Σημειώσεις στην UML

Δρ. Κωνσταντινος Λιαγκούρας
Τμήμα Πληροφορικής
Πανεπιστήμιο Πειραιώς

1. Η UML (Unified Modeling Language), είναι πρότυπη γλώσσα μοντελοποίησης για την ανάπτυξη λογισμικού και συστημάτων.
2. Η μοντελοποίηση βοηθάει στην καλύτερη τεκμηρίωση και επικοινωνία σχετικά με τις σημαντικότερες απόψεις του συστήματος.
3. Η UML μοντελοποιεί τη στατική και τη δυναμική όψη ενός Συστήματος Λογισμικού.
4. Η στατική όψη αφορά τα είδη των πιο σημαντικών αντικειμένων του Συστήματος, και τις μεταξύ τους συσχετίσεις.
5. Η δυναμική όψη, περιγράφει την εξέλιξη των αντικειμένων χρονικά, καθώς και τη μεταξύ τους επικοινωνία.

- Οι **κλάσεις**, **τα αντικείμενα** και οι μεταξύ τους **συσχετίσεις** είναι τα πρωταρχικά στοιχεία μοντελοποίησης στην αντικειμενοστραφή θεώρηση.
- Οι **κλάσεις** και τα **αντικείμενα** περιγράφουν τι υπάρχει μέσα στο σύστημα που περιγράφουμε.
- Οι **συσχετίσεις** μεταξύ τους περιγράφουν πως δομούνται το ένα συστατικό σε σχέση με το άλλο.
- Η **κλάση** αποτελεί μια περιγραφή ενός τύπου **αντικειμένου**: περιγράφει τα χαρακτηριστικά και τη συμπεριφορά του συγκεκριμένου τύπου **αντικειμένου.**

Διαγράμματα Κλάσεων

- Είναι τύπος στατικού μοντέλου.
- Περιγράφουν τη στατική άποψη ενός συστήματος με κλάσεις και συσχετίσεις
- Ένα Class diagram μπορεί να περιγραφεί ως ακολούθως:

Class name
Attributes
Methods ή Operations

Η class αποτελεί μια περιγραφή ενός συνόλου αντικειμένων (objects) που μοιράζονται κοινά χαρακτηριστικά (attributes), συμπεριφορά (operations ή methods) και relationships. Το όνομα της κλάσης (class) είναι το μόνο απαραίτητο στοιχείο στην γραφική αναπαράσταση μια κλάσεως και εμφανίζεται στην κορυφή της κλάσεως.

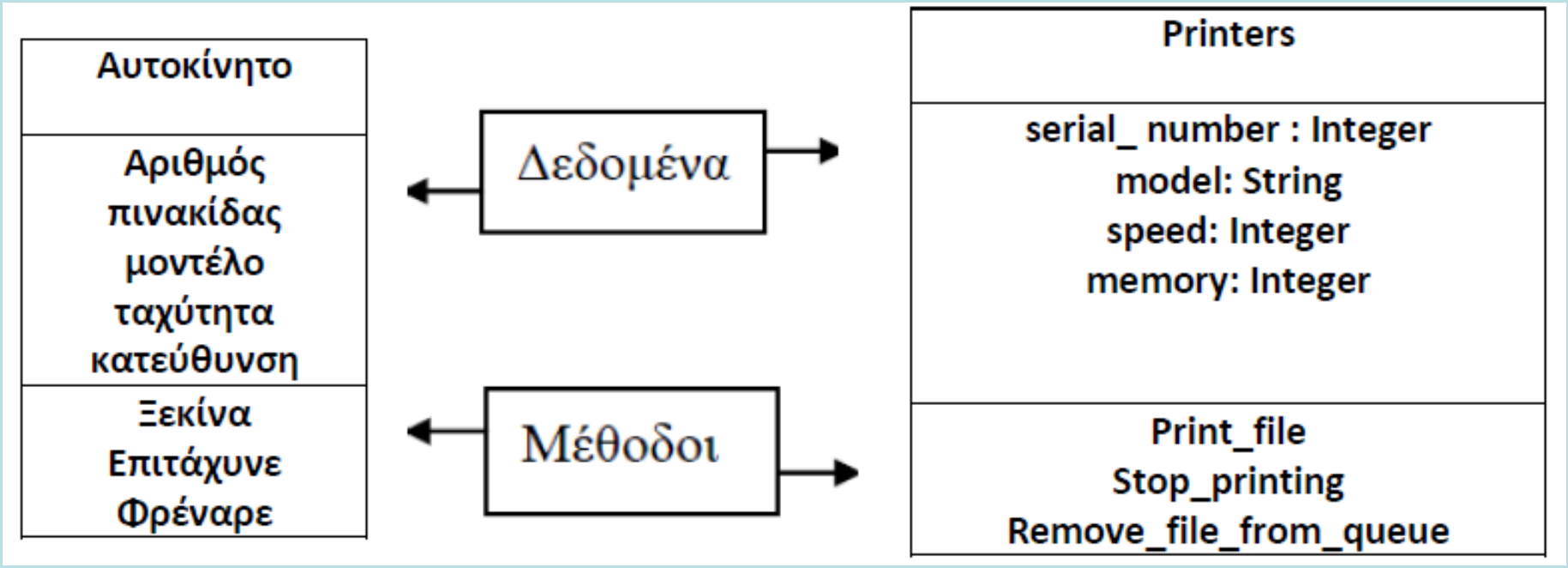
Διαγράμματα Κλάσεων

Person
name: String address: Address birthdate: Date passport: Id
methods

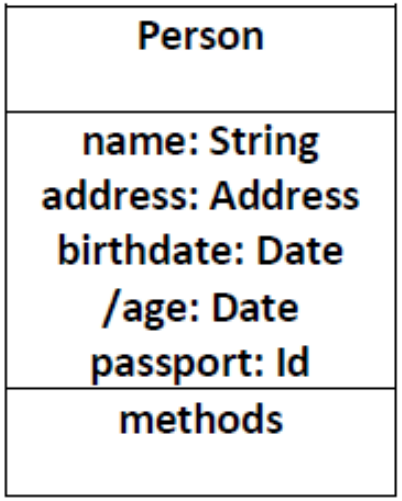
Τα attributes περιγράφουν τα αντικείμενα (objects) μιας κλάσεως.

Τιμολόγιο
ποσό: πραγματικός αριθμός ημερομηνία: Ημερομηνία = Τρέχουσα αριθμός τιμολογίου: Ακέραιος
Methods

Διαγράμματα Κλάσεων

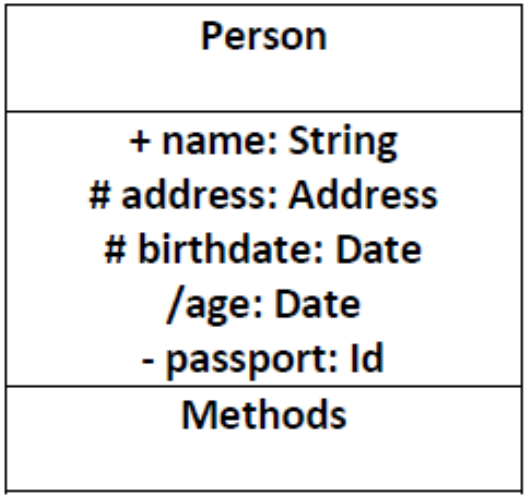


Διαγράμματα Κλάσεων



derived attribute (/) ονομάζεται οποία υπολογίζεται από άλλα attributes.

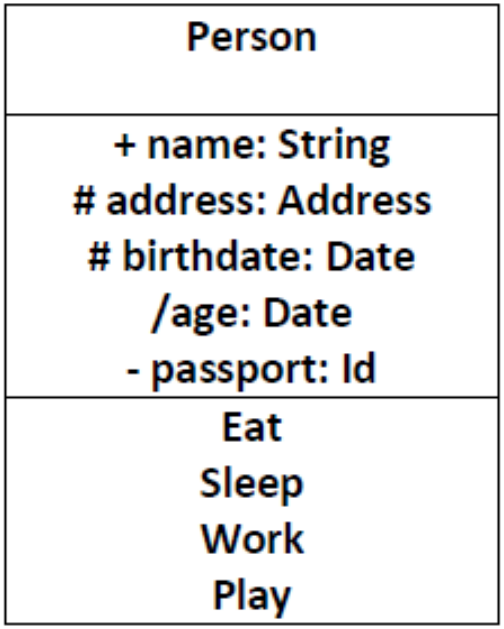
π.χ. Η ηλικία του Person μπορεί να υπολογιστεί από την ημερομηνία γέννησης (birthdate).



Τα attributes μπορεί να είναι:

+ public
# protected
- private
/ derived

Διαγράμματα Κλάσεων



Οι methods περιγράφουν τη συμπεριφορά της κλάσης και εμφανίζονται στο 3° διαμέρισμα:

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
Να κάνουν κάποιες ενέργειες πάνω στο αντικείμενο ή/και να καλούν άλλα αντικείμενα.

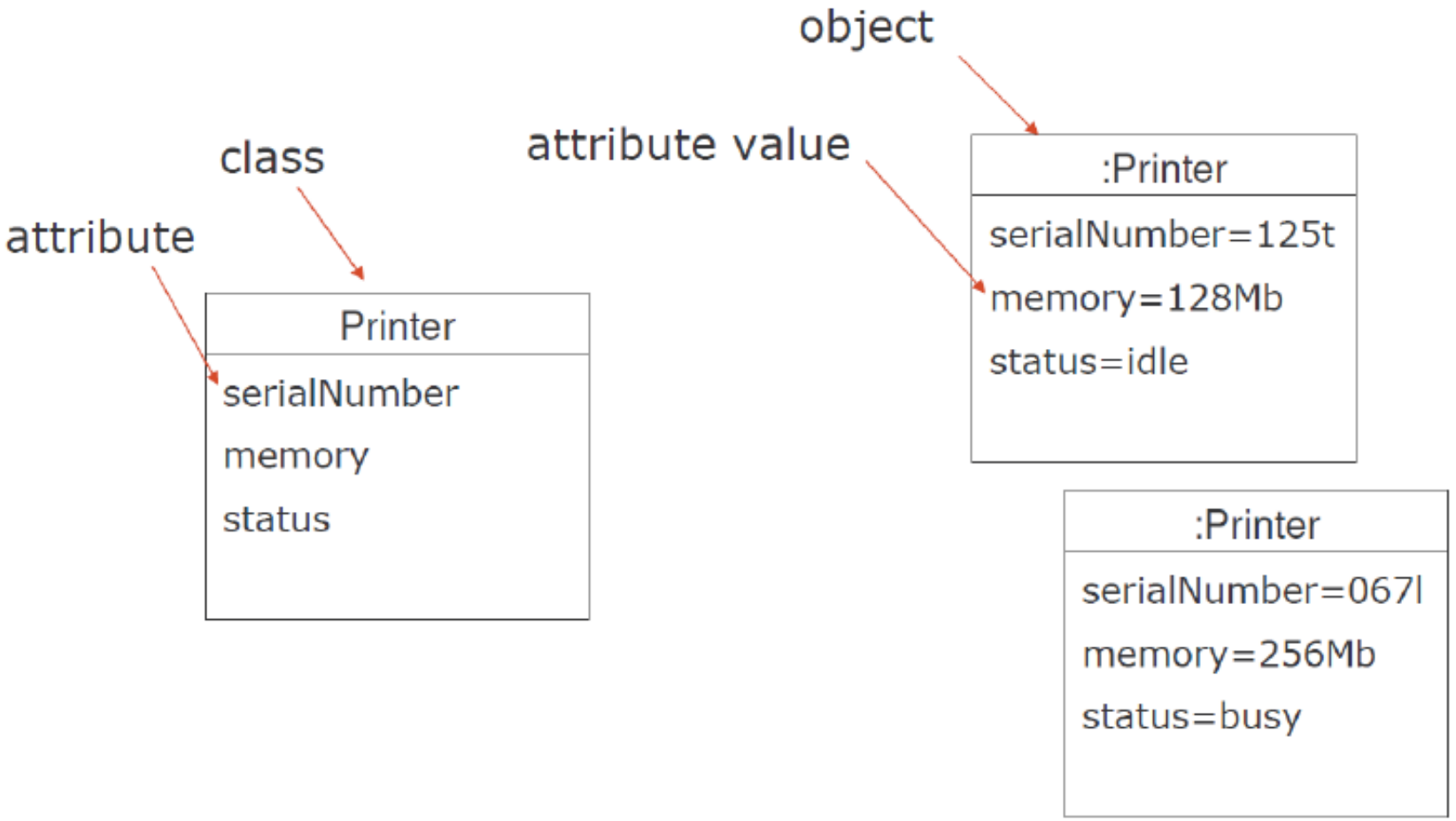
Παράδειγμα κλάσης με διάφορες μεθόδους

Αυτοκίνητο
+αριθμός πινακίδας: Συμβολοσειρά +ταχύτητα: Ακέραιος +κατεύθυνση: Κατεύθυνση -δεδομένα: Δεδομένα_Αυτοκινήτου
+οδήγησε (ταχύτητα: ακέραιος, κατεύθυνση: Κατεύθυνση) ανάκτησε_δεδομένα():Δεδομένα_Αυτοκινήτου

Η μέθοδος **οδήγησε** δέχεται δύο παραμέτρους, ταχύτητα και κατεύθυνση.

