- ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ ΜΗΧΑΝΙΚΗΣ ΛΟΓΙΣΜΙΚΟΥ
- ΠΡΟΣΧΕΔΙΑΣΜΕΝΟΣ & ΕΥΕΛΙΚΤΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

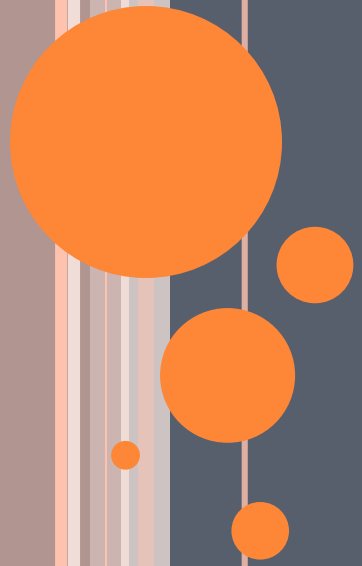
- ADVANCED SOFTWARE ENGINEERING
- PLAN-DRIVEN & AGILE PROGRAMMING

Prerequisite & Desirable Courses

- *Procedural Programming (C),*
- *Principles of Software Engineering*
- *OOD*
- *Data Bases,*
- *HCI*

Περιεχομενα

1. *Modeling Languages*
2. *The UML*



Modeling Languages

Modelling Languages
Definition

Modelling Language

- *A modeling language is any artificial language that can be used to express information or knowledge or systems in a structure that is defined by a consistent set of rules. The rules are used for interpretation of the meaning of components in the structure.*

Modelling Languages
Definition

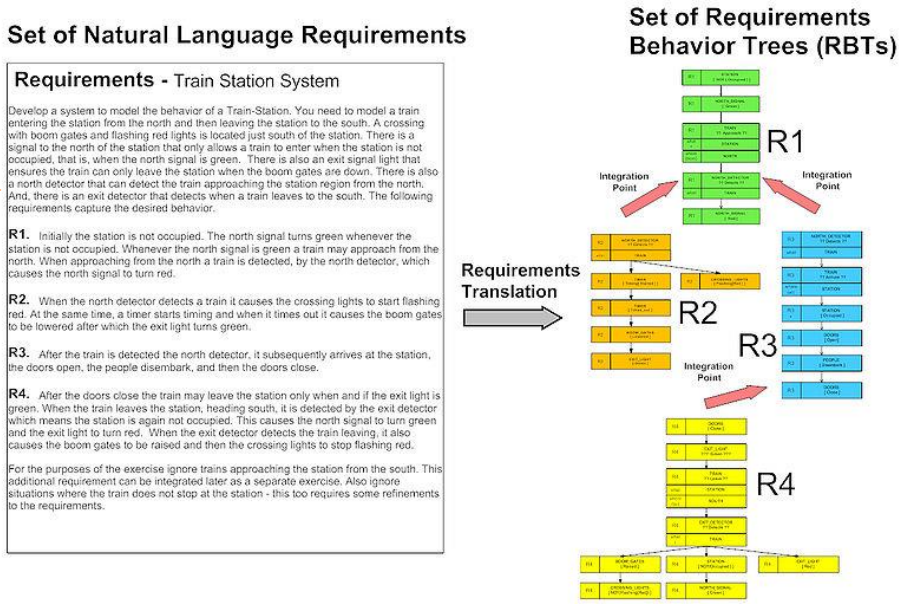
A modeling language can be **graphical** or **textual**.

- ***Graphical*** modeling languages use a diagram technique with named symbols that represent concepts and lines that connect the symbols and represent relationships and various other graphical annotation to represent constraints.
- ***Textual*** modeling languages typically use standardized keywords accompanied by parameters to make computer-interpretable expressions.

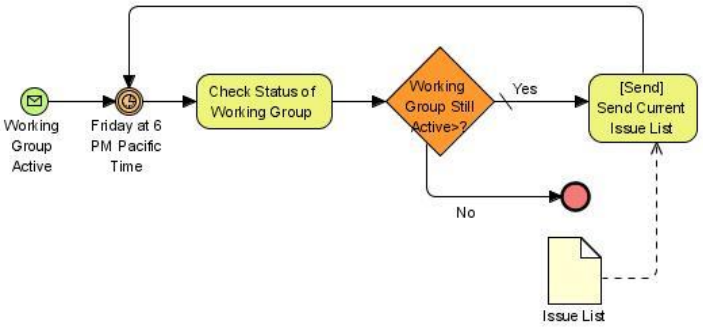
Modelling Languages

Graphical modelling languages

Behavior Trees
are a formal, graphical modelling language used primarily in systems and software engineering.



Business Process Modeling Notation (BPMN, and the XML form BPML) is an example of a Process Modeling language.



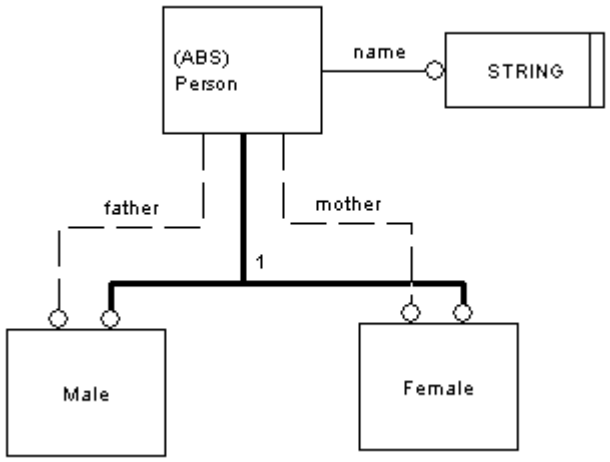
Modelling Languages

Graphical modelling languages

EXPRESS and EXPRESS-G (ISO 10303-11) is an international standard general-purpose data modeling language.

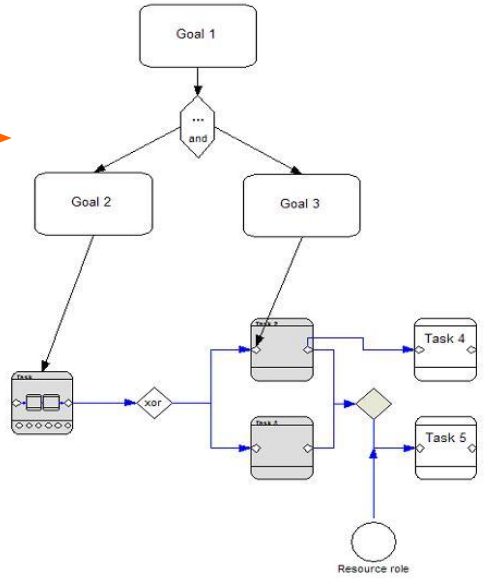
Extended Enterprise Modeling Language (EEML) is commonly used for business process modeling across a number of layers.

```
SCHEMA Family;
ENTITY Person
  ABSTRACT SUPERTYPE OF (ONEOF (Male, Female));
    name: STRING;
    mother: OPTIONAL Female;
    father: OPTIONAL Male;
END_ENTITY;

ENTITY Female
  SUBTYPE OF (Person);
END_ENTITY;

ENTITY Male
  SUBTYPE of (Person);
END_ENTITY;

END_SCHEMA;
```

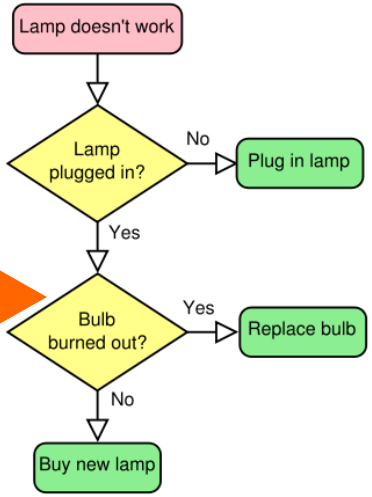


Modelling Languages
Graphical modelling languages

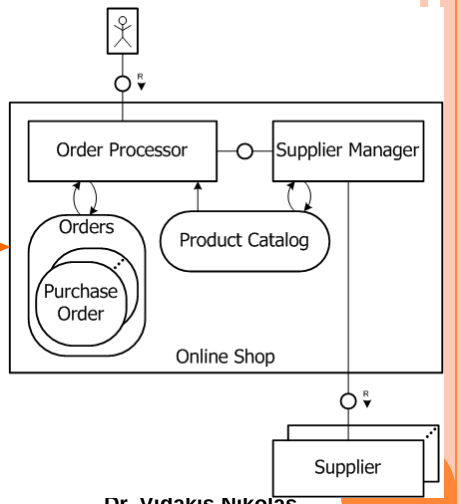
Extended Enterprise Modeling Language (EEML) is commonly used for business process modeling across a number of layers.



Flowchart is a schematic representation of an algorithm or a stepwise process,



Fundamental Modeling Concepts (FMC) modeling language for software-intensive systems.

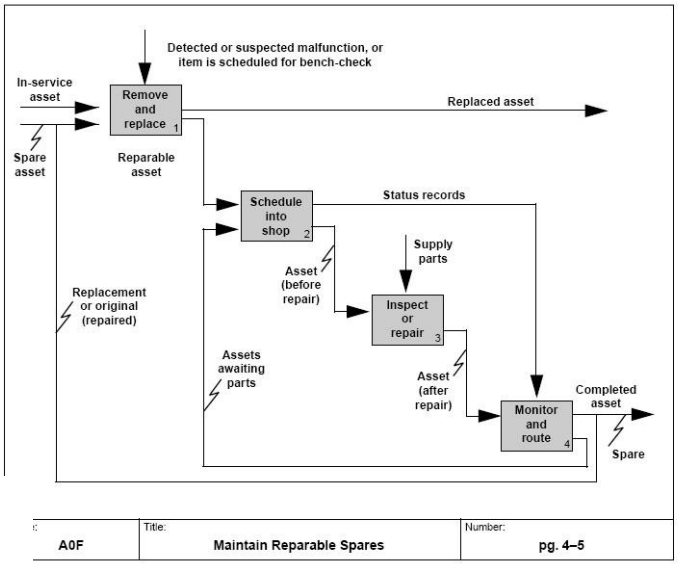


Modelling Languages

Graphical modelling languages

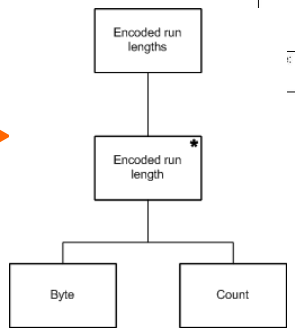
IDEF

is a family of modeling languages, which include IDEF0 for functional modeling, IDEF1X for information modeling, IDEF3 for business process modelling, IDEF4 for Object-Oriented Design and IDEF5 for modeling ontologies.



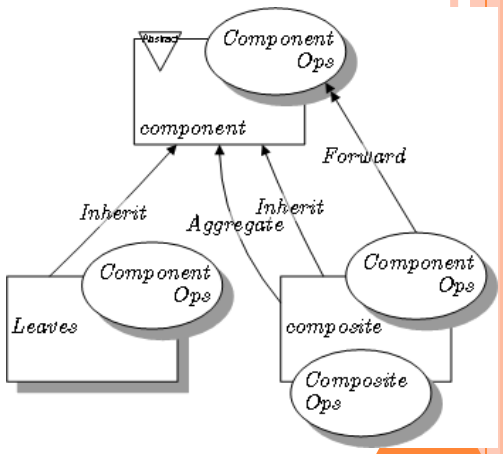
Jackson Structured Programming

(JSP) is a method for structured programming based on correspondences between data stream structure and program structure



LePUS3

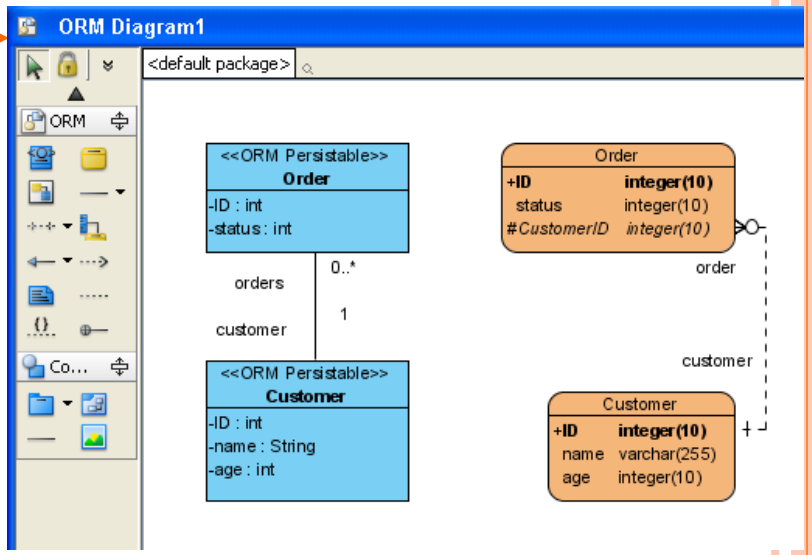
is an object-oriented visual Design Description Language and a formal specification language that is suitable primarily for modelling large object-oriented (Java, C++, C#) programs and design patterns.



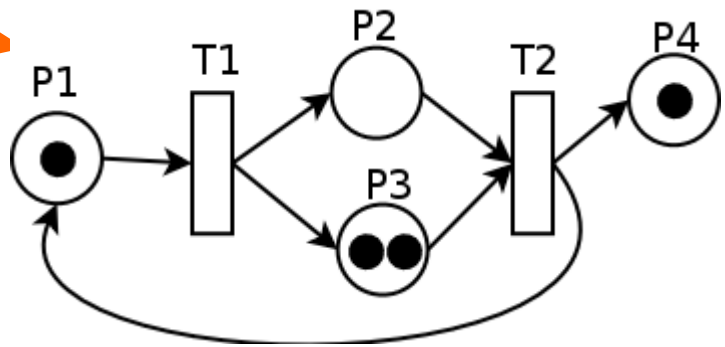
Modelling Languages

Graphical Modelling Languages

Object Role Modeling (ORM) in the field of software engineering is a method for conceptual modeling, and can be used as a tool for information and rules analysis.



Petri nets use variations on exactly one diagramming technique and topology, namely the bipartite graph. The simplicity of its basic user interface easily enabled extensive tool support over the years, particularly in the areas of model checking, graphically-oriented simulation, and software verification.

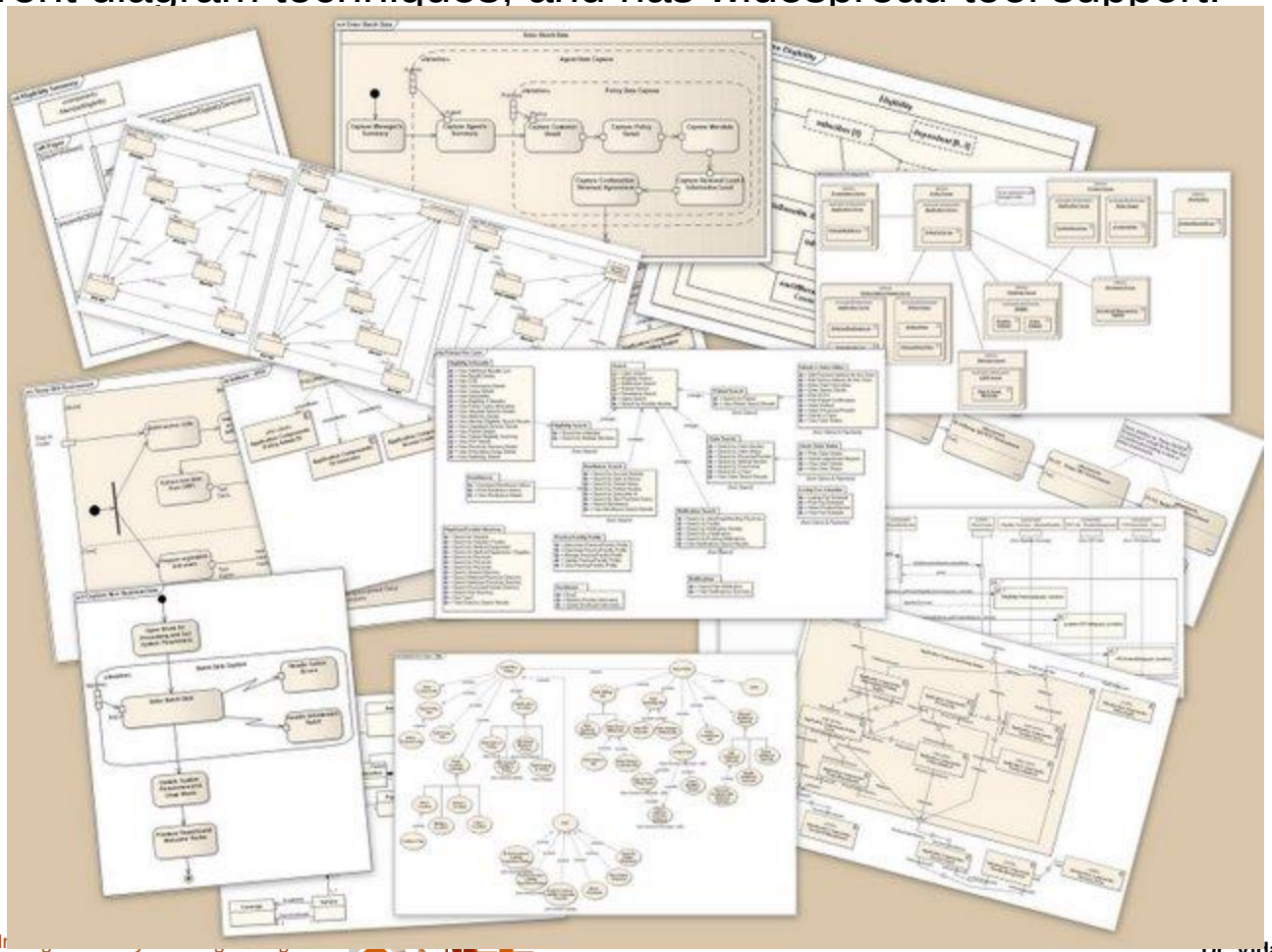


UML

UML Graphical modelling languages

Unified Modeling Language (UML)

is a general-purpose modeling language that is an industry standard for specifying software-intensive systems. UML 2.0, the current version, supports thirteen different diagram techniques, and has widespread tool support.



UML – What is it

Τι είναι η UML (Unified Modeling Language)

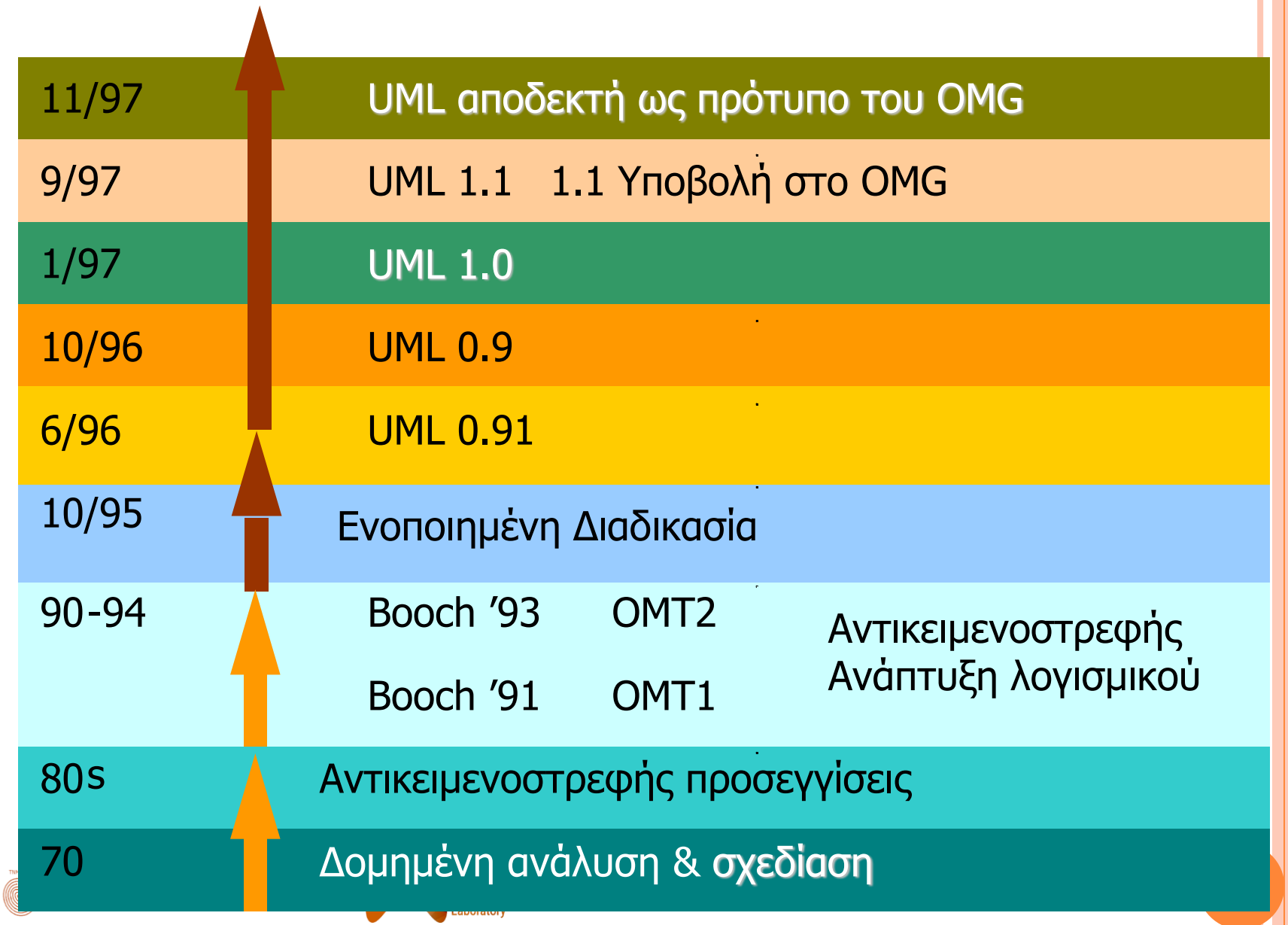
Η UML είναι μια γλώσσα για:

- Κατάρτιση προδιαγραφών λογισμικού και τεκμηρίωση τμημάτων λογισμικού
- Αναπαράσταση με οπτικό τρόπο (visualization) τμημάτων λογισμικού
- Μοντελοποίηση εταιρικών και άλλων συστημάτων που δεν αφορούν λογισμικό

UML – Overview

- Η UML είναι μια γλώσσα οπτικού σχεδιασμού
  - Οπτικός Σχεδιασμός: Η αντιστοιχία των διαδικασιών του πραγματικού κόσμου ενός υπολογιστικού συστήματος με μία γραφική αναπαράσταση ή με ένα σχέδιο
- Η UML ως γλώσσα απαρτίζεται από
  - Στοιχεία μοντέλων
    - Οντότητες
    - Γνωρίσματα
    - Μεθόδους
    - Συσχετίσεις
    - Τύπους οντοτήτων
  - Σύμβολα για την απόδοση των στοιχείων των μοντέλων
  - Οδηγίες πρακτικής και ανάπτυξης μοντέλων
- Ευρύ φάσμα εφαρμογής
  - Προσδιορισμό προδιαγραφών συστήματος
  - Ανάλυση & σχεδιασμό αντικειμενοστραφών συστημάτων λογισμικού
  - Δημιουργία & τεκμηρίωση τμημάτων ενός συστήματος λογισμικού

UML – Flashback

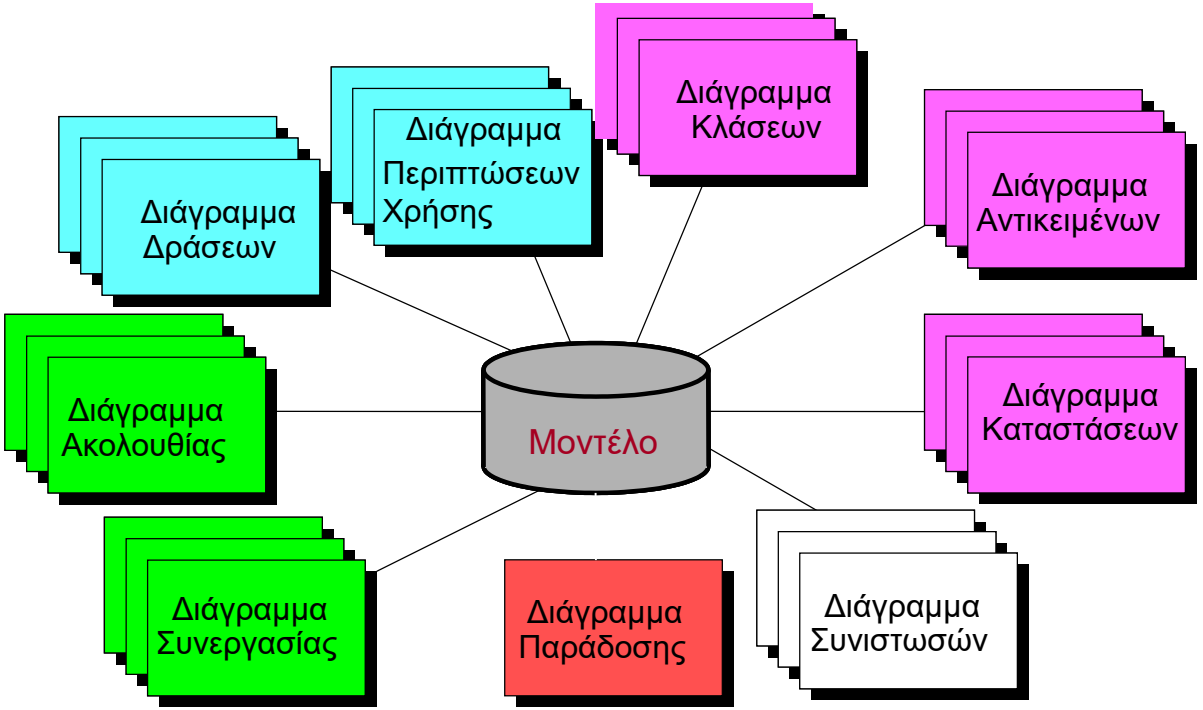


UML – Why

- Από το 1997 είναι ανοικτό πρότυπο (Open Standard) του Object Management Group (OMG) & χρησιμοποιείται ευρέως τόσο από την βιομηχανία όσο και από την ακαδημαϊκή & ερευνητική κοινότητα
- Προάγει τη κατανόηση και την επικοινωνία μεταξύ εμπλεκομένων (π.χ. χρήστες, σχεδιαστές & αναλυτές συστημάτων, προγραμματιστές)
- Ενσωματώνει πολλά από τα στοιχεία που απαιτεί μια ποιοτική διαδικασία ανάπτυξης λογισμικού
  - Πολλαπλές όψεις του συστήματος
  - Περιγραφή της χρήσης του λογισμικού
  - Τρόπο που «δουλεύει» το λογισμικό
  - Πως πρέπει να αναπτυχθεί
- Υποστηρίζεται από πολλά CASE εργαλεία (π.χ. Rose, ArgoUML, UMLi, Poseidon

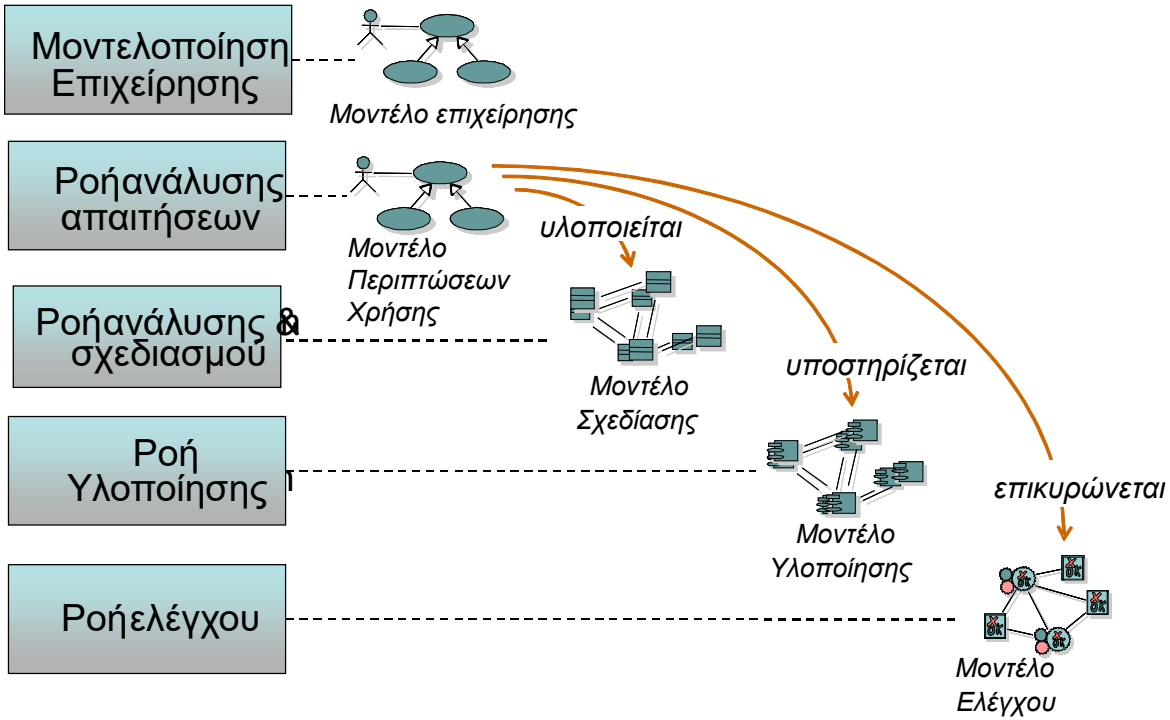
UML – Why

- Γραφική γλώσσα / τεχνοτροπία που υποστηρίζει οριοθέτηση προδιαγραφών, οπτική αναπαράσταση, κατασκευή, καταγραφή και τεκμηρίωση συστημάτων λογισμικού



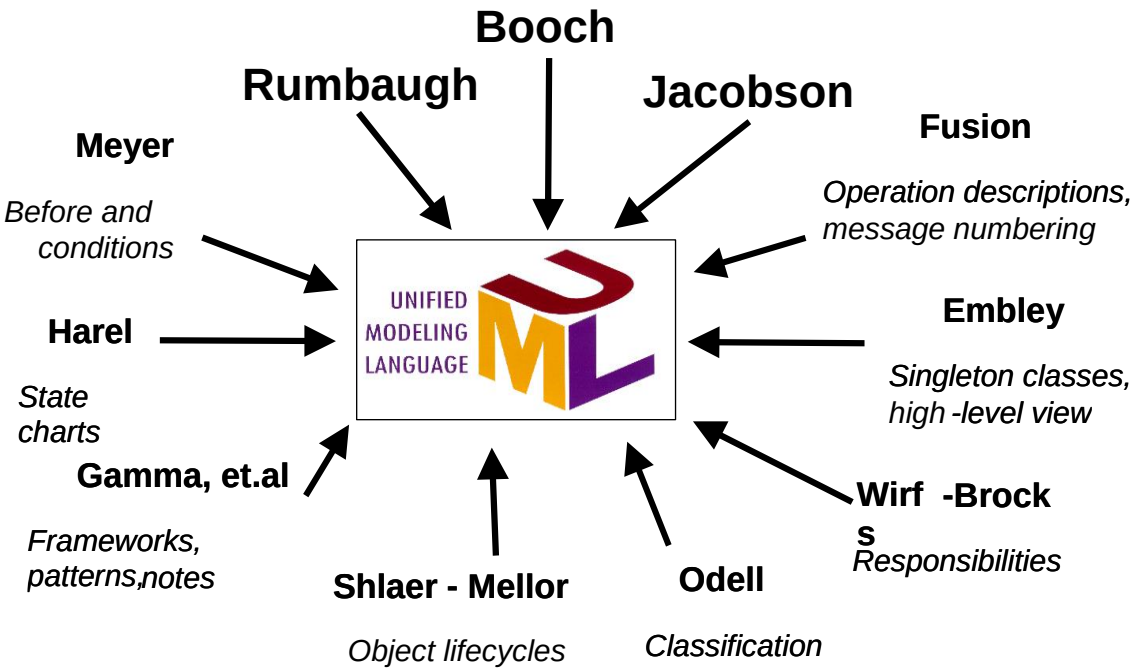
UML – Why

- Ευρεία χρήση της γλώσσας από αρχικά στάδια του κύκλου ζωής ενός προϊόντος έως τη λεπτομερή ανάλυση, τον σχεδιασμό την υλοποίηση και τον έλεγχό του



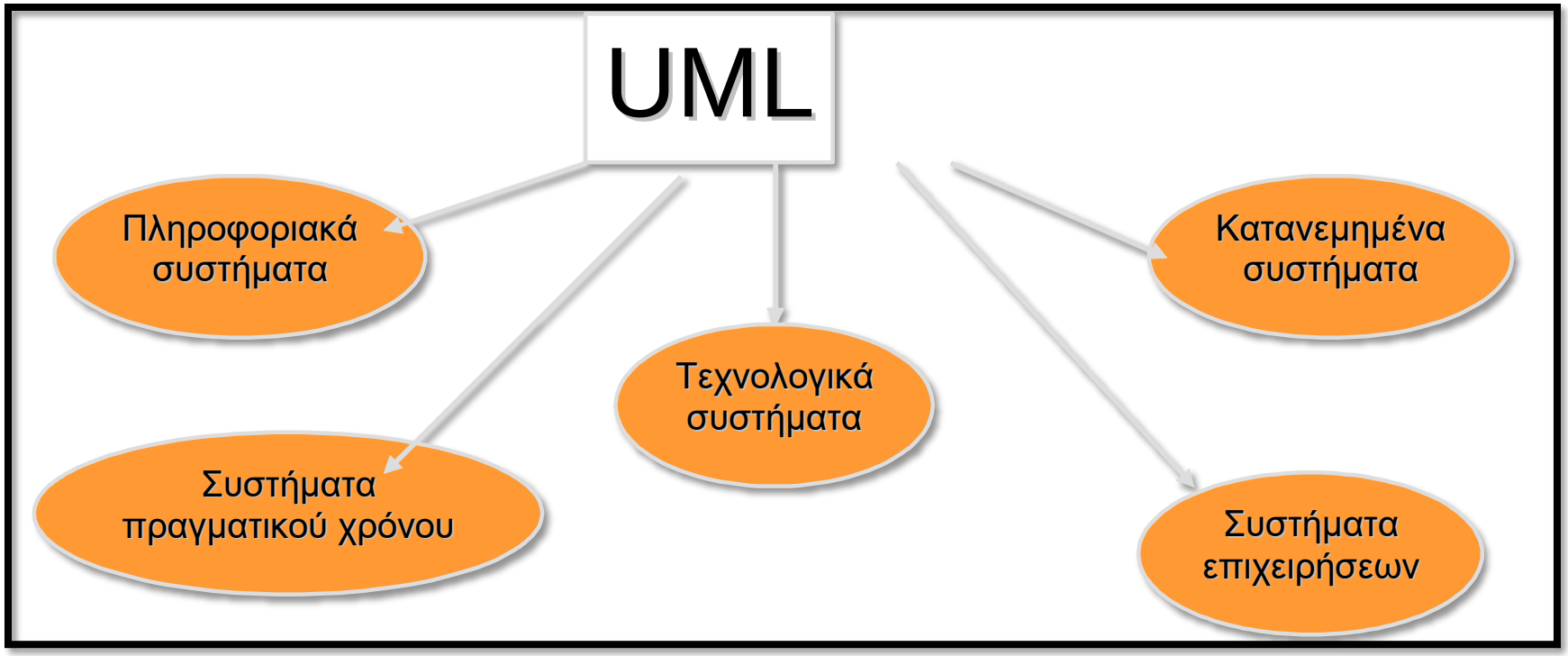
UML – Why

- Βασίζεται και προέκυψε από εμπειρία και επαφή με την κοινότητα της μηχανικής λογισμικού



- Η UML δεν είναι προαπαιτούμενο κάποιας συγκεκριμένης διαδικασίας
  - Μπορεί επομένως να χρησιμοποιηθεί για διάφορους τύπους έργων και διαφορετικές διαδικασίες ανάπτυξης

UML - Use


UML
Πληροφοριακά συστήματα
Τεχνολογικά συστήματα
Κατανεμημένα συστήματα
Συστήματα πραγματικού χρόνου
Συστήματα επιχειρήσεων

UML Aims

- Η μοντελοποίηση συστημάτων με βάση τις αρχές των αντικειμενοστραφών μοντέλων
- Η δημιουργία μιας μοντελοποιημένης γλώσσας που μπορεί να χρησιμοποιηθεί τόσο από τον άνθρωπο όσο κι από τις μηχανές

UML Abstraction

Η έννοια της Αφαίρεσης

- Εστίαση σε επιλεγμένα στοιχεία και αγνόηση υπόλοιπων λεπτομερειών
- Τα μοντέλα μπορούν να εκφραστούν σε διαφορετικά επίπεδα πιστότητας και λεπτομέρειας
- Σύνθετα τμήματα περιγράφονται καλύτερα με μικρά σύνολα ανεξάρτητων όψεων

UML Model Characteristics

Χαρακτηριστικά των Μοντέλων

- Ακρίβεια - περιγράφουν με σωστό τρόπο το σύστημα.
- Συνέπεια – διαφορετικές όψεις δεν έρχονται σε σύγκρουση μεταξύ τους.
- Διευκολύνουν την επικοινωνία
- Ευμετάβλητα
- Κατανοητά

UML - Sections

- Οψεις: Δείχνουν διαφορετικά χαρακτηριστικά του συστήματος που μοντελοποιούνται. Μια όψη αποτελείται από ένα σύνολο διαγραμμάτων
- Διαγράμματα: Περιγράφουν τα περιεχόμενα μιας όψης. Υπάρχουν εννέα διαφορετικά διαγράμματα που χρησιμοποιούνται σε συνδυασμό
- Στοιχεία μοντέλου: Είναι οι έννοιες που χρησιμοποιούνται στα διαγράμματα για να αναπαραστήσουν τις κλάσεις, τα αντικείμενα και τις μεταξύ τους συσχετίσεις
- Πρακτικές μοντελοποίησης: Η UML συνδυάζει διαφορετικές πρακτικές μοντελοποίησης
  - Data Modeling concepts (Μοντέλο Οντοτήτων –Συσχετίσεων)
  - Business Modeling (Ροές εργασίας)
  - Object Modeling (Μοντελοποίησης αντικειμένων)
  - Μοντελοποίησης συνιστωσών λογισμικού (components)

UML - What can you Model with UML?

