτητής μπορεί να κάνει κράτηση.
- Η κράτηση(reservation) καταχωρίζεται στον τίτλο του βιβλίου. Ένας φοιτητής μπορεί επίσης να επεκτείνει τον δανεισμό του, εφόσον το βιβλίο δεν έχει ήδη κρατηθεί από άλλον.
- Όταν ένα αντίτυπο του συγκεκριμένου βιβλίου γίνει διαθέσιμο, ο βιβλιοθηκάριος το δεσμεύει(reserve) για τον φοιτητή και ο φοιτητής ειδοποιείται ότι το αντίτυπό του είναι έτοιμο για παραλαβή.



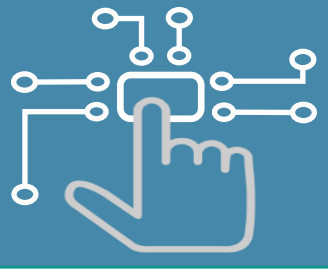
Exercise 1
University online Library system Case Study1

- Κάθε **φοιτητής(student)** μπορεί να δανειστεί έως και 3 βιβλία ταυτόχρονα.
- Οι φοιτητές μπορούν να περιηγηθούν σε έναν ηλεκτρονικό κατάλογο βιβλίων (browse an online catalogue for books).
- Οι δανεισμοί(Loans) πρέπει να καταγράφονται για συγκεκριμένα αντίτυπα βιβλίων και όχι απλώς για τους τίτλους τους, καθώς μπορεί να υπάρχουν πολλαπλά αντίτυπα του ίδιου βιβλίου.
- Εάν όλα τα αντίτυπα ενός βιβλίου (book copies) είναι δανεισμένα, ο φοιτητής μπορεί να κάνει κράτηση.
- Η κράτηση(reservation) καταχωρίζεται στον τίτλο του βιβλίου. Ένας φοιτητής μπορεί επίσης να επεκτείνει τον δανεισμό του, εφόσον το βιβλίο δεν έχει ήδη κρατηθεί από άλλον.
- Όταν ένα αντίτυπο του συγκεκριμένου βιβλίου γίνει διαθέσιμο, ο **βιβλιοθηκάριος** το δεσμεύει(reserve) για τον φοιτητή και ο φοιτητής ειδοποιείται ότι το αντίτυπό του είναι έτοιμο για παραλαβή.