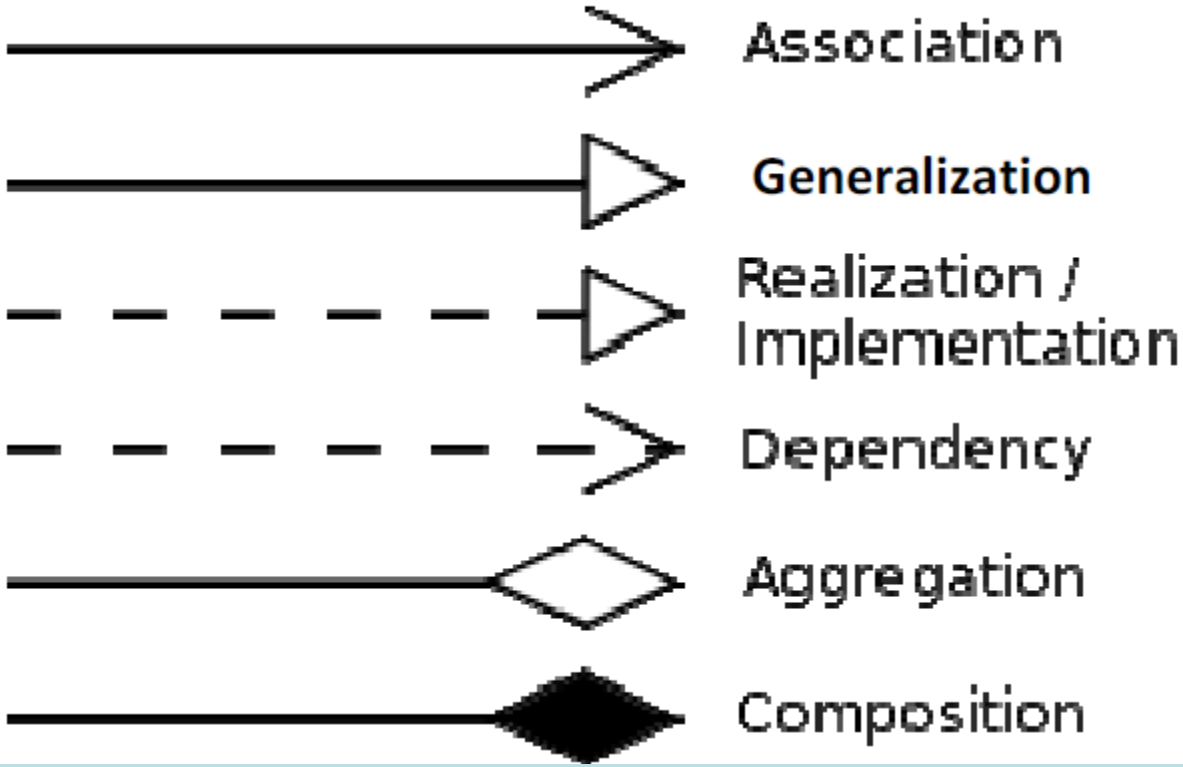
Η μέθοδος **ανάκτησε_δεδομένα** έχει έναν τύπο επιστροφής, Δεδομένα_Αυτοκινήτου

Το Object Diagram δείχνει τη σχέση μεταξύ των objects. Σε αντίθεση με τις κλάσεις τα objects έχουν state.



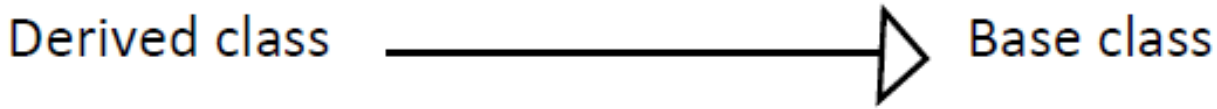
Στη UML ορίζονται οι ακόλουθες Σχέσεις μεταξύ κλάσεων

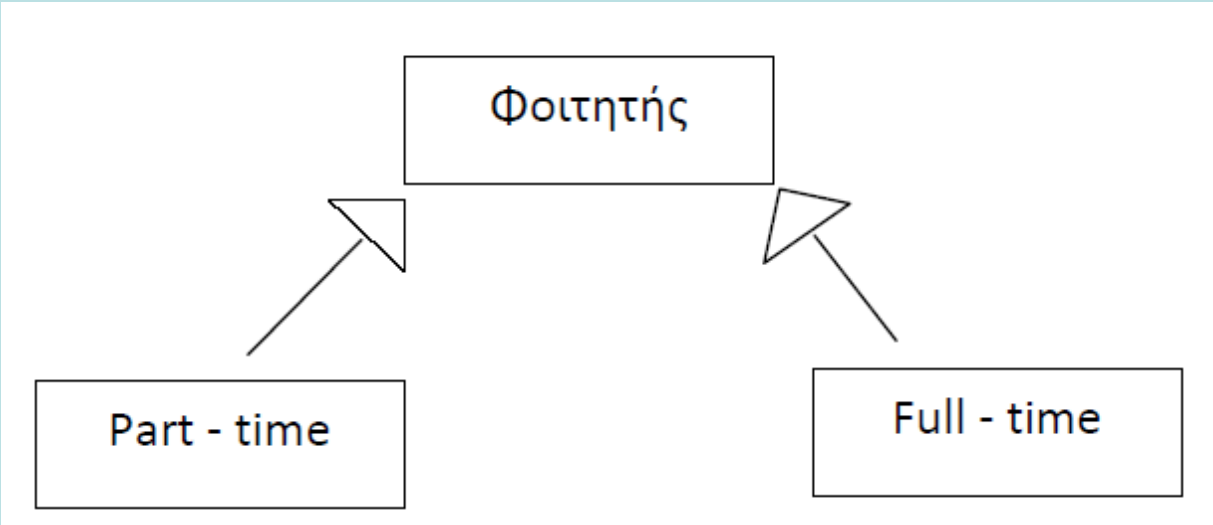
Οι σχέσεις αναδεικνύουν πως τα στοιχεία σχετίζονται μεταξύ τους, και περιγράφουν τη λειτουργικότητα μίας εφαρμογής.

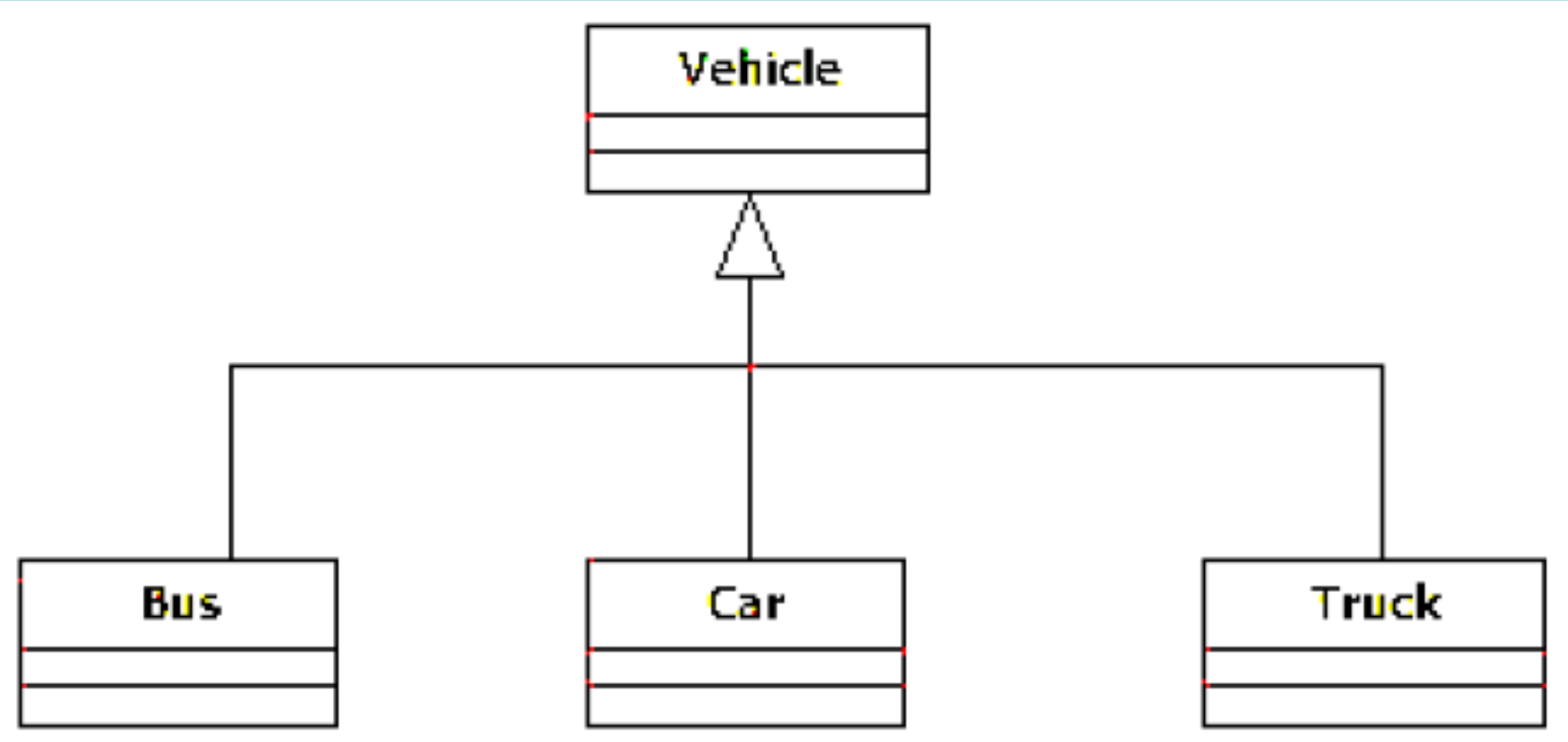


Υπάρχουν τα ακόλουθα είδη σχέσεων:

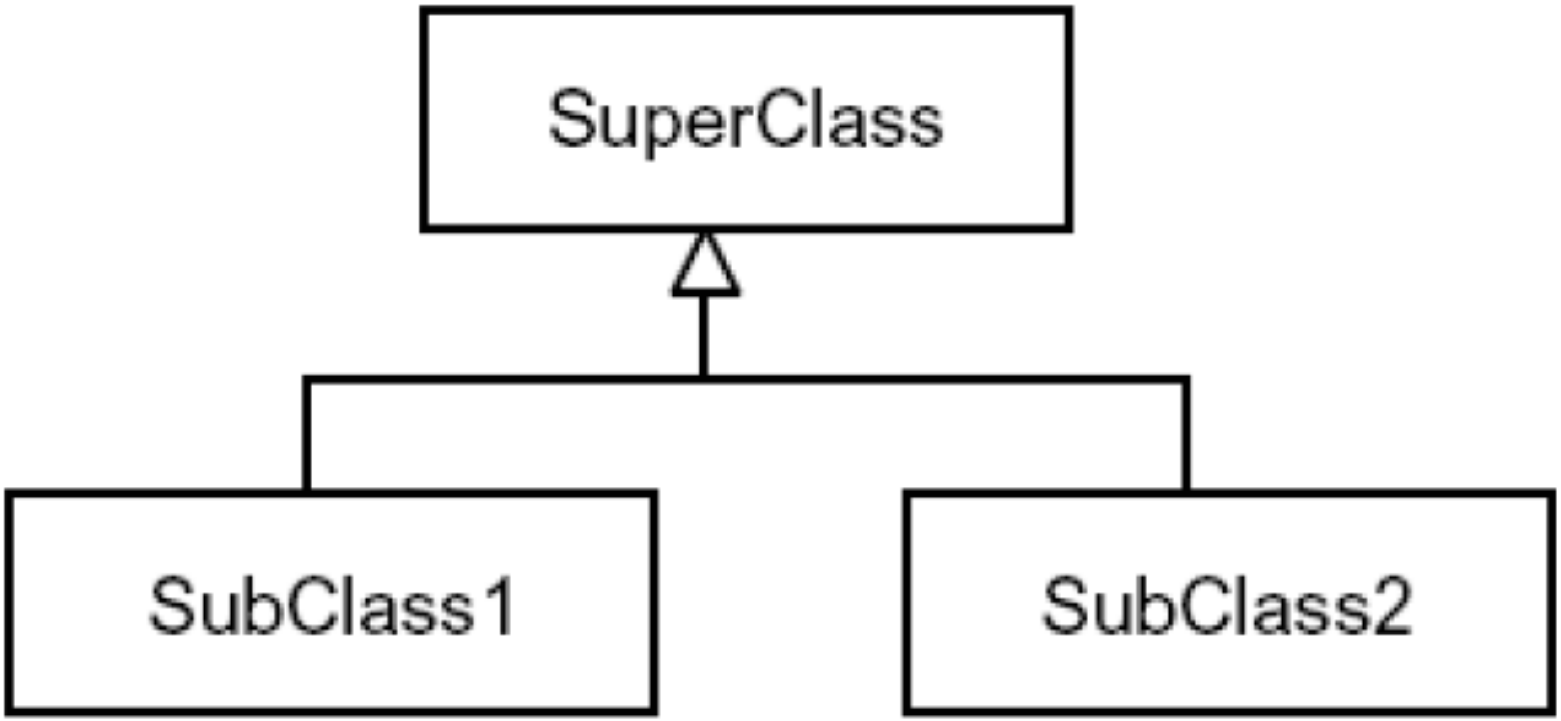
1) Γενίκευση (generalization): Συνδέει ένα ειδικότερο στοιχείο με ένα γενικότερο. Περιγράφει σχέσεις κληρονομικότητας μεταξύ αντικειμένων. Παριστάνεται με μια συνεχή γραμμή με κλειστό βέλος που δείχνει προς τη βασική κλάση. Το ειδικό στοιχείο θα πρέπει να περιέχει μόνο επιπρόσθετες πληροφορίες.