Structure Diagrams:

- Class Diagram,
- Object Diagram,
- Component Diagram,
- Composite Structure Diagram,
- Package Diagram,
- and Deployment Diagram.

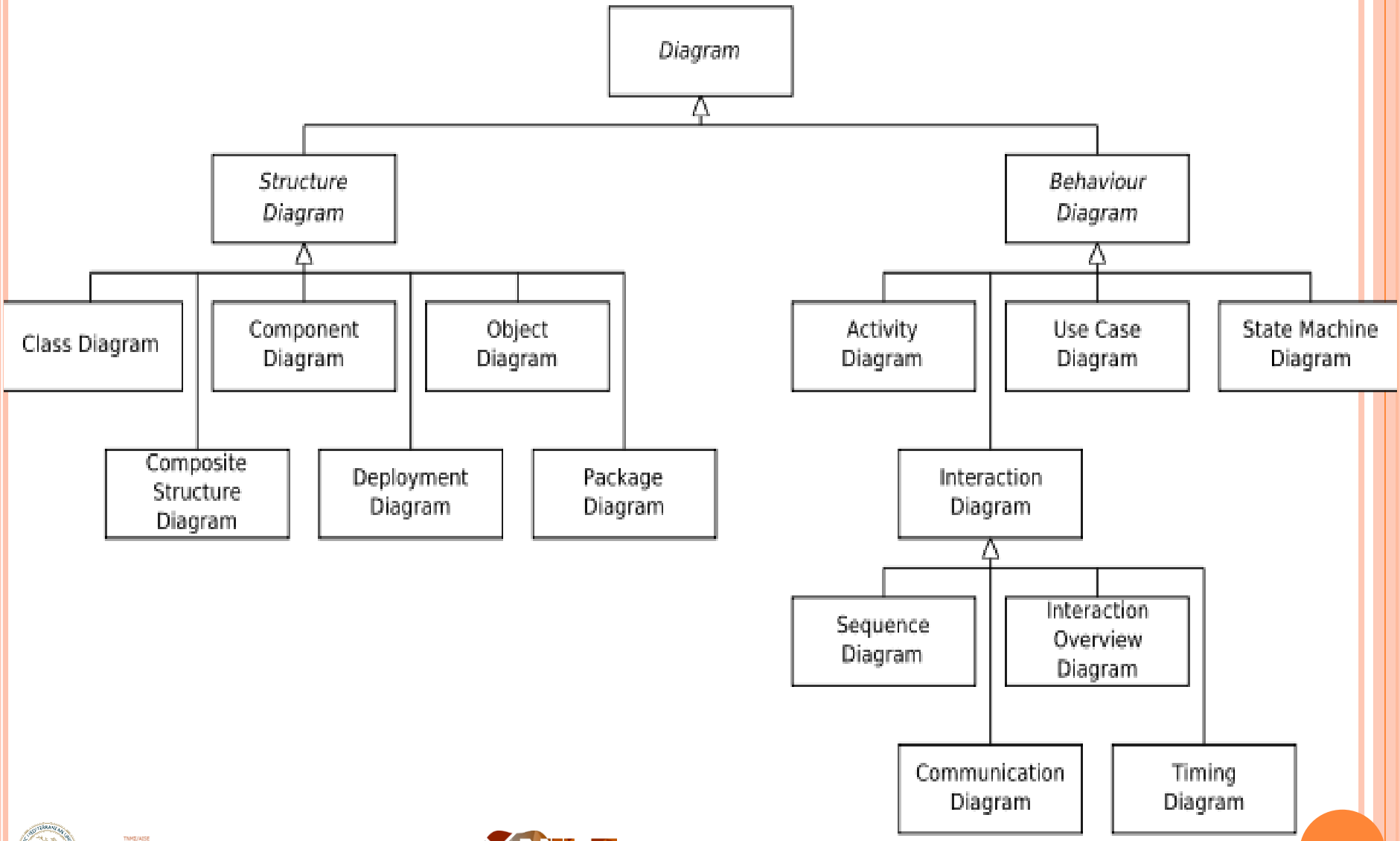
Behavior Diagrams:

- Use Case Diagram (used by some methodologies during requirements gathering);
- Activity Diagram,
- and State Machine Diagram.

Interaction Diagrams all derived from the more general Behavior Diagram,

- Sequence Diagram,
- Communication Diagram,
- Timing Diagram,
- and Interaction Overview Diagram.

UML
Diagrams Categorization Tree



UML - Structure Diagrams

Structure / Static diagrams

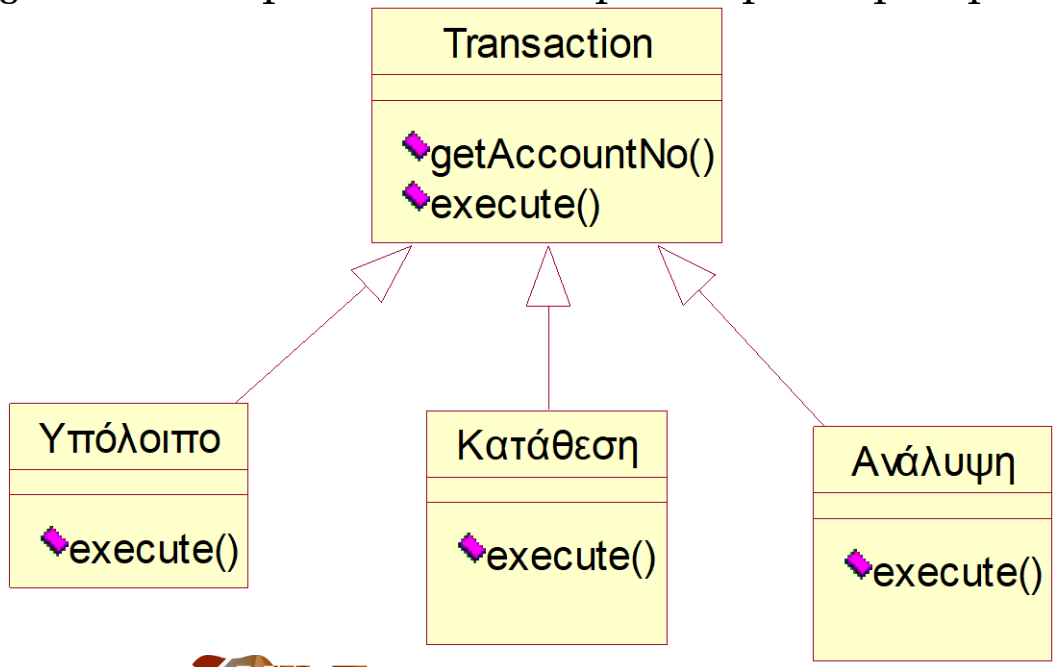
Τα διαγράμματα που ανήκουν στην κατηγορία *Structure* δίνουν έμφαση στην δομή του συστήματος , δηλαδή τι περιέχει το σύστημα

- Class diagram: describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes.
- Component diagram: depicts how a software system is split up into components and shows the dependencies among these components.
  (Δείχνει τα συστατικά μέρη του κώδικα και τις εξαρτήσεις τους)
- Composite structure diagram: describes the internal structure of a class and the collaborations that this structure makes possible.
- Deployment diagram serves to model the hardware used in system implementations, the components deployed on the hardware, and the associations between those components.
  (Δείχνει τους υπολογιστές και τις συσκευές (κόμβους) καθώς και τον τύπο των συνδέσεων)
- Object diagram: shows a complete or partial view of the structure of a modeled system at a specific time.
- Package diagram: depicts how a system is split up into logical groupings by showing the dependencies among these groupings.

UML - Structure Diagrams
Class Diagram

(see "Lecture 01 extra-UML-Class-Diagrams" file for full description of Class Diagrams)

- Παράσταση των κλάσεων που απαρτίζουν το σύστημα και των σχέσεών τους
  - Στατικό διάγραμμα
  - Παριστά μονάδες που αλληλεπιδρούν και τα γνωρίσματά τους
  - Δεν εστιάζει στο τι συμβαίνει κατά την αλληλεπίδραση

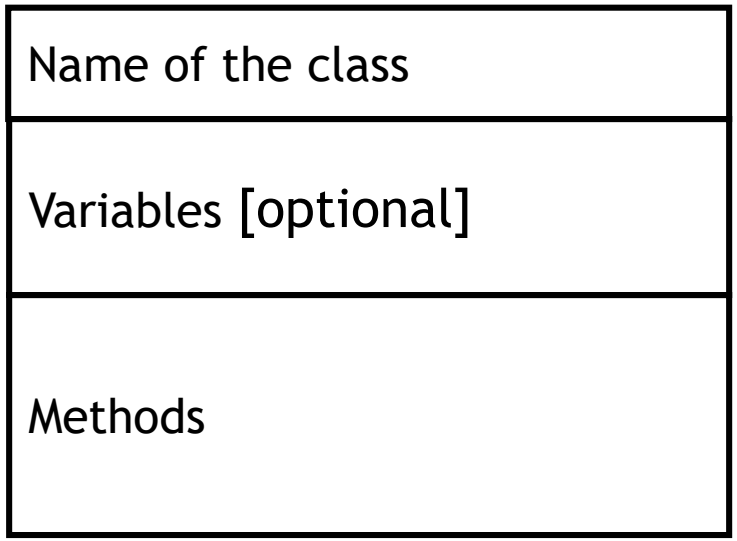


UML - Structure Diagrams
Class Diagram

- Χρήσιμο στο να κατανοηθεί η συνολική αρχιτεκτονική των συνιστωσών τμημάτων του συστήματος (system components)
- Ένα διάγραμμα κλάσης μπορεί να σχεδιαστεί από τρεις όψεις
  - Εννοιολογικό (conceptual)
  - Προδιαγραφών (specification)
  - Υλοποίησης (implementation)

UML - Structure Diagrams
Class Diagram - Class

- Μια κλάση σχεδιάζεται ως ένα ορθογώνιο με τρία τμήματα
- Τα ονόματα των ιδιοτήτων και μεθόδων της κλάσης μπορούν να έχουν ένα πρόθεμα ανάλογα με την ορατότητά τους
  - +
  - Public (δημόσια)
  - #
  - protected (προστατευμένη)
  - -
  - Private (ιδιωτική)



UML - Structure Diagrams
Class Diagram – Class variables

- Μια μεταβλητή καταγράφεται ως

  visibility name : type

  όπου visibility είναι:
  - **+** = public visibility
  - **#** = protected visibility
  - **-** = private visibility
  - *<blank>* = default (package) visibility
- Παράδειγμα

  +length:int

UML - Structure Diagrams
Class Diagram – Class methods

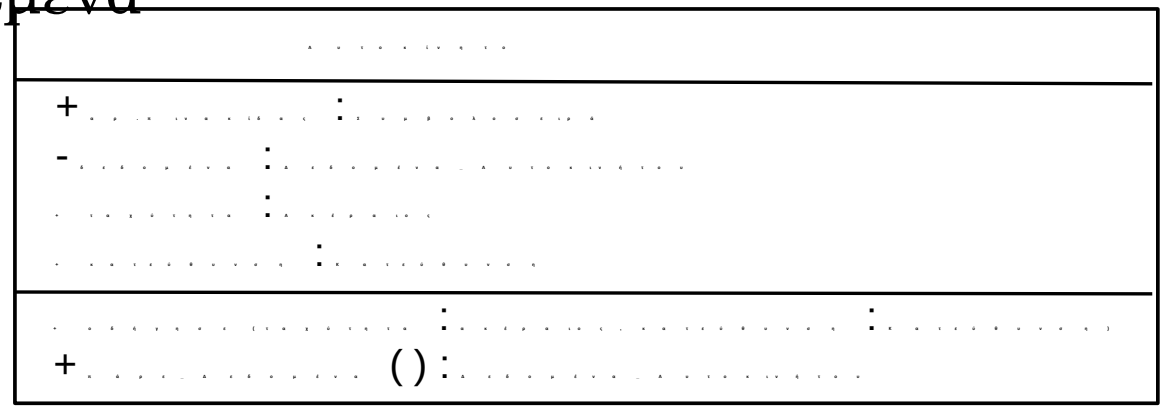
- Μέθοδοι καταγράφονται ως:

  visibility name (parameters) : returnType

  όπου
  - ***visibility*** είναι ένα από **+**, **-**, **#**, ***blank***
  - Παράμετροι καταγράφονται ως ***name:type***
  - Αν ***returnType*** είναι ***void*** παραλείπεται

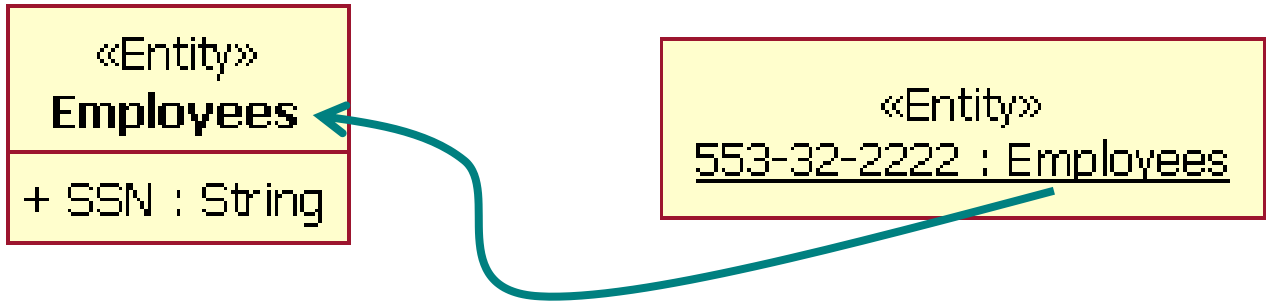
UML - Structure Diagrams
Class Diagram – Class methods

- Οι μέθοδοι χρησιμοποιούνται για να διαχειριζόμαστε τα χαρακτηριστικά ή να εκτελούμε συγκεκριμένες ενέργειες
- Οι μέθοδοι περιγράφουν τι υπηρεσίες προσφέρει η κάθε κλάση και κάποιες από αυτές παρέχουν την κατάλληλη διασύνδεση.
- Οι μέθοδοι μπορούν να παίρνουν πληροφορίες, να ενημερώνουν γνωρίσματα και να καλούν άλλα αντικείμενα



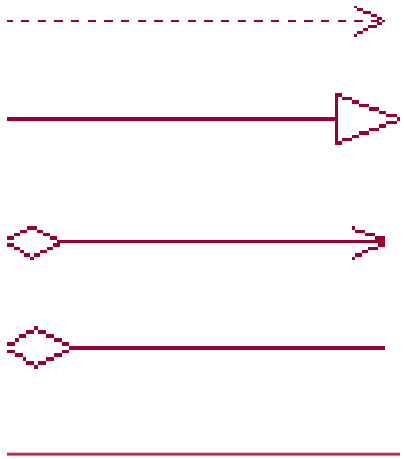
UML - Structure Diagrams
Class Diagram – Class Object-Instance

- Η αναπαράσταση ενός αντικειμένου προκύπτει από την κλάση του
- Συνήθως ένα αντικείμενο εμφανίζεται με υπογραμμισμένο το όνομά του



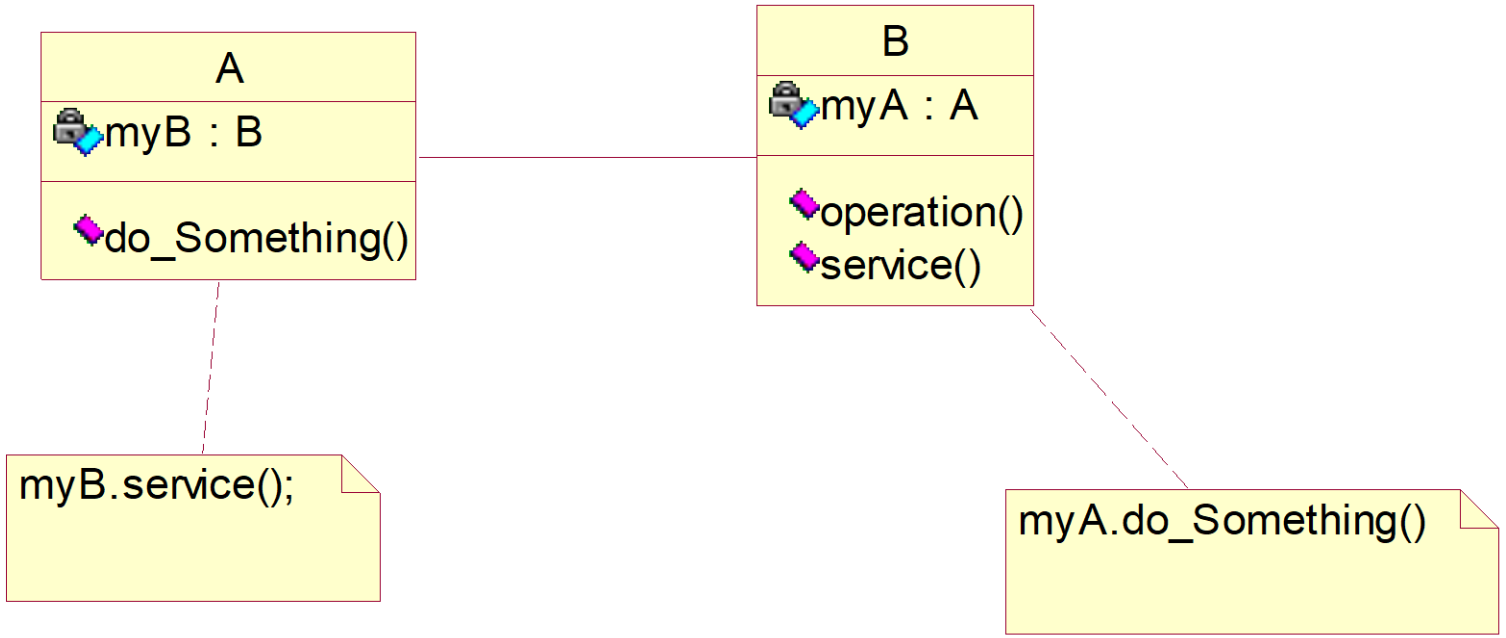
UML - Relationships

- Στη UML ορίζονται τέσσερις βασικές σχέσεις
  - Εξάρτηση (dependency)
  - Γενίκευση (generalisation)
  - Συγκρότηση (aggregation)
  - Σύνθεση (composition)
  - Σύνδεση (association)



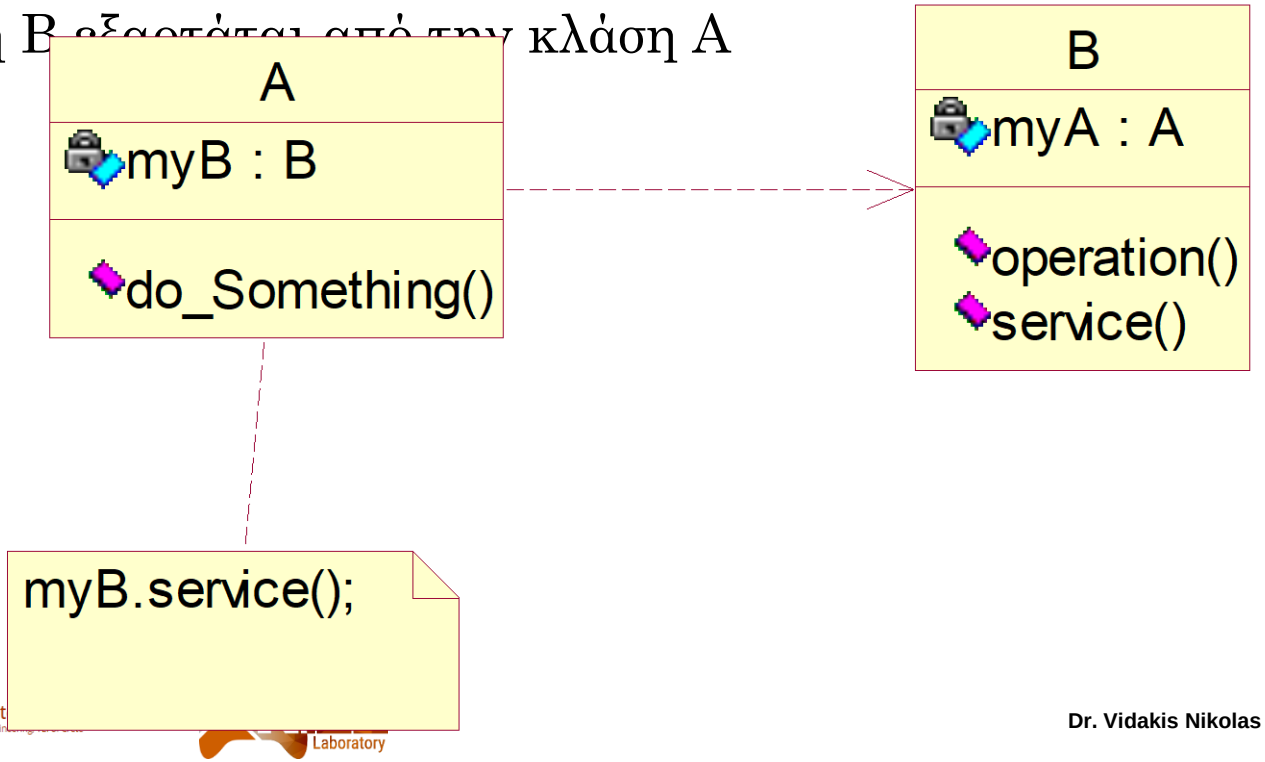
UML – Relationships Association

- Δυαδική συσχέτιση
  - Στη δυαδική συσχέτιση και οι δύο κλάσεις γνωρίζουν τι μεθόδους υλοποιεί η άλλη κλάση



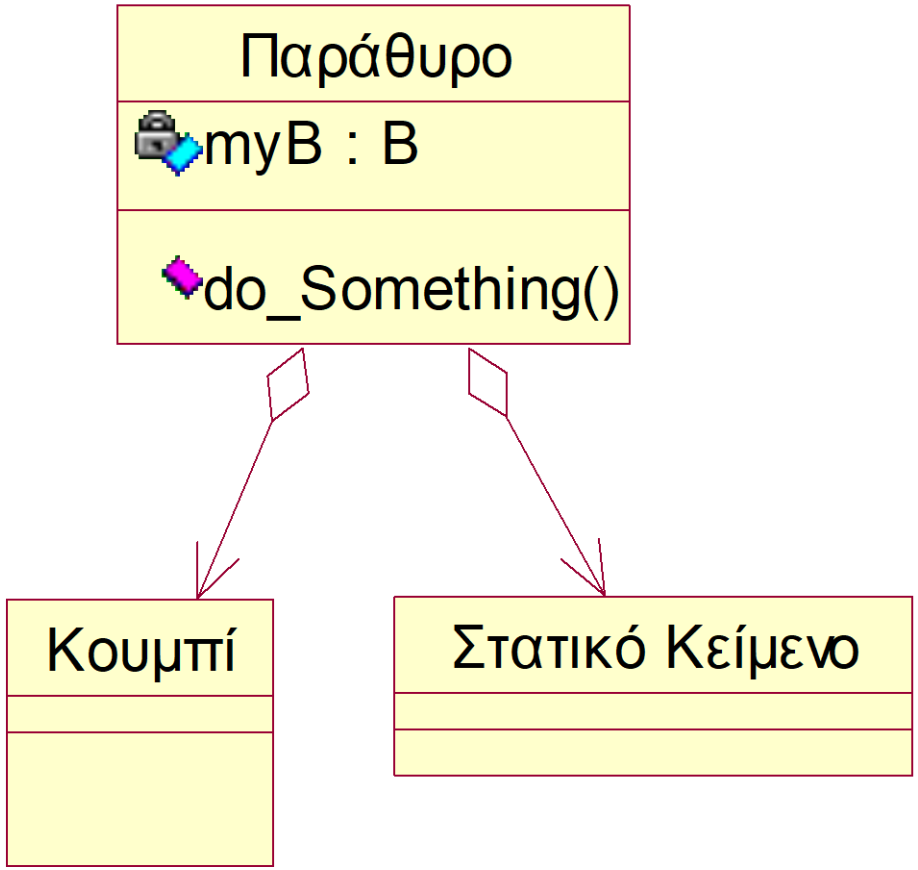
UML – Relationships Dependency

- Τυπική ερμηνεία είναι ότι μια Κλάση Α γνωρίζει για μια κλάση Β, αλλά η Β δεν γνωρίζει για την Α
  - Η κλάση Α μπορεί να 'καλέσει' την κλάση Β
  - Κλάση Β εξαρτάται από την κλάση Α



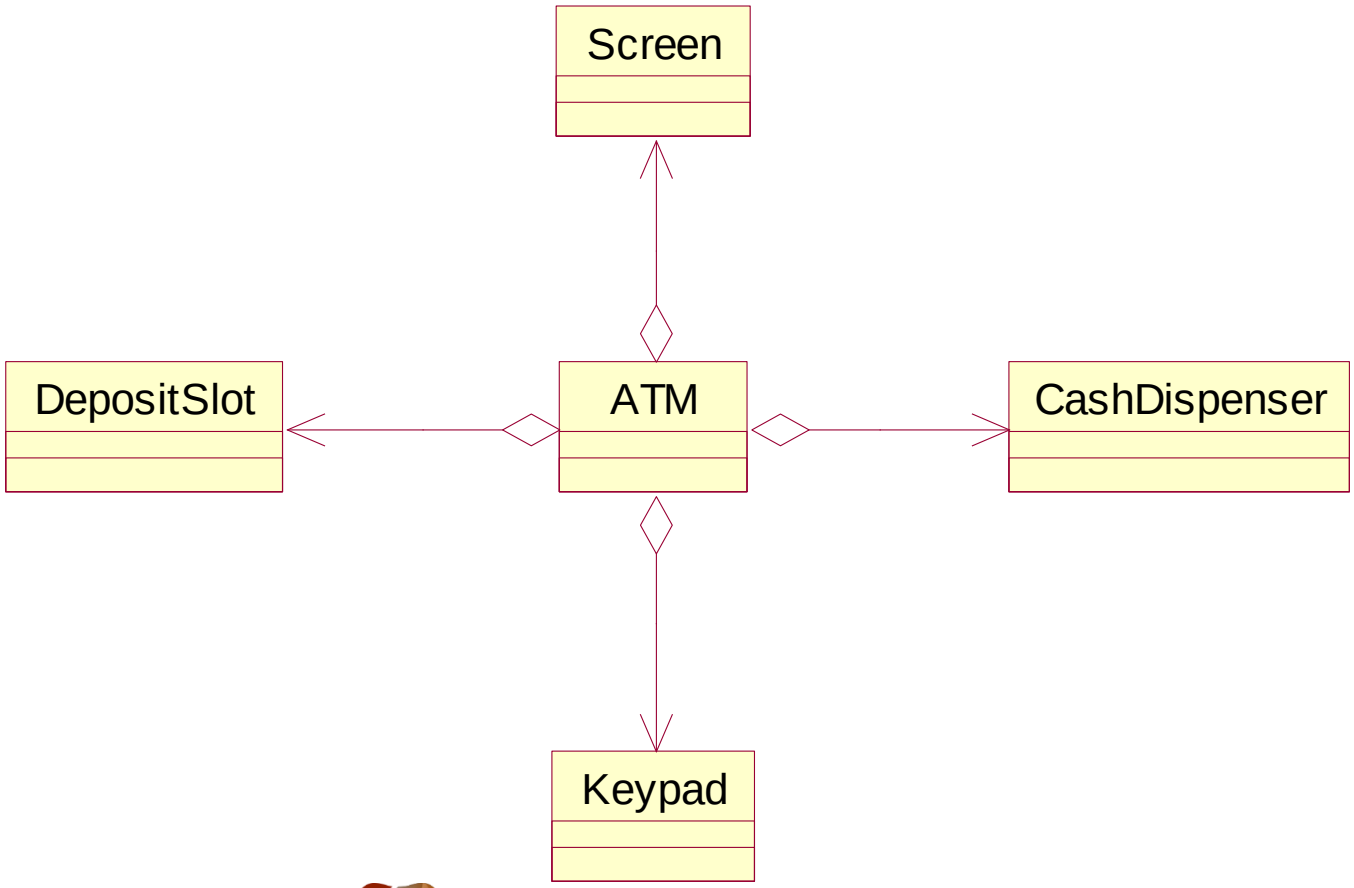
UML – Relationships Composition

- Η συνάθροιση δηλώνει σχέση “ομάδας - μέλους”



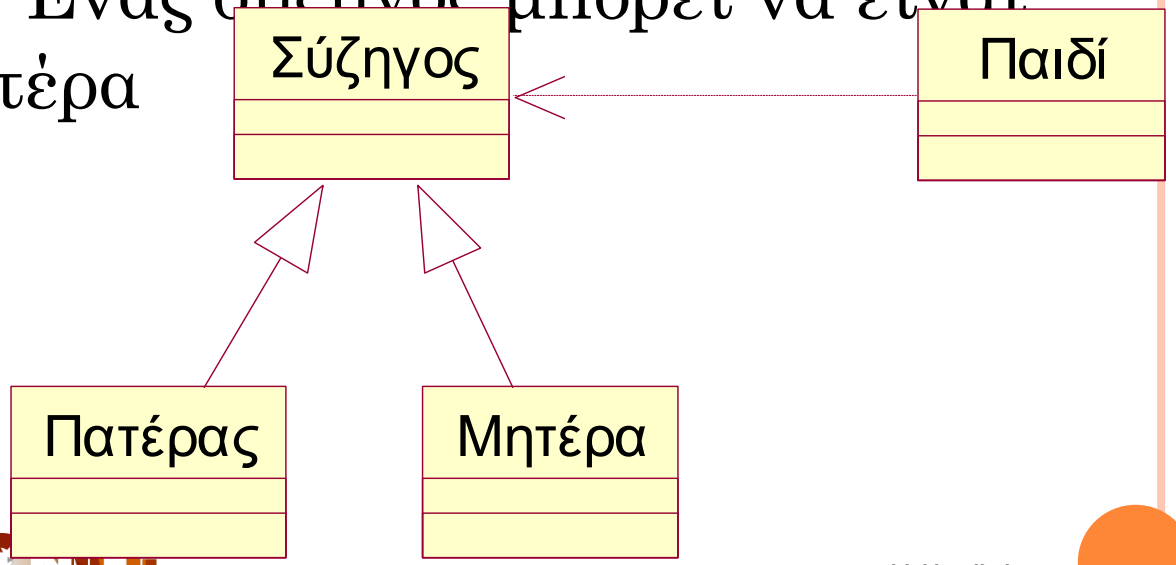
UML – Relationships Composition

- Το παράδειγμα του ATM

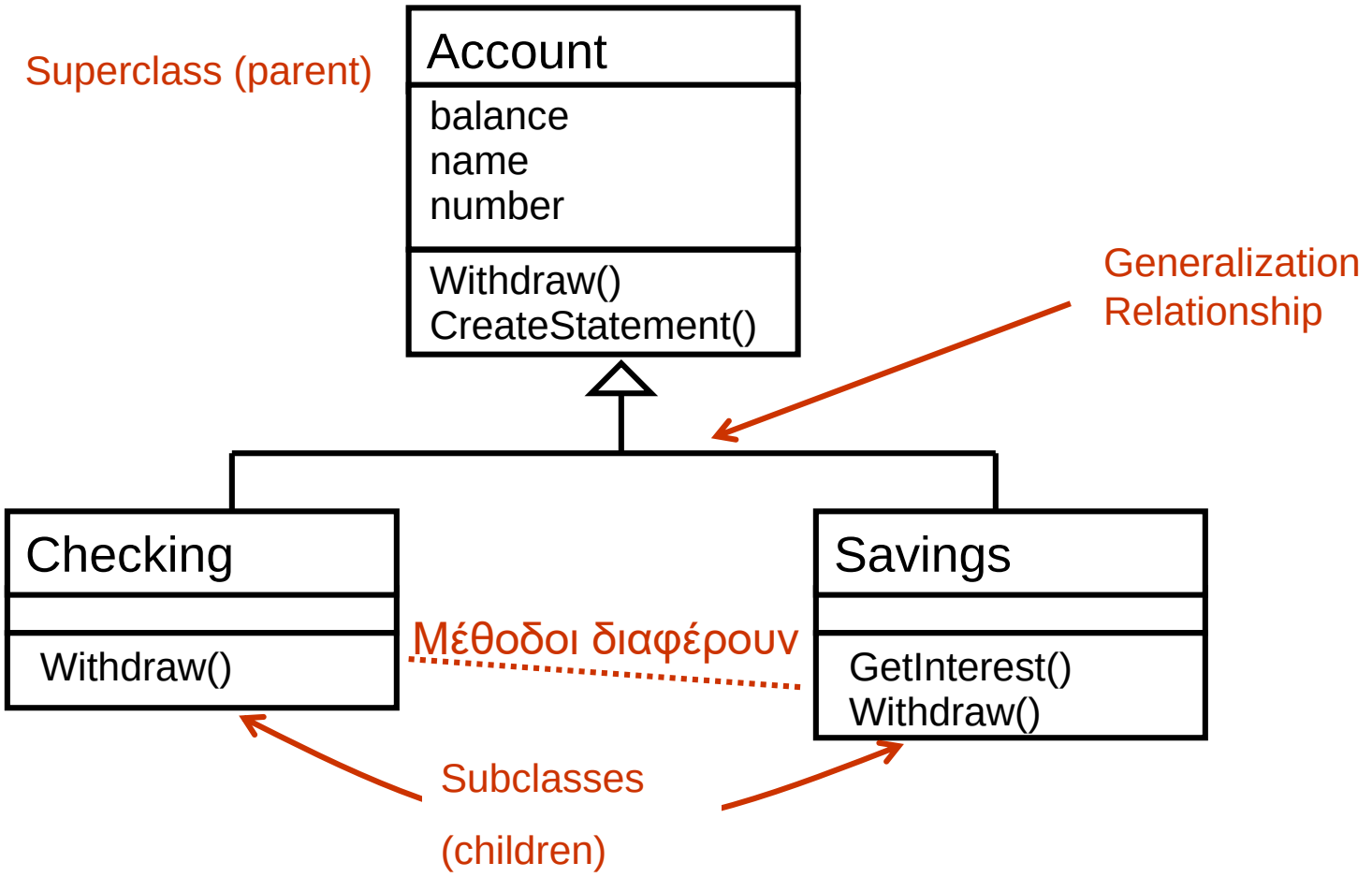


UML – Relationships Generalisation

- Γενίκευση δηλώνει μια σχέση ανάμεσα σε κάτι γενικό (τη βασική κλάση ή αλλιώς super-κλάση) και κάτι ειδικό (μια υποκλάση ή sub-κλάση)
  - Δηλώνει και κληρονομικότητα
- Παράδειγμα - Ένας σύζυγος μπορεί να είναι πατέρας ή μητέρα