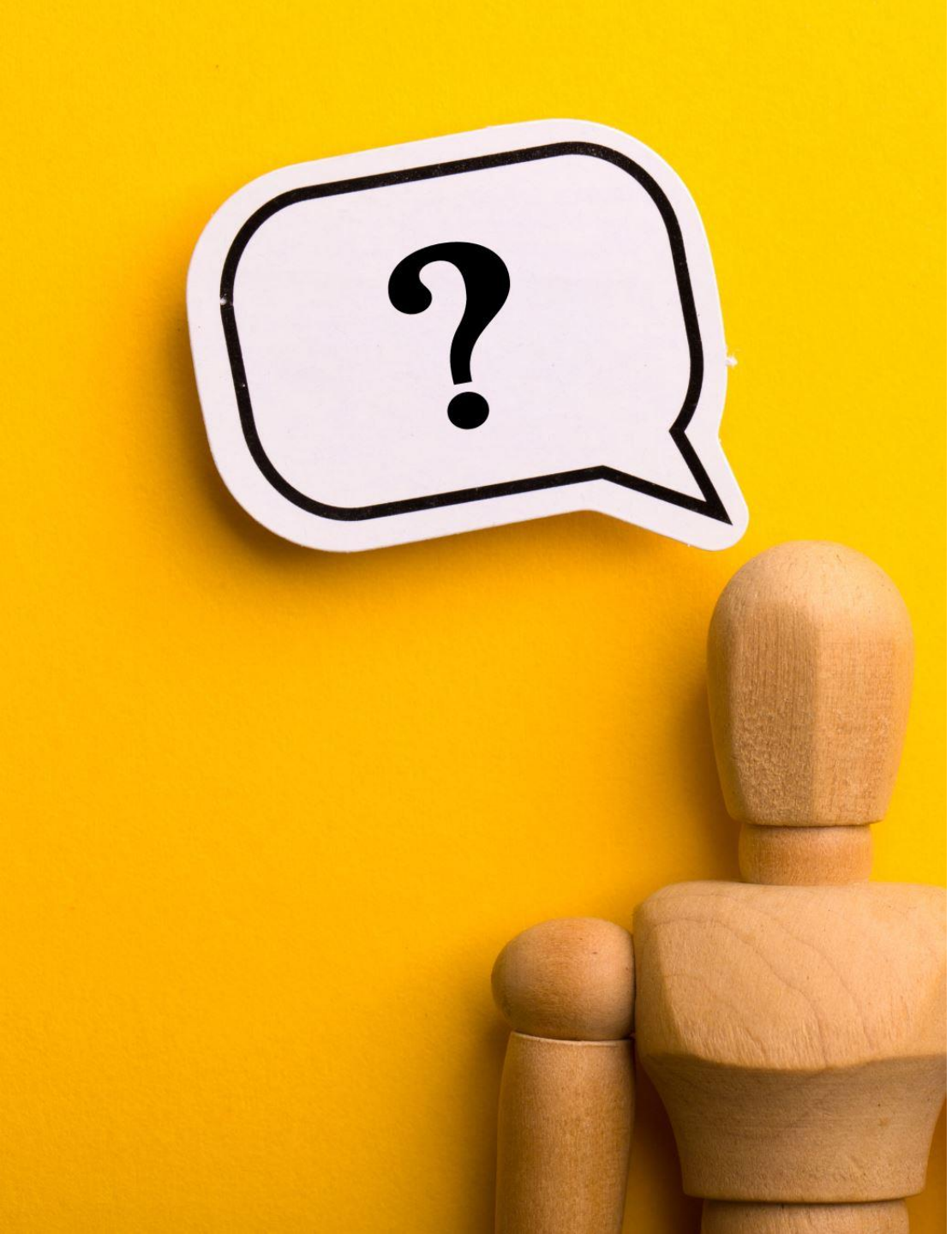
Exercise 1
University online Library system Case Study1

- Κάθε **φοιτητής(student)** μπορεί να **δανειστεί (borrow)** έως και 3 βιβλία ταυτόχρονα.
- Οι φοιτητές μπορούν να περιηγηθούν σε έναν ηλεκτρονικό κατάλογο βιβλίων (**browse** an **online catalogue** for books).
- Οι δανεισμοί(Loans) πρέπει να καταγράφονται για συγκεκριμένα αντίτυπα βιβλίων και όχι απλώς για τους τίτλους τους, καθώς μπορεί να υπάρχουν πολλαπλά αντίτυπα του ίδιου βιβλίου.
- Εάν όλα τα αντίτυπα ενός βιβλίου (book copies) είναι δανεισμένα, ο φοιτητής μπορεί να κάνει **κράτηση (place a reservation).**
- Η κράτηση(reservation) καταχωρίζεται στον τίτλο του βιβλίου. Ένας φοιτητής μπορεί επίσης να **επεκτείνει τον δανεισμό του (extend their loan)**, εφόσον το βιβλίο δεν έχει ήδη κρατηθεί από άλλον.
- Όταν ένα αντίτυπο του συγκεκριμένου βιβλίου γίνει διαθέσιμο, ο **βιβλιοθηκάριος** το **δεσμεύει(a copy is reserved)** για τον φοιτητή και ο **φοιτητής ειδοποιείται(student is notified)** ότι το αντίτυπό του είναι έτοιμο για παραλαβή.

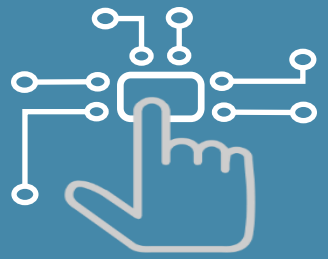


University online Library system
Sample solution

Solution ?