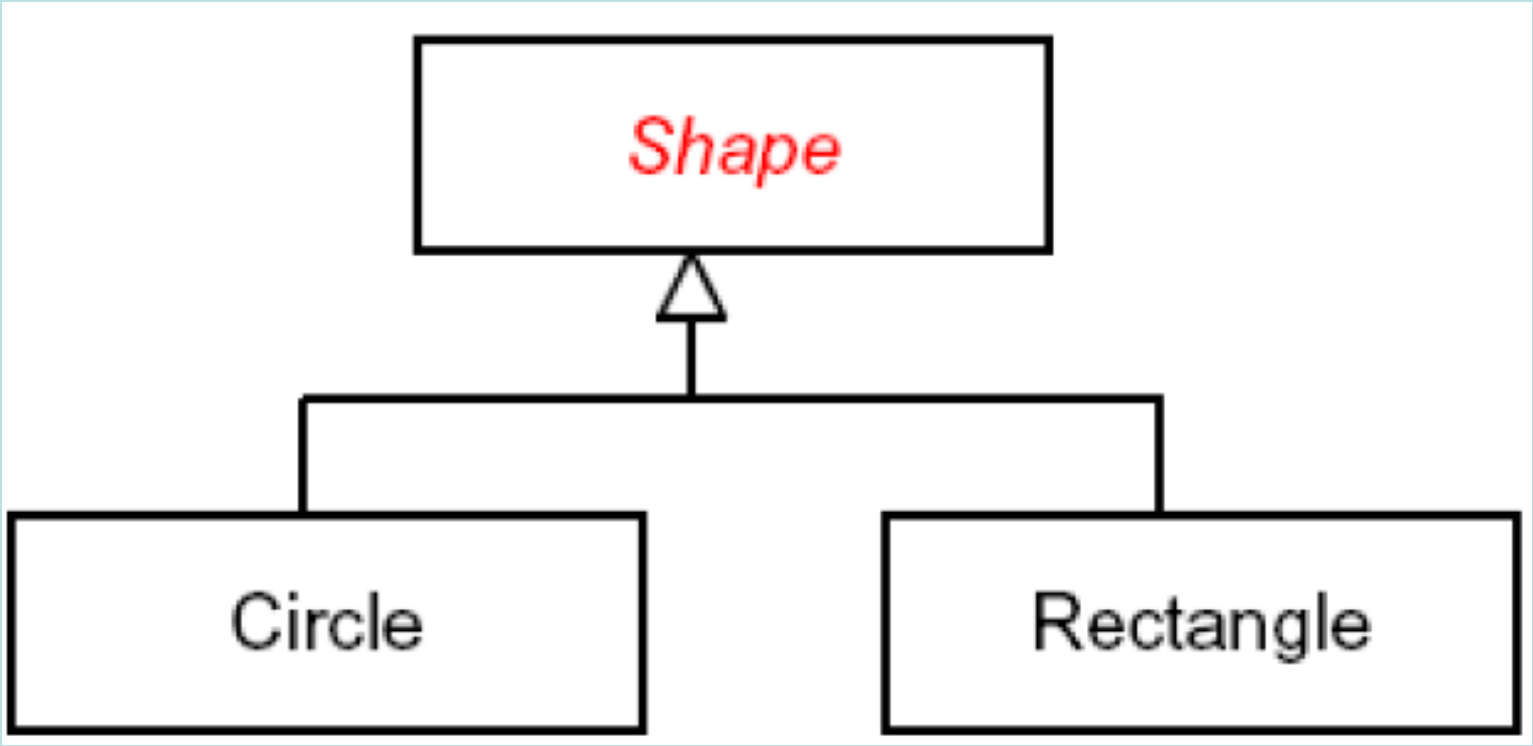



Φοιτητής
Part - time
Full - time

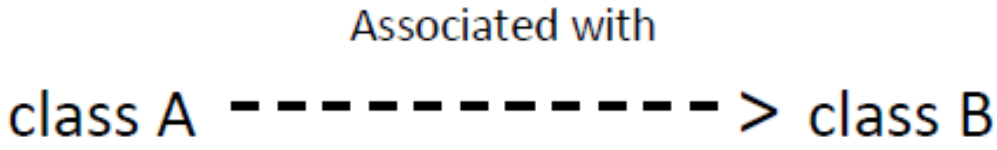

Vehicle
Bus
Car
Truck

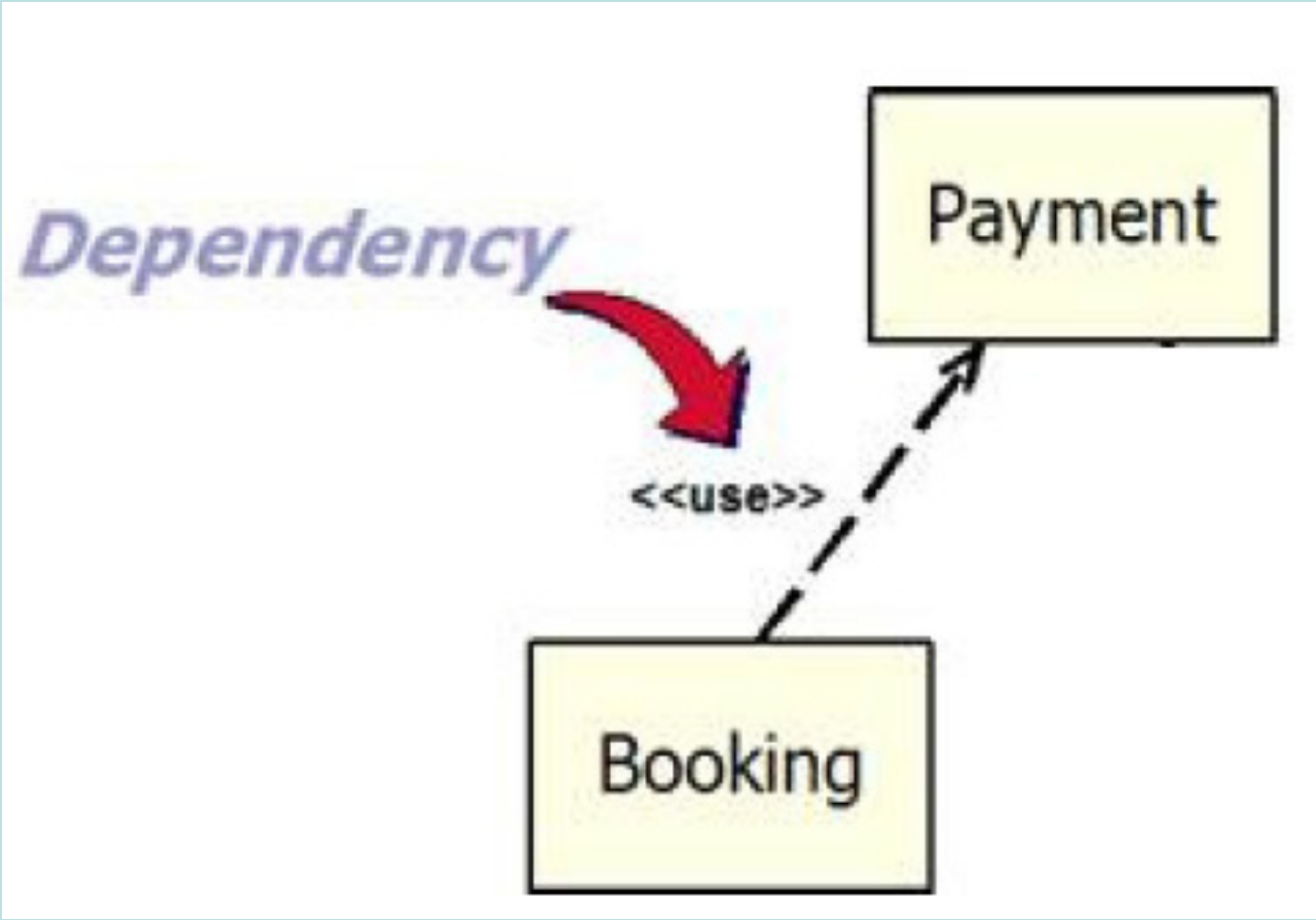
Γενίκευση: Child class είναι μια ειδική περίπτωση της parent class

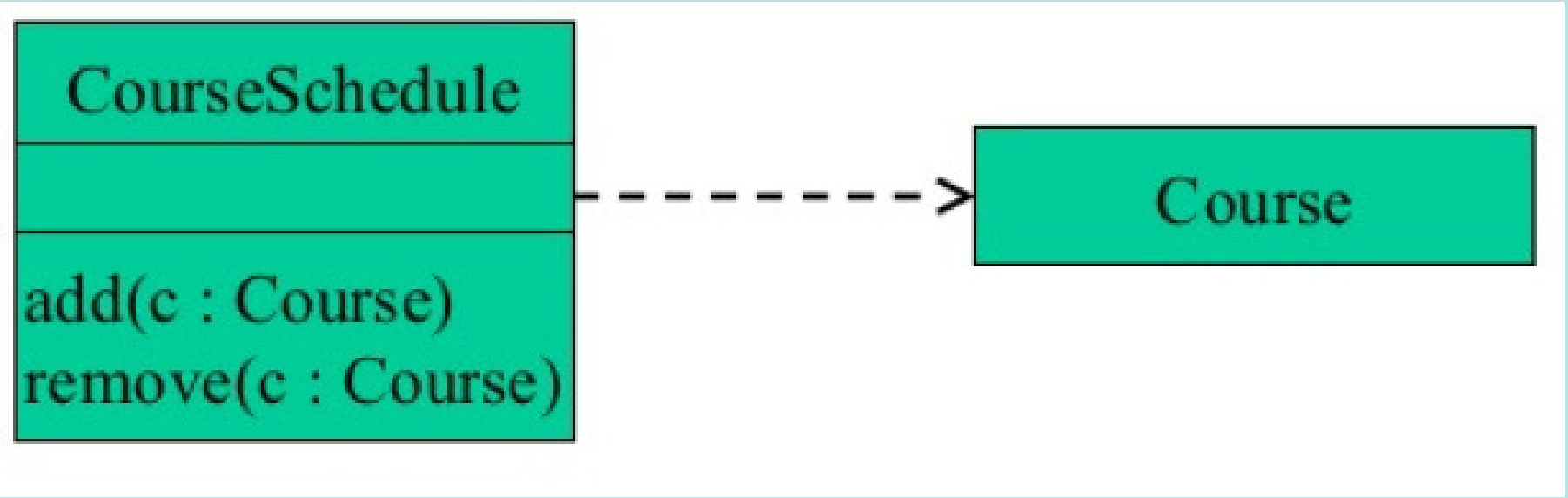



Shape
Circle
Rectangle

2) Εξάρτηση (dependency): Η σχέση αυτή, υφίσταται μεταξύ δυο στοιχείων, και η αλλαγή σε ένα στοιχείο επηρεάζει το άλλο στοιχείο, αλλά όχι απαραίτητα και το αντίστροφο. Το βέλος δείχνει προς την οντότητα που υπάρχει η εξάρτηση (π.χ. το A: the client εξαρτάται από το B: the supplier). Οποιαδήποτε μεταβολή στην supplier class έχει επίδραση στην client class, δηλαδή μια ενδεχόμενη αλλαγή στο ανεξάρτητο στοιχείο θα επηρεάσει το εξαρτημένο στοιχείο.



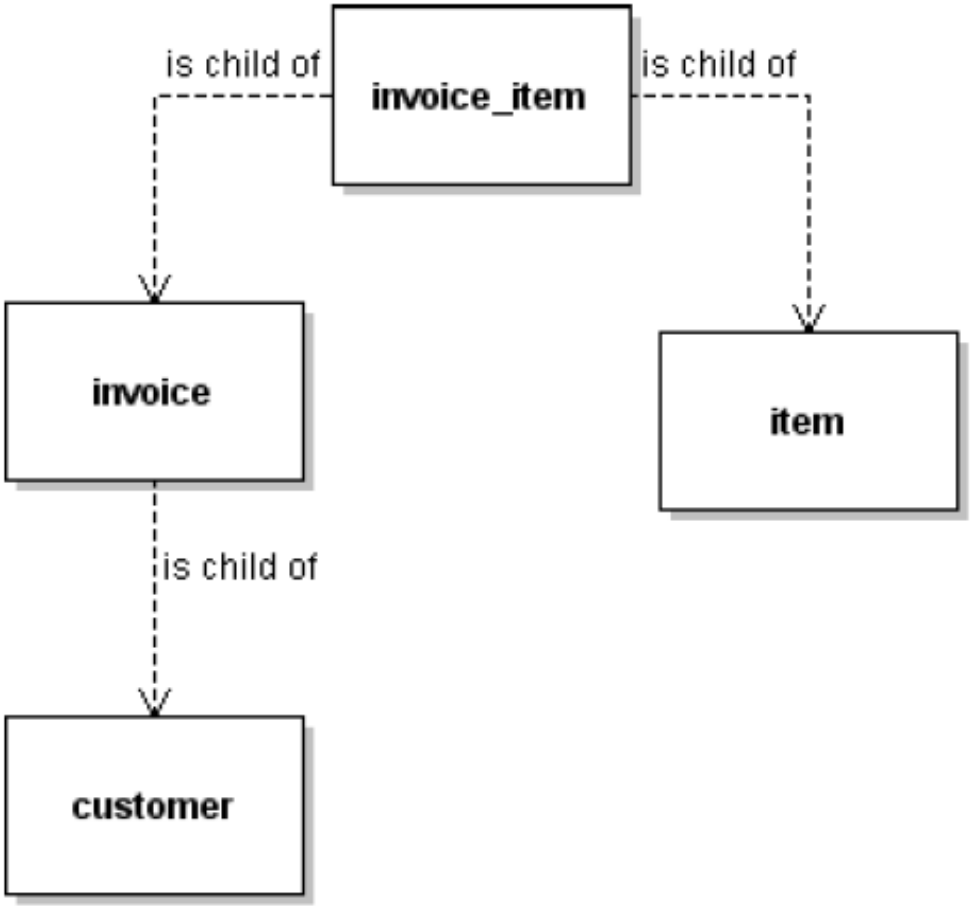

Dependency
Payment
<<use>>
Booking


CourseSchedule
add(c : Course)
remove(c : Course)
Course

Παράδειγμα dependency graph

Child tables είναι: το invoice και invoice_item.