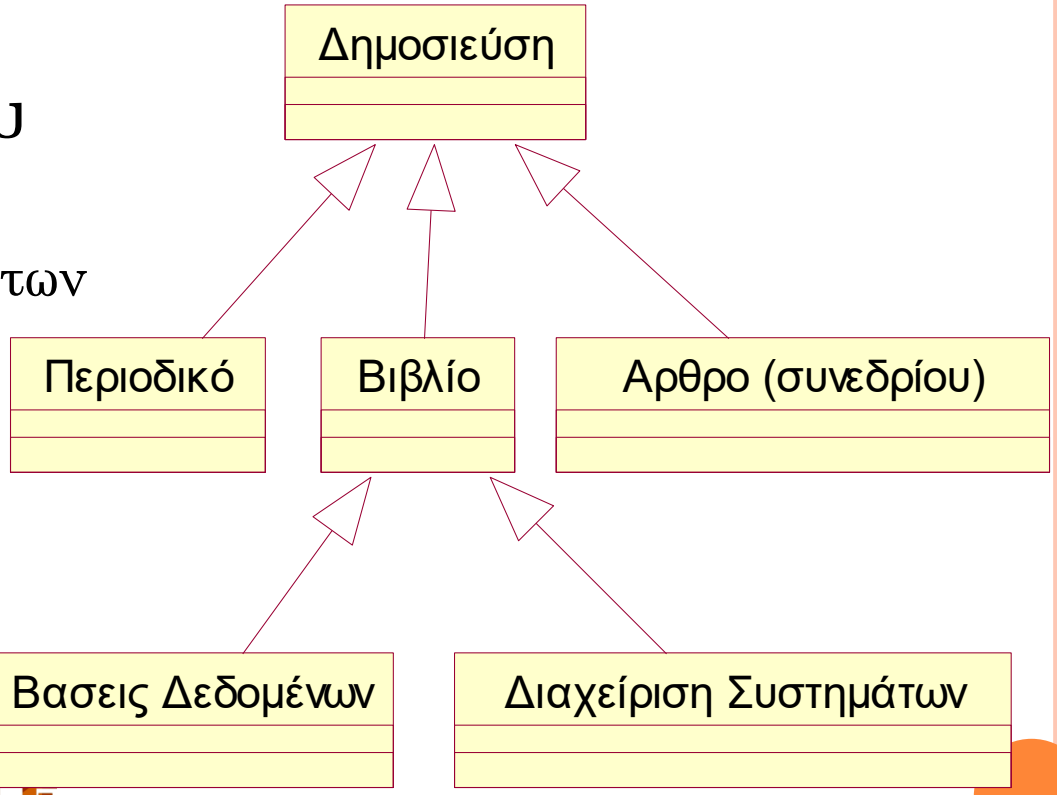


UML – Relationships Generalisation SuperClass & SubClass



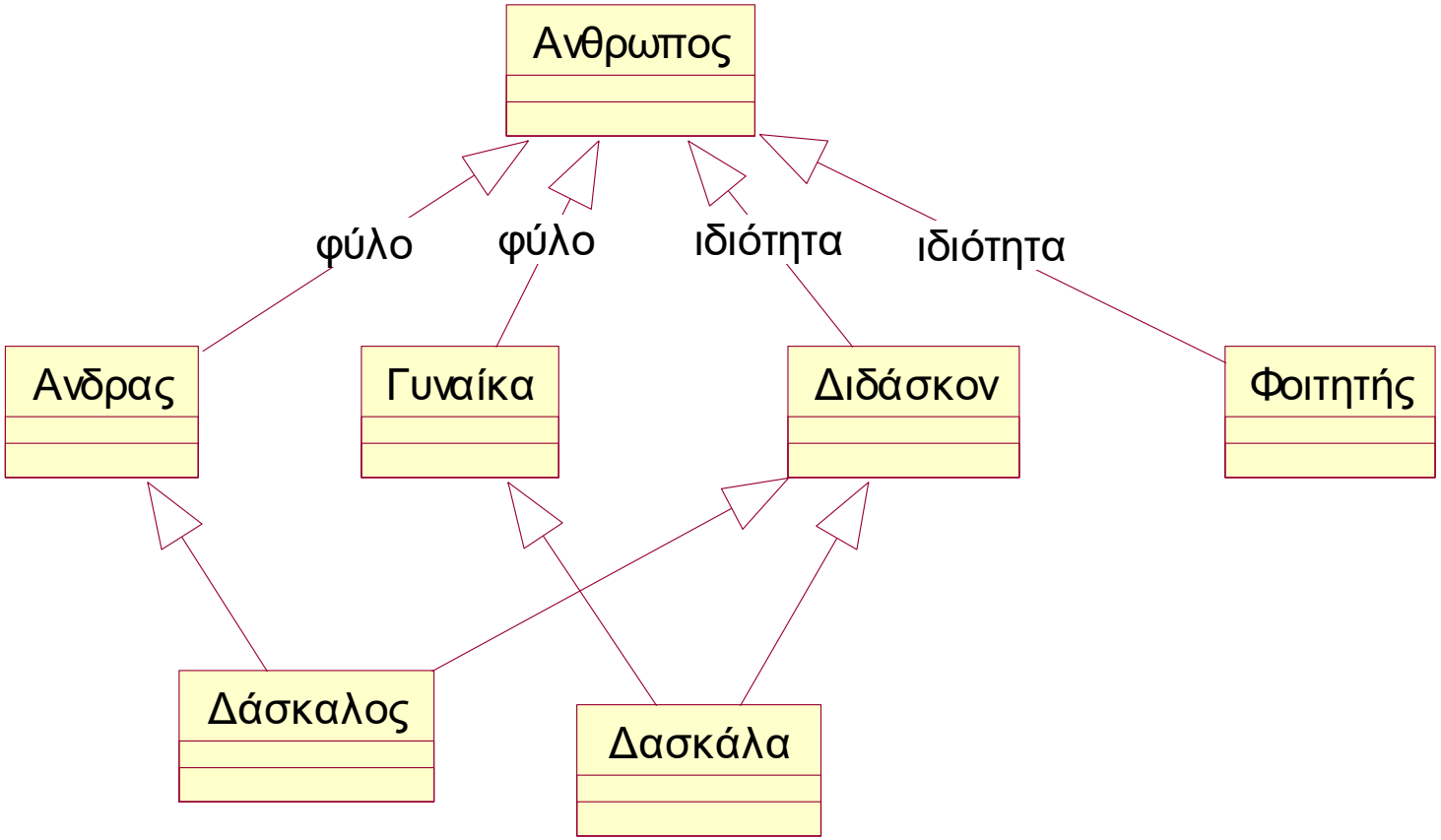
UML – Relationships Generalisation Example

- Μια δημοσίευση μπορεί να είναι πολλών τύπων:
  - Περιοδικό (δημοσίευμα σε περιοδικό)
  - Βιβλίο
  - Άρθρο (συνεδρίου)
- Θέμα του βιβλίου
  - Βάσεις δεδομένων
  - Διαχείριση συστημάτων



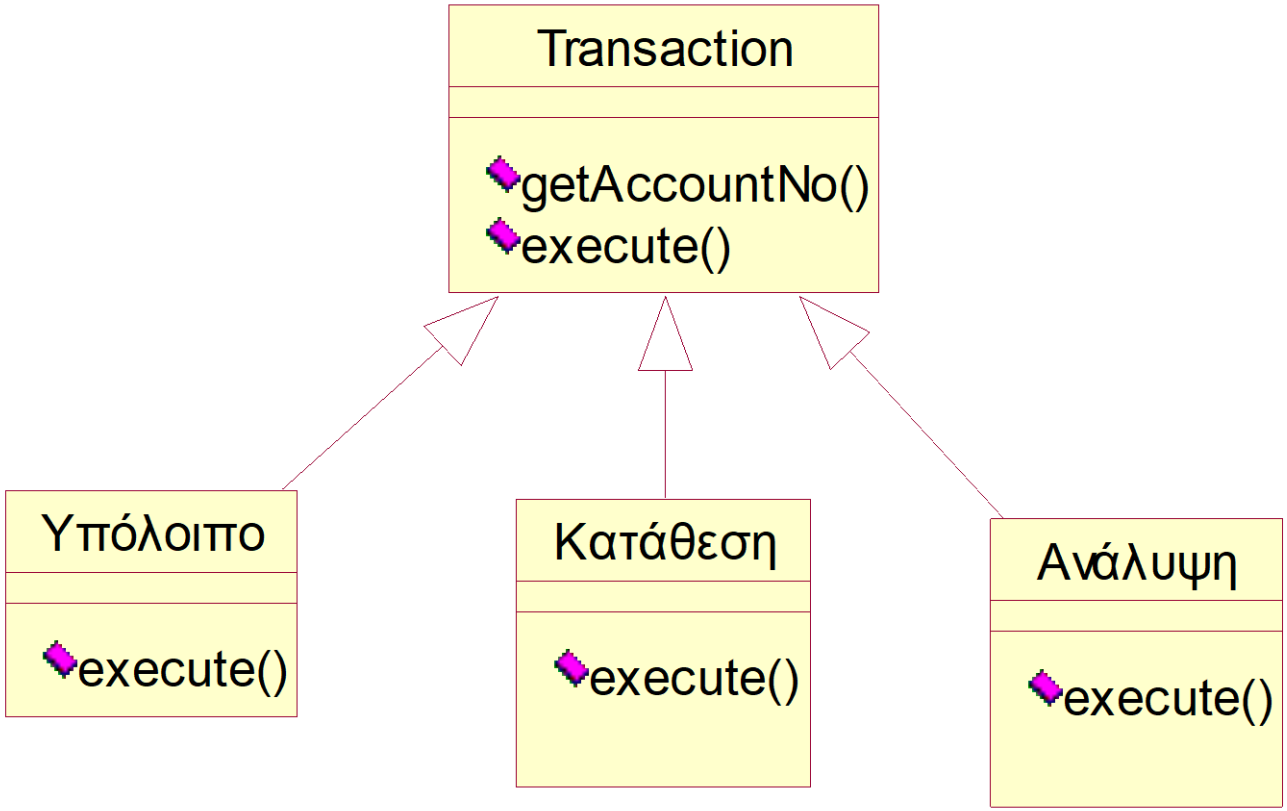
UML – Relationships Generalisation Example

- Πολλαπλή κληρονομικότητα



UML – Relationships Generalisation Example

- Τύποι συναλλαγών με ATM

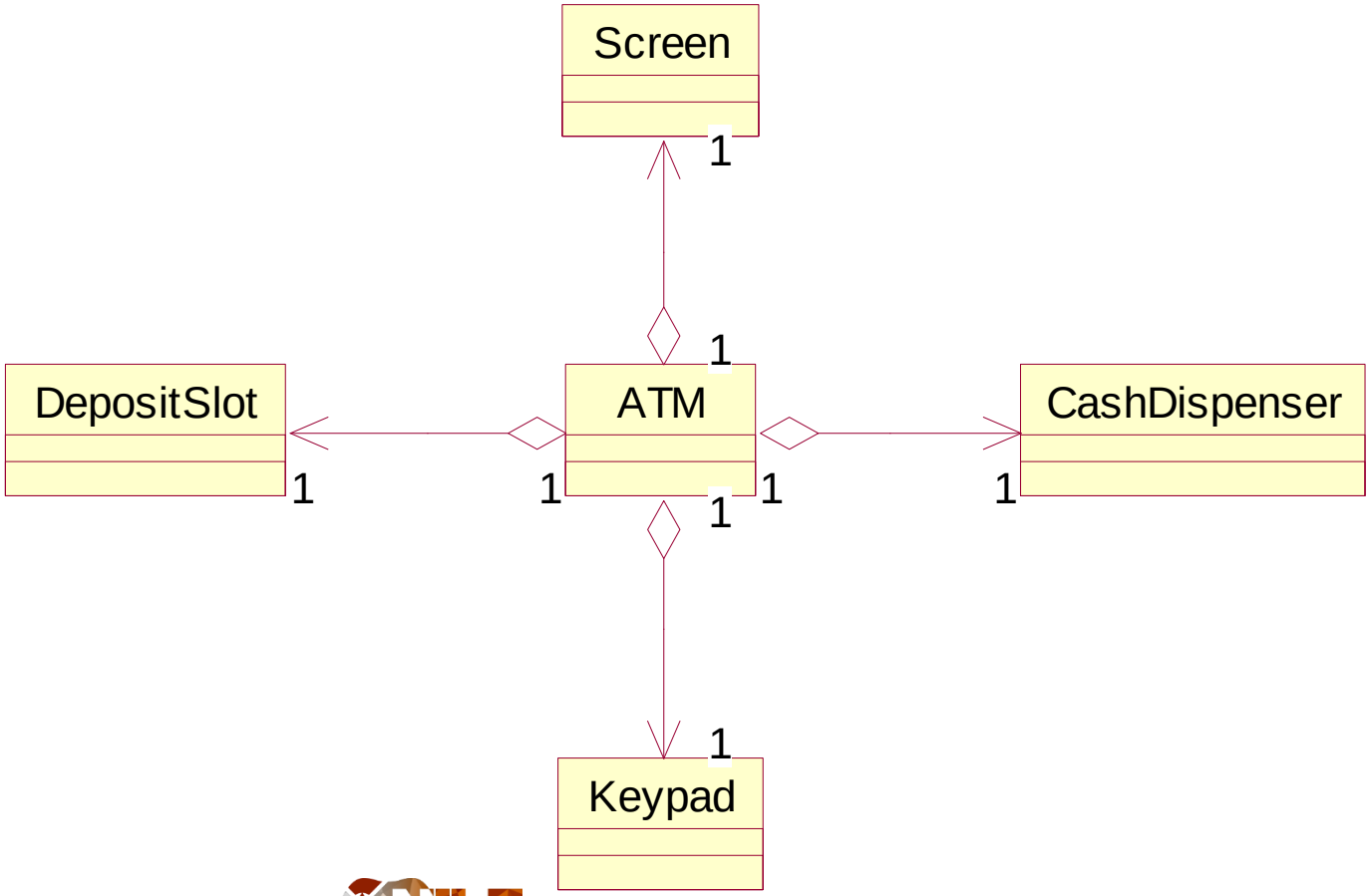


UML – Relationships Multiplicity

Πληθικότητα	Εμηνεία
0..1	Μηδέν ή ένα.
0..* *or* *****	Μηδέν έως πολλά
1	Ακριβώς ένα
1..*	Τουλάχιστον 1 (ένα ή περισσότερα)

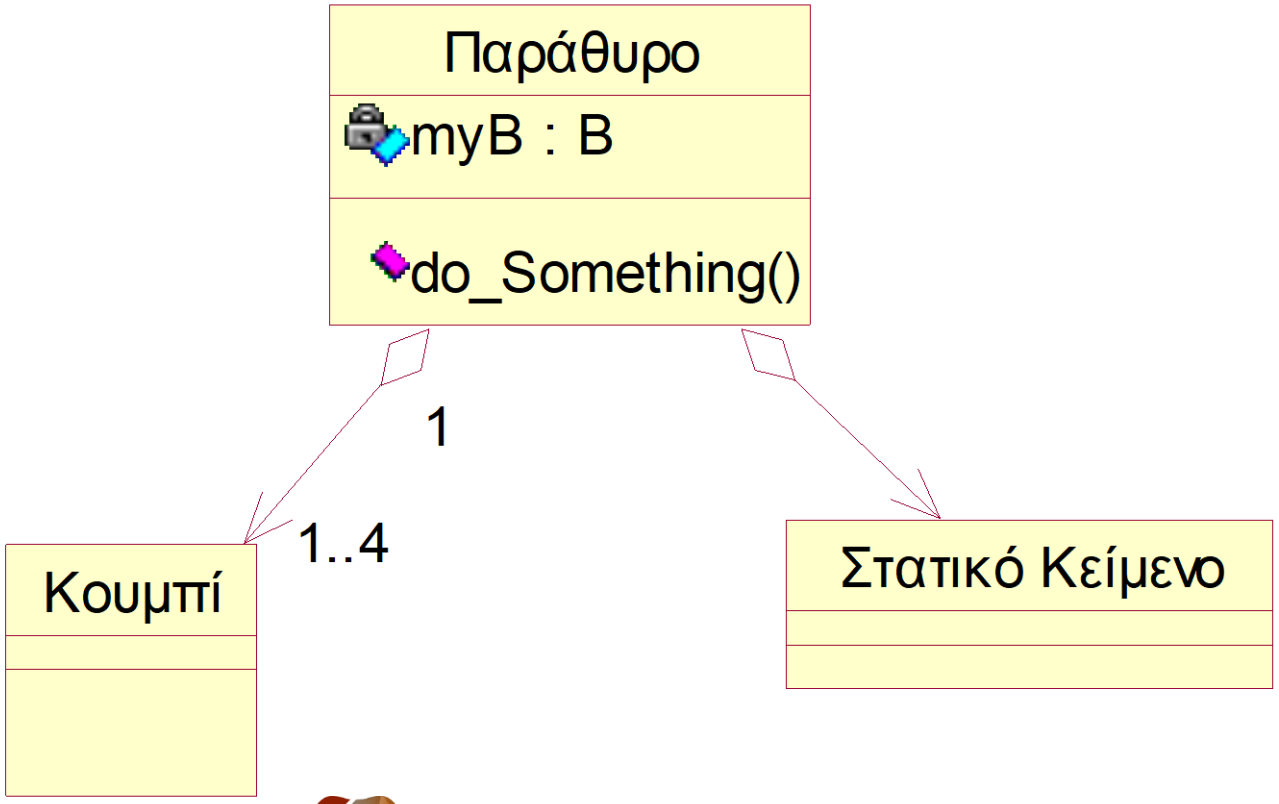
UML – Relationships Multiplicity Example

- Το παράδειγμα του ATM



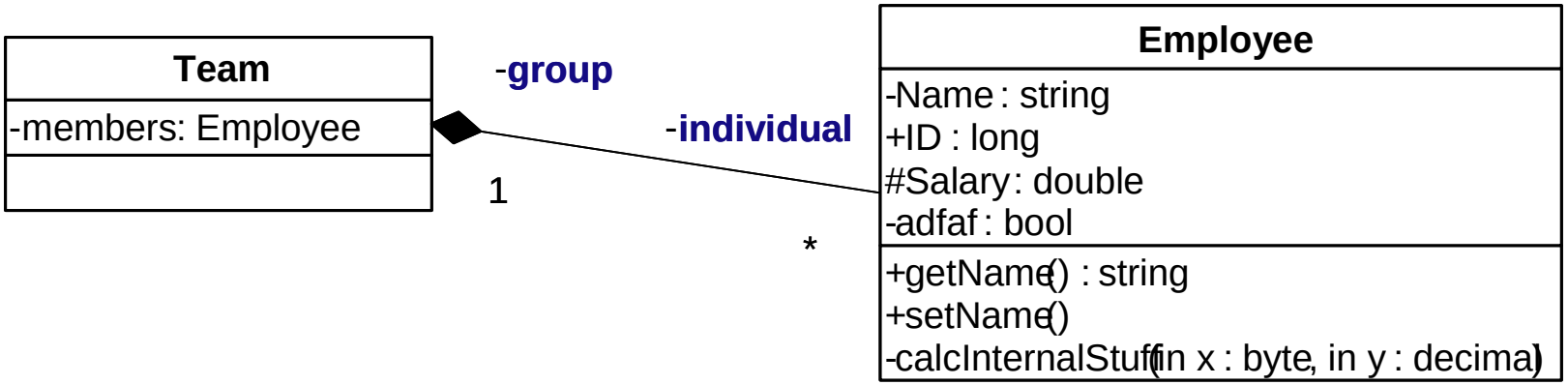
UML – Relationships Multiplicity Example

- Ένα παράθυρο μπορεί να περιέχει μέχρι τέσσερα αντικείμενα τύπου ‘κουμπί’



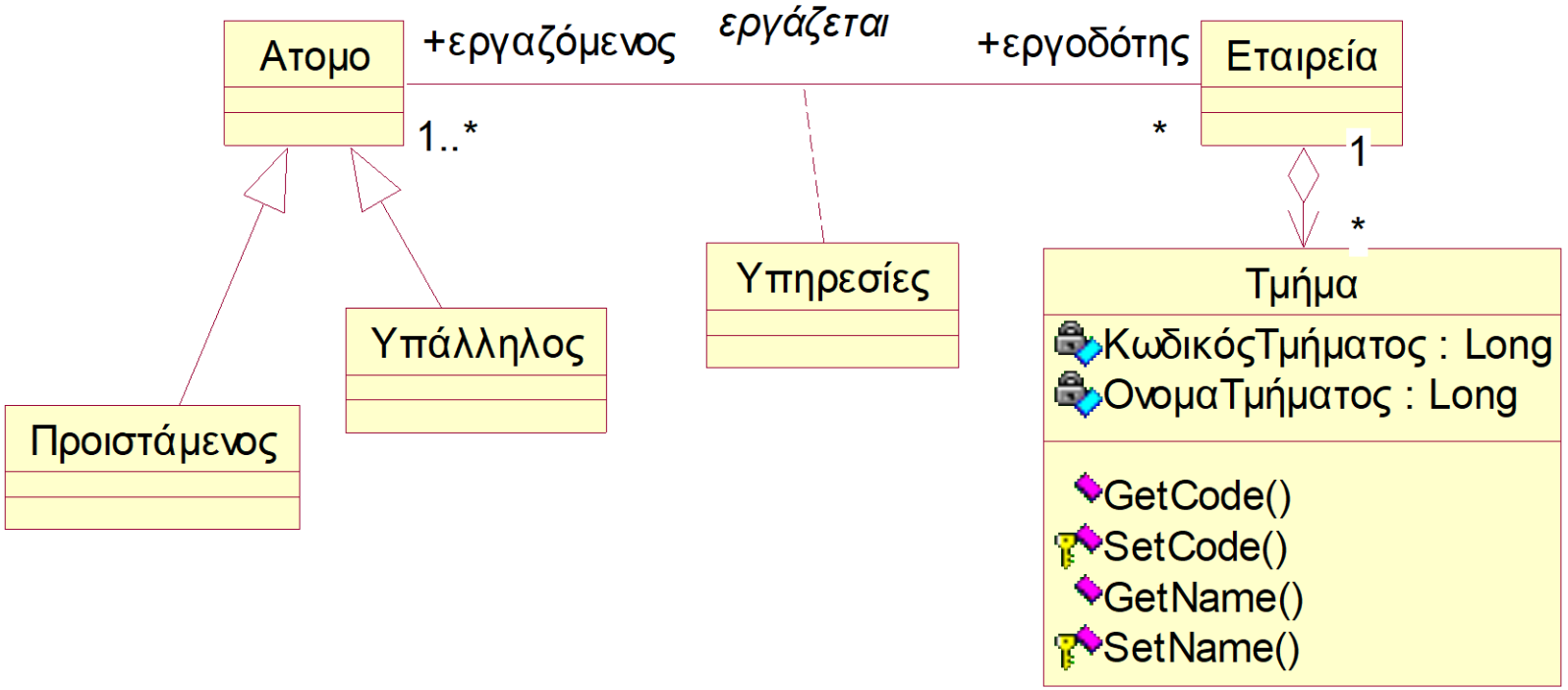
UML – Relationships Class Roles

- Ονομασία σχέσεων για επιπλέον πληροφορία

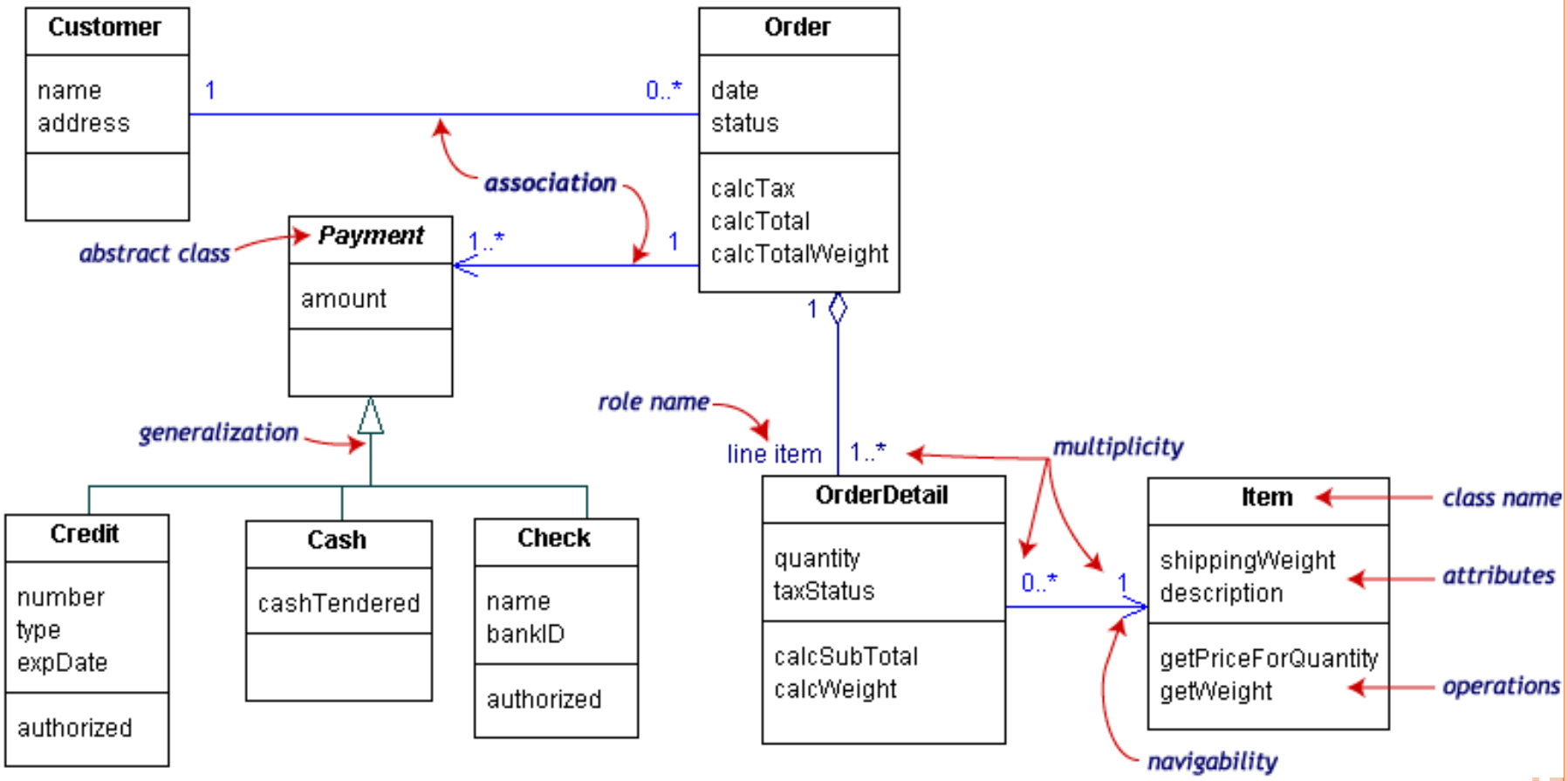


UML – Relationships Class Roles

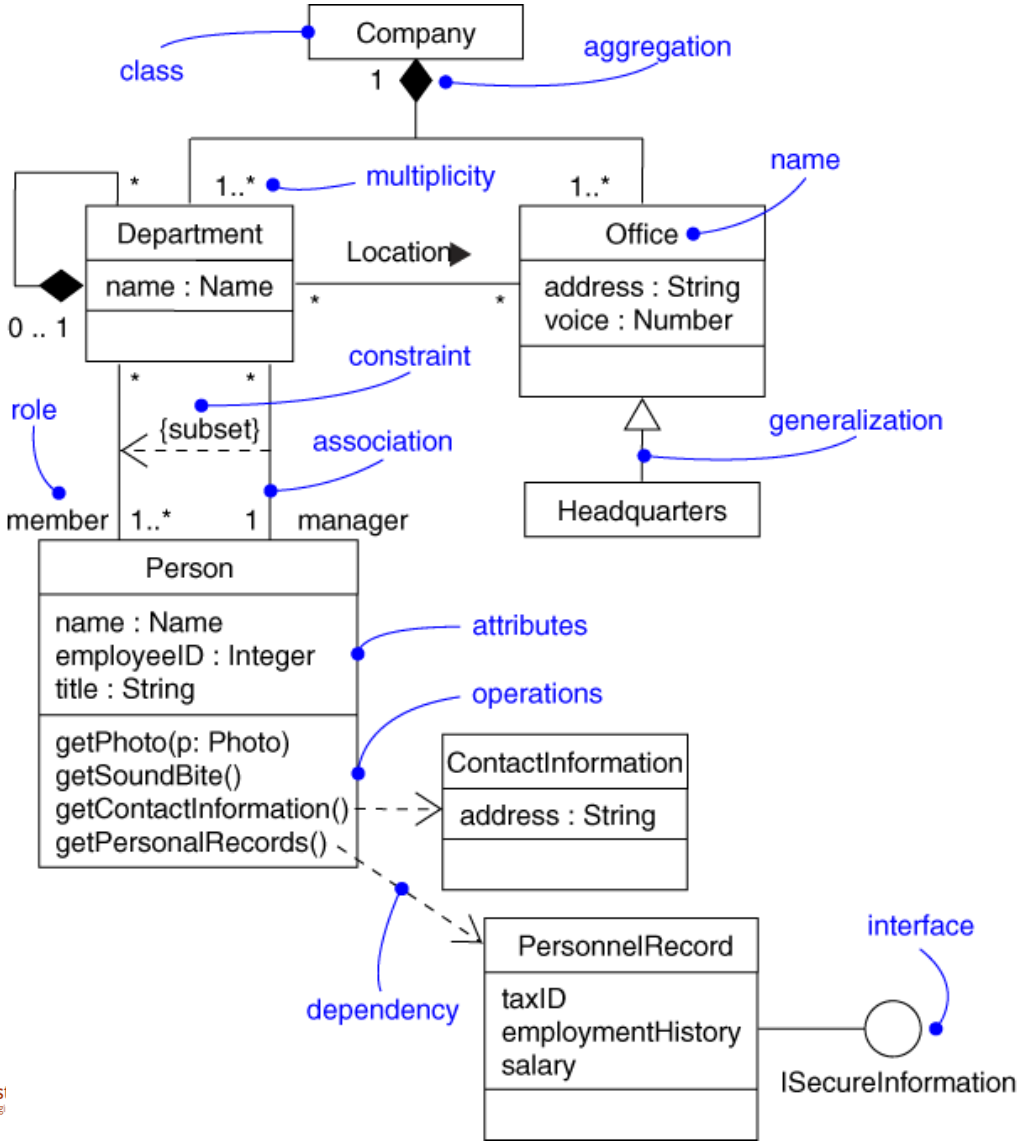
- Μια εταιρεία συγκροτείται από πολλά τμήματα



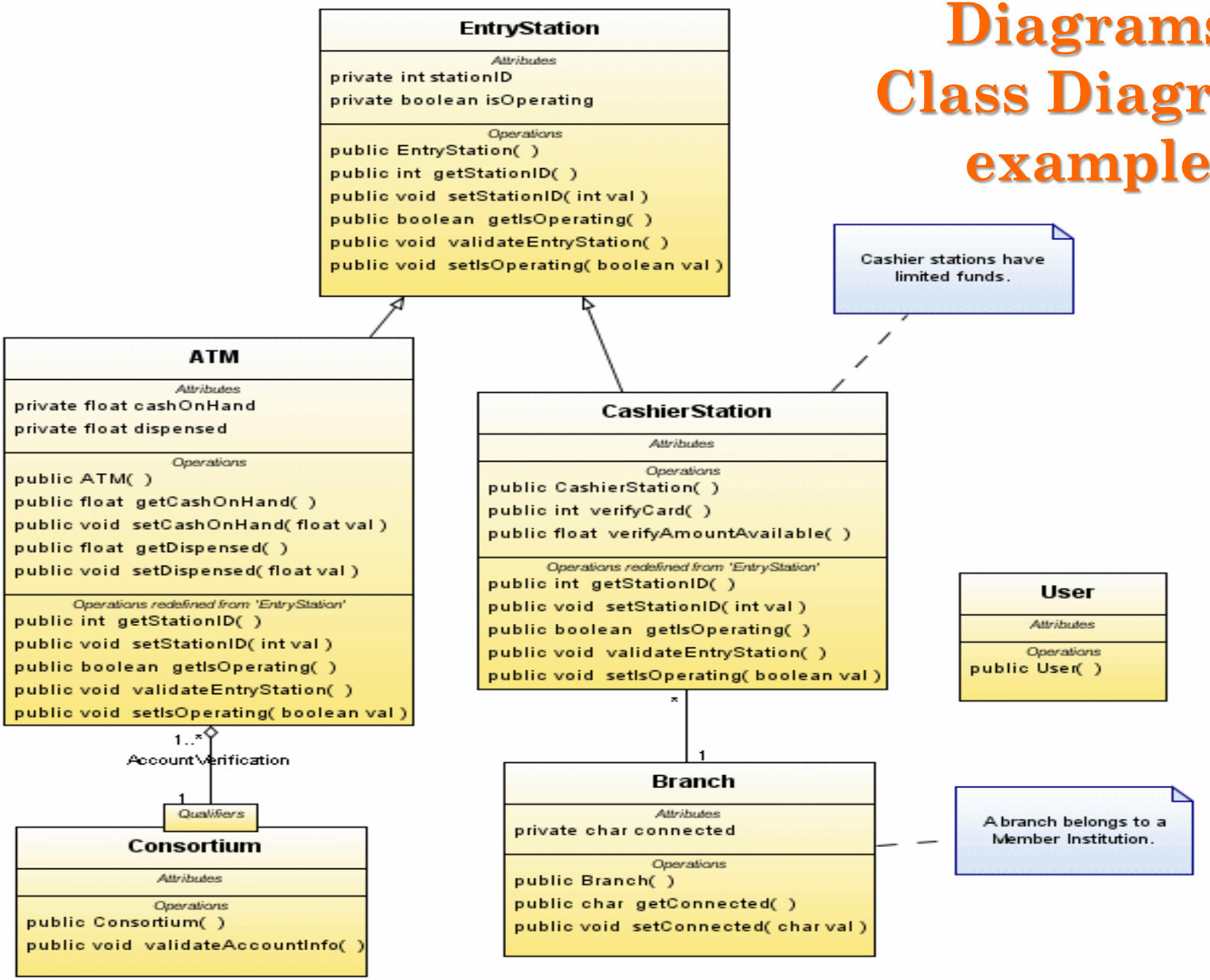
UML - Structure Diagrams
Class Diagram example