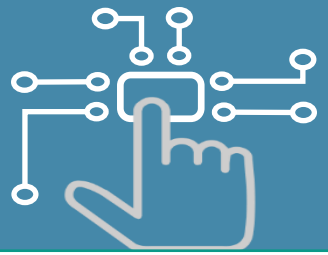


BREAKOUT SESSION



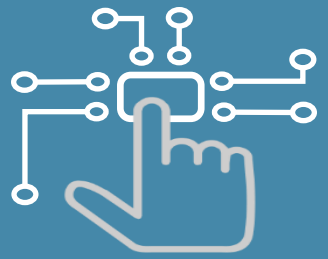
Exercise 2
Something for Everyone – SFE Case Study

- SFE is a shop which sells various products.
- **Customers** can purchase products from clerks.
- In order to **make a sale**, **clerks** scan items, calculate the total and obtain the payment from the customer.
- A **payment** can be done by cash or credit card.
- A customer can refund a purchase. For this to happen the clerk will ask for the receipt and will refund the payment (cash or credit).



Example – SFE

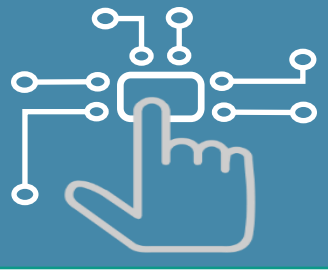
Solution ?



Example – SFE

SFE System

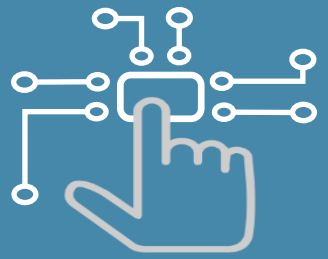
Solution ?



Example – SFE

SFE System

Solution ?



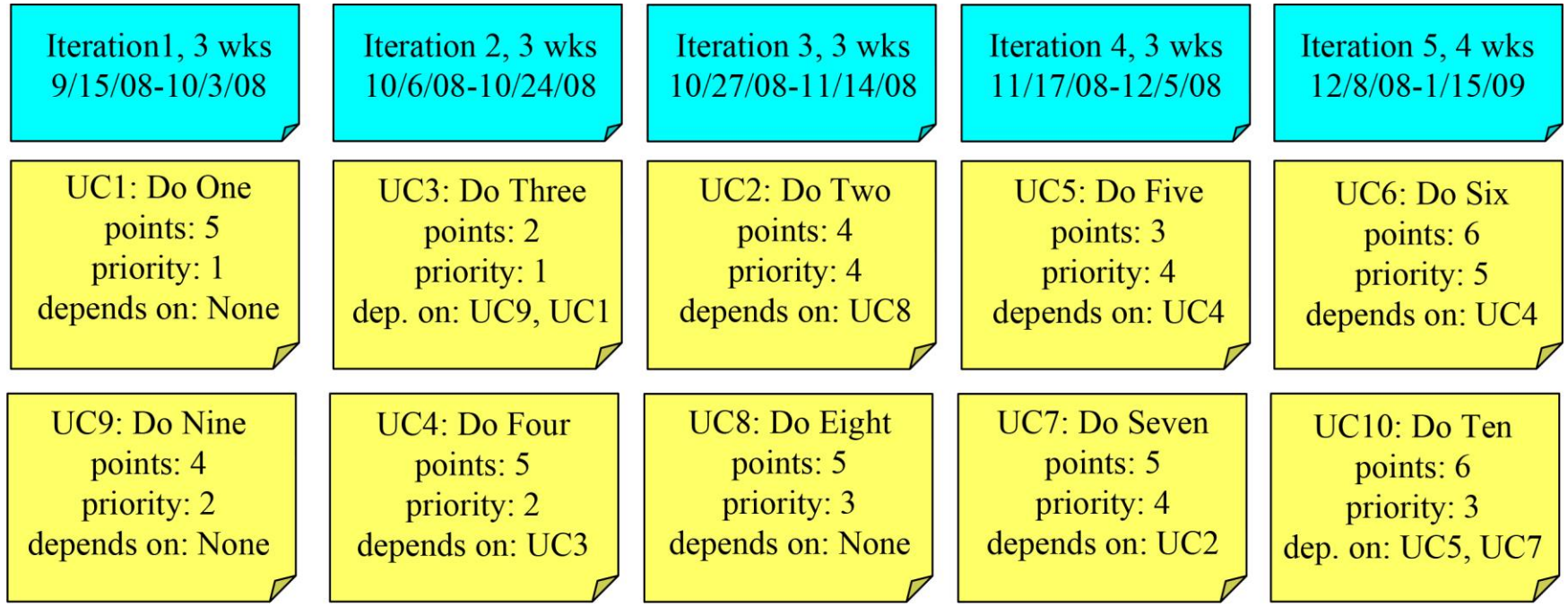
Ποτέ χρησιμοποιούμε Use-Cases?