Parent tables είναι: customer, invoice, and item.

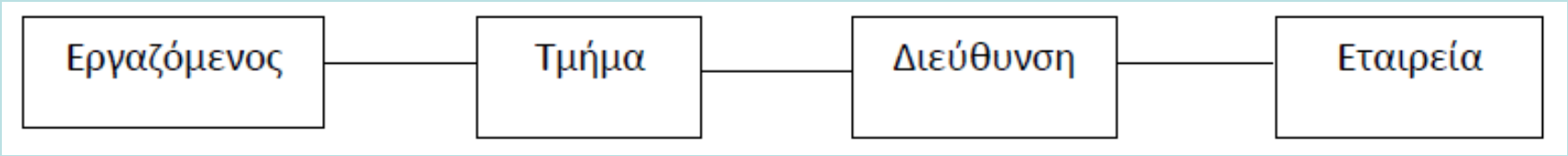


3) Σύνδεση (Association): Αναφέρεται σε αντικείμενα τα οποία συνδέονται με κάποιο τρόπο με άλλα. Η σύνδεση παριστάνεται με μια ευθεία γραμμή ανάμεσα στα 2 αντικείμενα.

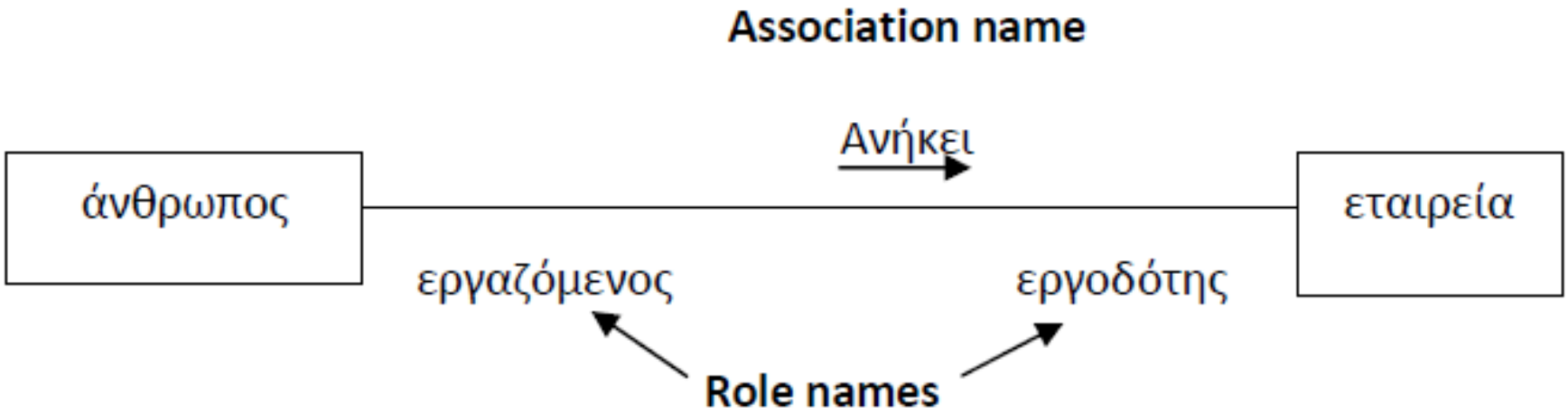
- Αν η σύνδεση δεν είναι αμφίδρομη τότε η κατεύθυνση μπορεί να οριστεί με ένα βέλος

- Το όνομα της σύνδεσης μπορεί να γραφτεί πάνω από τη γραμμή.

- Ο αριθμός που δηλώνει πόσα αντικείμενα αντιστοιχούν σε κάθε αντικείμενο στην άλλη άκρη της σχέσης (πολλαπλότητα – multiplicity) δηλώνεται από ένα αριθμό (π.χ. 5) ή μια περιοχή αριθμών (π.χ. 1...* για ένα έως πολλά).

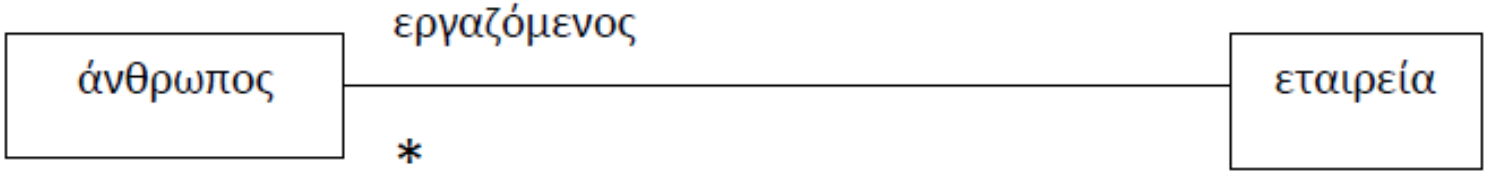

Εργαζόμενος
Τμήμα
Διεύθυνση
Εταιρεία

Association name και Role names

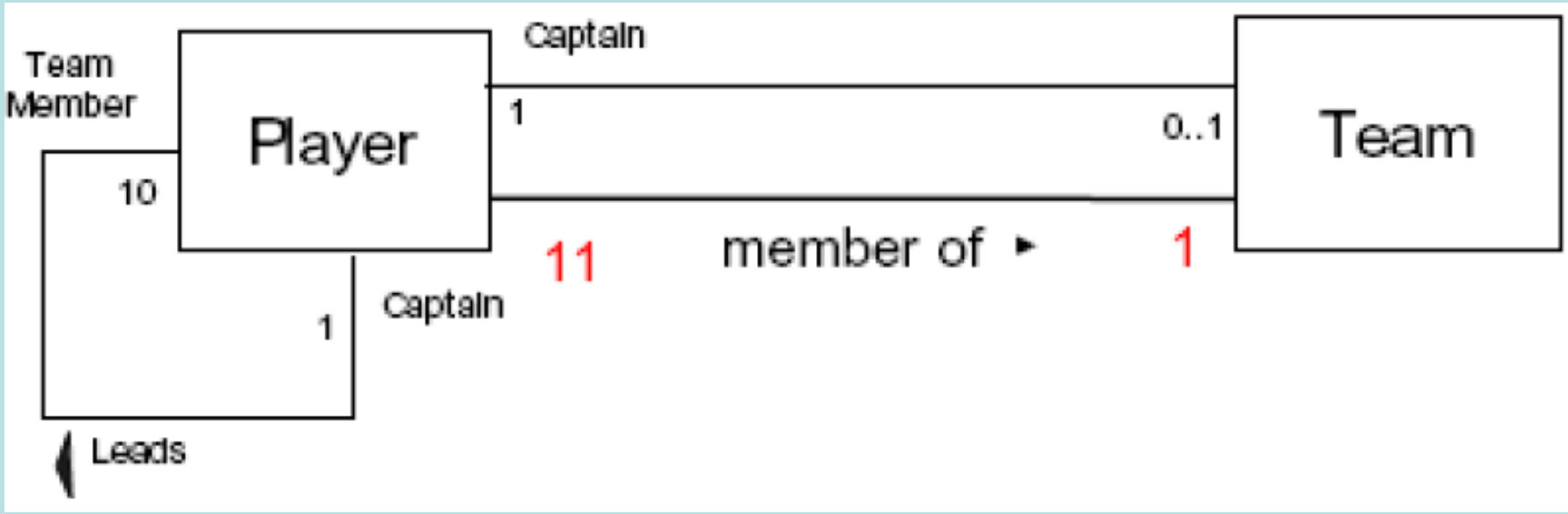


Πολλαπλότητα (Multiplicity)

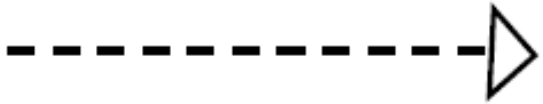
Καθορίζει πόσα αντικείμενα μετέχουν σε μια συσχέτιση (από 0 έως οποιοδήποτε πλήθος).



Ακριβώς ένα	1
Μηδέν ή ένα	0..1
Μηδέν ή περισσότερα	0...*
Ένα ή περισσότερα	1..*
Από....Έως	3...10
Πολλαπλά διαφορετικά όρια	2, 5..9, 10


Team
Member
Player
Captain
1
0..1
Team
10
11
member of
1
Captain
1
Leads

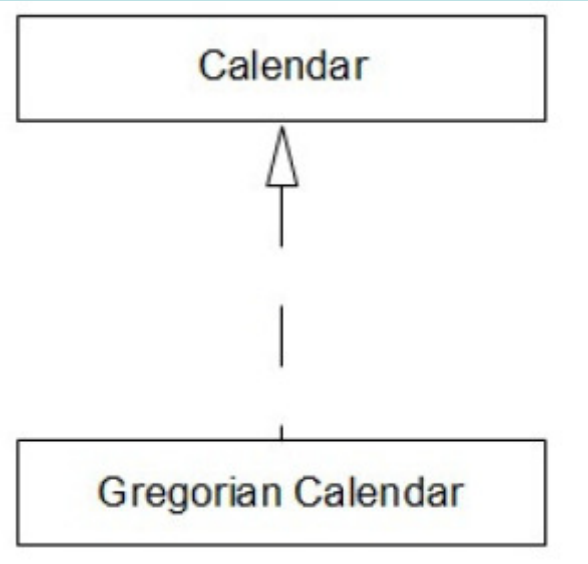
4) Υλοποίηση (realisation): Η υλοποίηση δηλώνει πως ο εξυπηρετούμενος υποστηρίζει τη διεπαφή. Μια σχέση υλοποίησης υπάρχει μεταξύ 2 κλάσεων όταν η μία από αυτές (the client) πραγματοποιεί ή εφαρμόζει τη συμπεριφορά (behaviour) που καθορίζεται από την άλλη (the supplier).