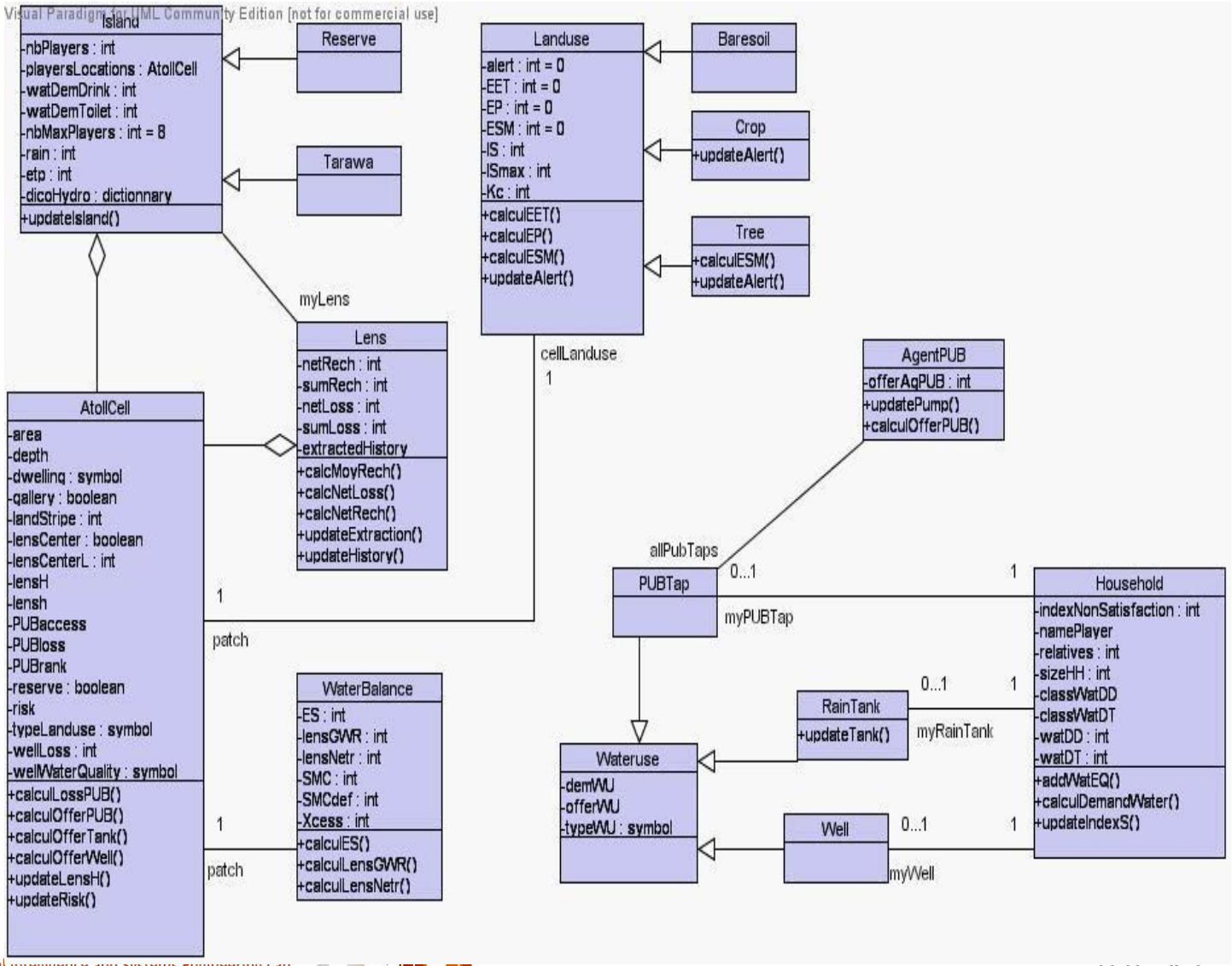
Παράδειγμα διαγράμματος κλάσεων



UML - Structure Diagrams
Class Diagram example

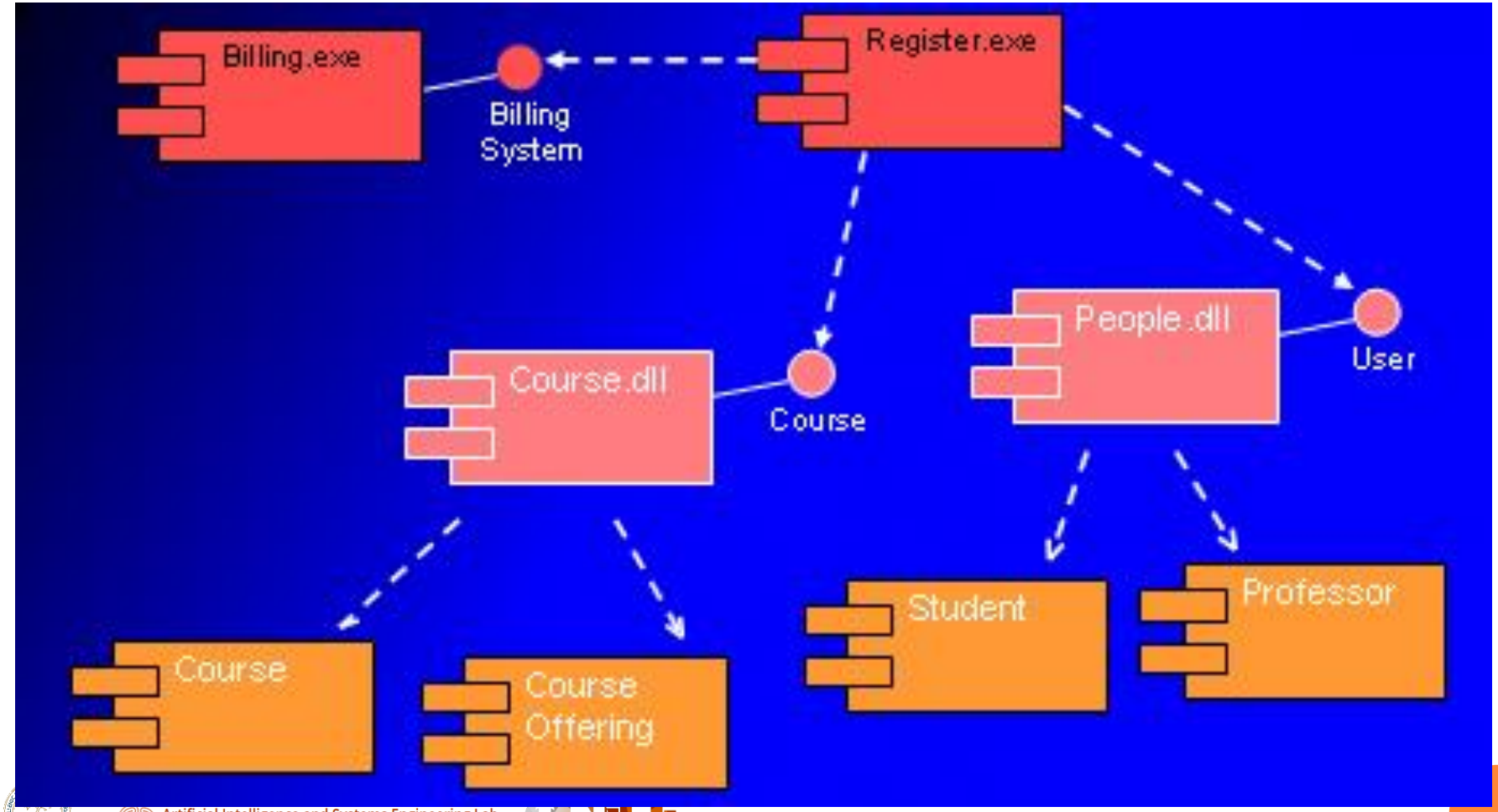


UML - Structure Diagrams
Class Diagram example

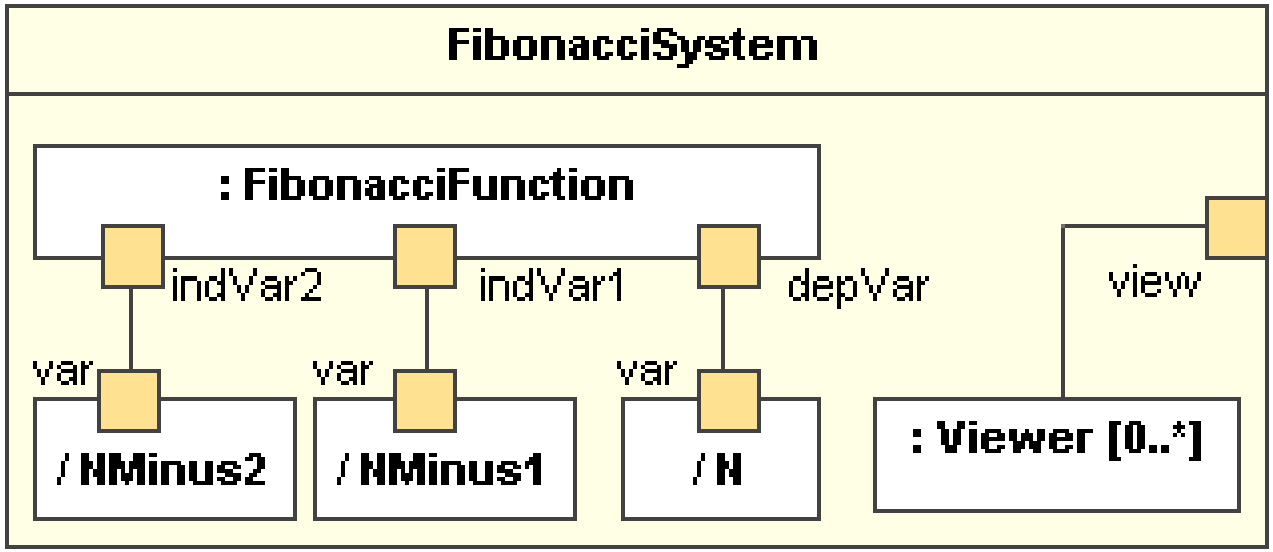


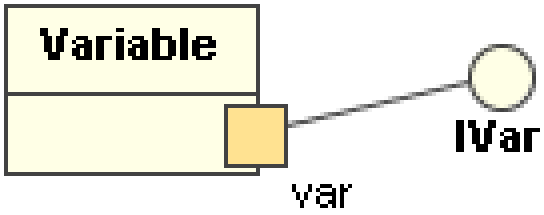
UML - Structure Diagrams
Component Diagram

Παραδείγματα Component Diagram

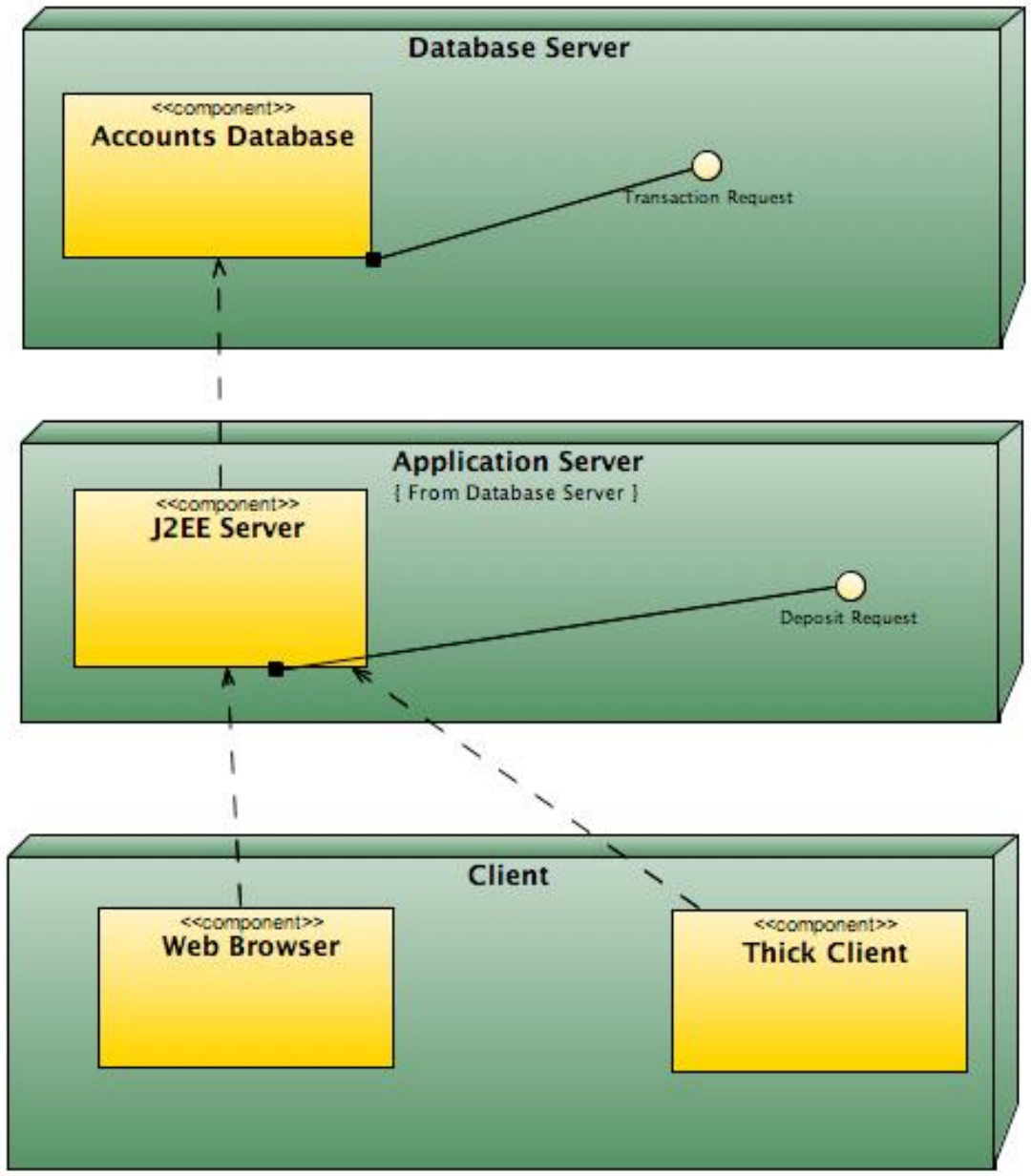


UML - Structure Diagrams
Composite structure Diagram





UML - Structure Diagrams (Deployment Diagram)



UML - Behaviour Diagrams

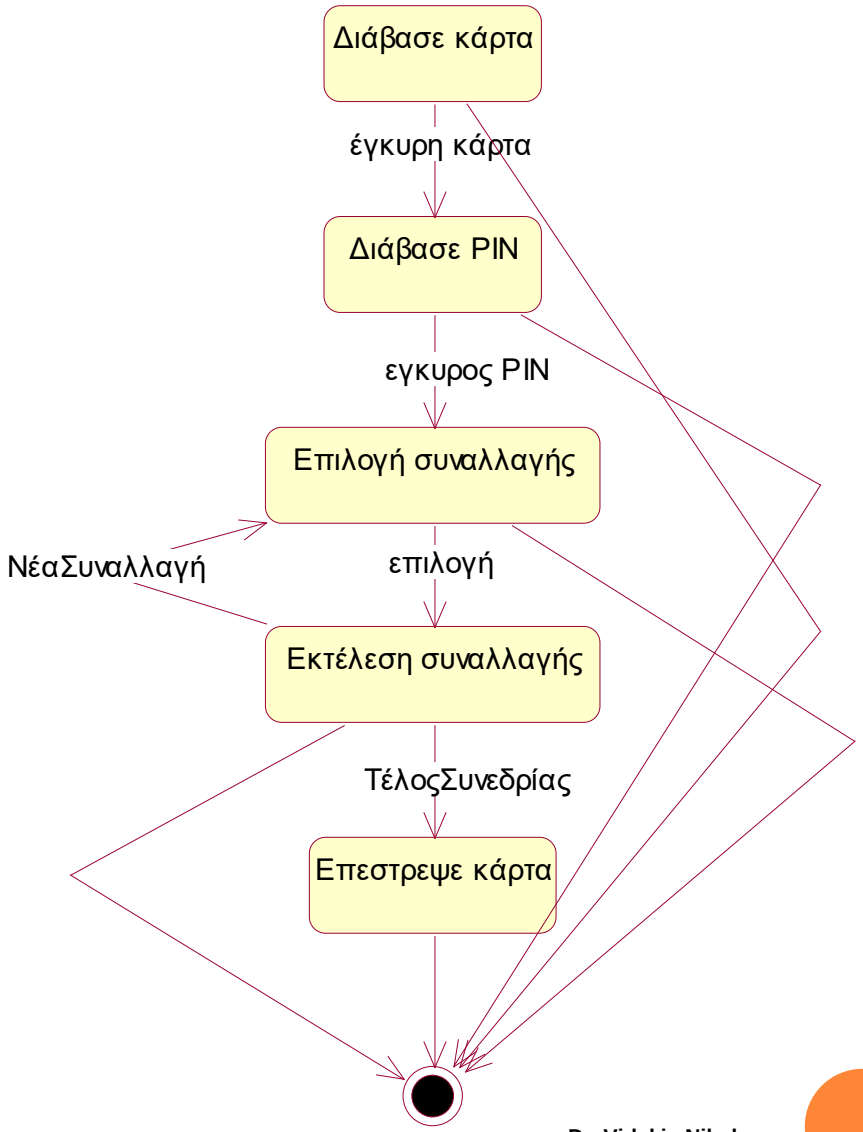
Behavior / Dynamic diagrams

Τα διαγράμματα Behavior δίνουν έμφαση στο τι πρέπει να γίνει στο σύστημα.

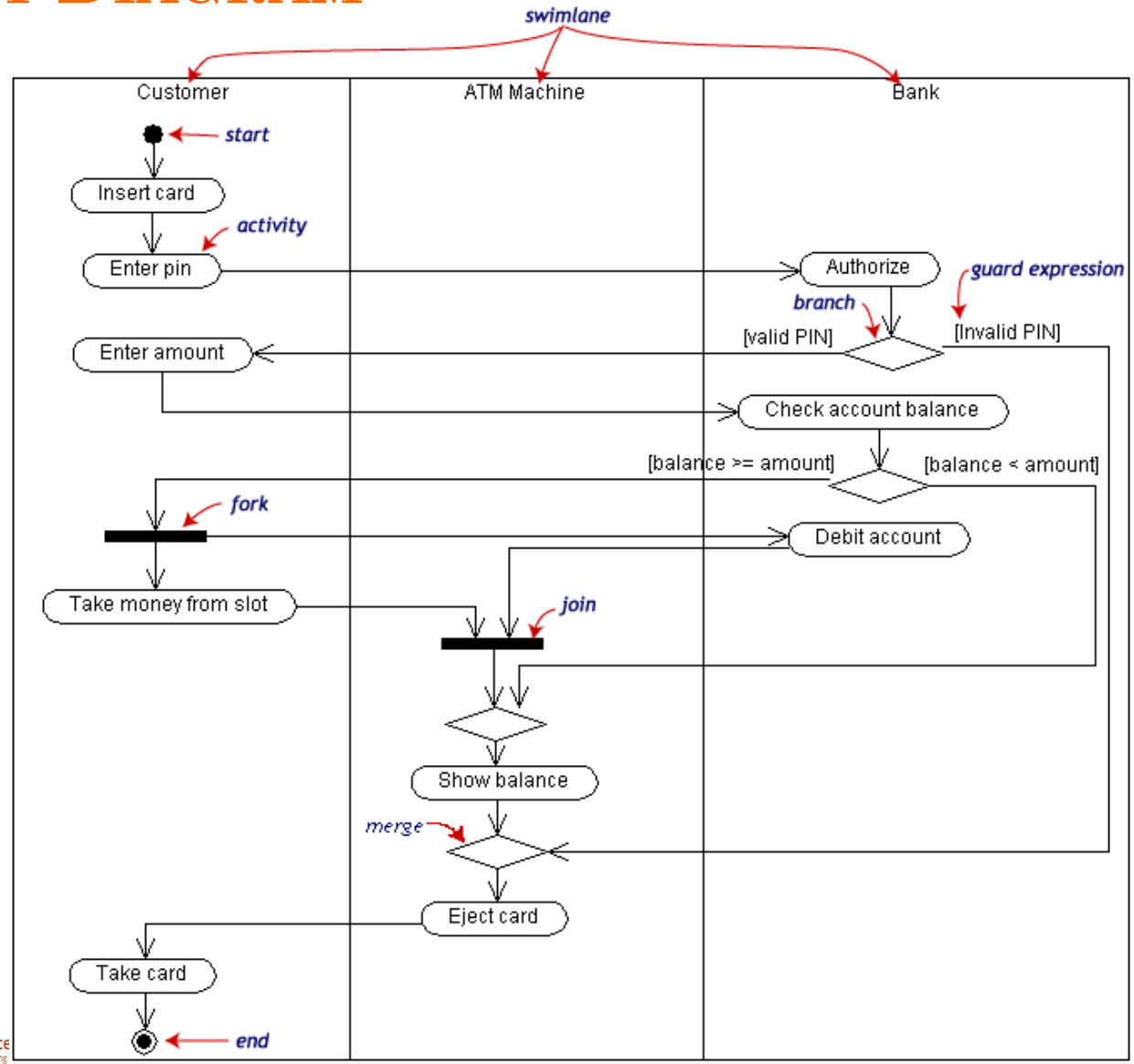
- Activity diagram: represents the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.
  (Δίνει την ακολουθιακή ροή των δραστηριοτήτων, Περιέχει προσδιορισμούς των μηνυμάτων που στέλνονται)
- State diagram: standardized notation to describe many systems, from computer programs to business processes.
- Use case diagram: shows the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies between those use cases.
  (Περιγράφει τη λειτουργικότητα του συστήματος όπως αυτή γίνεται αντιληπτή από τον χρήστη)

UML - Structure Diagrams
Activity Diagram

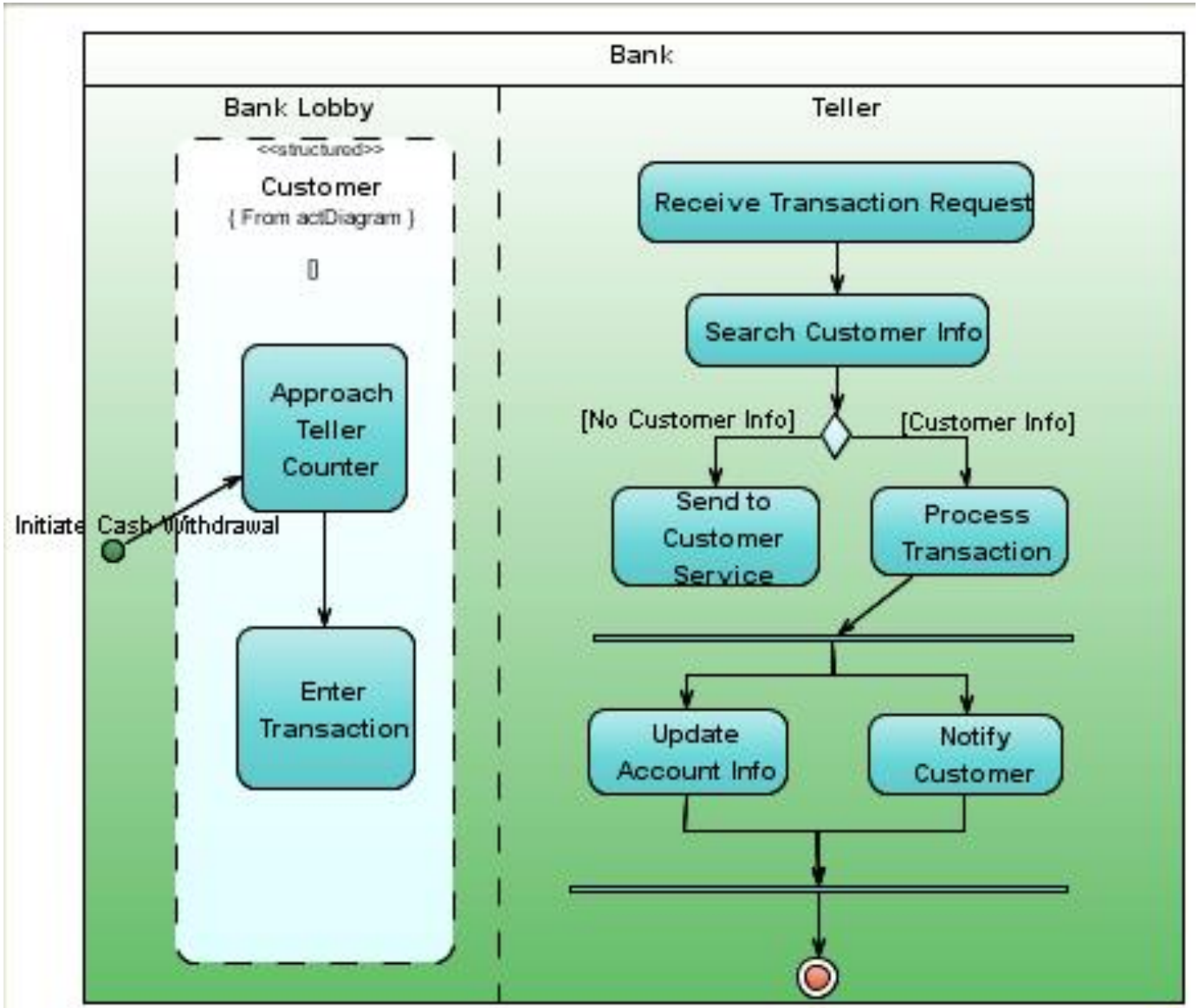
- Χρησιμοποιείται να καταγράψει ενέργειες που γίνονται (ροές δεδομένων) κατά την εκτέλεση ενός λειτουργικού τμήματος (π.χ. μιας περίπτωσης χρήσης)



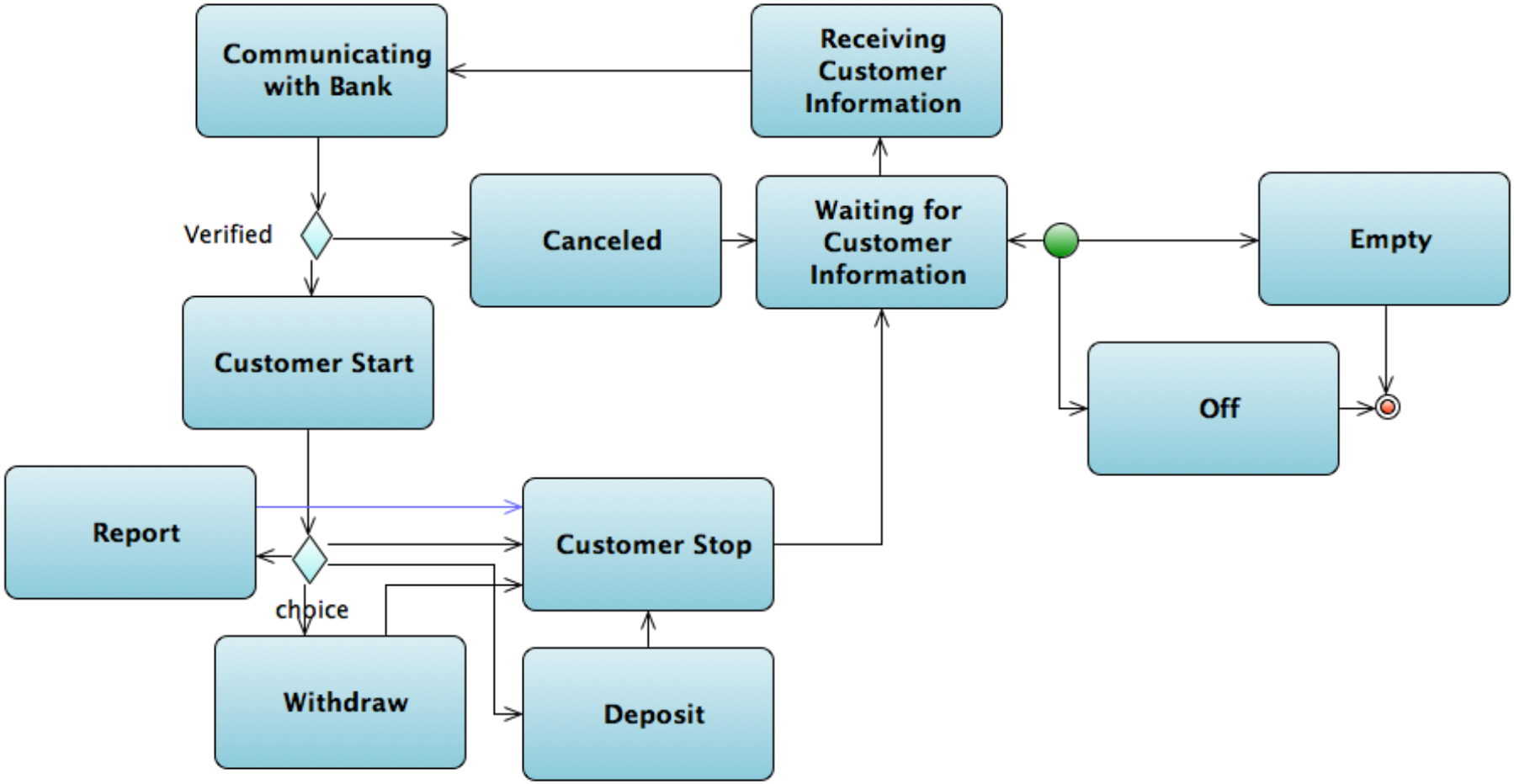
UML - Structure Diagrams
Activity Diagram



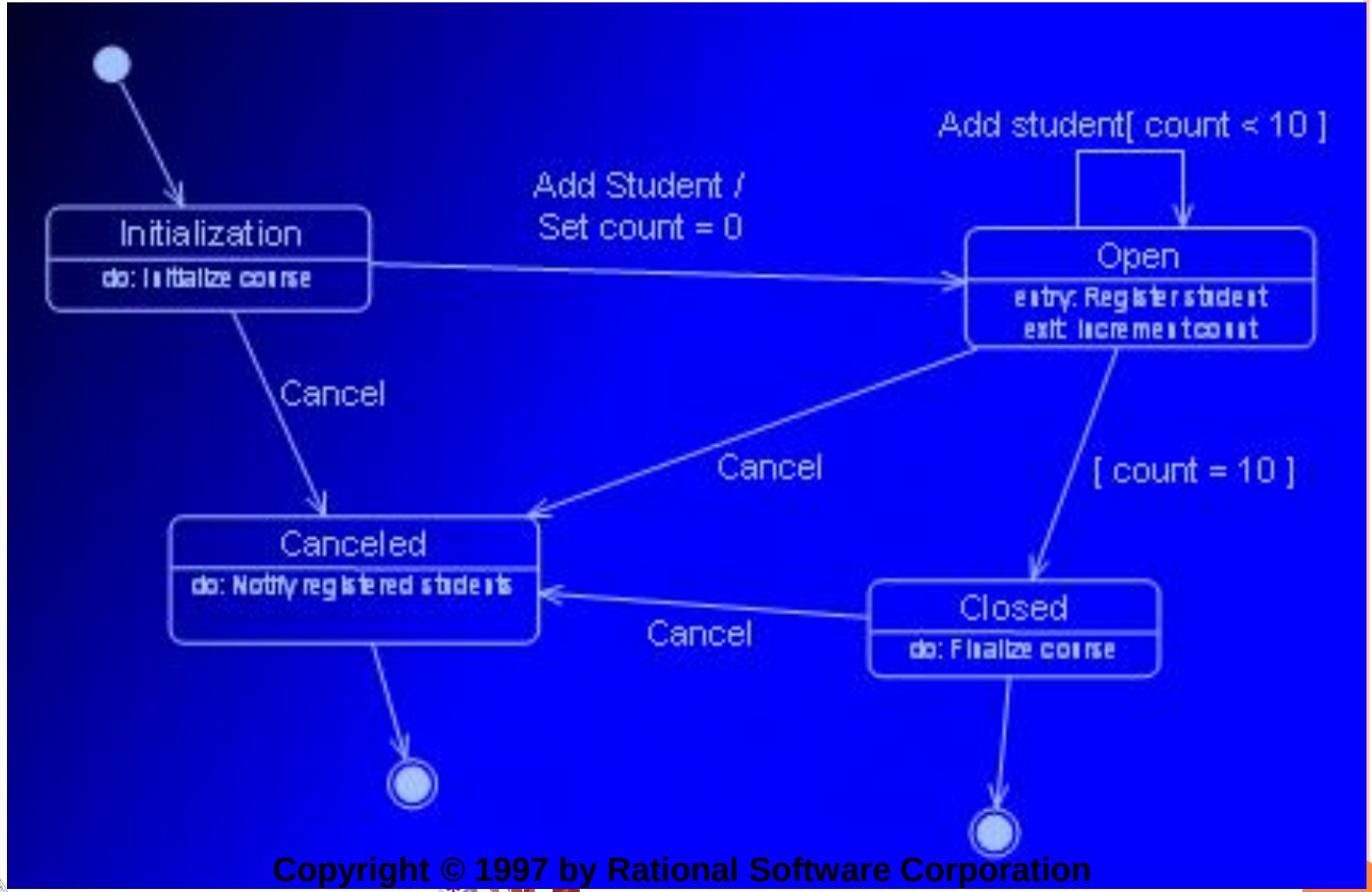
UML - Behaviour Diagrams
Activity Diagram



UML - Behaviour Diagrams
State Diagram



UML - Behaviour Diagrams
State Diagram

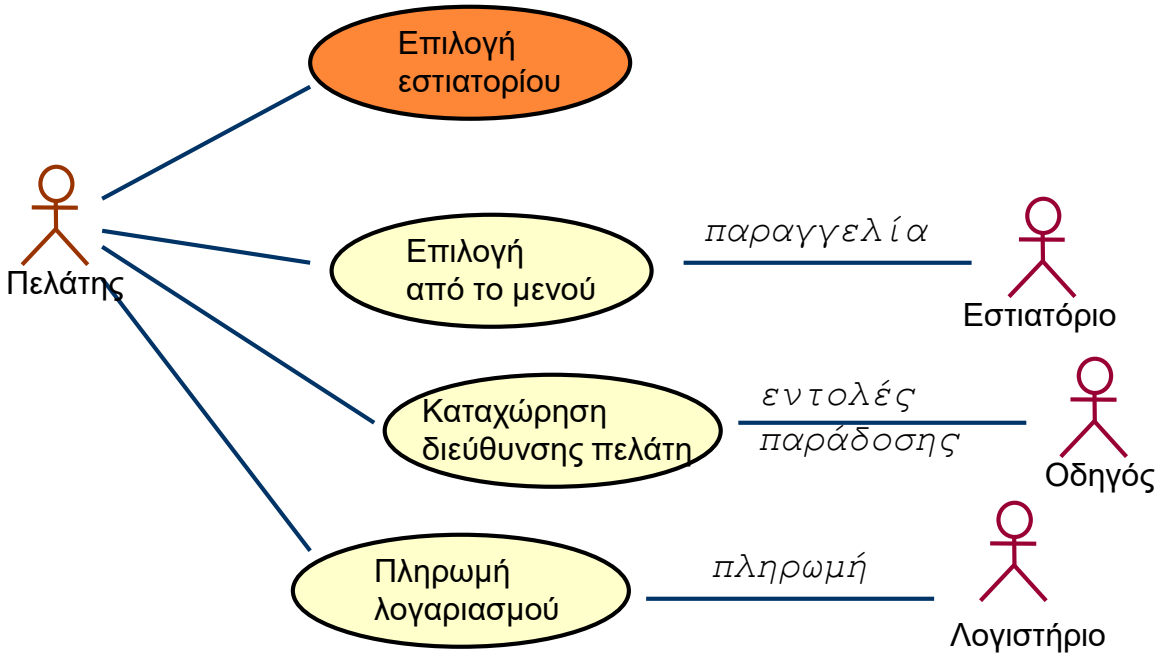


UML - Behaviour Diagrams
Use Case Diagram

- Μια περίπτωση χρήσης αναπαριστά ένα τμήμα λειτουργικότητας του συστήματος που έχει νόημα σε ένα ή περισσότερους εξωτερικούς χρήστες (actors)
- Συμβολισμός
  - Actors αναπαριστώνται σχηματικά με ανθρωπάκια και εκφράζουν ρόλους
  - Η περίπτωση χρήσης αναπαριστάται από έλλειψη
  - Επικοινωνία ή χρήση της περίπτωσης χρήσης από actors αναπαριστάται με γραμμές

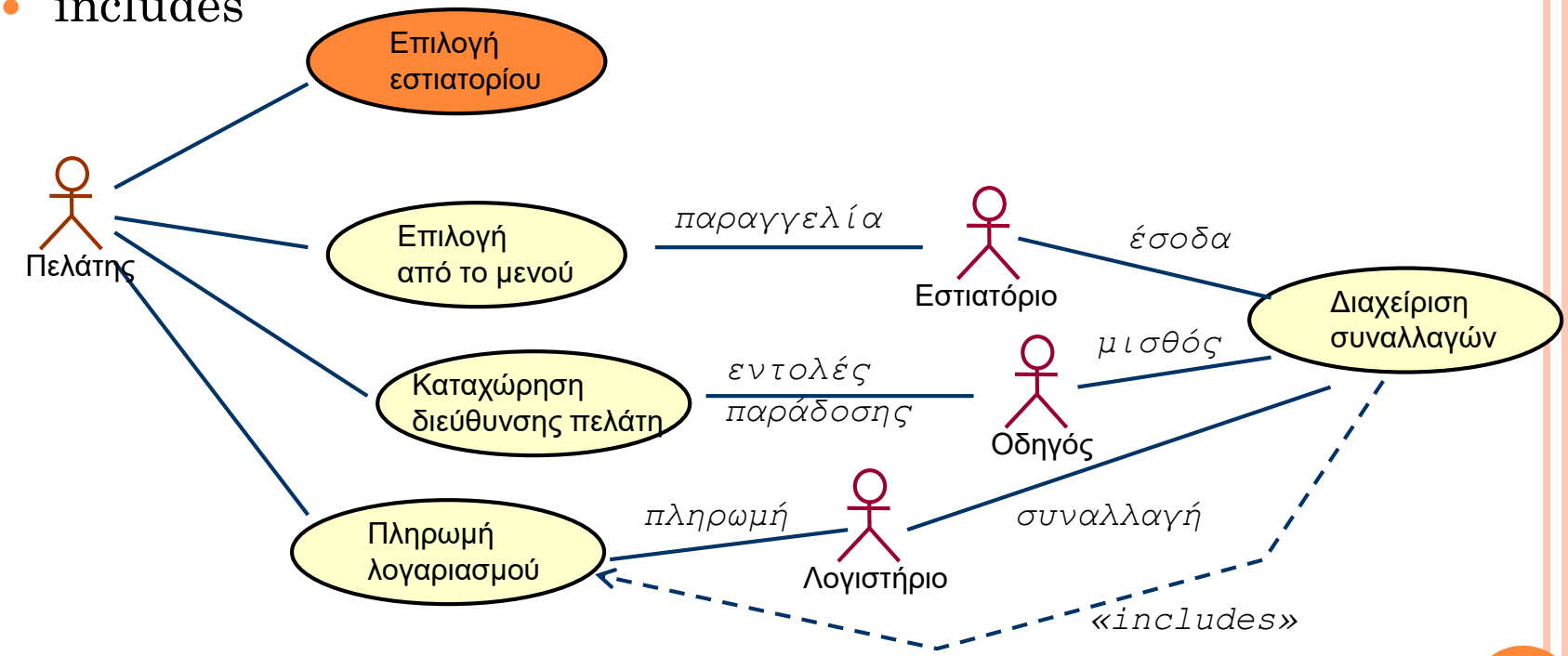
UML - Behaviour Diagrams
Use Case Diagram

- Συνοψίζει μια ενότητα ή σύνολο σχετικών περιπτώσεων χρήσης
- Συνεχής γραμμές δηλώνουν ποιοι actors μέσω ποιών χρήσεων αλληλεπιδρούν με το σύστημα