- Με λίγα λόγια, πάντα!!!
- Οι απαιτήσεις είναι το πιο δύσκολο κομμάτι της ανάπτυξης λογισμικού
- Τα διαγράμματα περιπτώσεων χρήσης είναι ένα ισχυρό εργαλείο για την κατανόηση του
  - ποιοι είναι οι χρήστες (συμπεριλαμβανομένων των αλληλεπιδρώντων συστημάτων)
  - Ποιες λειτουργίες πρέπει να παρέχει το σύστημα
  - Πώς λειτουργούν αυτές οι λειτουργίες σε υψηλό επίπεδο

User-centred design

Να αφιερώνετε πάντοτε επαρκή χρόνο στις απαιτήσεις και στη φάση της επεξεργασίας τους

ΕΥΕΛΙΚΤΟΣ ΣΧΕΔΙΑΣΜΟΣ - AGILE PLANNING



- Οι ευέλικτες ομάδες χρησιμοποιούν διαδραστικά, ανεπίσημα μέσα για την εκτέλεση του ευέλικτου σχεδιασμού (agile planning).

APPLYING AGILE PRINCIPLES

1. Συνεργαστείτε στενά με τους πελάτες και τους χρήστες για να κατανοήσετε τις επιχειρηματικές διαδικασίες και τις προτεραιότητές τους και να τους βοηθήσετε να προσδιορίσουν τις πραγματικές τους ανάγκες.

2. Τα μέλη της ομάδας θα πρέπει να συνεργαστούν για να προσδιορίσουν τις περιπτώσεις χρήσης, τους actors και τα υποσυστήματα και να καθορίσουν τις περιπτώσεων χρήσης.

3. Οι απαιτήσεις εξελίσσονται αλλά το χρονοδιάγραμμα είναι σταθερό.

4. Επικεντρωθείτε στη συχνή παράδοση μικρών προσαυξήσεων, κάθε μία από τις οποίες αναπτύσσει μόνο μερικές περιπτώσεις χρήσης.

5. Μην επιχειρήσετε να δημιουργήσετε ένα άριστο σύνολο περιπτώσεων χρήσης. Το Αρκετά καλό είναι αρκετό.

Wrap up

- Μοντελοποίηση περιπτώσεων χρήσης με UML
- Σχέσεις μεταξύ actors, use cases, includes, extends and generalisation
- Κατασκευή διαγραμμάτων περιπτώσεων χρήσης
- Δημιουργία σεναρίων.

Την επόμενη εβδομάδα: Μοντέλο αντικειμένων: Διάγραμμα κλάσεων (Conceptual Class Diagram)

ΠΩΣ ΤΑ ΠΗΓΑΜΕ ΑΥΤΗ ΤΗΝ ΕΒΔΟΜΑΔΑ? ΤΙ ΜΑΘΑΜΕ?

Reply at:
PollEv.com/avgoustakyri977

1. Go to **PollEv.com**
2. Enter **AVGOUSTAKYRI977**
3. Respond to activity

1. Text **AVGOUSTAKYRI977** to **22333**
2. Text in your message

Further Reading

1. Kung, D.C (2024) Software Engineering, 2nd Edition, McGraw Hill – Chapter 7,8

2. Pressman, R.S. (2020). Software Engineering: A practitioner's approach, 9th Edition, Mc-Graw Hill – Chapter 8