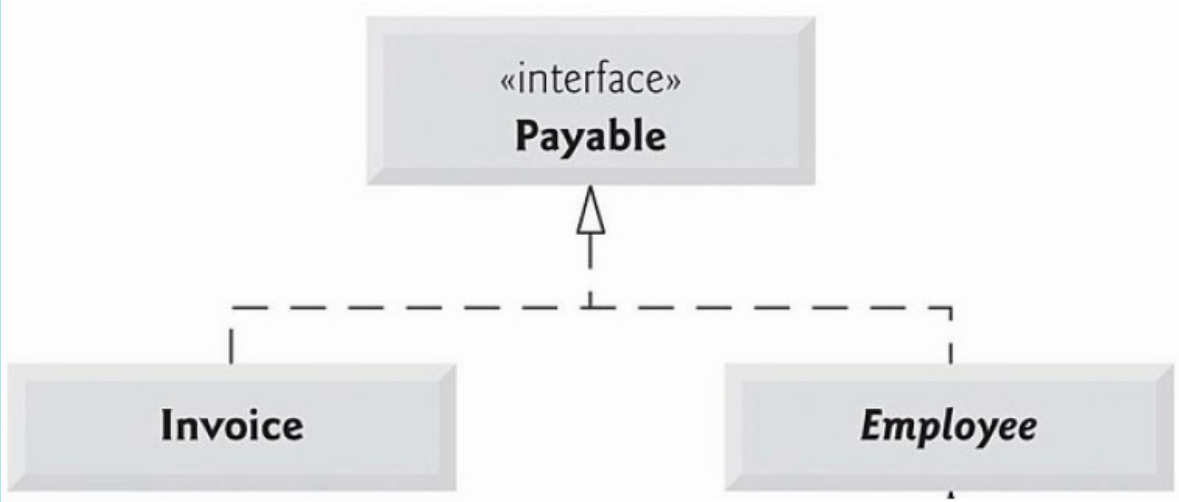


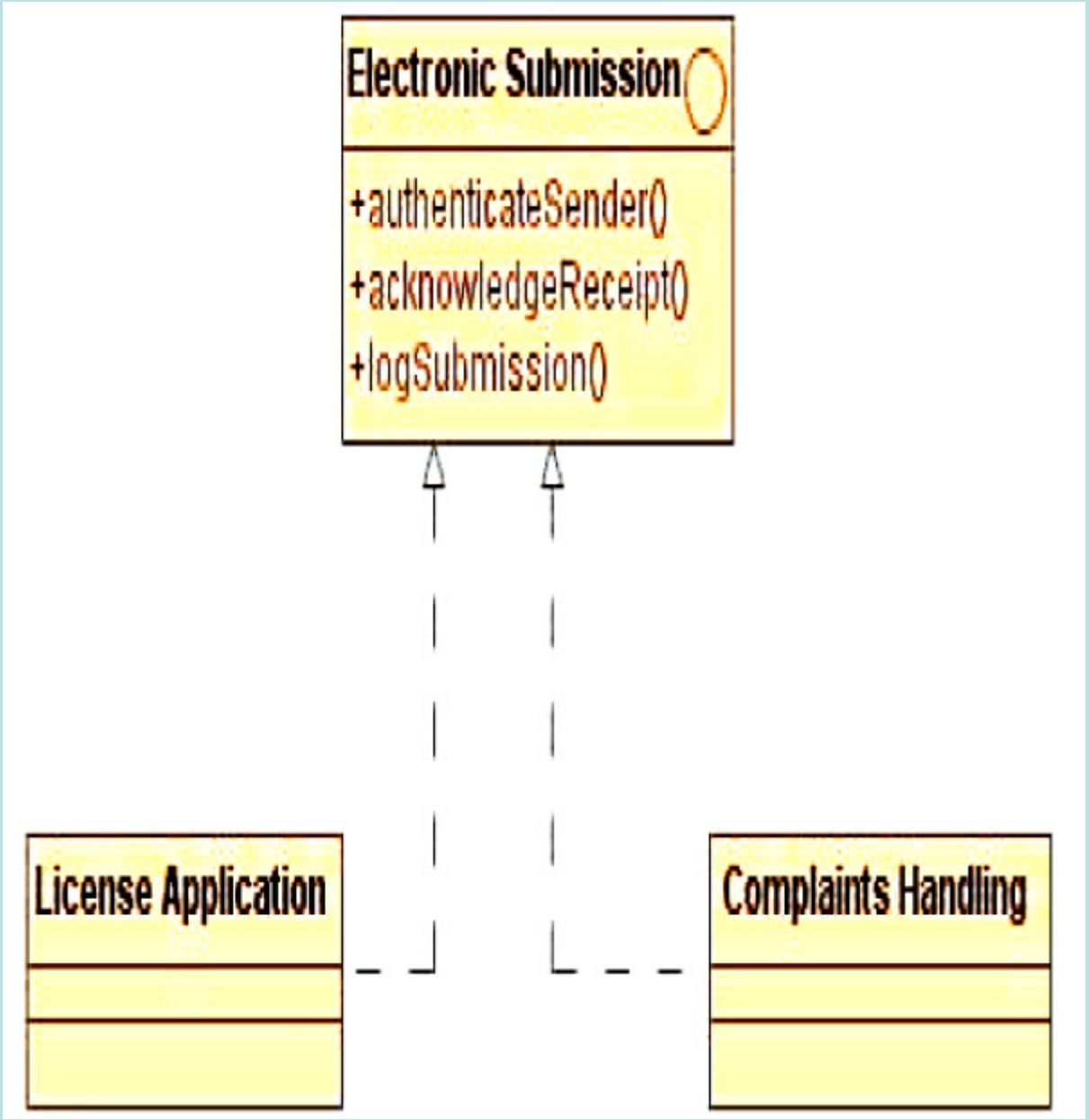
Υλοποίηση - Realization

1. Επιτρέπει σε μία κλάση να κληρονομεί από μια interface class, χωρίς να είναι sub-class της interface class.

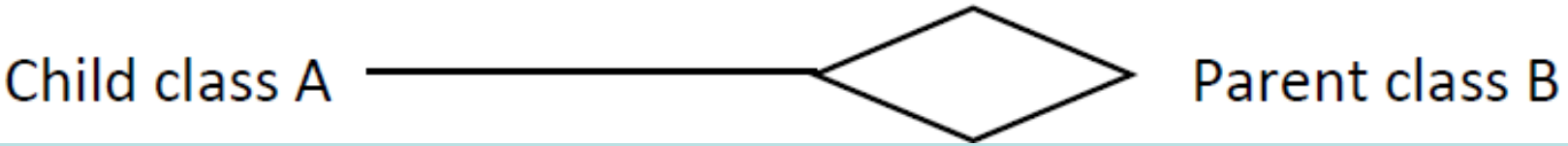
2. Κληρονομεί μόνο operations

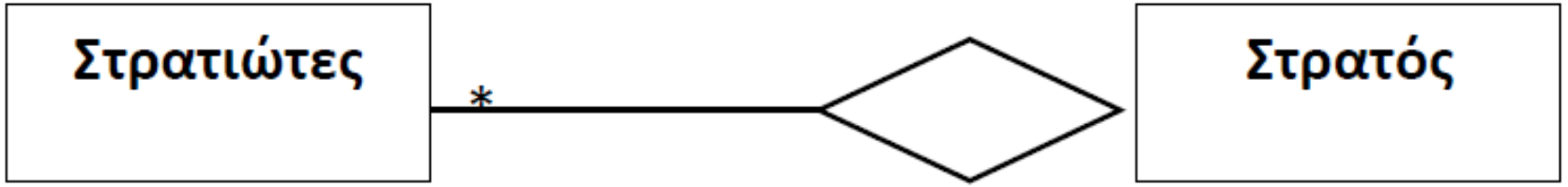




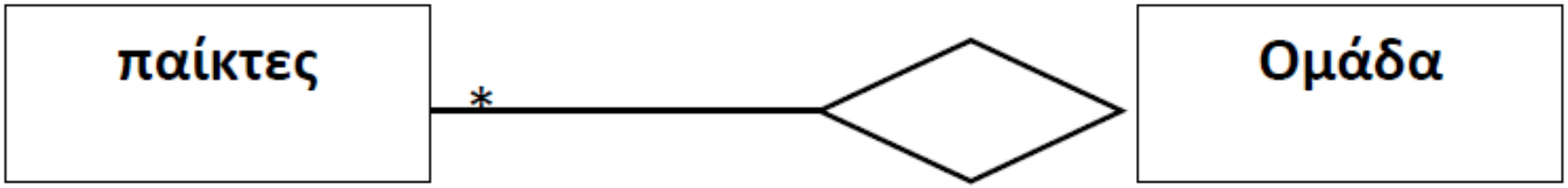

Electronic Submission
+authenticateSender()
+acknowledgeReceipt()
+logSubmission()
License Application
Complaints Handling

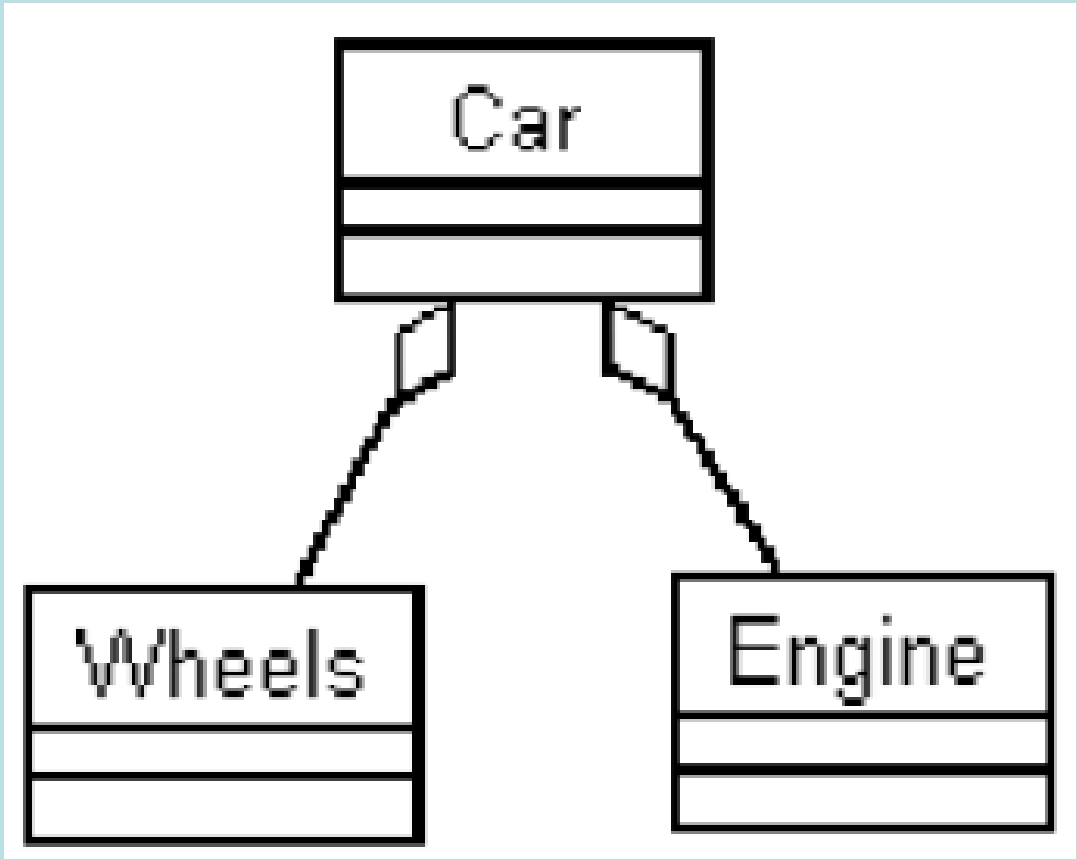
5) Συγκρότημα (Aggregation): Αν σε μια σχέση τα αντικείμενα απαρτίζουν τμήματα ενός όλου, τότε αυτή απεικονίζεται ως συγκρότημα (aggregation).





Οι στρατιώτες (τμήματα) συνθέτουν το Στρατό (σύνολο). Τα τμήματα (οι στρατιώτες) υπάρχουν, έστω κι αν το σύνολο καταστραφεί.