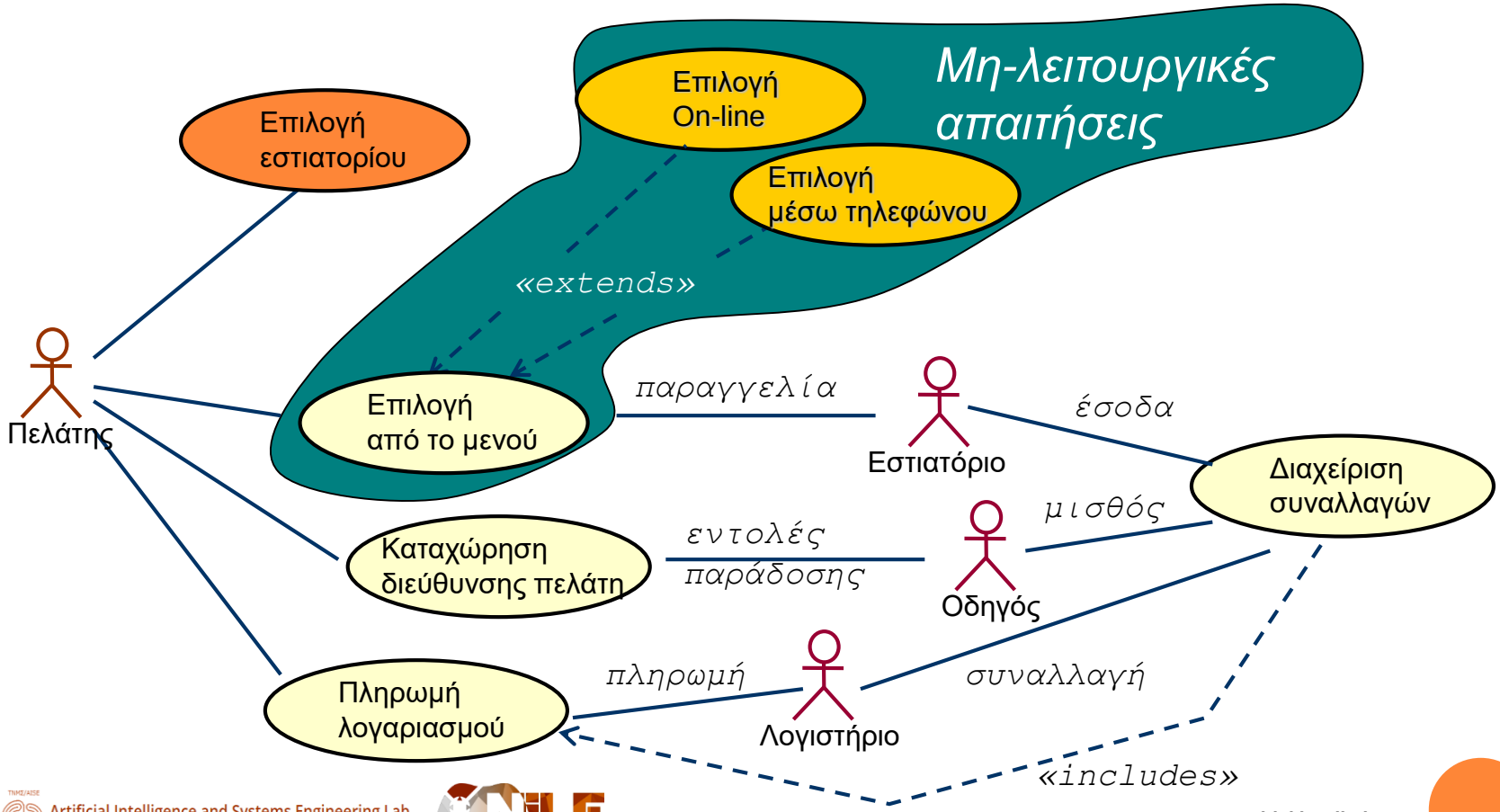
UML - Behaviour Diagrams
Use Case Diagram Extend - Includes

- Διακεκομμένη γραμμή δηλώνει εξαρτήσεις μεταξύ περιπτώσεων χρήσης
- Δύο τύποι εξάρτησης
  - extends
  - includes



UML - Behaviour Diagrams
Use Case Diagram Usage

- Καθορισμό απαιτήσεων συστήματος
  - Λειτουργικές & μη λειτουργικές απαιτήσεις

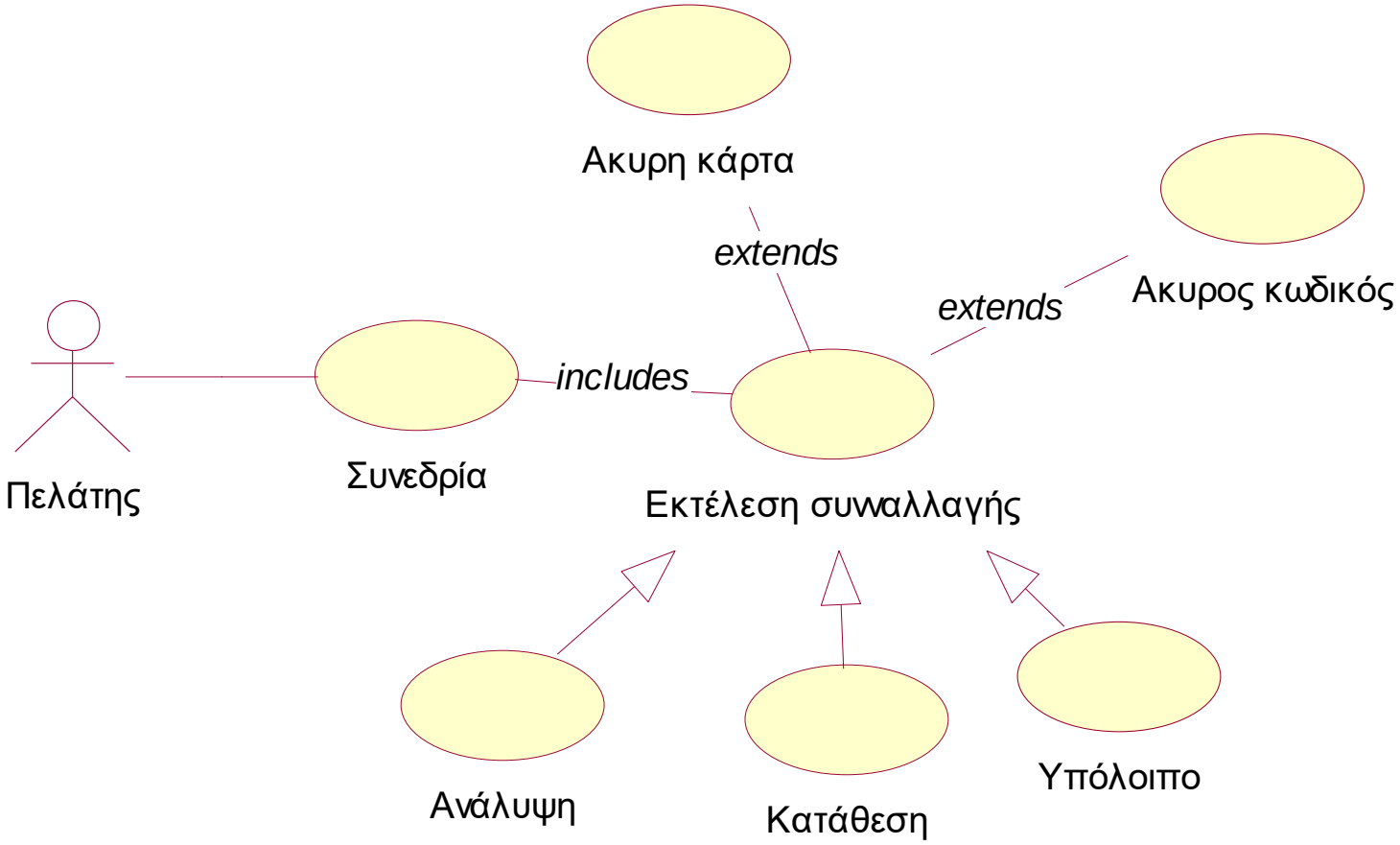


UML - Behaviour Diagrams
Use Case Diagram Usage

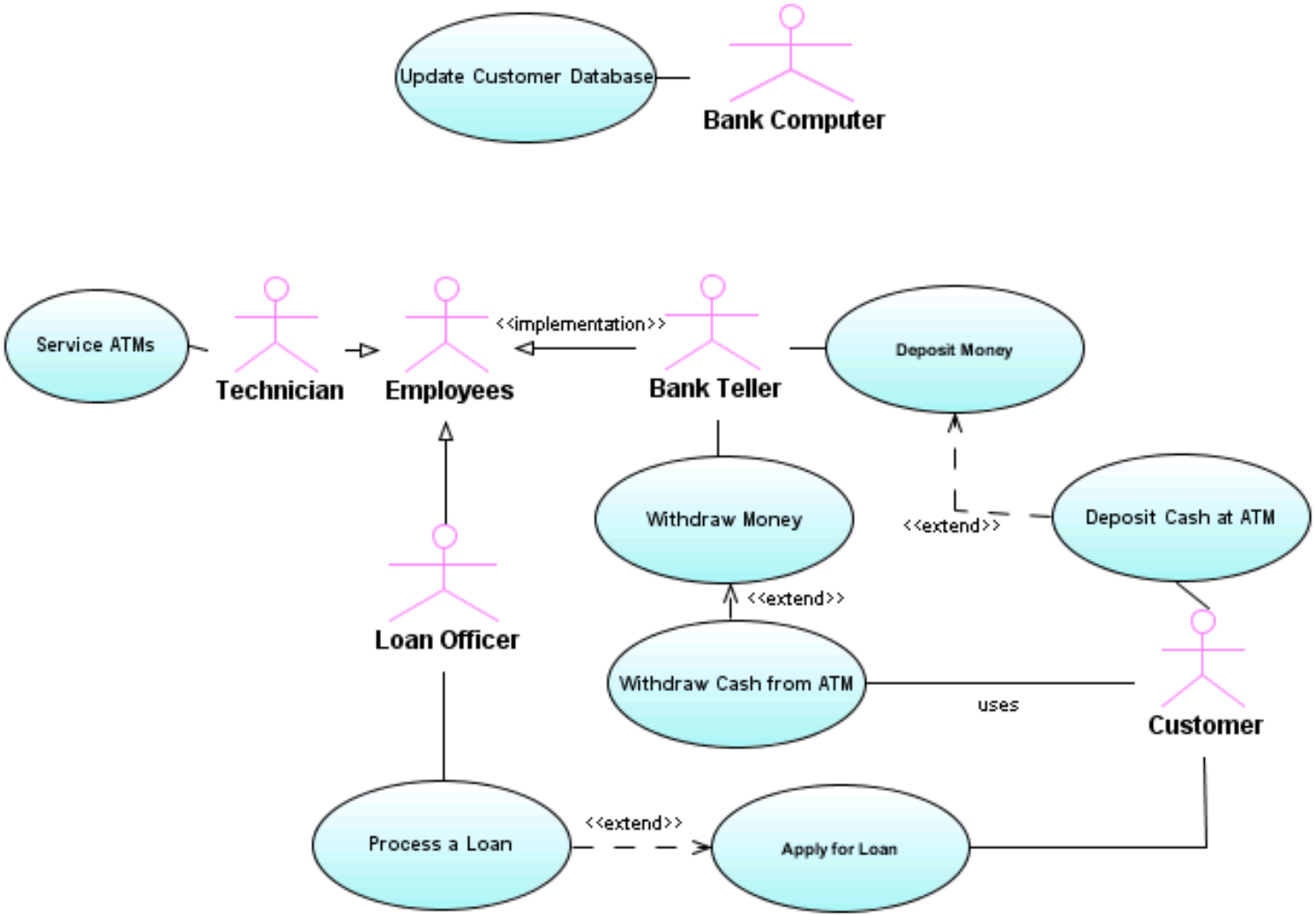
- Επικοινωνία με πελάτες
  - Τα διαγράμματα χρήσης είναι απλά και βοηθούν την ανταλλαγή απόψεων αναφορικά με τον καθορισμό βασικών λειτουργικών και μη-λειτουργικών τμημάτων ενός συστήματος
    - Ειδικότερα για μη-λειτουργικές απαιτήσεις όπως (portability,, scalability, adaptability, etc), τα διαγράμματα περίπτωσης χρήσης είναι ιδιαίτερα χρήσιμα αφού τονίζουν την χρήση του συστήματος και μπορούν να εξηγήσουν δύσκολους όρους
- Καθορισμός σημείων ελέγχου
  - Βοηθούν στον κατοπινό έλεγχο του συστήματος οδηγώντας στην οριοθέτησης σημείων ελέγχου (test cases)

UML - Behaviour Diagrams
Use Case Diagram Example

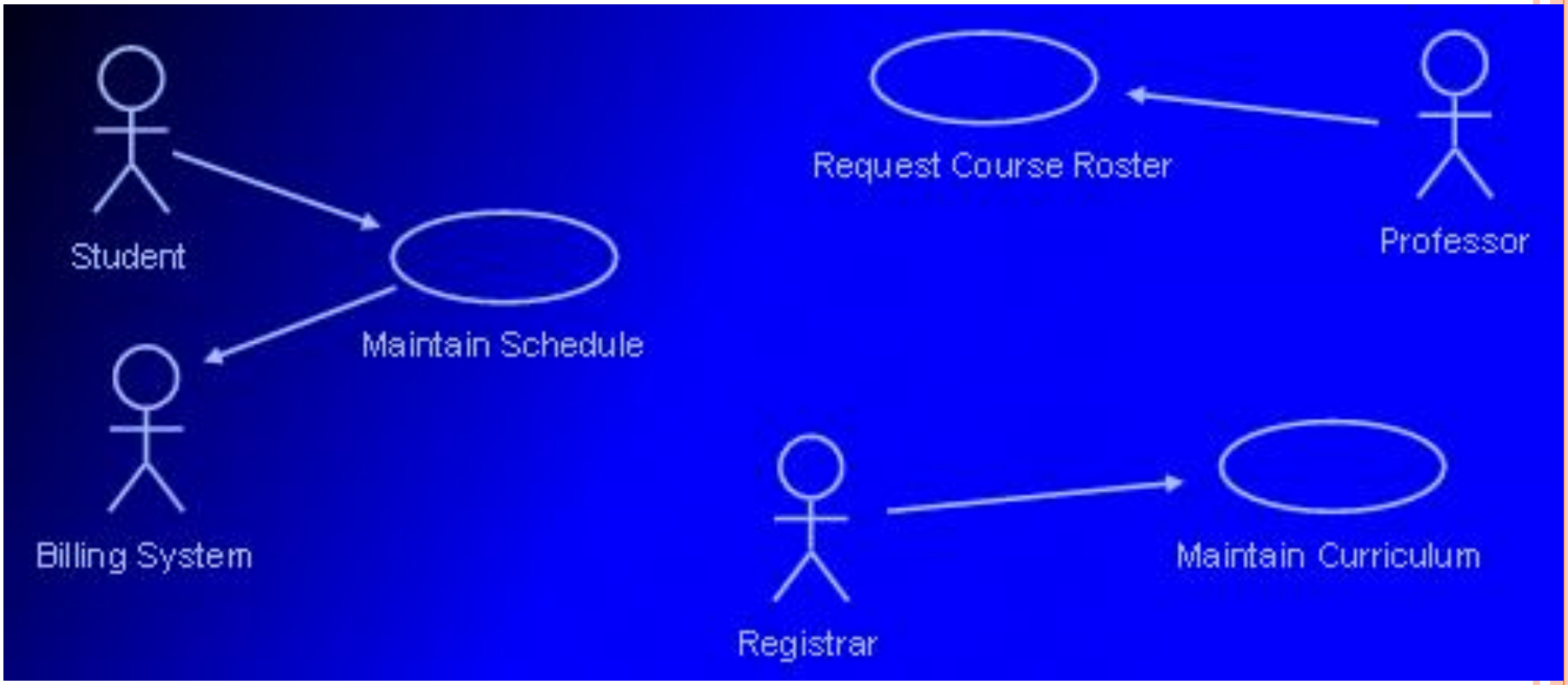
- Περιπτώσεις χρήσης ενός ATM



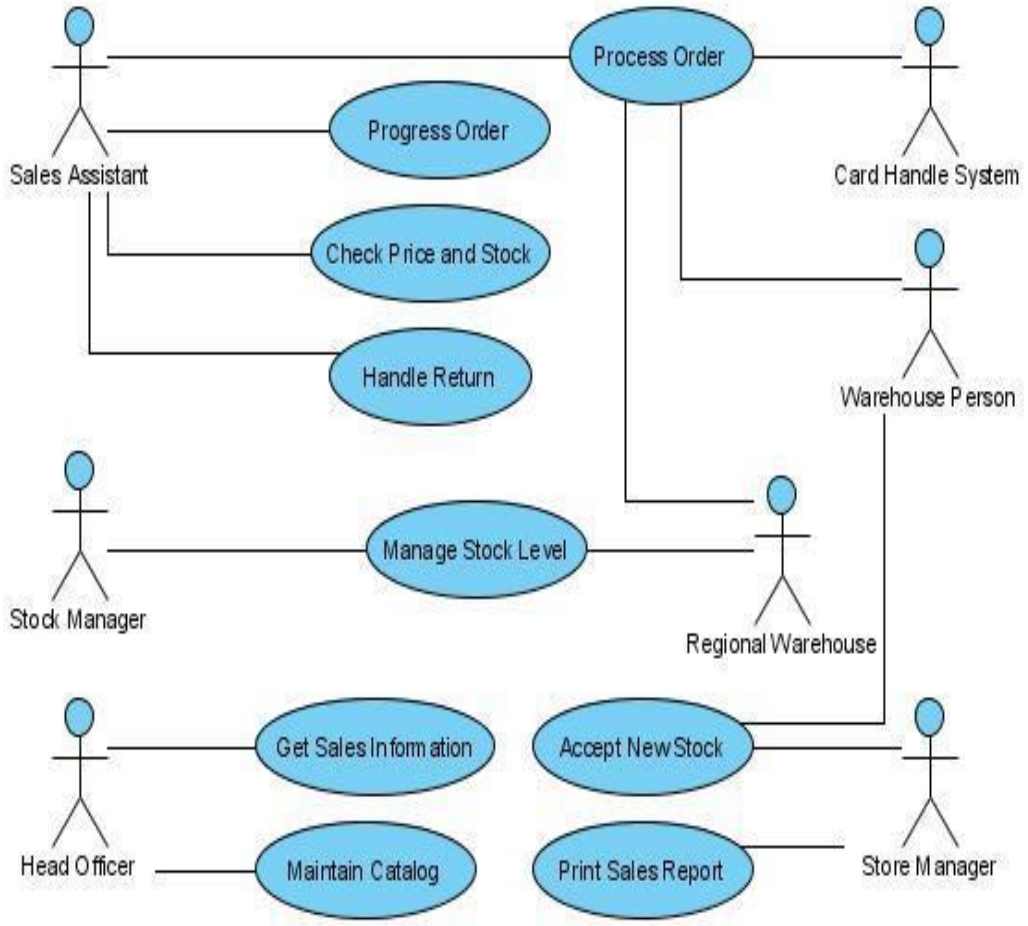
UML - Behaviour Diagrams
Use Case Diagram Example



UML - Behaviour Diagrams
Use Case Diagram Example



UML - Behaviour Diagrams
Use Case Diagram Example



UML - Interaction Diagrams

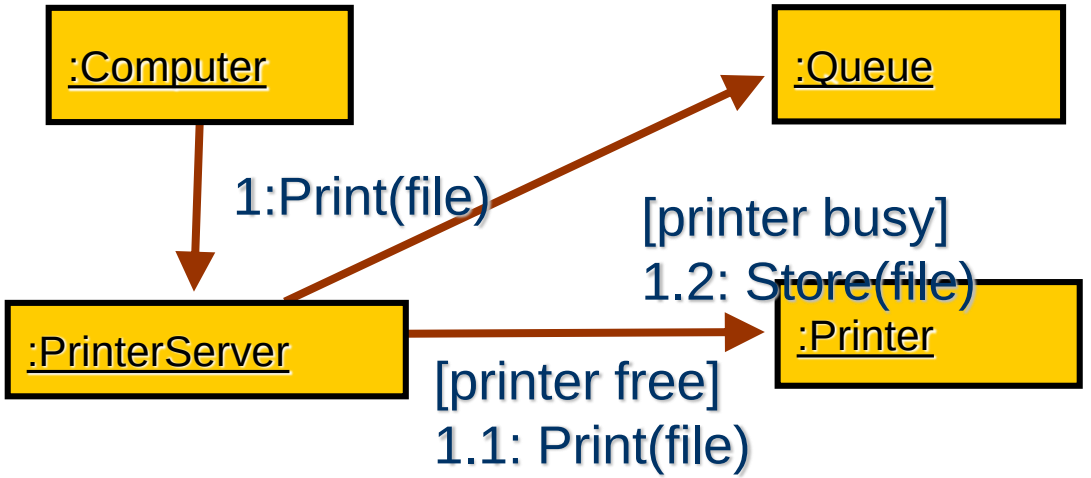
Interaction diagrams

Τα διαγράμματα Interaction, τα οποία μπορεί να θεωρηθούν ως ένα υποσύνολο των διαγραμμάτων behavior, δίνουν έμφαση στον έλεγχο ροής και δεδομένων μεταξύ των οντοτήτων του συστήματος:

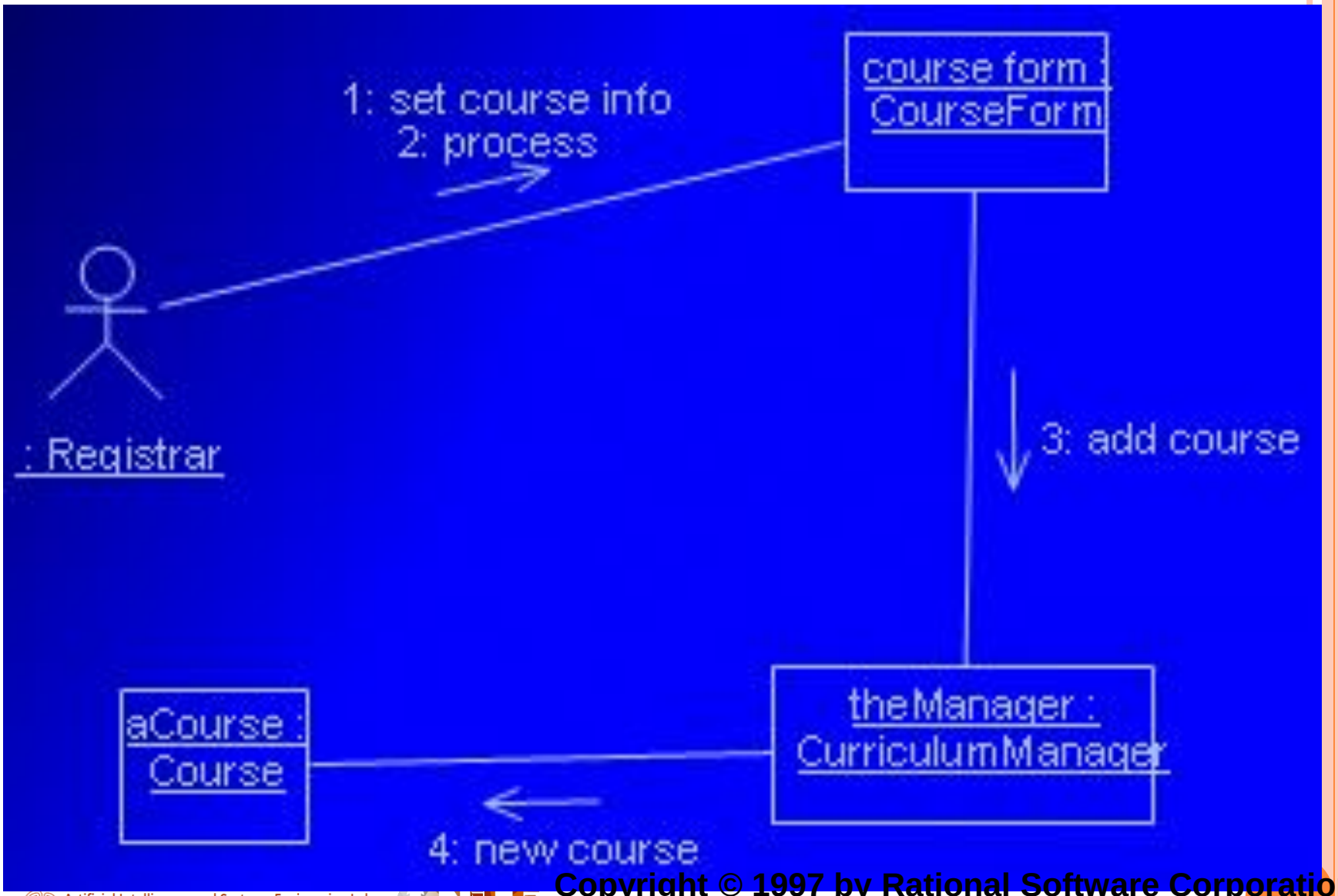
- Communication diagram shows the interactions between objects or parts in terms of sequenced messages. They represent a combination of information taken from Class, Sequence, and Use Case Diagrams describing both the static structure and dynamic behavior of a system.
- Interaction overview diagram: a type of activity diagram in which the nodes represent interaction diagrams.
- Sequence diagram: shows how processes operate with one another and in what order.
  (Δείχνει την ακολουθία μηνυμάτων ανάμεσα στα αντικείμενα, με την πάροδο του χρόνου)
- Timing diagrams are a specific type of interaction diagram, where the focus is on timing constraints.

UML - Behaviour Diagrams
Communication Diagram

- Ένα διάγραμμα συνεργασίας είναι μια εναλλακτική γραφικής αναπαράστασης της πραγματοποίησης ενός σεναρίου χρήσης
- Παρουσιάζει αλληλεπιδράσεις οργανωμένες γύρω από αντικείμενα και τις μεταξύ τους συνδέσεις
- Διαφορά μεταξύ διαγραμμάτων ακολουθίας και συνεργασίας
  - Ενώ το διάγραμμα ακολουθίας δείχνει την ακολουθία μηνυμάτων ανάμεσα στα αντικείμενα, με την πάροδο του χρόνου, το διάγραμμα συνεργασίας δίνει μεγαλύτερη έμφαση στο περιεχόμενο και στις σχέσεις μεταξύ των αντικειμένων



UML - Behaviour Diagrams
Communication Diagram



UML - Interaction Diagrams
Interaction overview Diagram

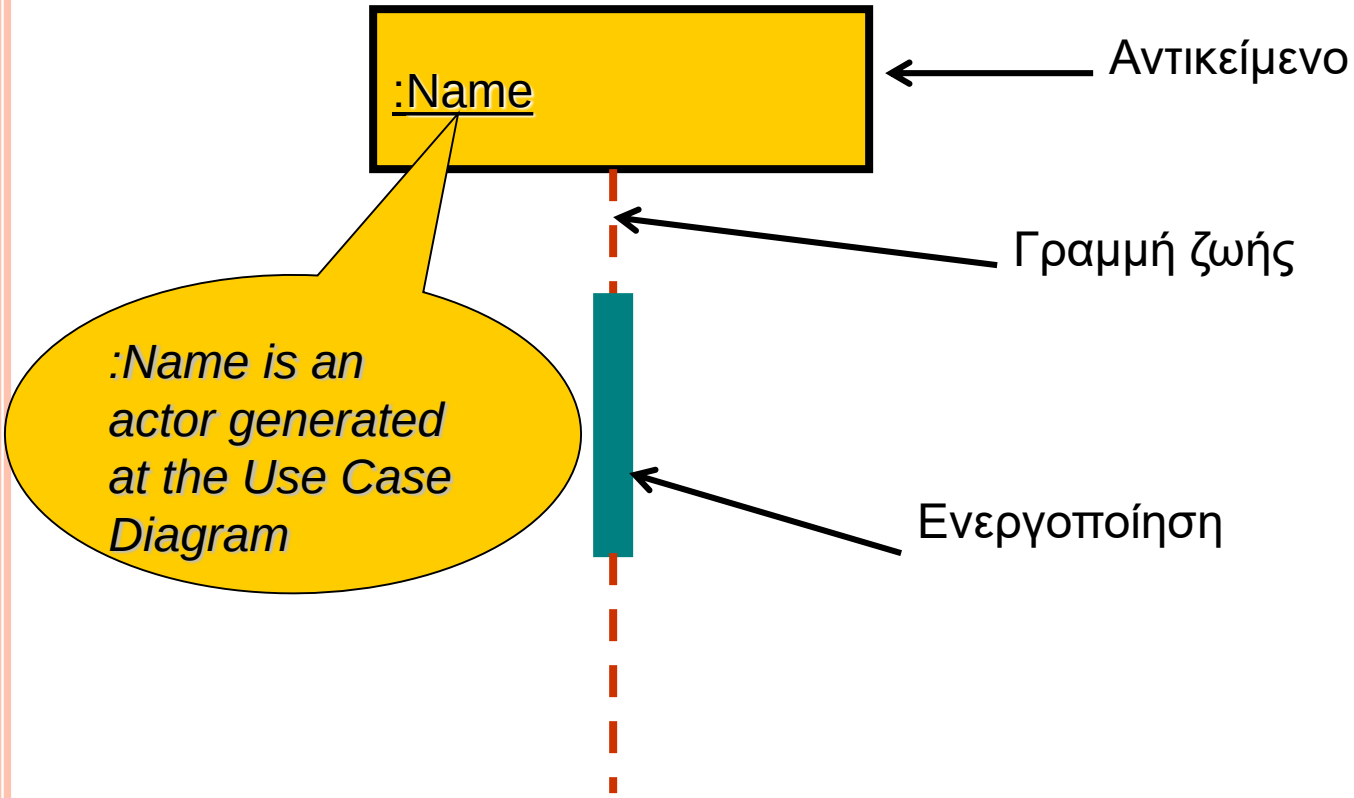
- Είναι δυναμικά διαγράμματα που καταδεικνύουν τον τρόπο που τα αντικείμενα επικοινωνούν και συνεργάζονται
- Δύο κατηγορίες
  - Διάγραμμα ακολουθίας
  - Διάγραμμα συνεργασίας

UML - Behaviour Diagrams
Sequence Diagram

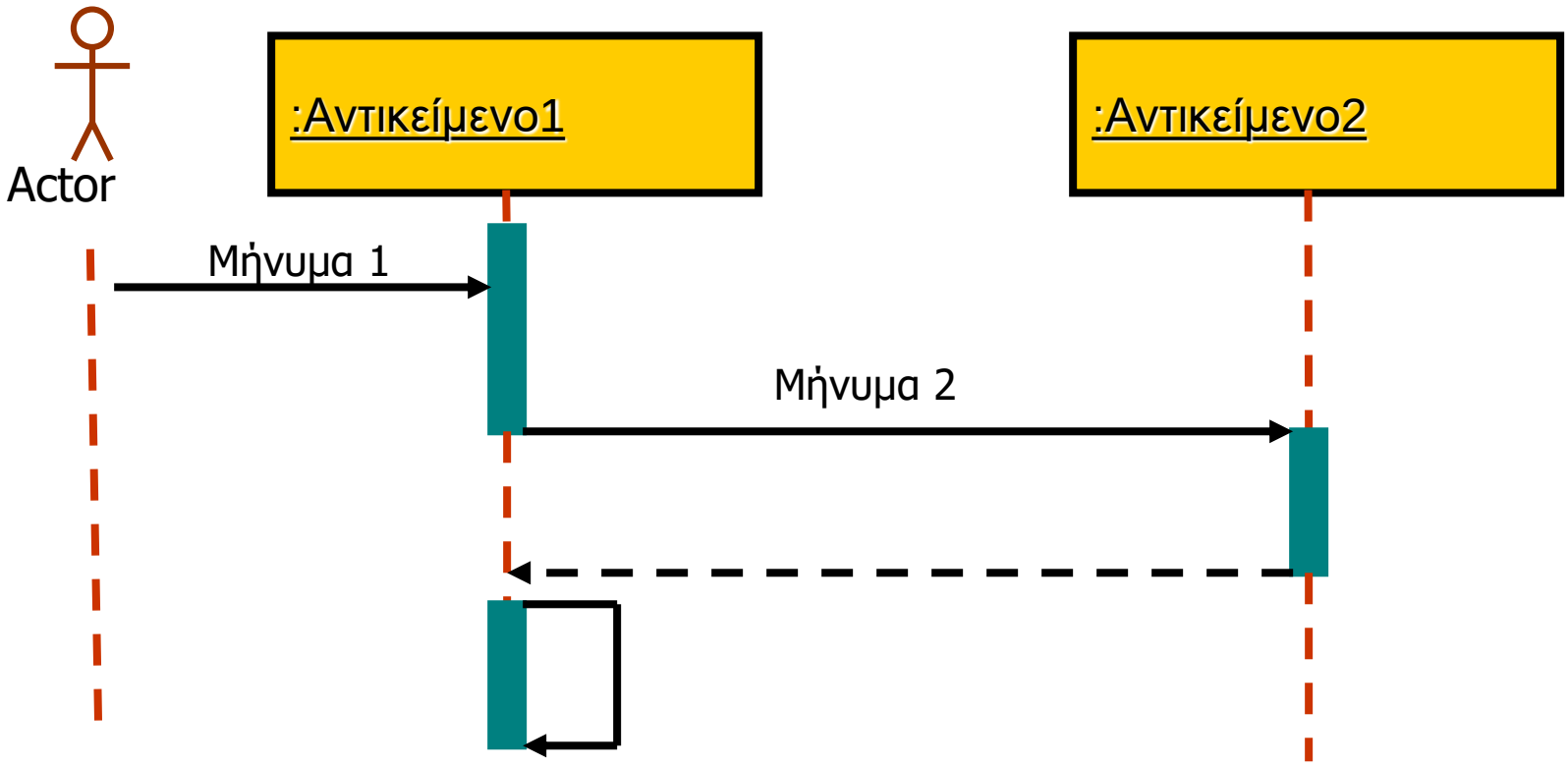
- Ένα διάγραμμα ακολουθίας δείχνει τι μηνύματα ανταλλάσσονται και πότε μεταξύ αντικειμένων
  - Τα μηνύματα καταγράφονται χρονολογικά
  - Τα αντικείμενα που εμπλέκονται καταγράφονται από αριστερά προς τα δεξιά
  - Μηνύματα στέλνονται από αριστερά προς τα δεξιά

UML - Behaviour Diagrams
Sequence Diagram Components

- Η γραμμή ζωής καταγράφει τι γίνεται σε ένα αντικείμενο σε χρονολογική σειρά

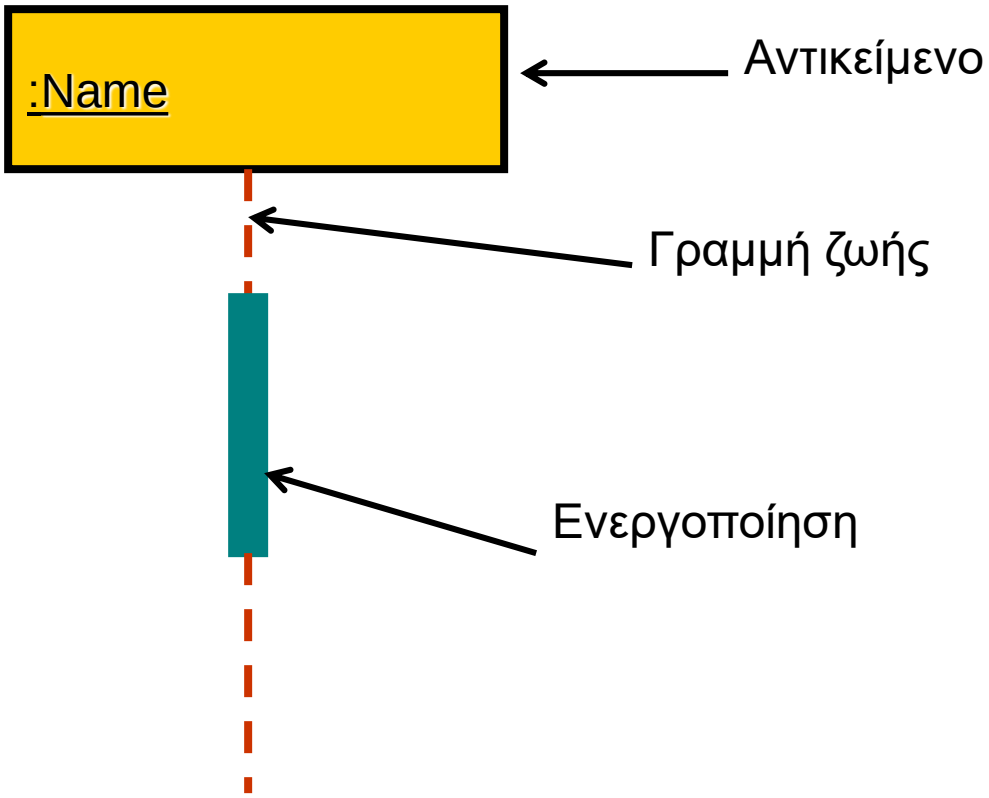


- Μηνύματα καταδεικνύουν επικοινωνία μεταξύ διαφορετικών ενεργών αντικειμένων



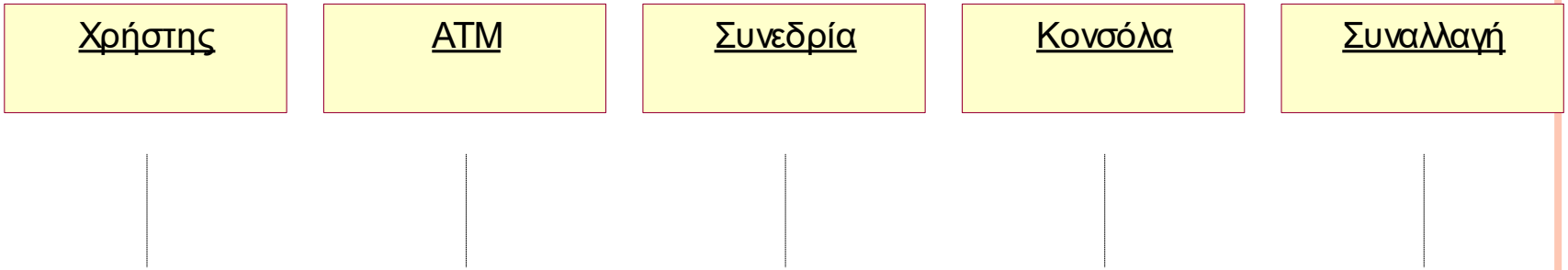
UML - Behaviour Diagrams
Sequence Diagram Messages