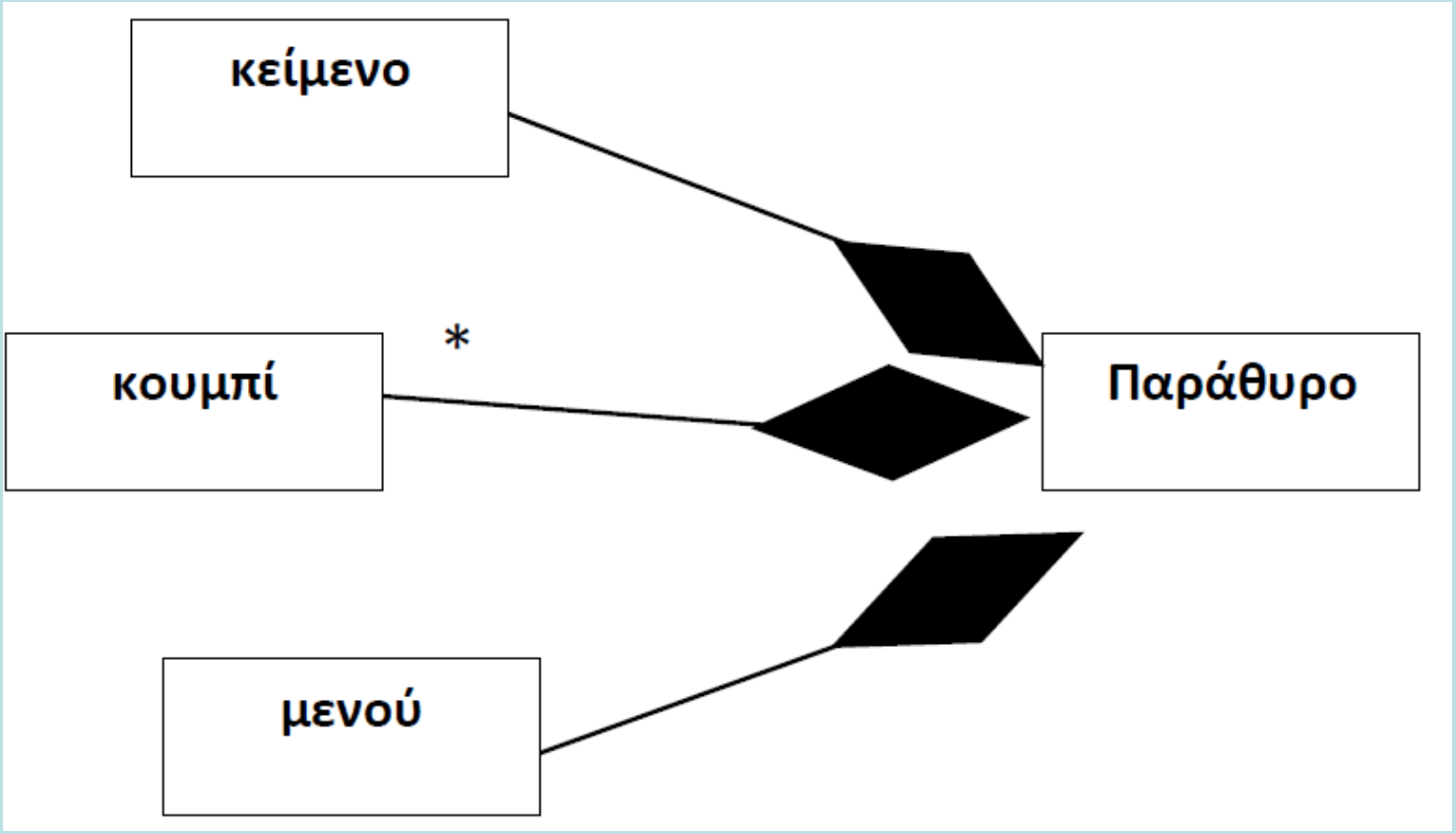



Car
Wheels
Engine

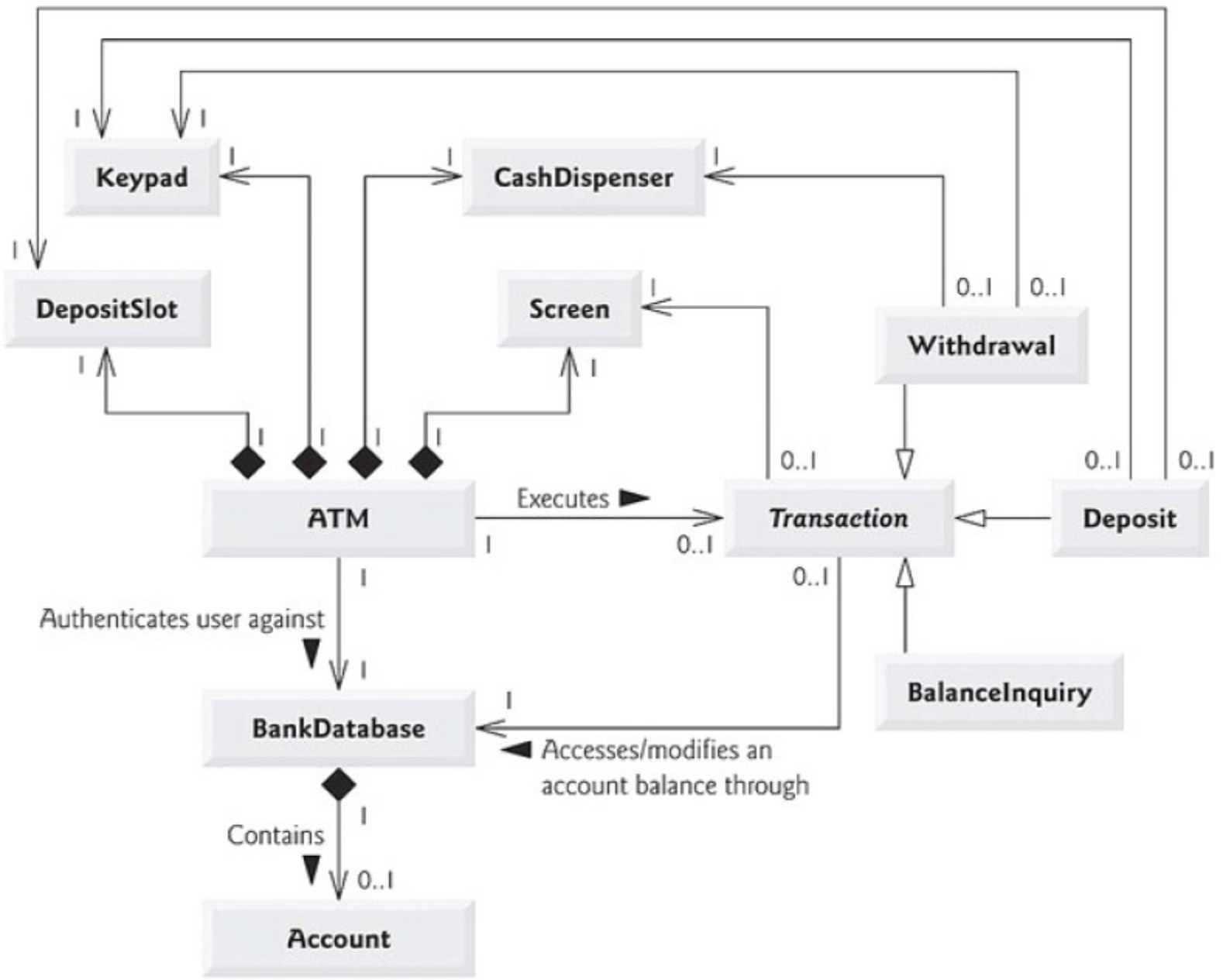
6) Συνθέσεις (Compositions): Η σύνθεση (composition) είναι μια σχέση όλου και μερών που το αποτελούν. Στη σύνθεση η διάρκεια ζωής των μερών περιέχεται στην διάρκεια ζωής του όλου, δηλαδή τα μέρη δεν ζουν χωρίς το όλο στο οποίο περιέχονται.

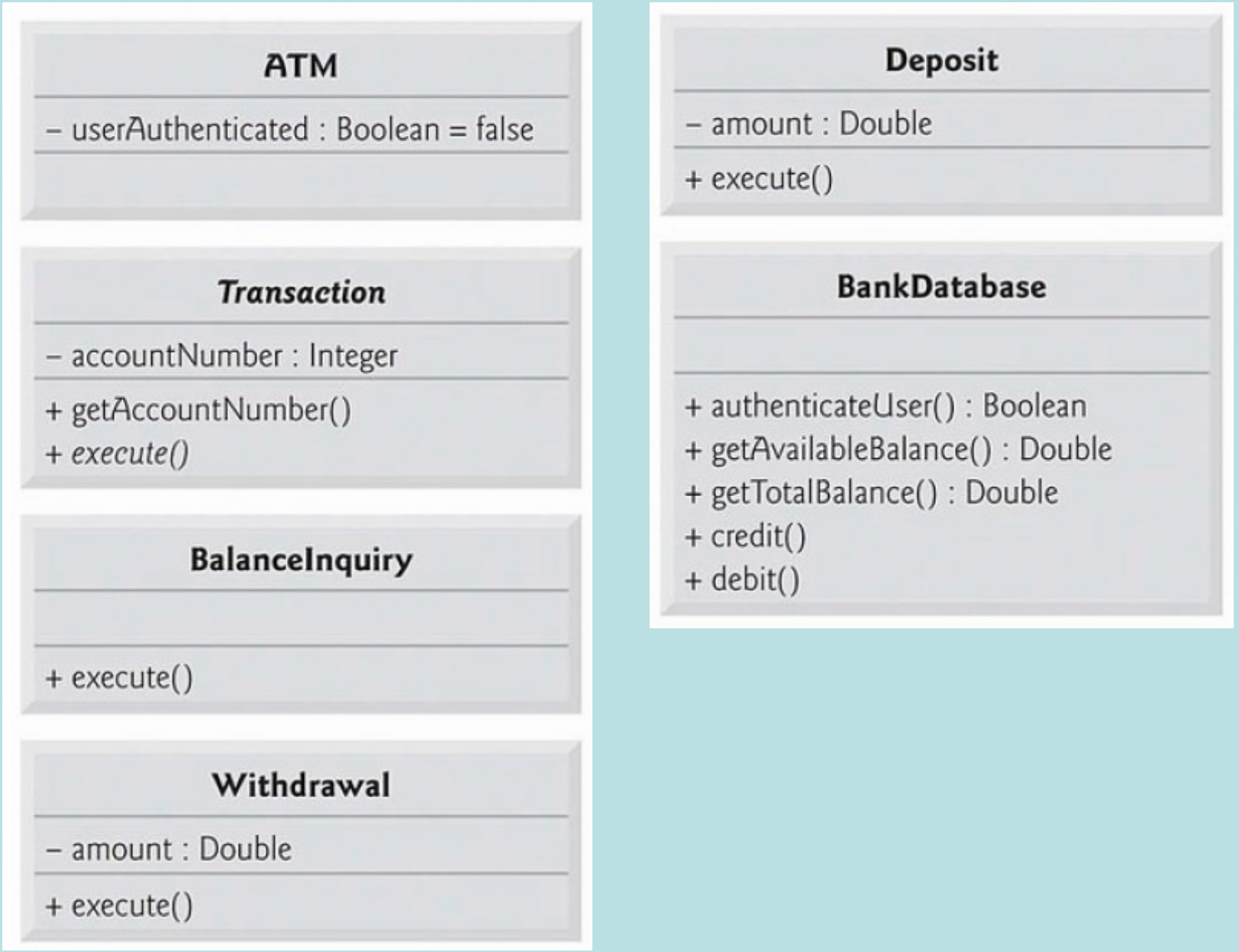
Το composition παρουσιάζει ισχυρή ιδιοκτησία των τμημάτων του.

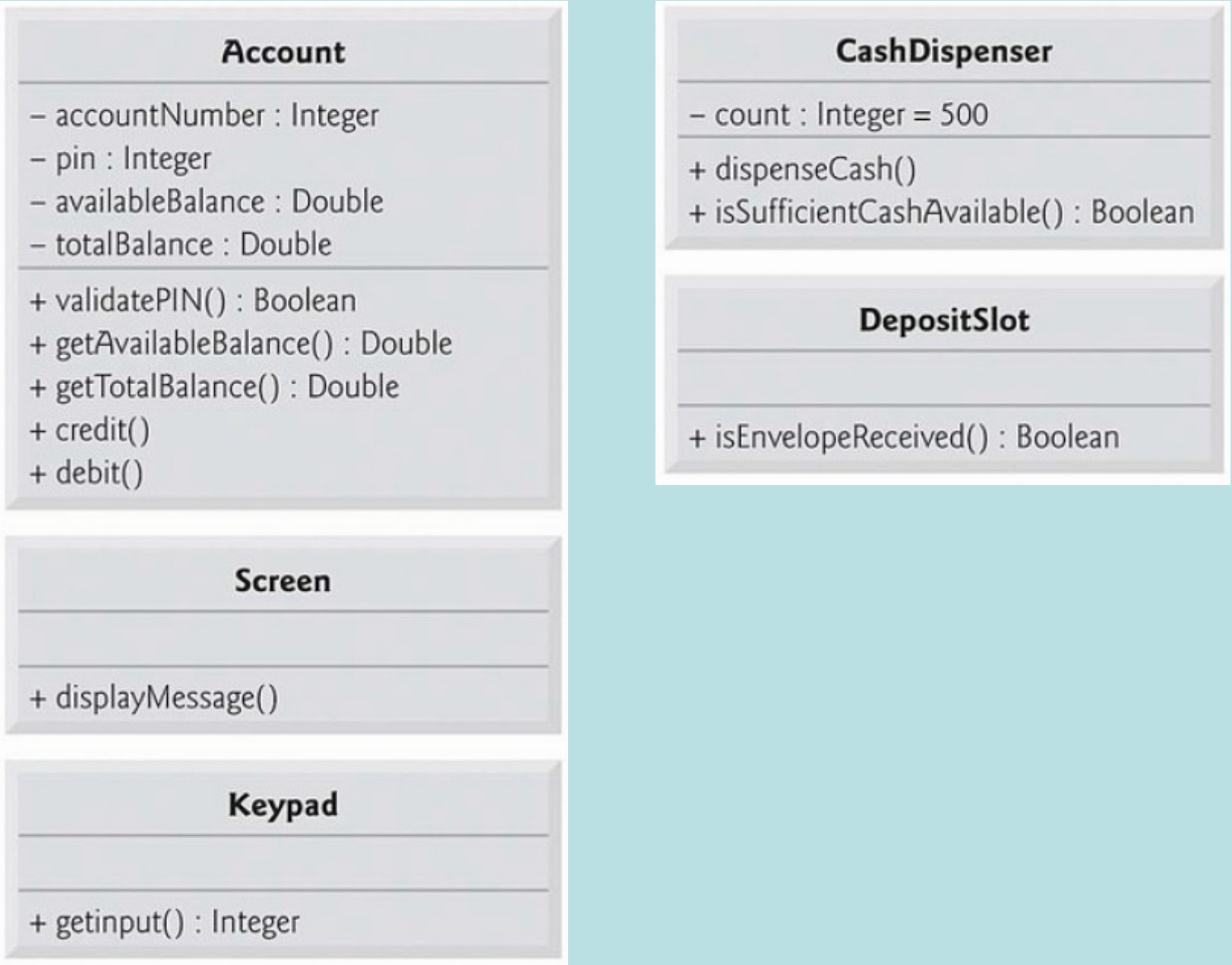
- Τα τμήματα που «ζουν» μέσα στο σύνολο θα καταστραφούν , όταν καταστραφεί και το σύνολο (π.χ. Το παράθυρο).
- Το διαμάντι δείχνει τη συνάθροιση σύνθεσης. Το παράθυρο περιέχει ένα μενού, πολλά κουμπιά και κείμενο.



ATM

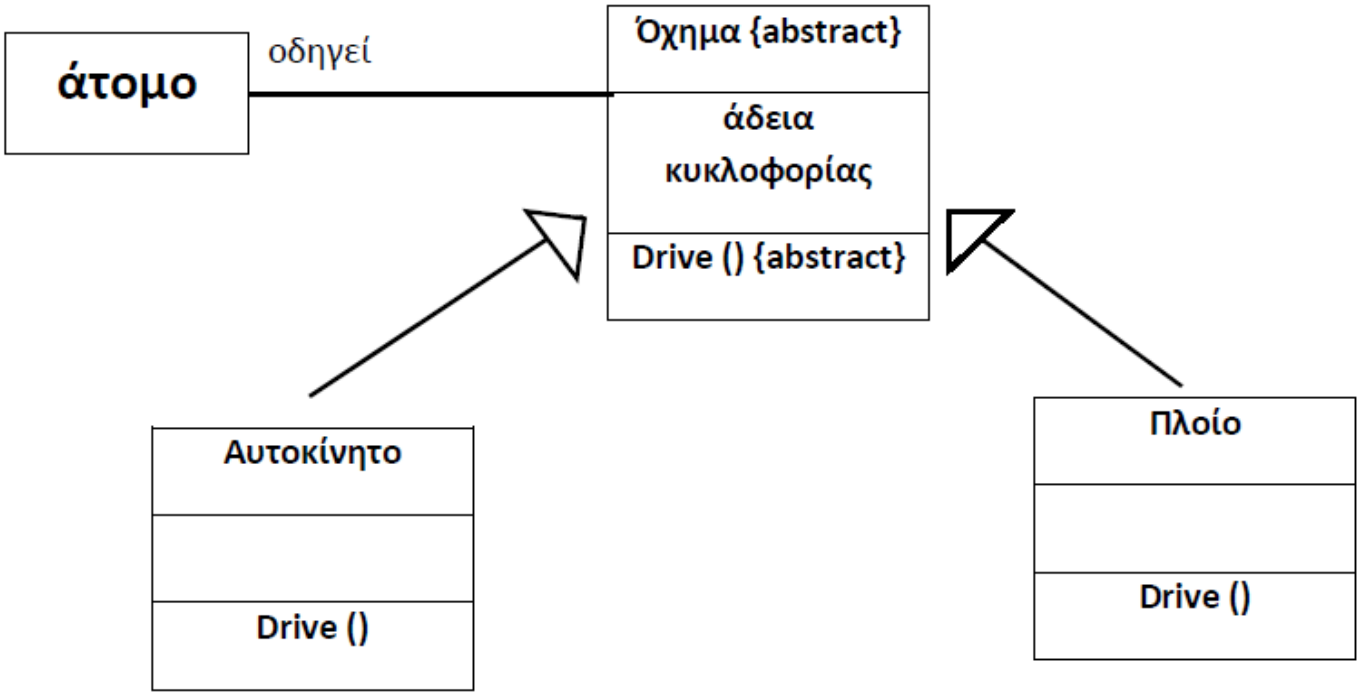

Keypad
CashDispenser
DepositSlot
Screen
Withdrawal
ATM
Executes
Transaction
Deposit
Authenticates user against
BalanceInquiry
BankDatabase
Accesses/modifies an account balance through
Contains
Account
1
0..1


ATM
– userAuthenticated : Boolean = false
Transaction
– accountNumber : Integer
+ getAccountNumber()
+ execute()
BalanceInquiry
+ execute()
Withdrawal
– amount : Double
+ execute()
Deposit
– amount : Double
+ execute()
BankDatabase
+ authenticateUser() : Boolean
+ getAvailableBalance() : Double
+ getTotalBalance() : Double
+ credit()
+ debit()


Account
– accountNumber : Integer
– pin : Integer
– availableBalance : Double
– totalBalance : Double
+ validatePIN() : Boolean
+ getAvailableBalance() : Double
+ getTotalBalance() : Double
+ credit()
+ debit()
Screen
+ displayMessage()
Keypad
+ getinput() : Integer
CashDispenser
– count : Integer = 500
+ dispenseCash()
+ isSufficientCashAvailable() : Boolean
DepositSlot
+ isEnvelopeReceived() : Boolean

Χαρακτηριστικά αφαιρετικής μεθόδου

- Δεν υπάρχει μέθοδος υλοποίησης στην κλάση που εμφανίζεται η συγκεκριμένη λειτουργία.
- Μια κλάση που έχει τουλάχιστον μια abstract μέθοδο, είναι μια abstract κλάση.



Αυτοκίνητο : Drive() αρχίζει τους τροχούς

Πλοίο : Drive () αρχίζει την προπέλα

Ένα άτομο οδηγά οχήματα. Όταν καλείται η μέθοδος οδήγησης, η επιλεγόμενη υλοποίηση εξαρτάται από το αν το χρησιμοποιούμενο αντικείμενο είναι αυτοκίνητο ή πλοίο. Το όχημα είναι μια abstract κλάση.

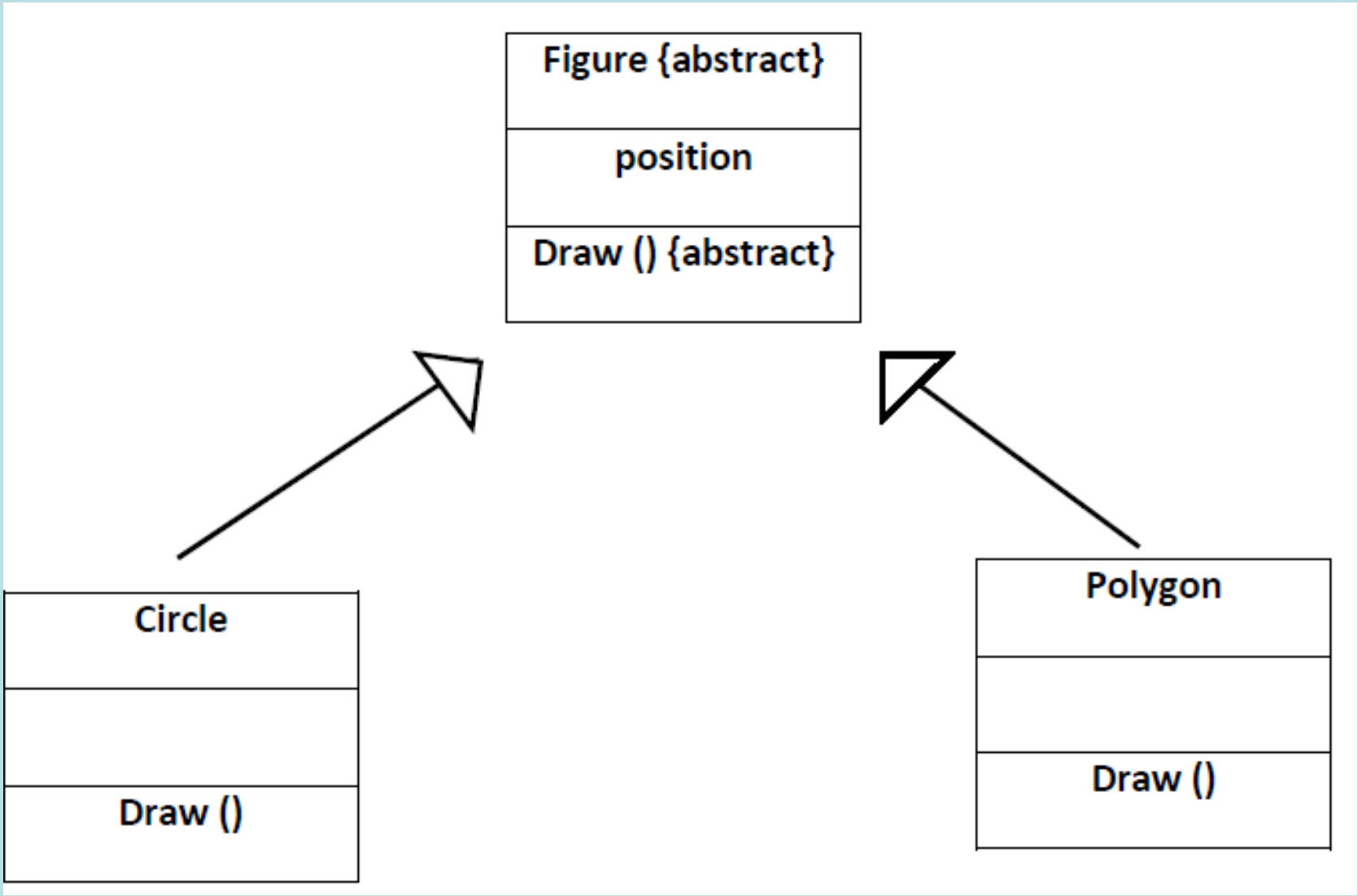


Figure {abstract}
position
Draw () {abstract}
Circle
Draw ()
Polygon
Draw ()

Στη UML , μπορεί να διακριθούν τα εξής τρία είδη μοντελοποίησης:

Α) Δομική Μοντελοποίηση.

Β) Μοντελοποίηση Συμπεριφοράς.

Γ) Αρχιτεκτονική Μοντελοποίηση.

Η Δομική Μοντελοποίηση:

- Περιλαμβάνει τα στατικά χαρακτηριστικά ενός συστήματος.
- Αποτελείται από: Διαγράμματα Κλάσεων (Class),
- Αντικειμένων (Object),
- Ανάπτυξης (Deployment),
- Συστατικών Στοιχείων (Component)
- Σύνθετης Δομής (Composite Structure).

Η Μοντελοποίηση Συμπεριφοράς:

- Περιγράφει τη διασύνδεση μεταξύ των δομικών διαγραμμάτων, και αναδεικνύει τη δυναμική φύση ενός συστήματος.
- Περιλαμβάνει τα διαγράμματα: Περίπτωσης Χρήσης (Use Case),
- Δραστηριοτήτων (Activity),
- Αλληλεπίδρασης (Interaction),
- Μηχανής Καταστάσεων (State Machine),
- Επικοινωνίας (Communication),
- Ακολουθίας (Sequence) και
- Χρονισμού (Timing).

Η Αρχιτεκτονική Μοντελοποίηση:

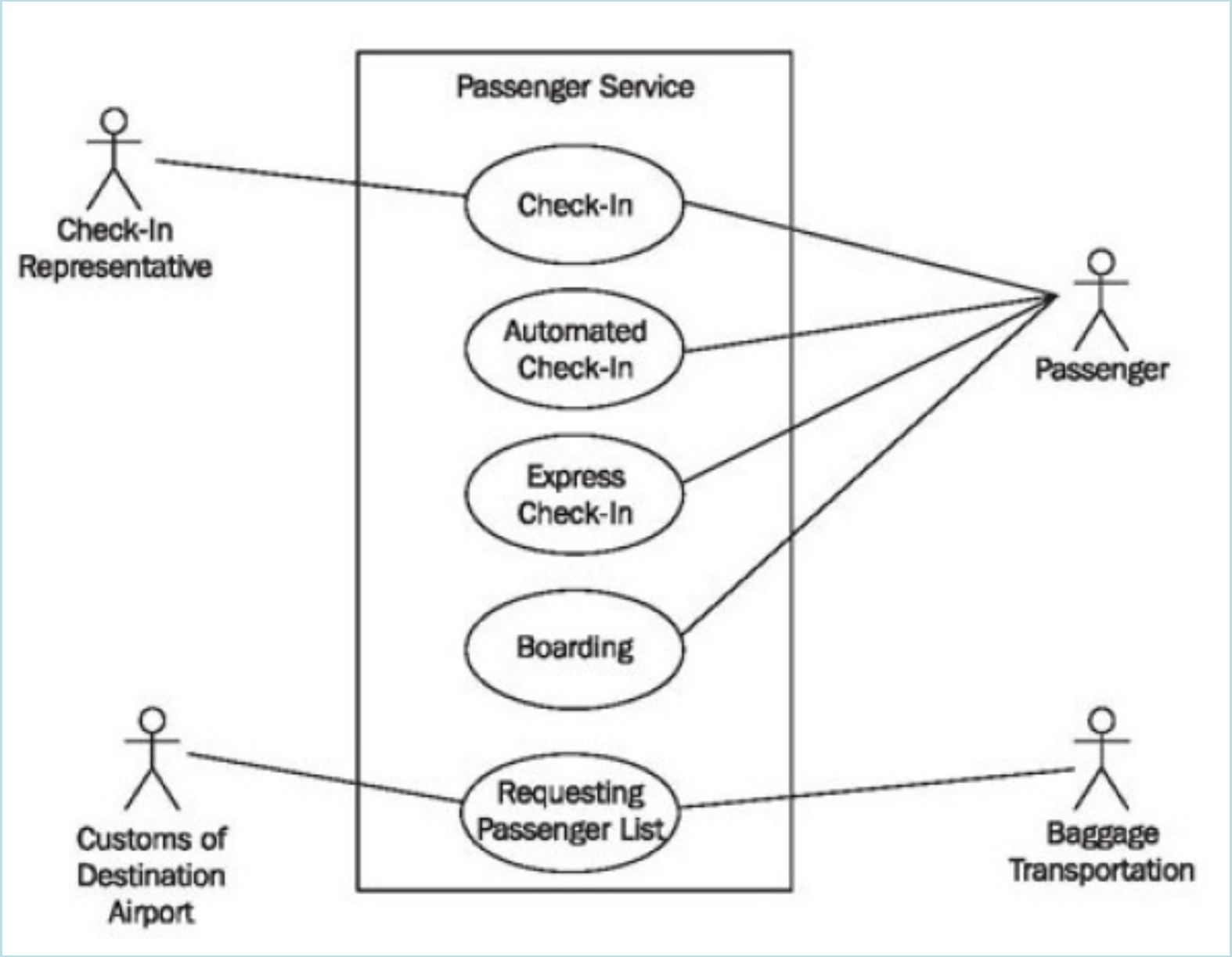
- Περιέχει δομικά στοιχεία, και στοιχεία συμπεριφοράς ενός συστήματος. Περιγράφει πως τα τμήματα του συστήματος είναι οργανωμένα σε υποσυστήματα.
- Μπορεί να θεωρηθεί ότι είναι το προσχέδιο ολόκληρου του συστήματος.
- Τα διαγράμματα Πακέτων (Package) εντάσσονται στην κατηγορία αυτή.