- Η γραμμή ζωής καταγράφει τι γίνεται σε ένα αντικείμενο σε χρονολογική σειρά

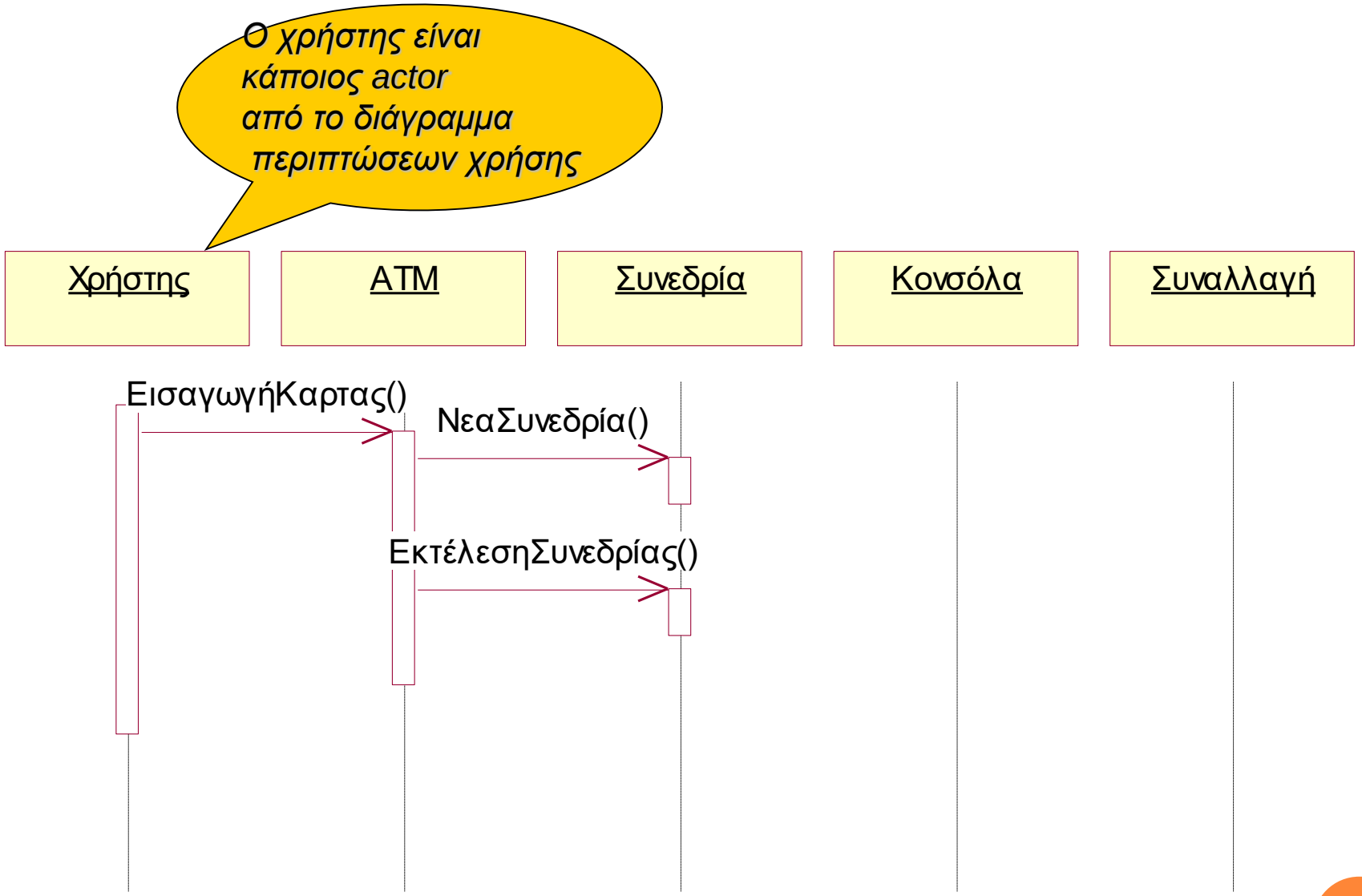


UML - Behaviour Diagrams
Sequence Diagram

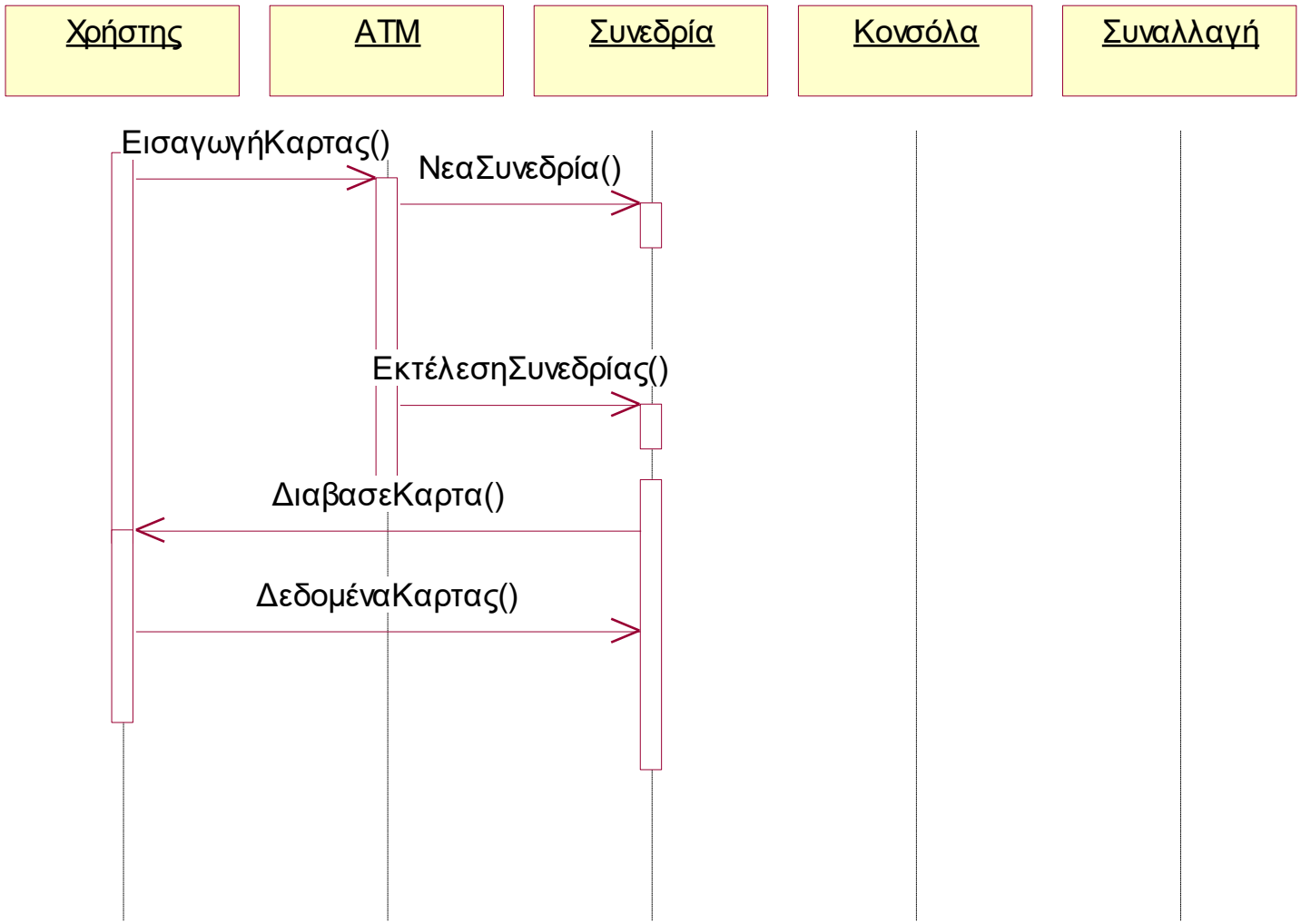
- Στην περίπτωση του ATM



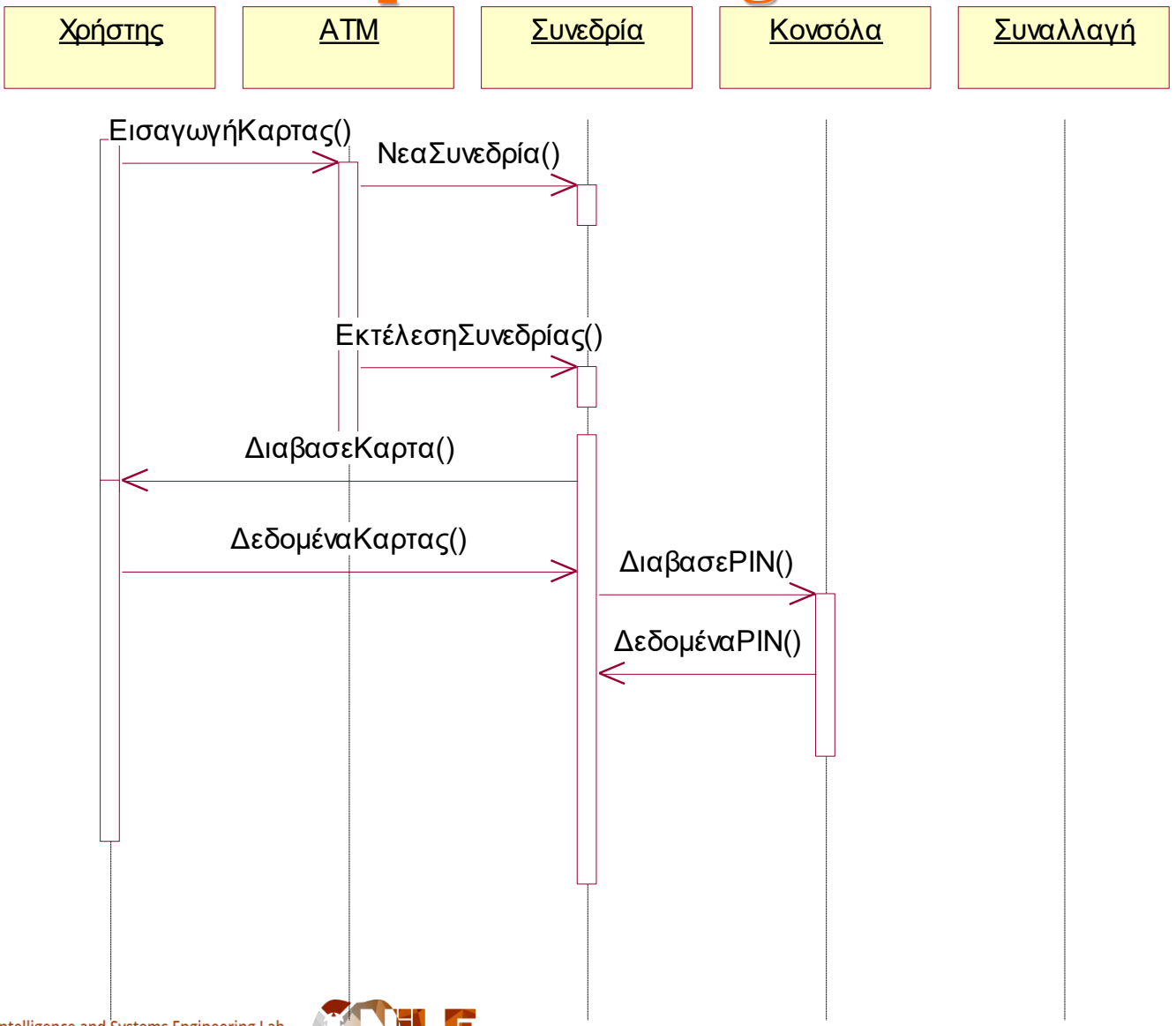
UML - Behaviour Diagrams
Sequence Diagram



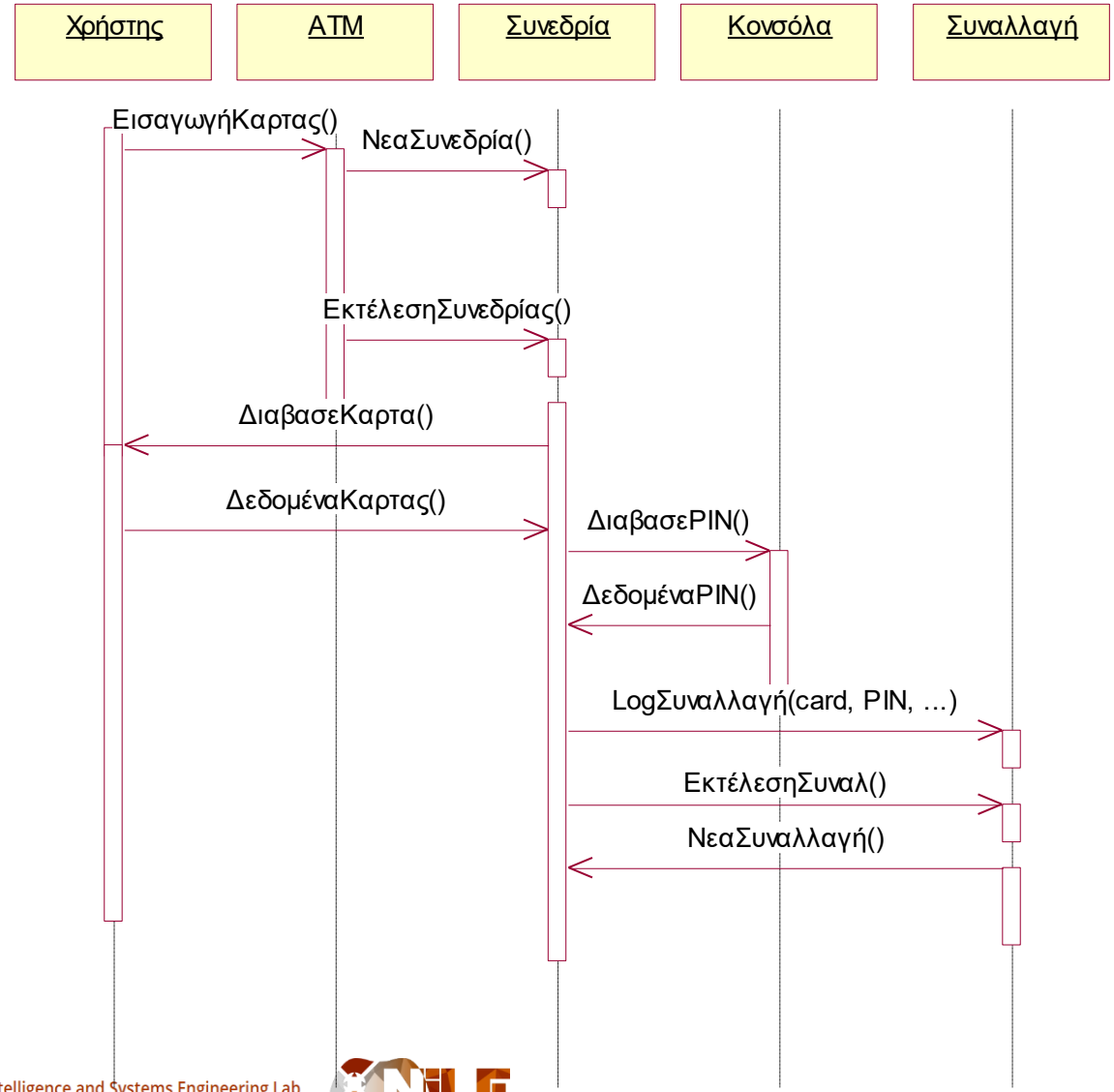
UML - Behaviour Diagrams
Sequence Diagram



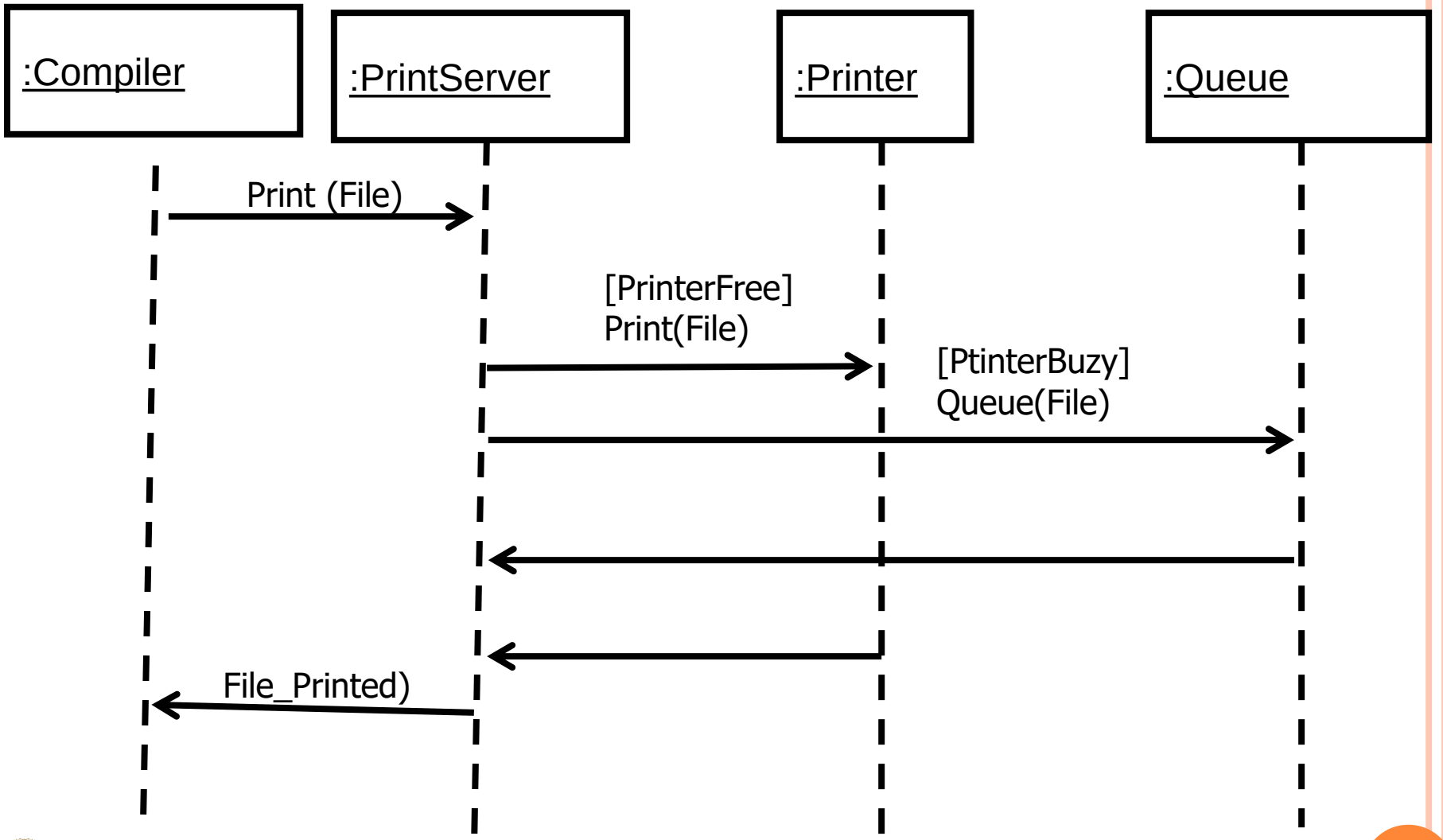
UML - Behaviour Diagrams
Sequence Diagram

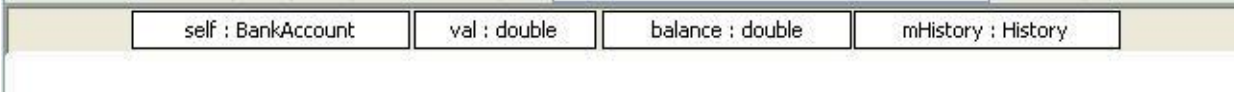


UML - Behaviour Diagrams
Sequence Diagram

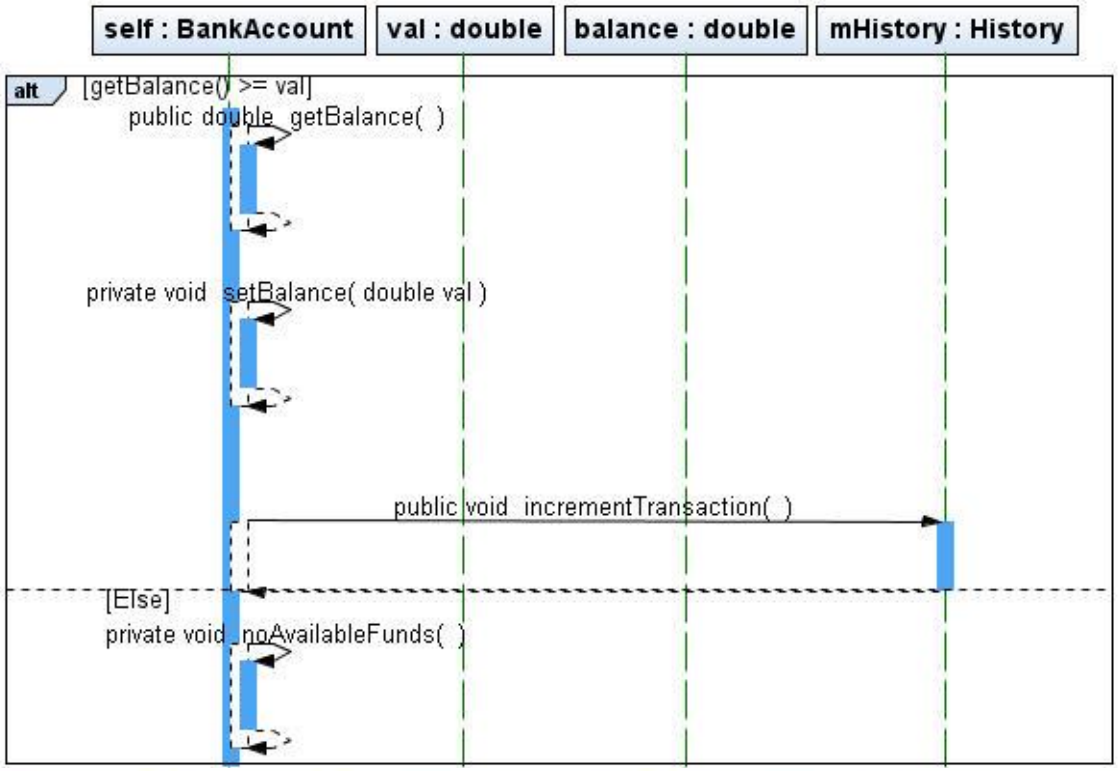


UML - Behaviour Diagrams
Sequence Diagram Example



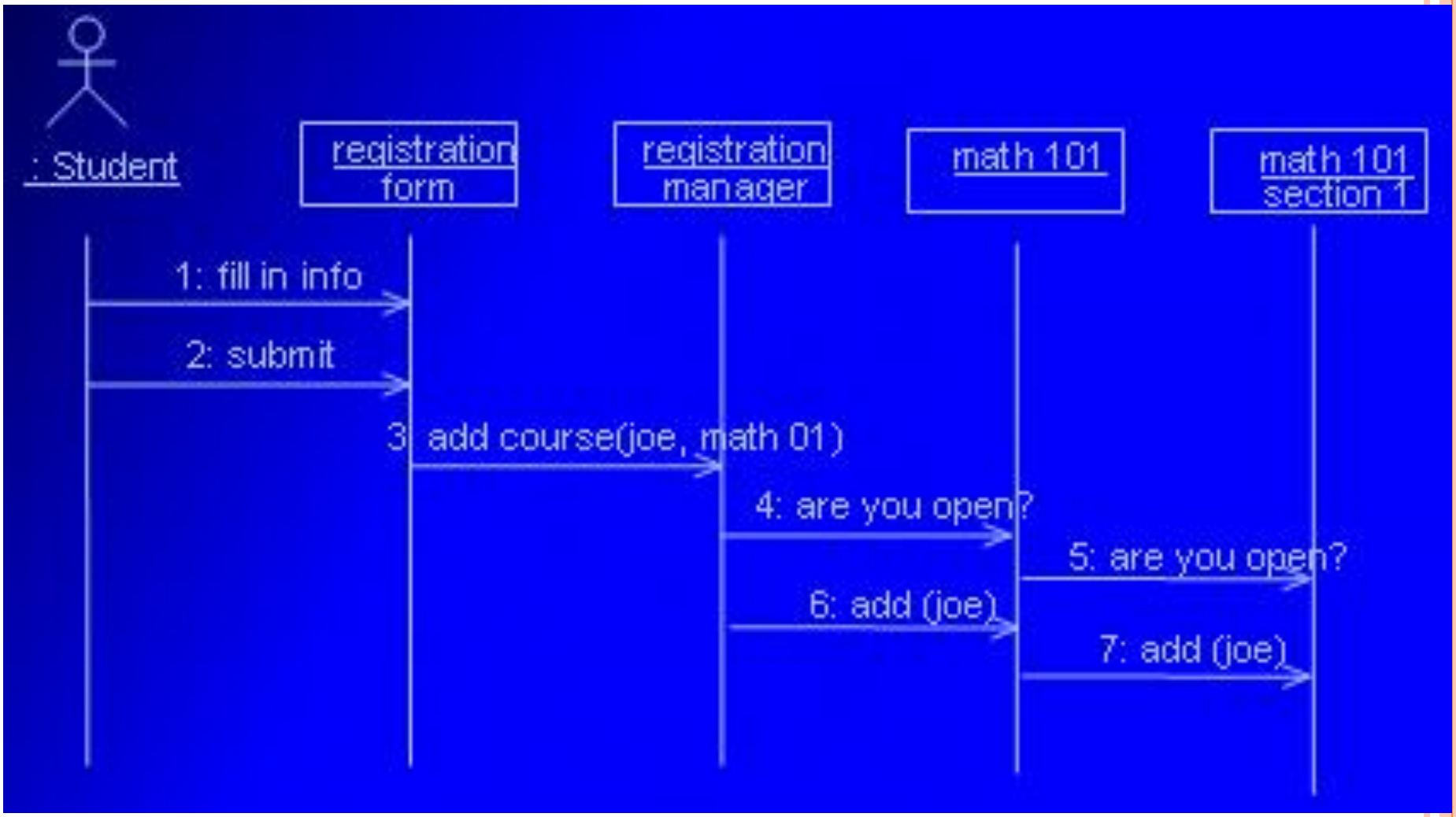


UML - Behaviour Diagrams Sequence Diagram Example

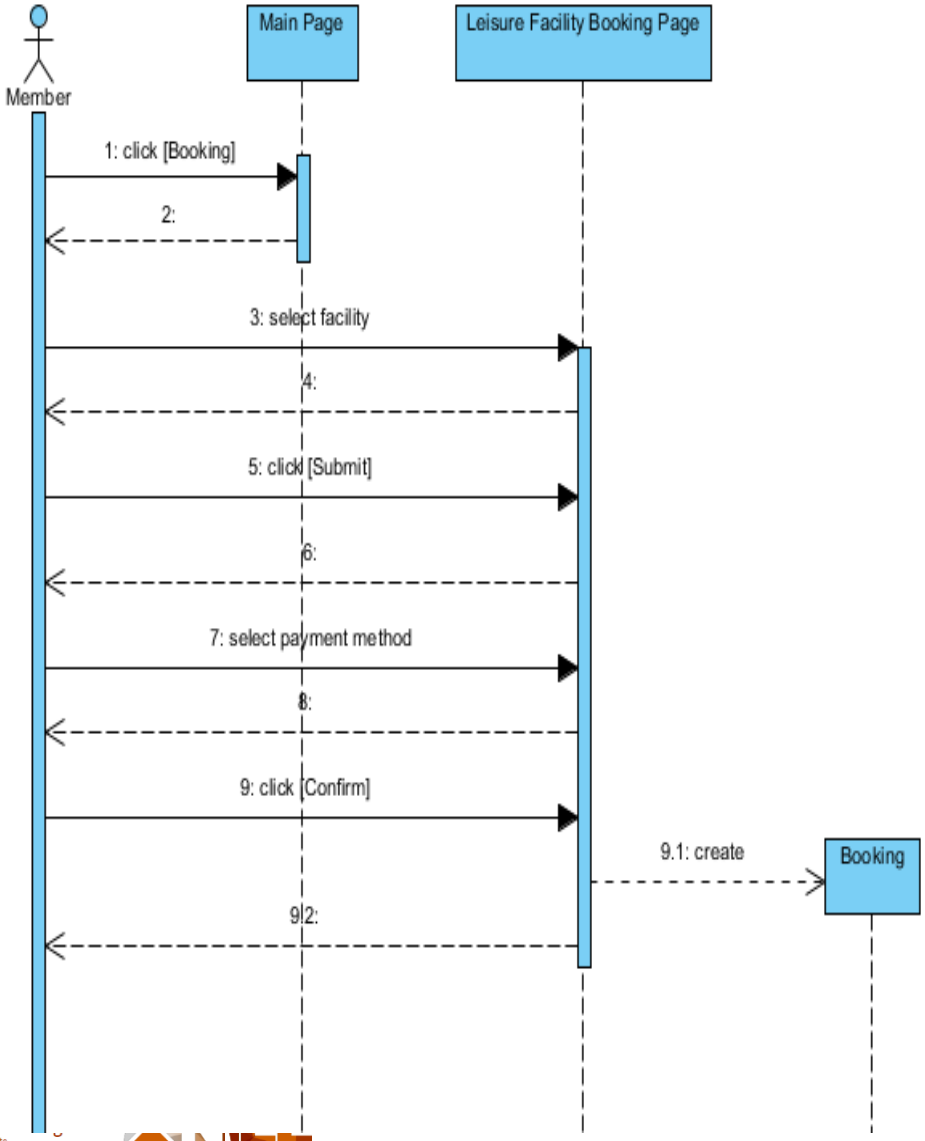



```
public void withdraw(double val) throws bankpack.NoAvailableFundsException{

    if (getBalance() >= val){
        setBalance(balance - val);
        mHistory.incrementTransaction();
    }
    else
        noAvailableFunds();
}
```

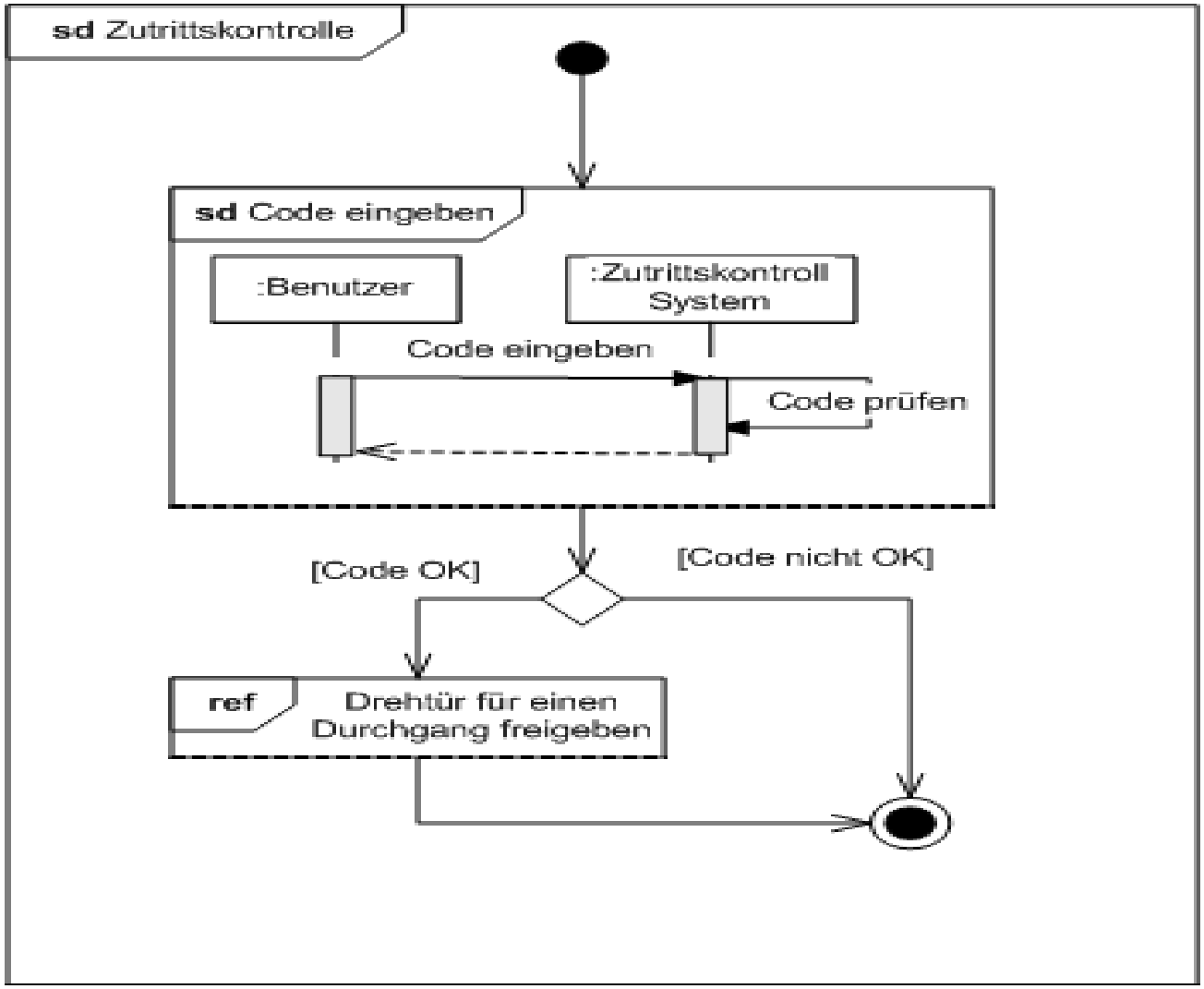
UML - Behaviour Diagrams
Sequence Diagram Example



UML - Behaviour Diagrams
Sequence Diagram Example in Visual Paradigm

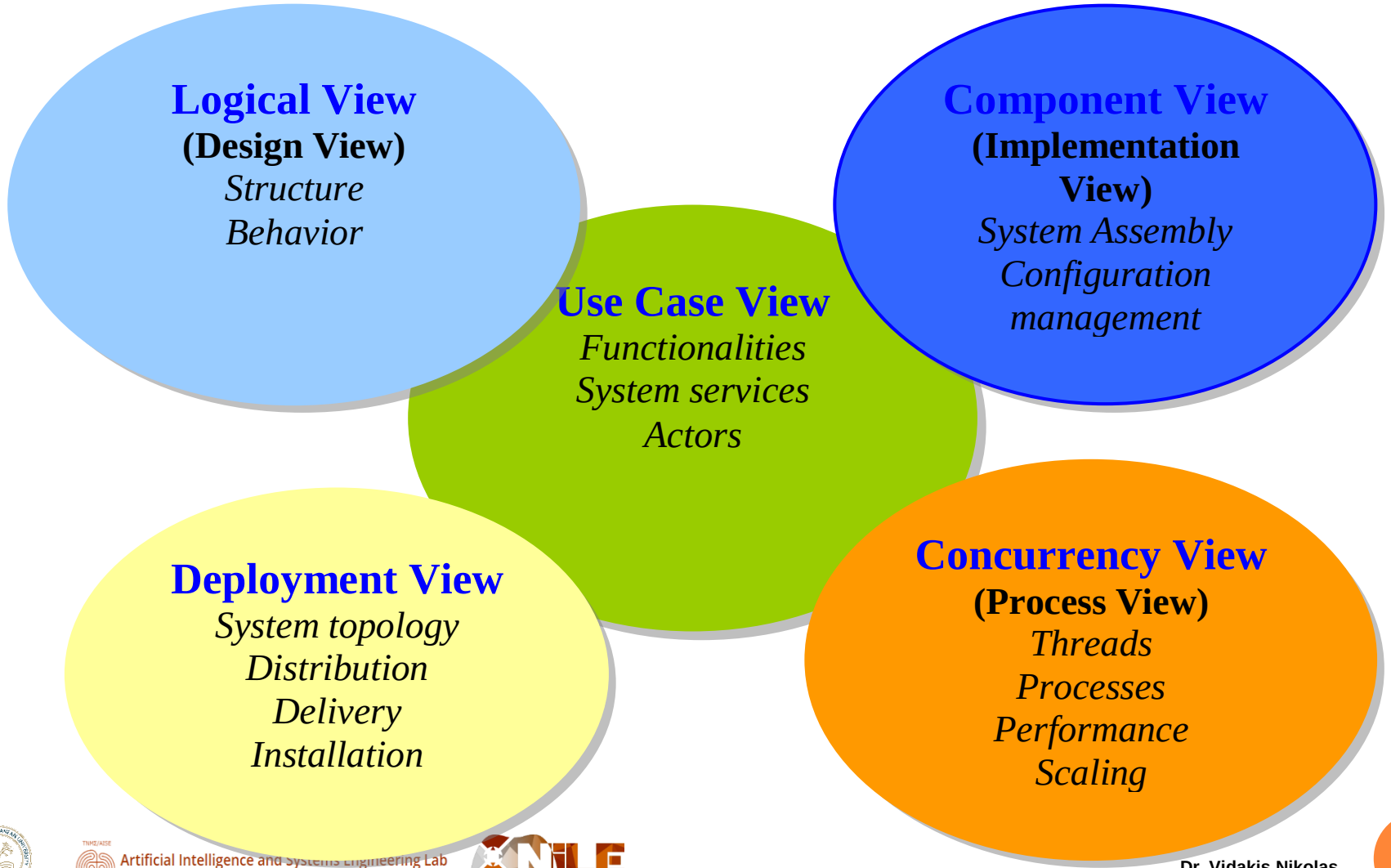


UML - Interaction Diagrams
Interaction overview Diagram



UML - VIEWS

4+1 View / Όψη / Σκοπιά of UML Diagrams



UML - VIEWS

Use Case View

- **Επιδεικνύει** την λειτουργικότητα του συστήματος όπως αυτή εννοείται από εξωτερικούς actors.
- Οι Actors μπορεί να είναι χρήστες ή άλλα συστήματα.
- **Χρησιμοποιεί / Περιγράφεται** από διαγράμματα *use case* και *activity*.
- Είναι η κεντρική προσέγγιση / σκοπιά (view) η οποία κατευθύνει την υλοποίηση τον υπόλοιπων.
- **Χρησιμοποιείται** από customers, designers, developers, testers.

UML - VIEWS

Logical View

- **Επιδεικνύει** πως σχεδιάζεται η λειτουργικότητα του συστήματος.
- **Χρησιμοποιεί** διαγράμματα *class* και *object* για να αναπαραστήσει την στατική δομή του συστήματος.
- **Χρησιμοποιεί** διαγράμματα *state*, *sequence*, *collaboration*, και *activity* για να αναπαραστήσει την δυναμική συμπεριφορά του συστήματος.
- **Χρησιμοποιείται** από designers and developers.

UML - VIEWS

Component View

- **Επιδεικνύει** την οργάνωση των κομματιών (component) κώδικα και την αλληλοεξάρτιση τους.
- **Χρησιμοποιεί / Περιγράφεται** από διαγράμματα *component*.
- **Χρησιμοποιείται** από developers.

Modelling Languages UML - Views

Concurrency View

- **Αναφέρεται** σε θέματα επικοινωνίας και συγχρονισμού συστημάτων concurrent.
- **Χρησιμοποιεί / Περιγράφεται** από διαγράμματα *state*, *sequence*, *collaboration*, *activity*, *deployment*, και *component*.
- **Χρησιμοποιείται** από developers και system integrators.

Modelling Languages UML - Views

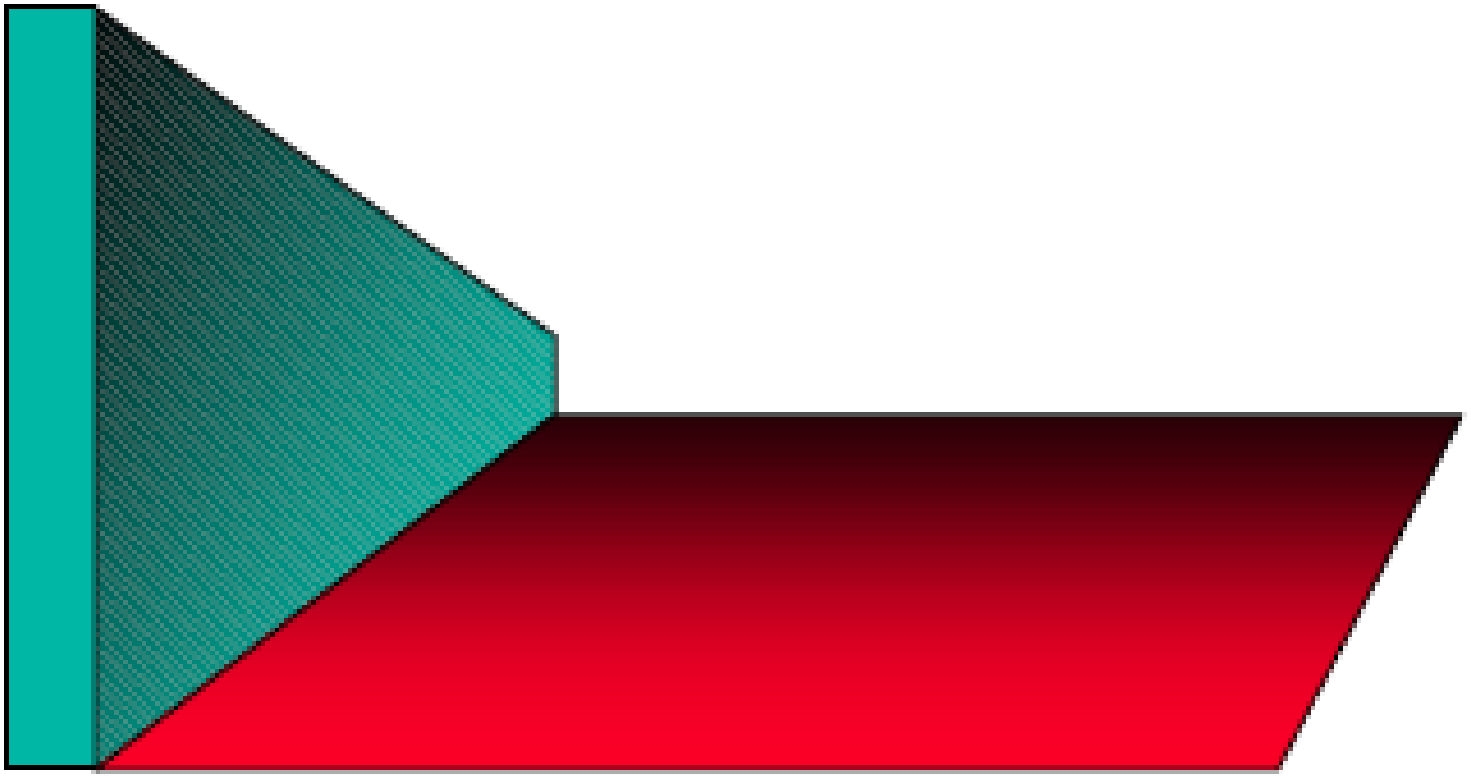
Deployment View.

- **Επιδεικνύει** την ανάπτυξη (deployment) ενός συστήματος μέσα στην αρχιτεκτονική σχεδίαση του με computers και devices.
- **Περιγράφεται** από διαγράμματα *deployment* .
- **Χρησιμοποιείται** από developers, system integrators, και testers.

UML

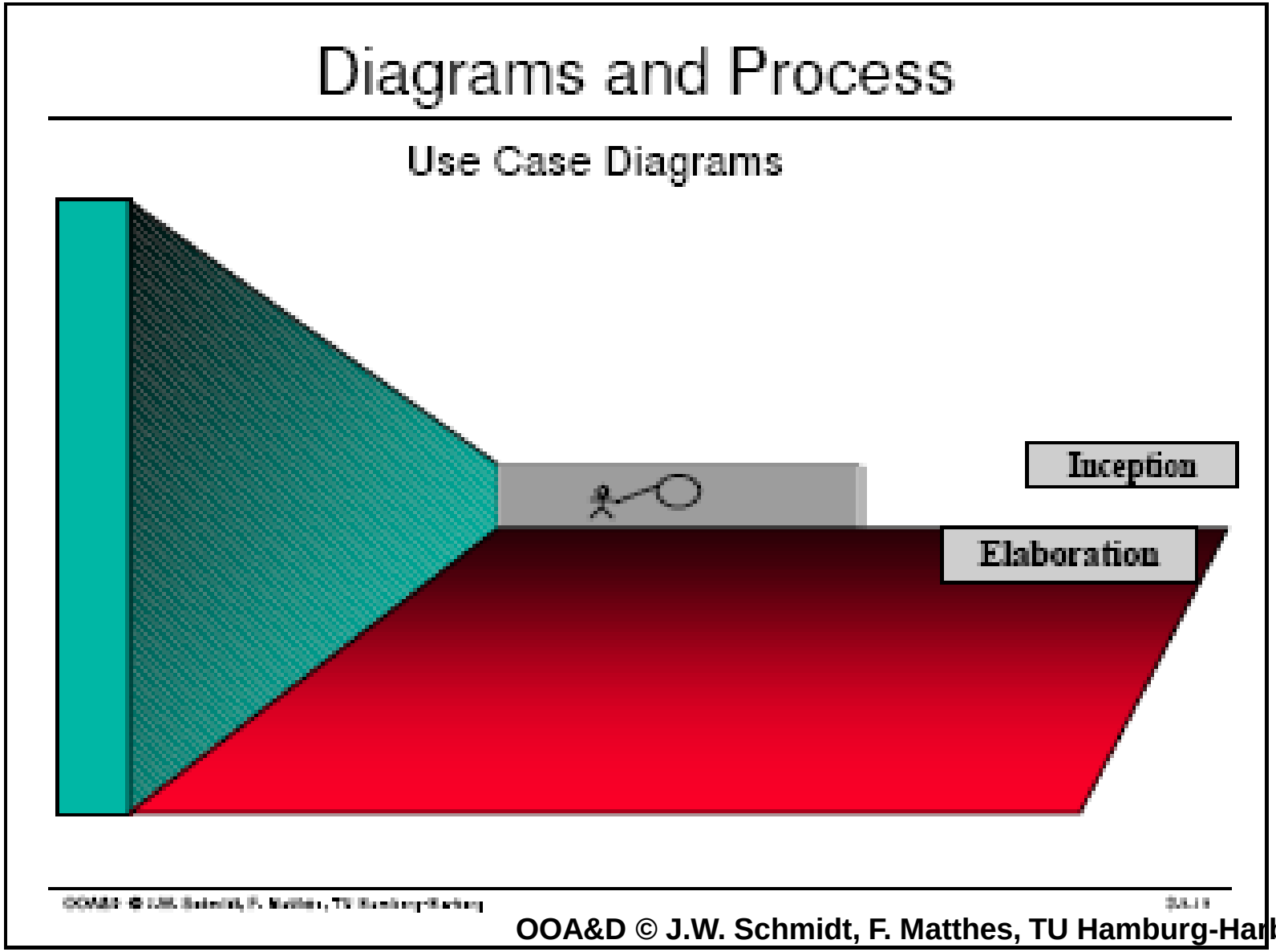
Exemplar usage of Diagrams

Diagrams and Process

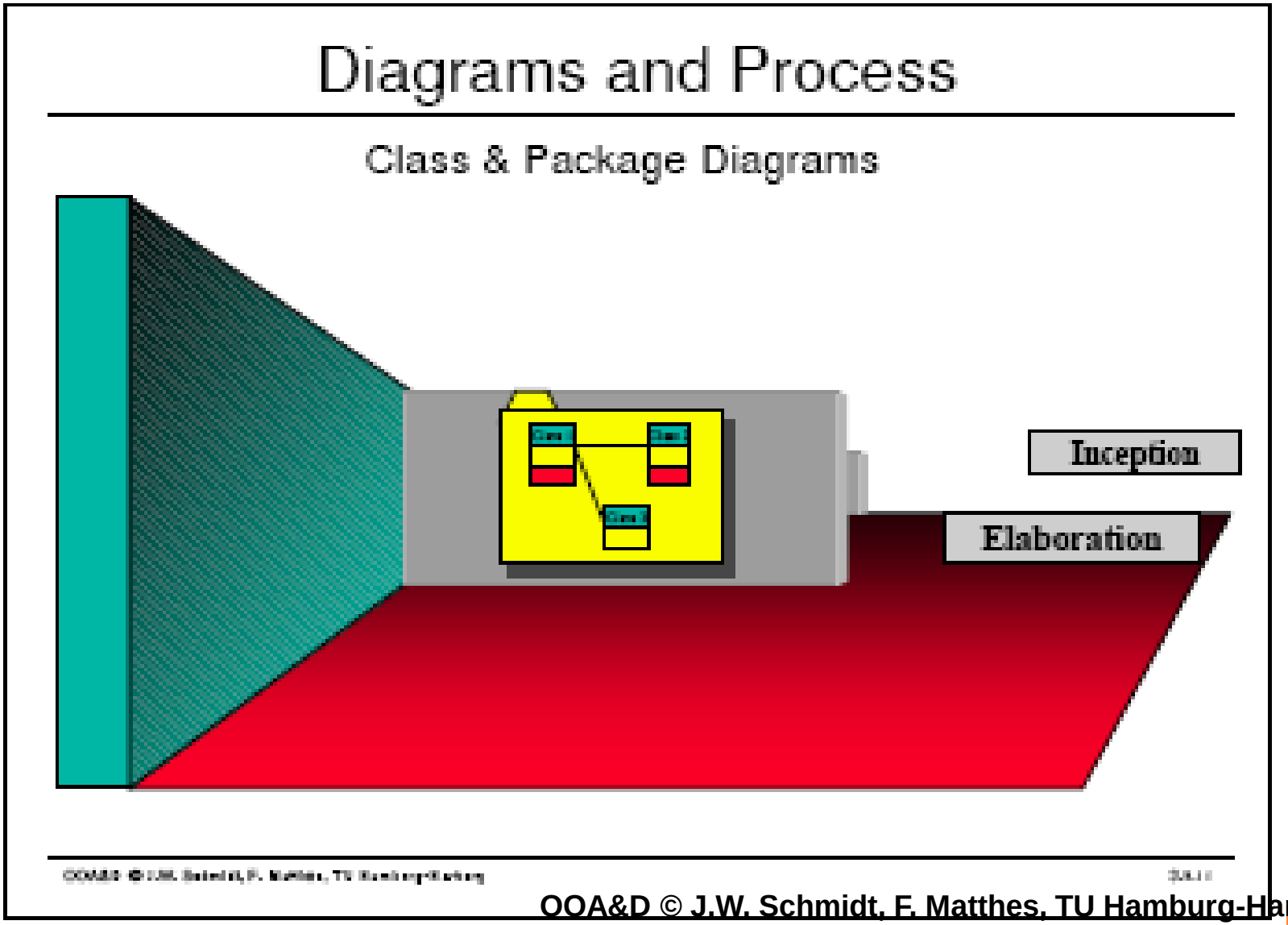


UML

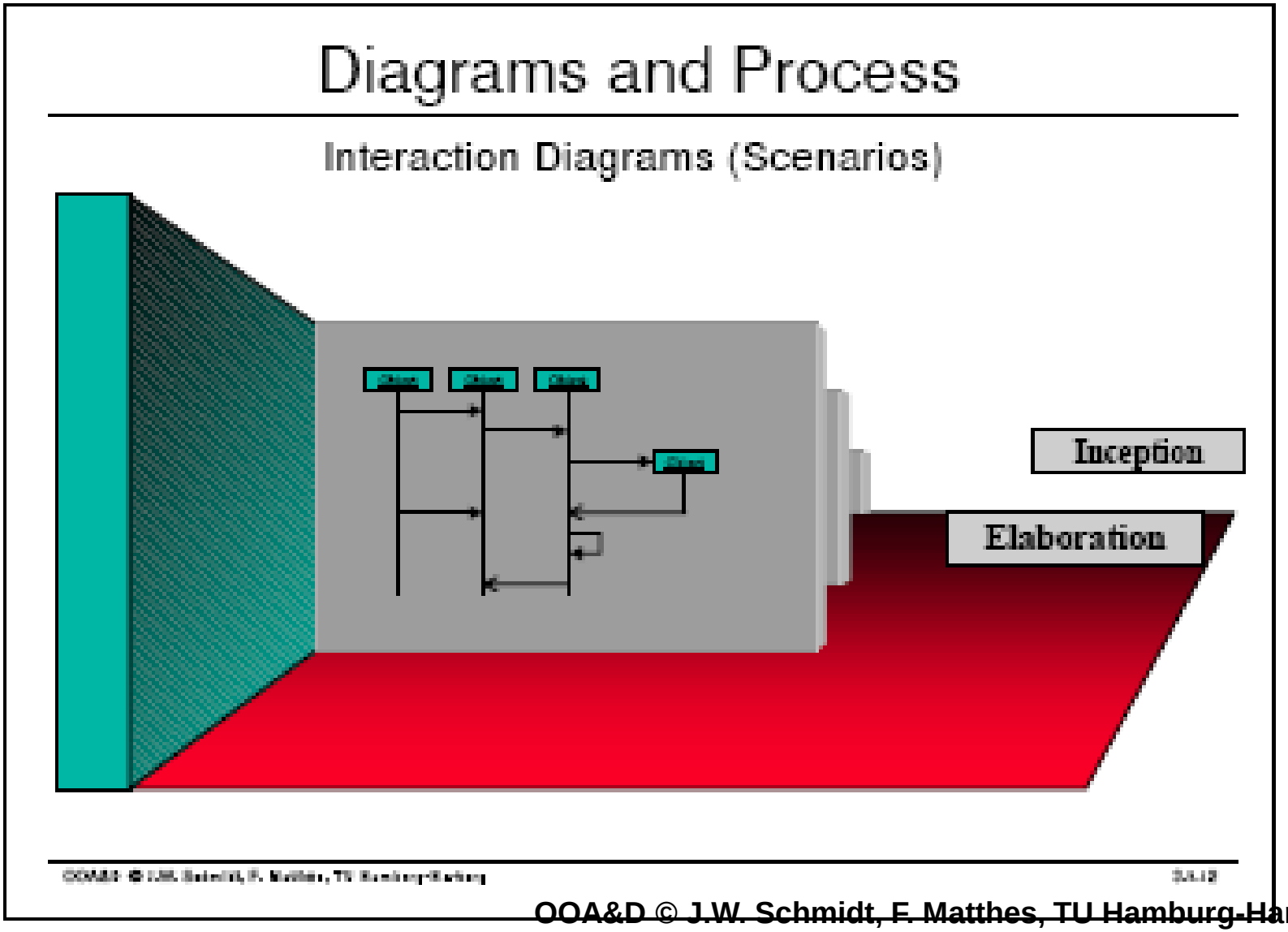
Exemplar usage of Diagrams



UML
Exemplar usage of Diagrams

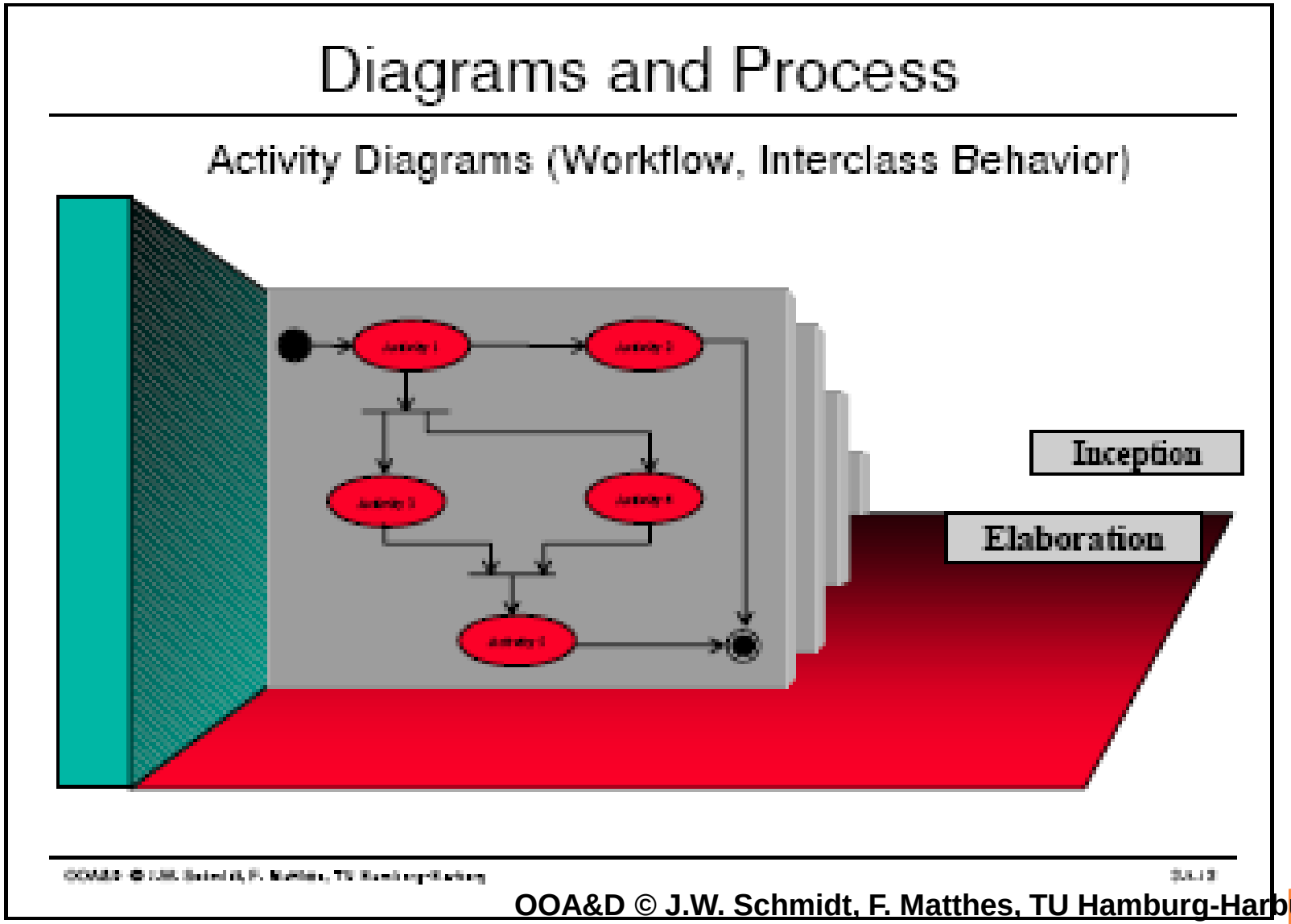


UML
Exemplar usage of Diagrams

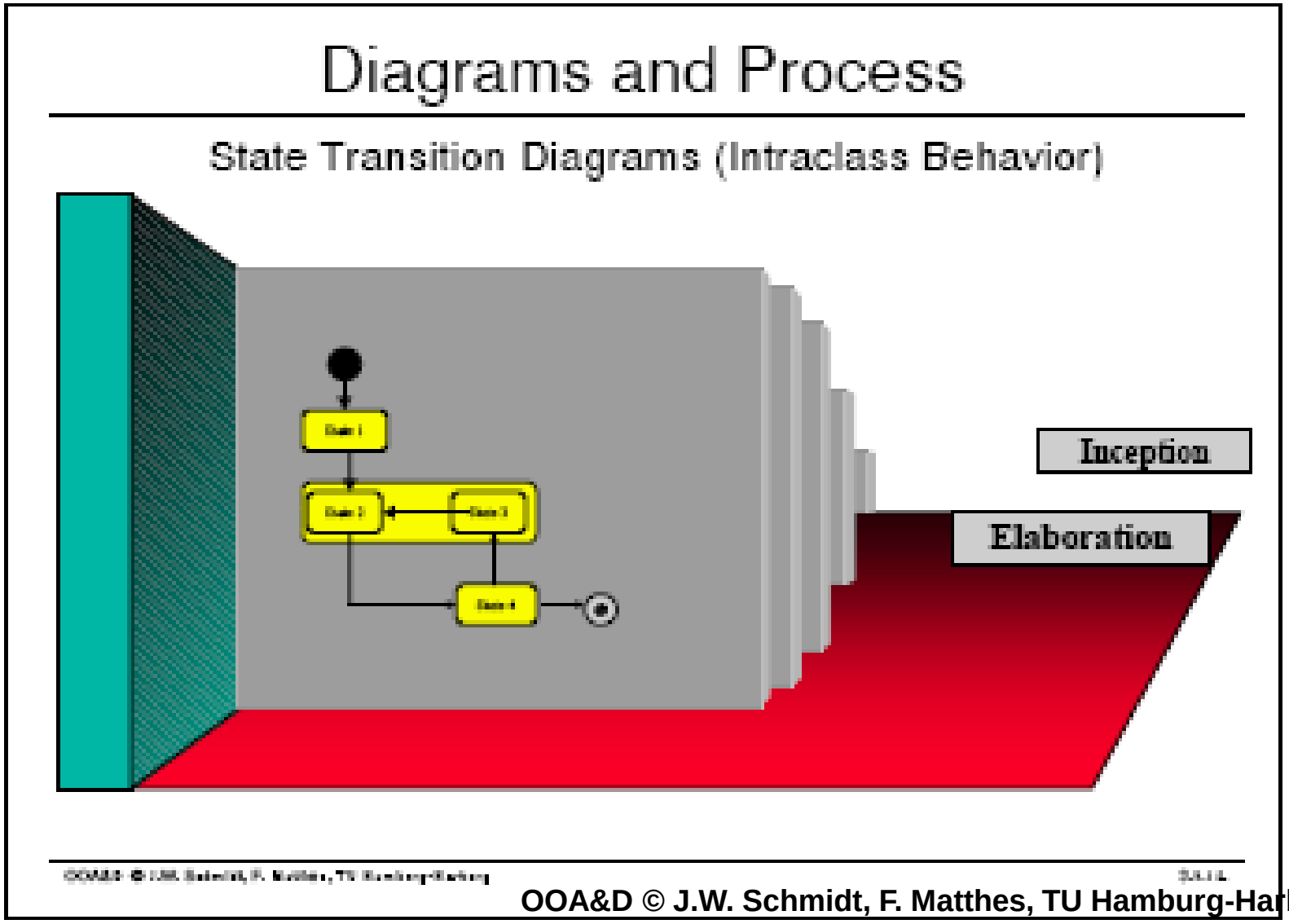


UML

Exemplar usage of Diagrams

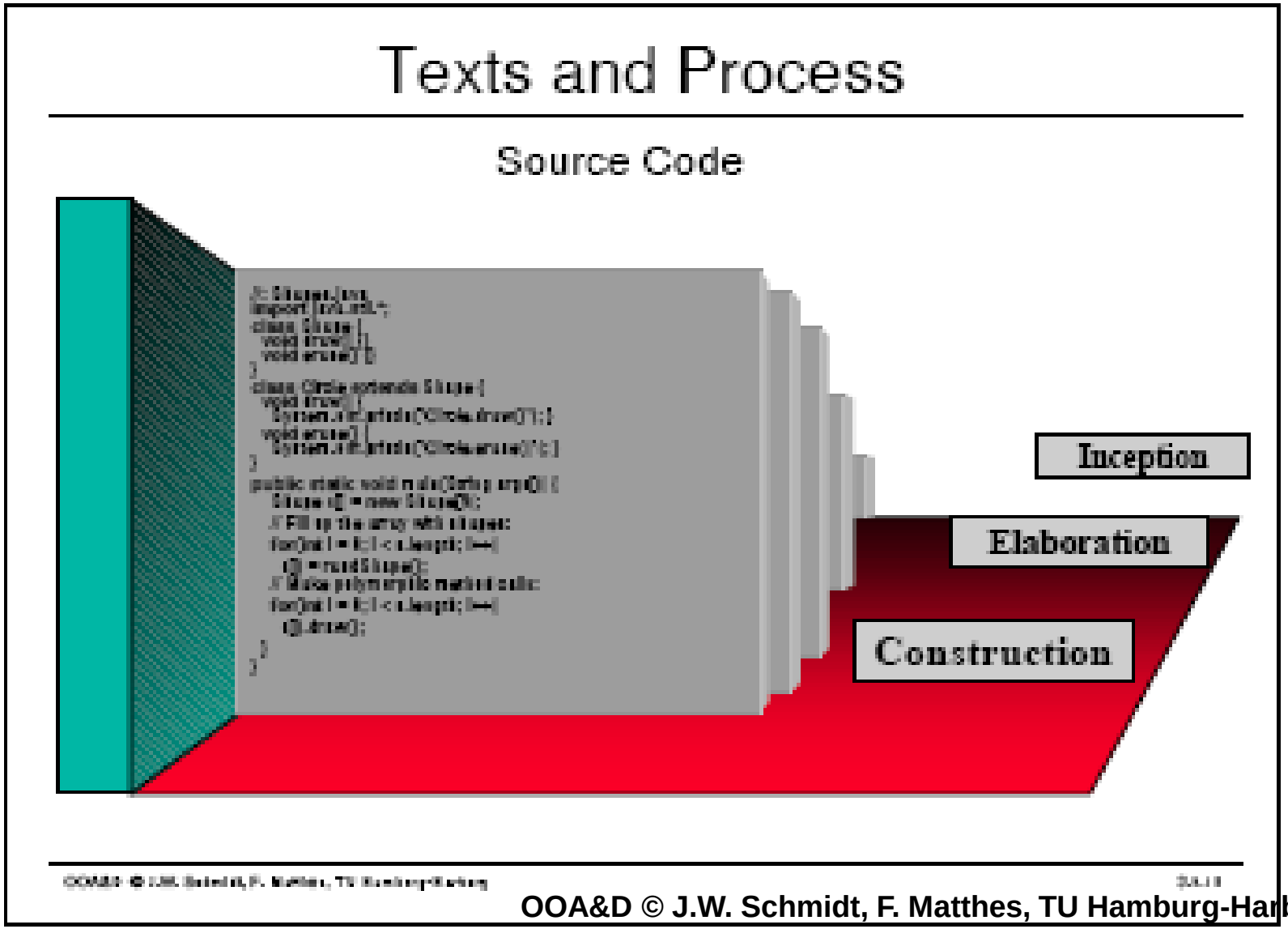


UML
Exemplar usage of Diagrams

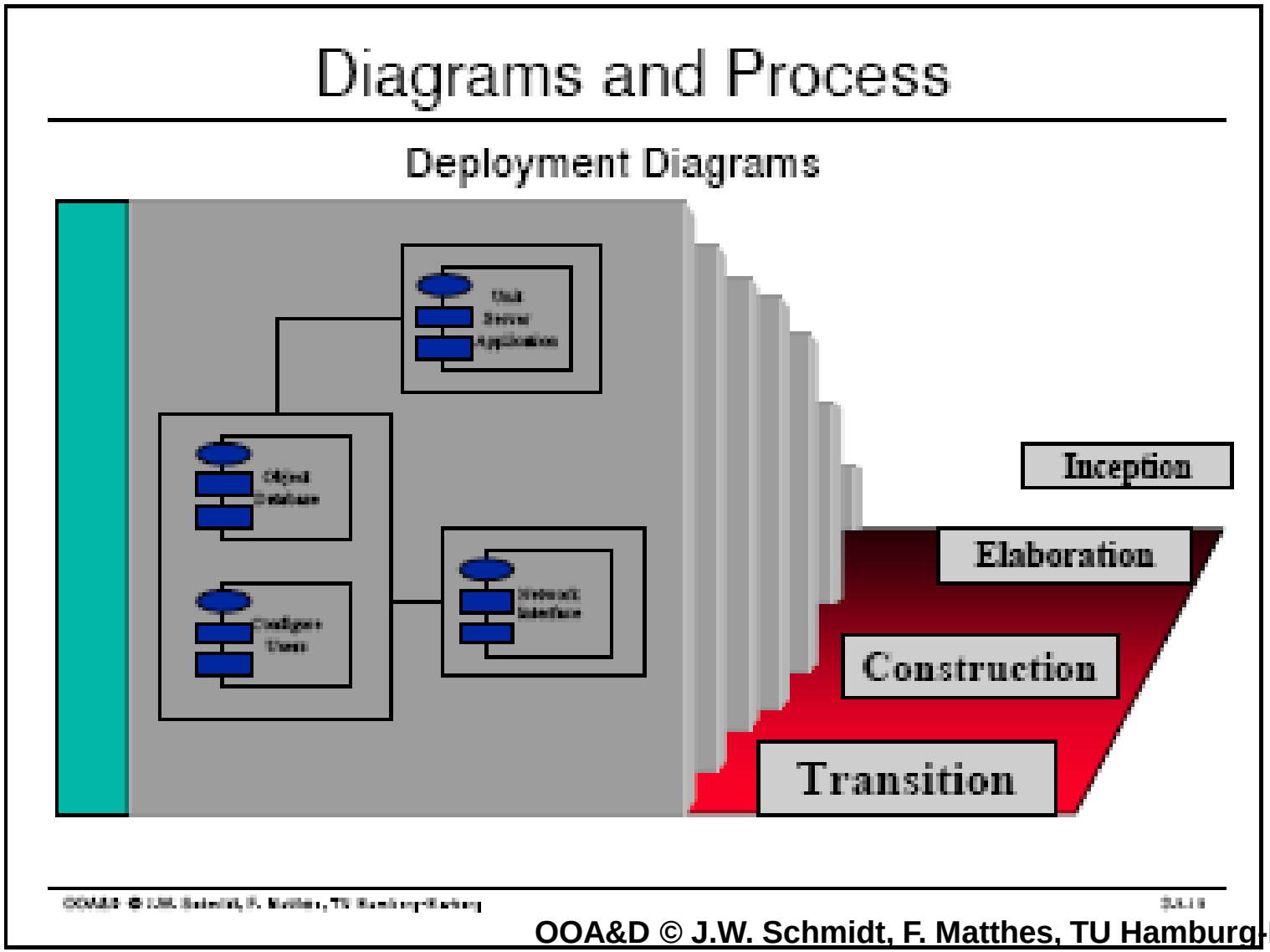


UML

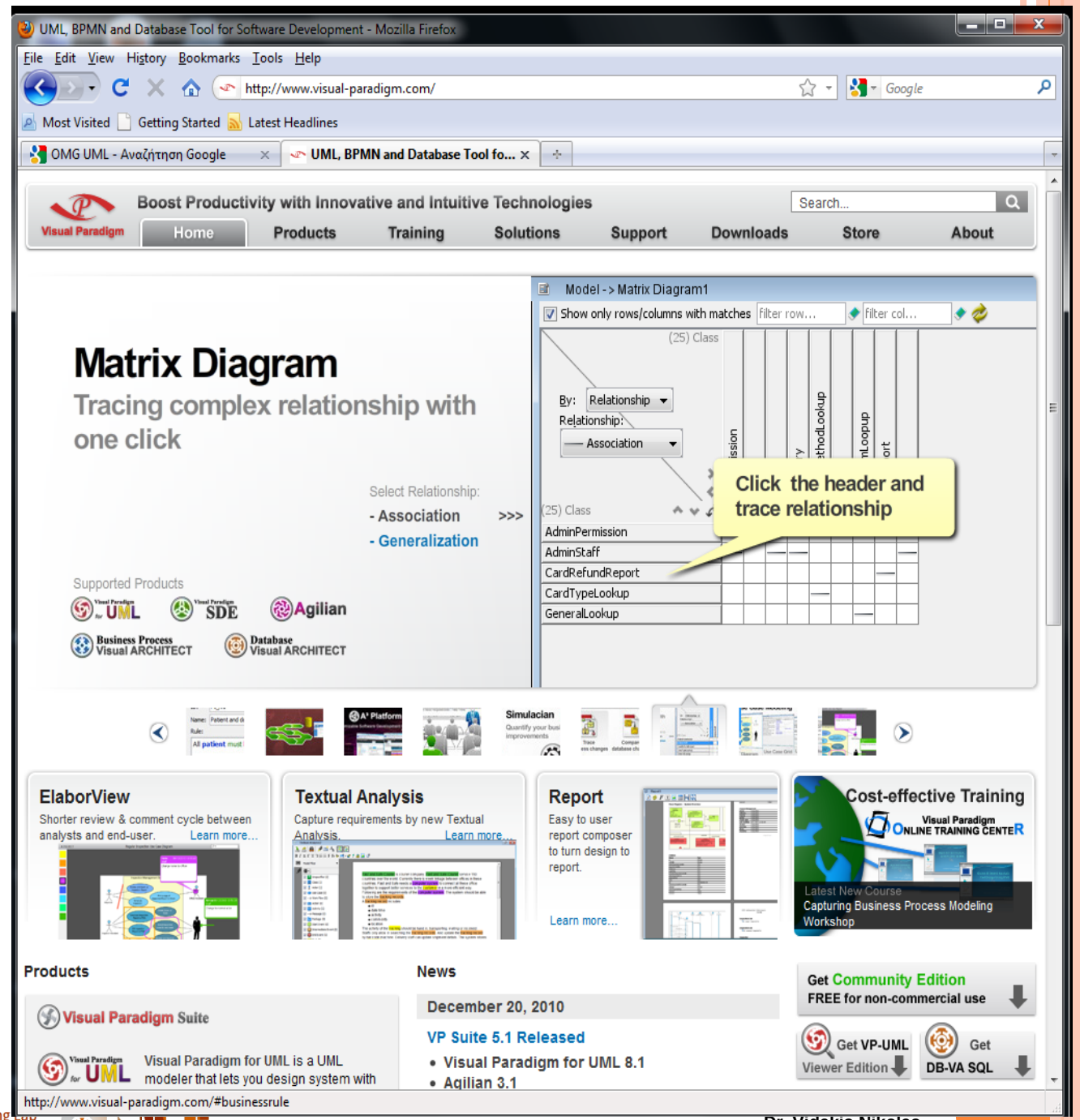
Exemplar usage of Diagrams



UML
Exemplar usage of Diagrams



Free UML Modeling Tools
http://www.visual-paradigm.com/

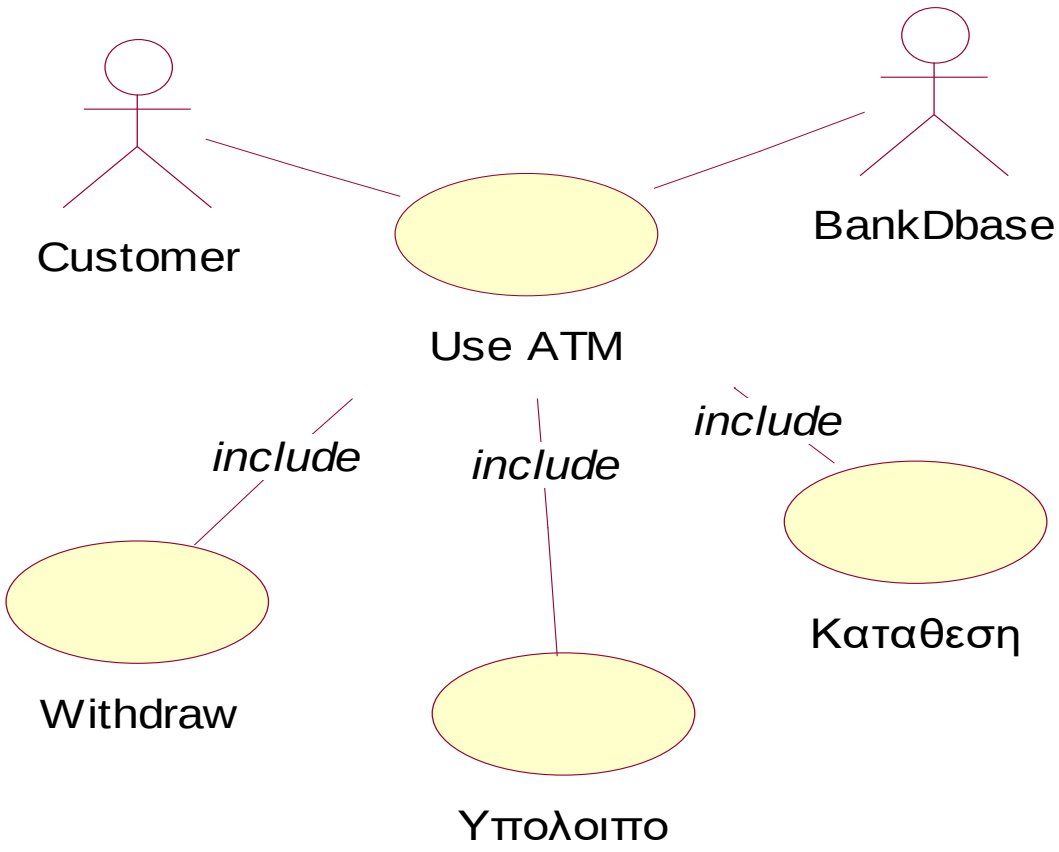


Example Reference Scenario

ATM

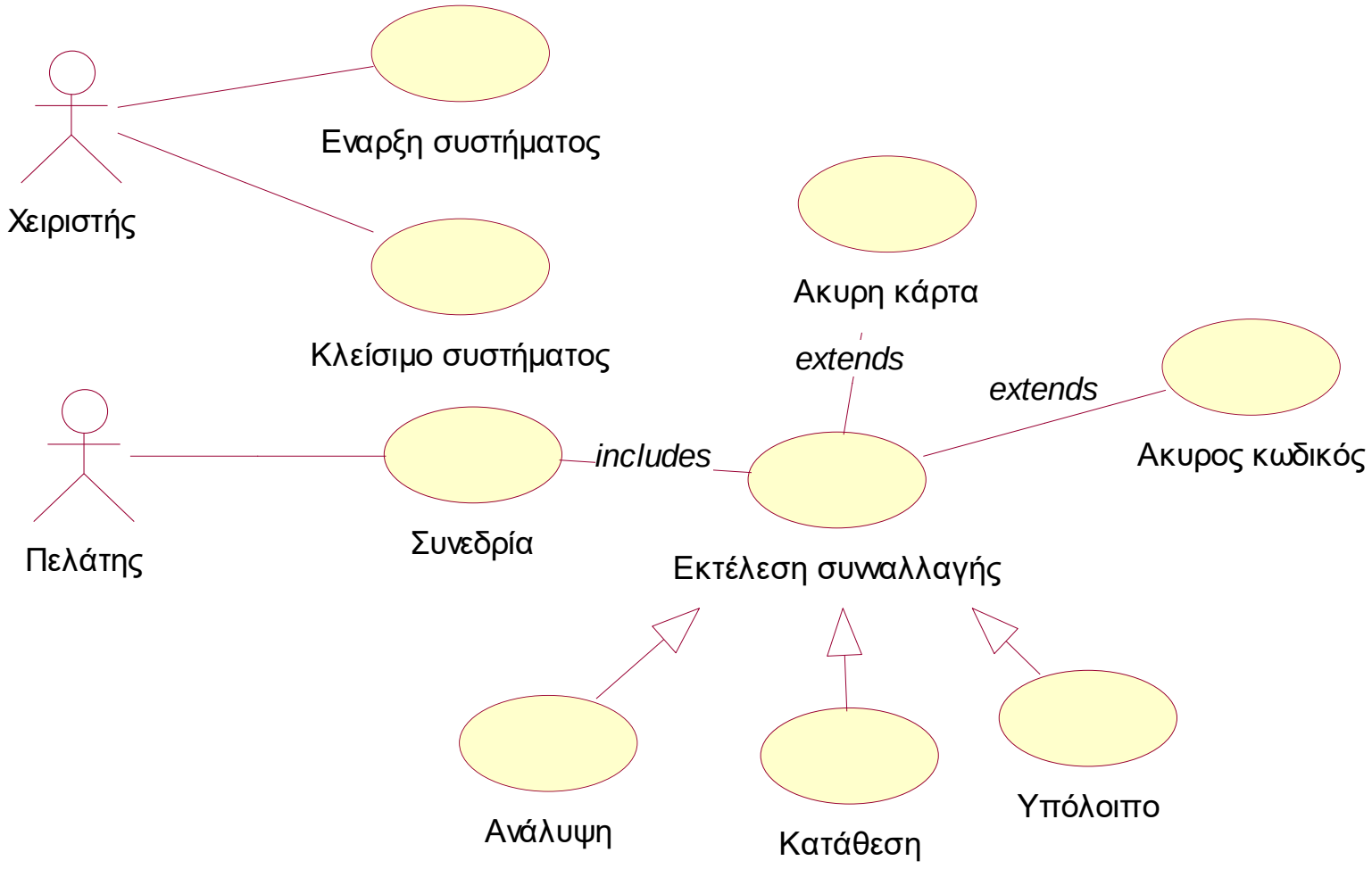
A Reference Scenario

Manage transactions through ATM



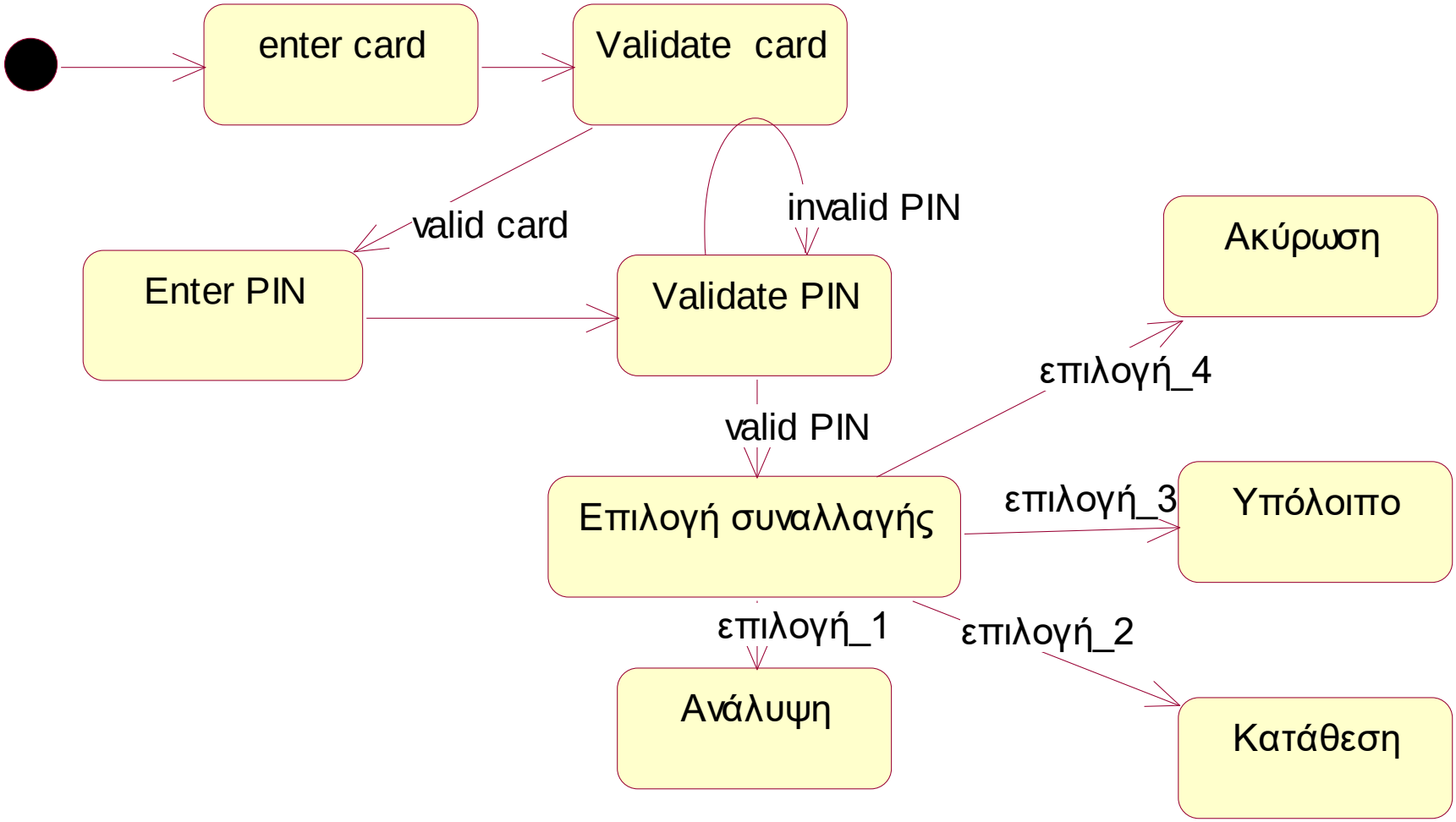
A Reference Scenario

Manage transactions through ATM - Another version



A Reference Scenario

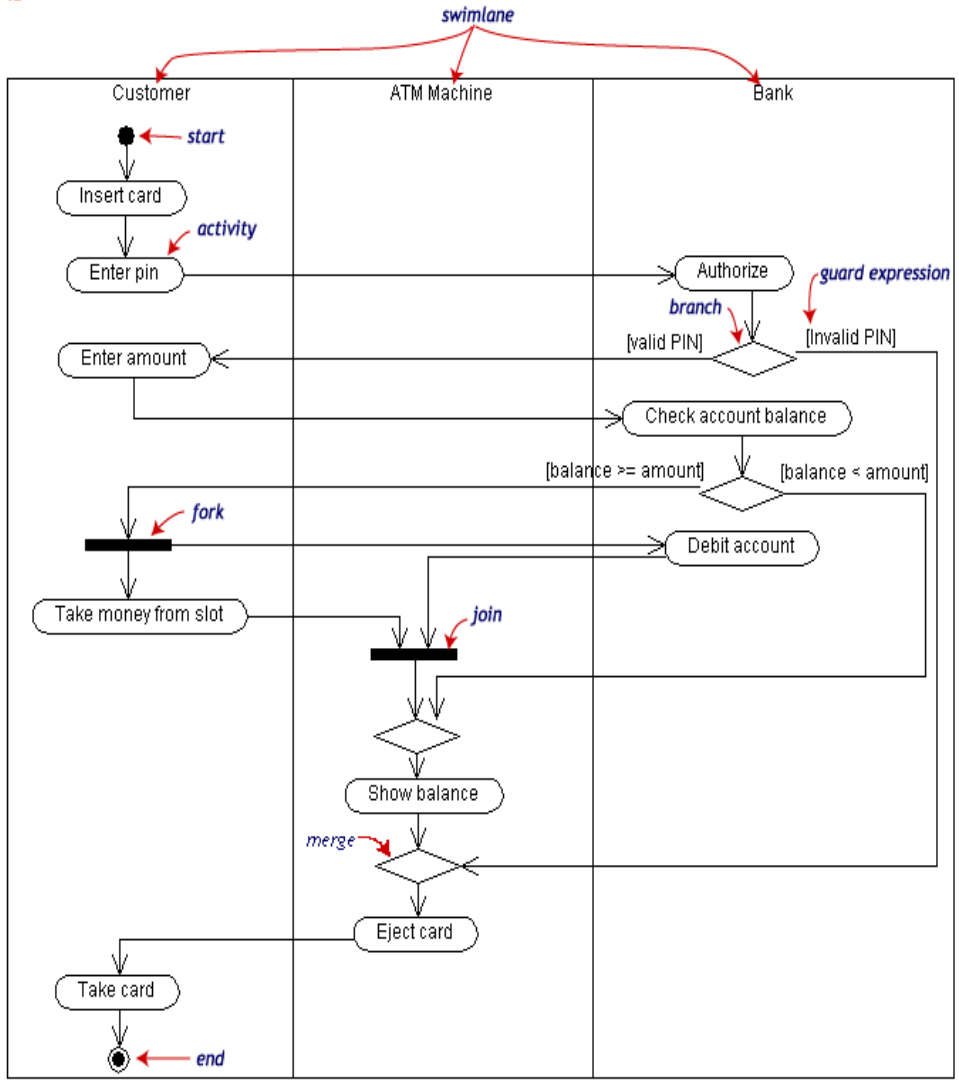
Manage transactions through ATM – Activity Diagram



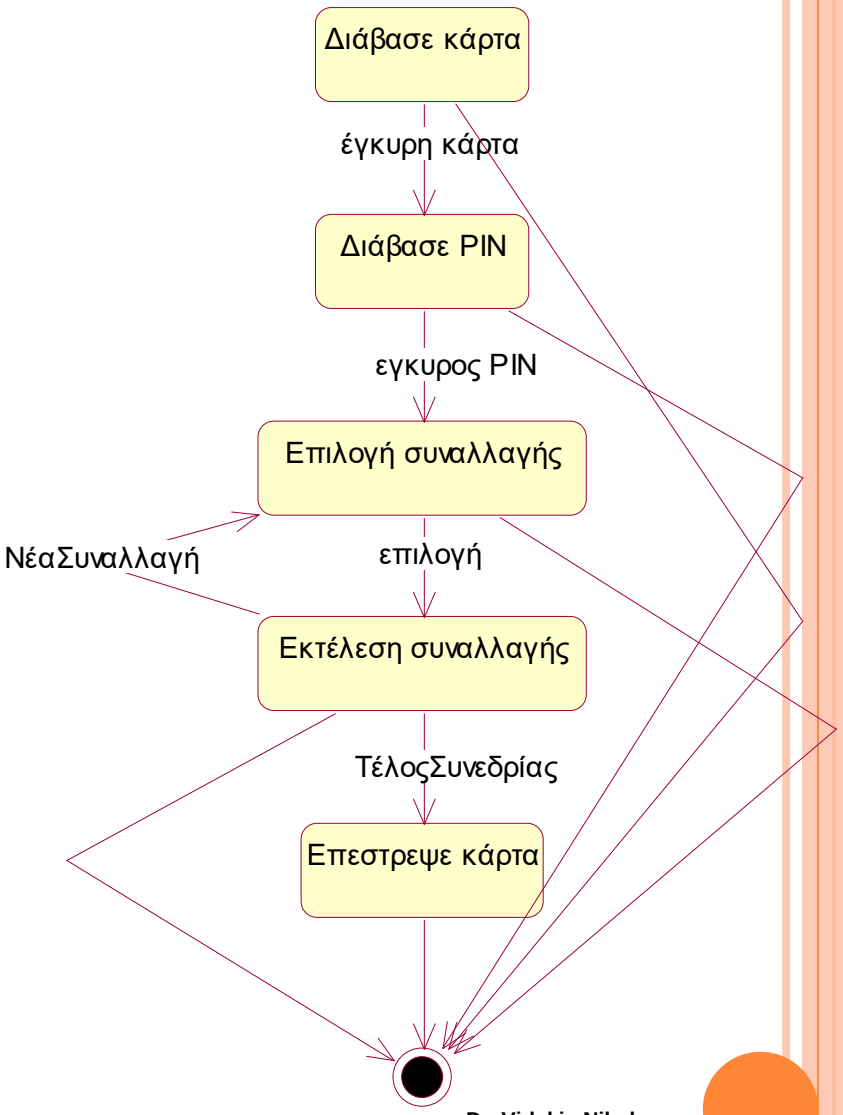
A Reference Scenario

Manage transactions through ATM – Activity Diagram

2nd Version



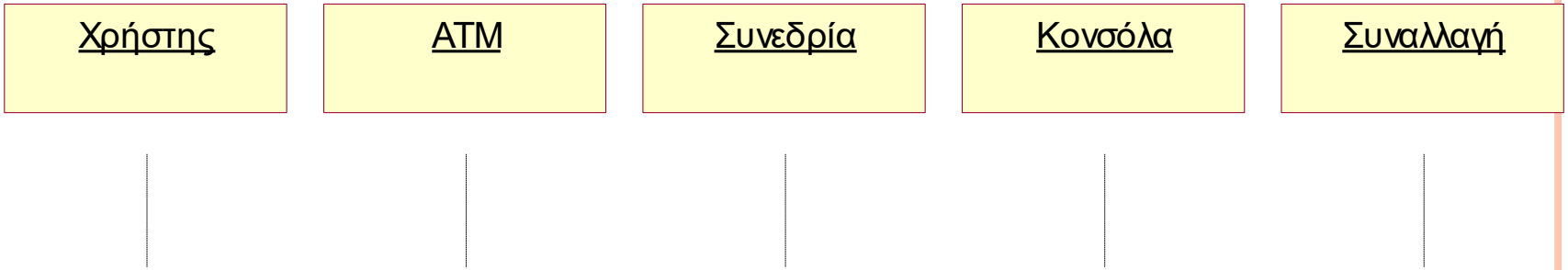
3nd Version



A Reference Scenario

Manage transactions through ATM – Sequence Diagram

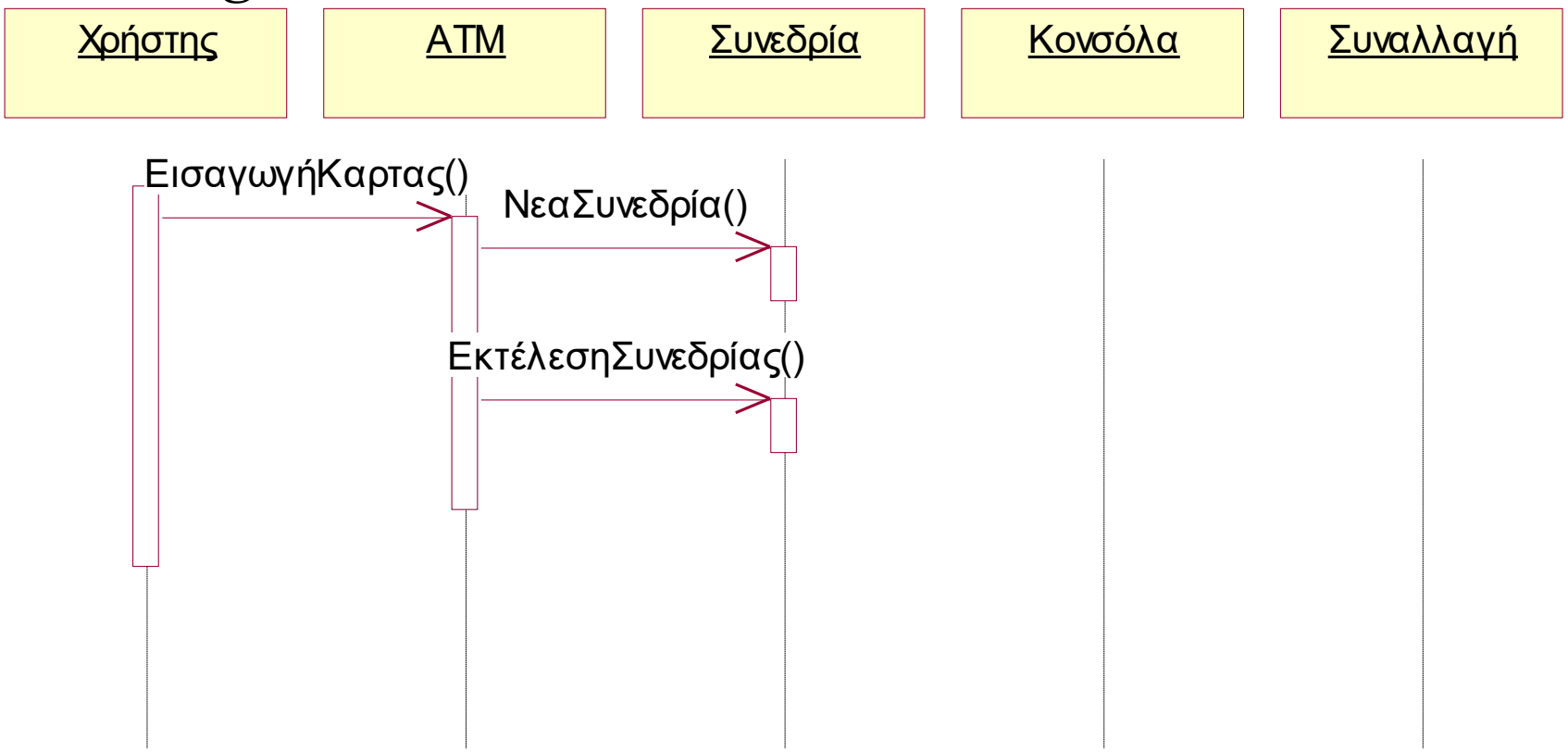
- Which Objects do you recognize?



A Reference Scenario

Manage transactions through ATM – Sequence Diagram

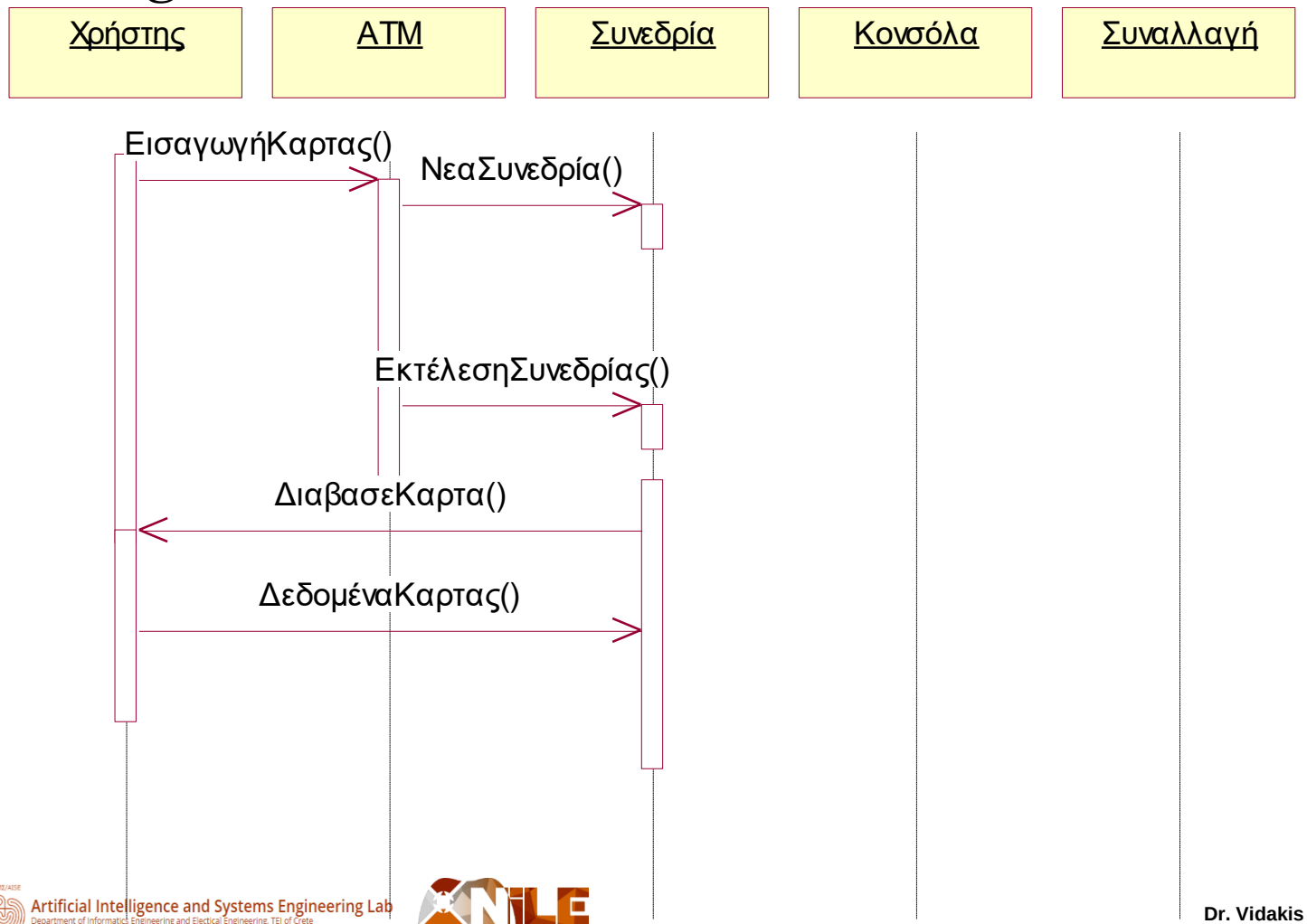
- Messages



A Reference Scenario

Manage transactions through ATM – Sequence Diagram

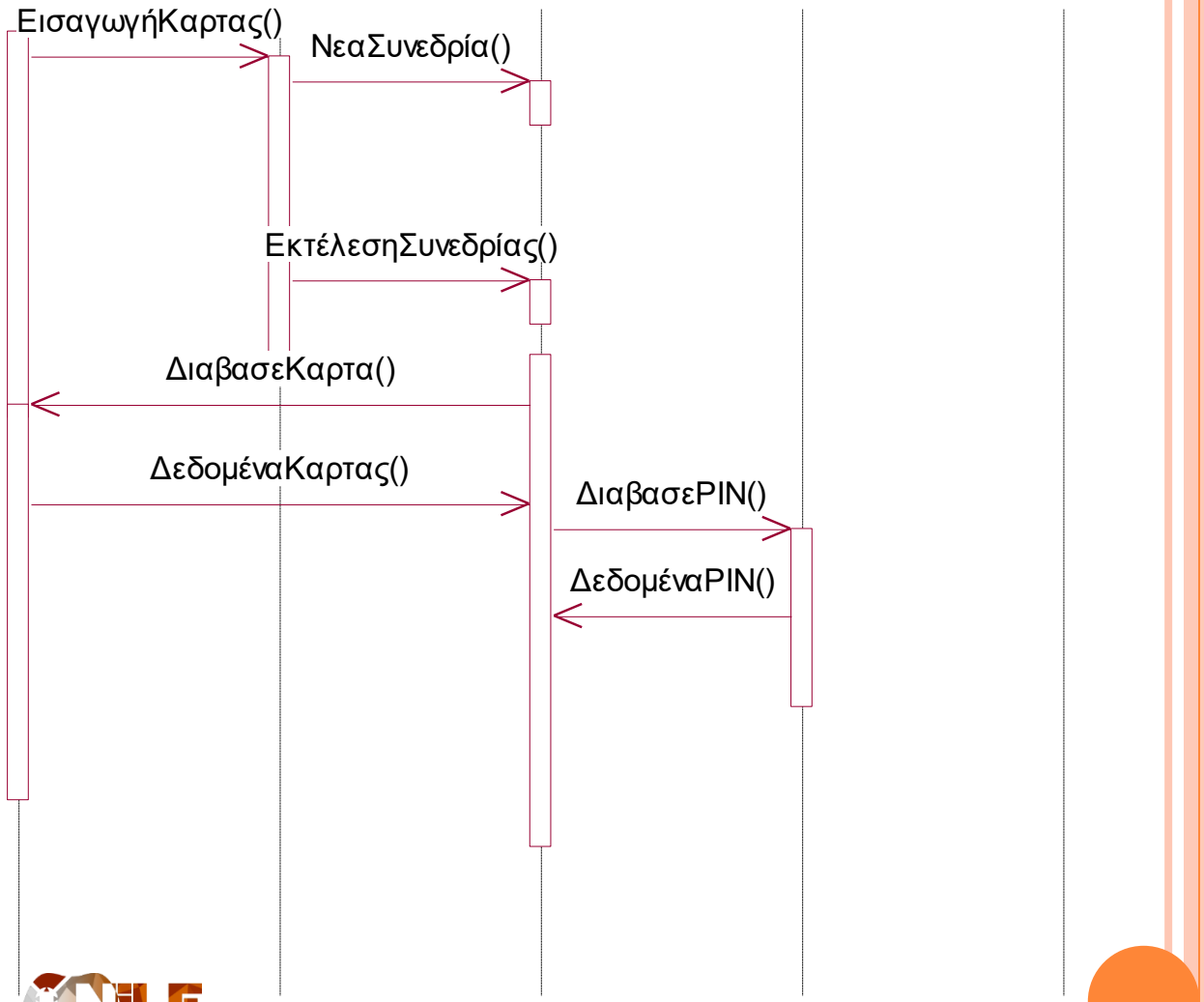
- Messages – Check Card



A Reference Scenario

Manage transactions through ATM – Sequence Diagram

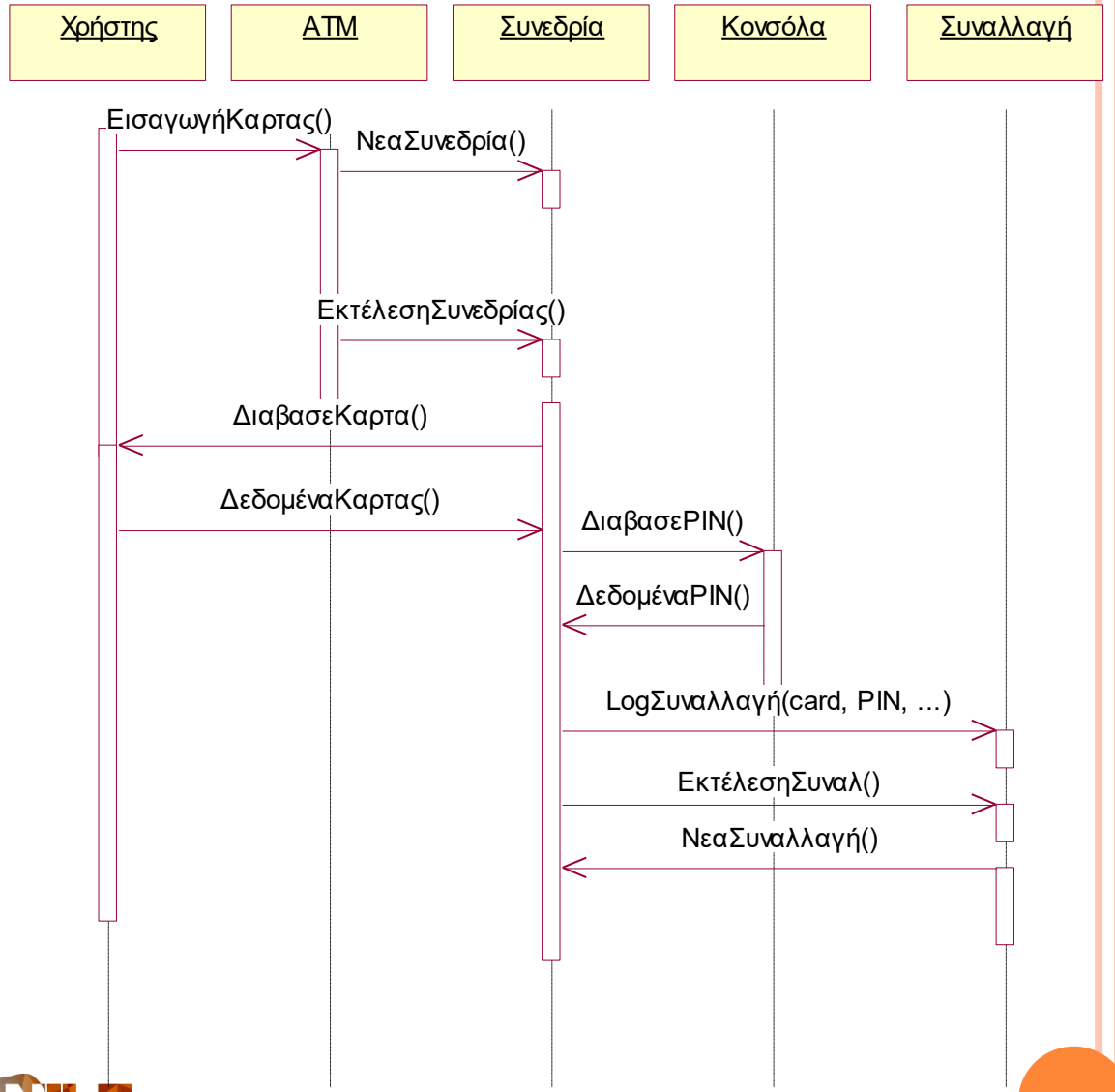
- Messages
  - Check PIN



A Reference Scenario

Manage transactions through ATM – Sequence Diagram

- Messages
  - Transaction



Further Reading

1. Robert Cecil Martin, The principles, Petterns and Practices of Agile Software Development, Prentice-Hall, 2003
2. Software Engineering: (Update) (8th Edition) (International Computer Science Series); Ian Sommerville
3. UML Distilled: A Brief Guide to the Standard Object Modeling Language
   Martin Fowler, Kendall Scott
4. IBM Rational http://www-306.ibm.com/software/rational/uml/
5. UML basics: The class diagram, An introduction to structure diagrams in UML 2, Donald Bell (bellds@us.ibm.com), IT Specialist, IBM, http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/index.html#N102EE
6. http://www.therationaledge.com/content/nov_03/t_modelinguml_db.jsp, Copyright Rational Software 2003
7. Practical UML --- A Hands-On Introduction for Developers
   http://www.togethersoft.com/services/practical_guides/umlonlinecourse/
8. Software Engineering Principles and Practice. Second Edition; Hans van Vliet.
9. http://www-inst.eecs.berkeley.edu/~cs169/
10. UML an overview By: DiGitAll
11. The UML Class Diagram: Part 1 By Mandar Chitnis, Pravin Tiwari, & Lakshmi Ananthamurthy
    http://www.developer.com/design/article.php/10925_2206791_1
12. Practical UML: A Hands-On Introduction for Developers By: Randy Miller, http://dn.codegear.com/article/31863#classdiagrams