Λίστα των κυριότερων διαγραμματικών τεχνικών της UML και των χρήσεων τους

- Use Case Diagram (διάγραμμα περιπτώσεων χρήσης)
- Activity Diagram (διάγραμμα δραστηριοτήτων)
- Class Diagram (διάγραμμα κλάσεων)
- Interaction Diagram (διάγραμμα αλληλεπίδρασης)
  - Sequence Diagrams (διαγράμματα αλληλουχίας)
  - Communication Diagrams (διαγράμματα επικοινωνίας)
- State Diagram (διάγραμμα καταστάσεων)
- Component Diagram (διάγραμμα εξαρτημάτων)
- Package Diagram (διάγραμμα πακέτων/συσκευασίας)
- Deployment Diagram (διαγράμματα παράταξης)

Use Case diagram

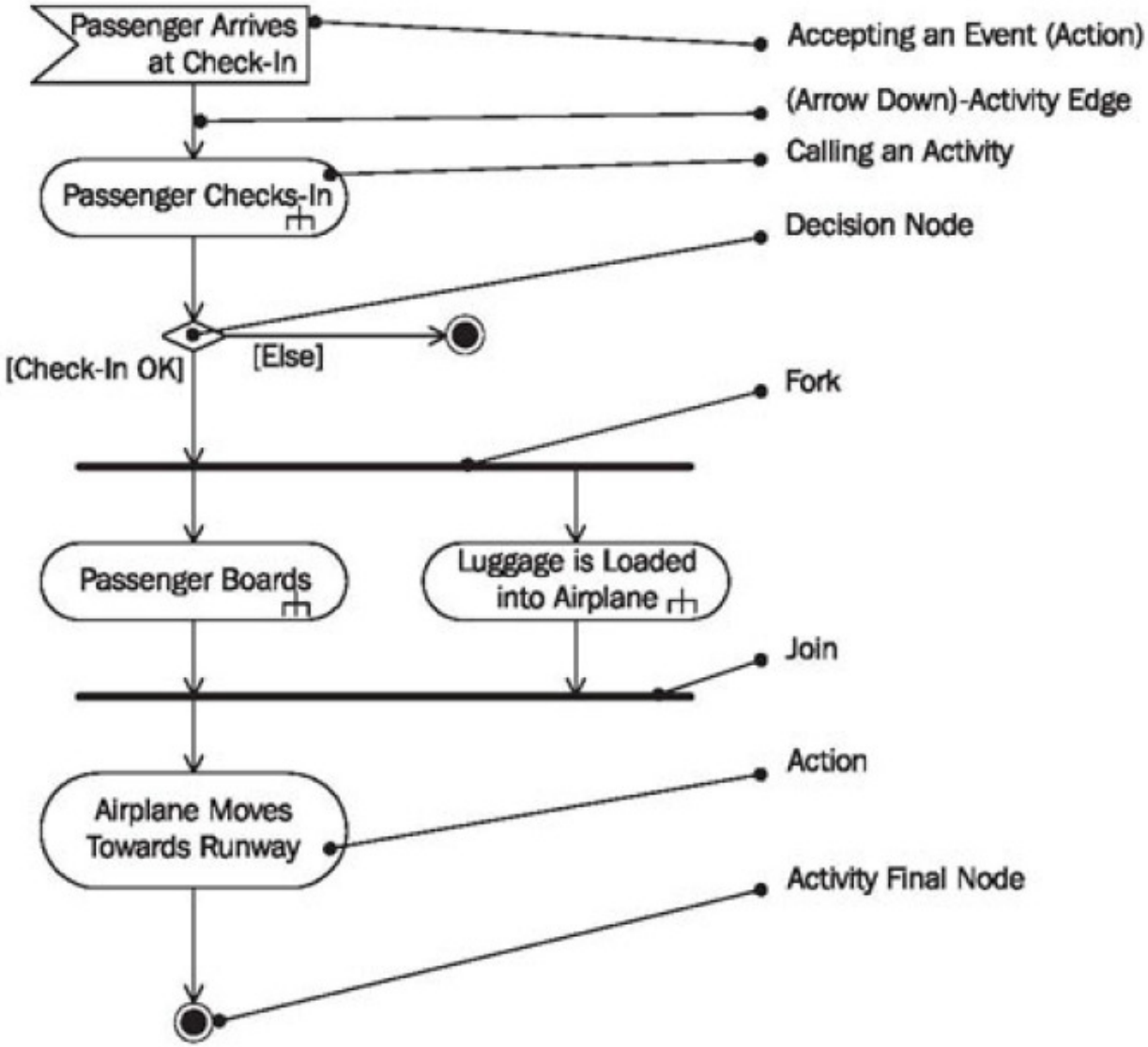
- Απαιτήσεις χρηστών
- Τεκμηρίωση συστήματος
- Χρήση σεναρίων για έλεγχο λειτουργιών


Passenger Service
Check-In Representative
Check-In
Automated Check-In
Express Check-In
Boarding
Passenger
Customs of Destination Airport
Requesting Passenger List
Baggage Transportation

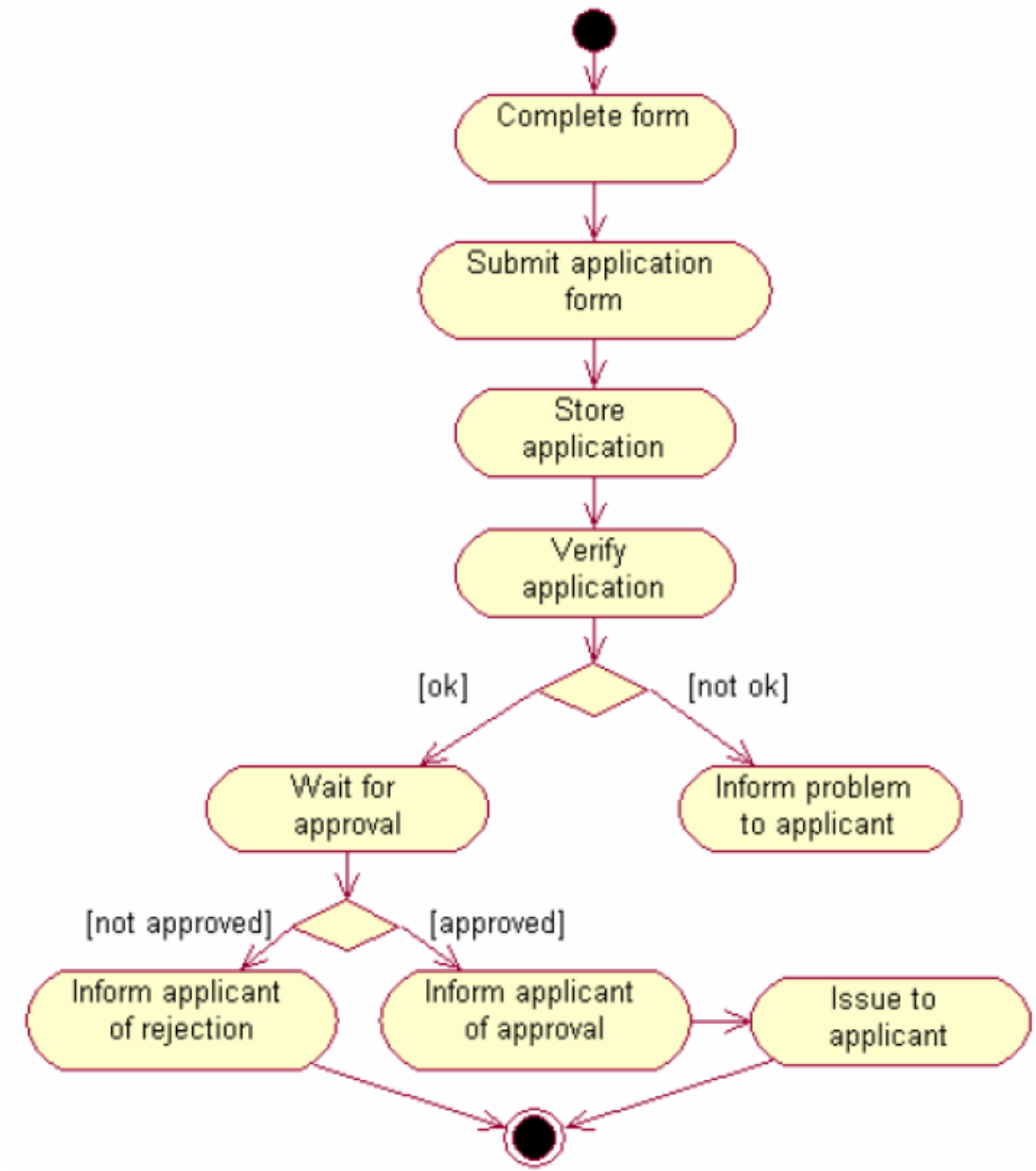
Activity Diagram

- Αναπαράσταση εκτέλεσης λειτουργιών
- Υλοποίηση use case σεναρίων
- Προβολή workflow σε διάφορα σημεία της εφαρμογής

Activity Diagram: check-In



Activity Diagram: Application form



Class diagram

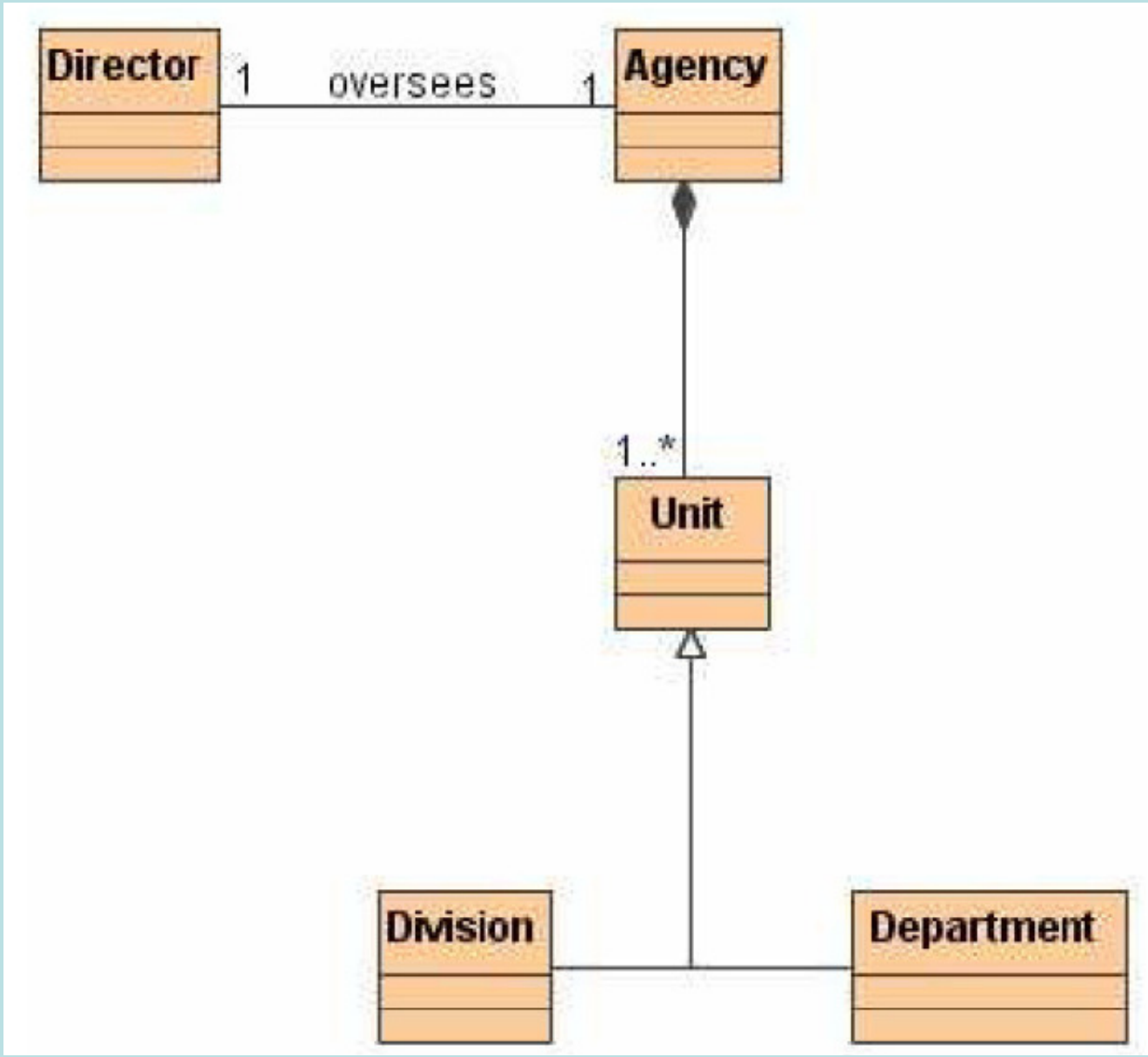
Ένα Class διάγραμμα περιγράφει στατικά τη δομή της εφαρμογής και περιλαμβάνει όλες τις οντότητες (τα αντικείμενα – κλάσεις) που θα χρησιμοποιηθούν σε αυτή. Συνήθως η δημιουργία ενός Class διαγράμματος ακολουθεί τη δημιουργία των use case και των activity διαγραμμάτων και έχει προέλθει μέσα από συνεργασία με τους πελάτες της εφαρμογής προκειμένου να ανακαλυφθούν οι κύριες οντότητες της εφαρμογής.

Από τα Use Case διαγράμματα προκύπτουν οι απαιτήσεις σε αντικείμενα (objects) που χρειάζονται για να επιτευχθούν οι στόχοι (goals) του συστήματος. Μέσα από την ανάλυση των διαγραμμάτων αυτών προσπαθούμε να ανακαλύψουμε τις κύριες οντότητες της εφαρμογής. Τα Activity διαγράμματα χρησιμεύουν στον καθορισμό της συμπεριφοράς (behavior) των αντικειμένων δηλαδή καθορίζουν το πως θα λειτουργούν τα αντικείμενα και ποια θα είναι η ροή της εργασία καθ' όλη τη διάρκεια της εφαρμογής προκειμένου να επιτευχθεί ο σκοπός της εφαρμογής.

Class diagram

Τα Class διαγράμματα περιγράφουν την κατασκευαστική δομή (structural view) του συστήματος, δηλαδή τους τύπους όλων των αντικειμένων που θα χρησιμοποιηθούν στο σύστημα, παρουσιάζοντας την στατική δομή των αντικειμένων αυτών και περιγράφει το είδος των συσχετίσεων που υπάρχουν μεταξύ των αντικειμένων αυτών. Με την ανάλυση του Class διαγράμματος βλέπουμε τις κλάσεις που δημιουργούνται για την υλοποίηση της εφαρμογής. Τα βασικότερα δομικά στοιχεία των Class διαγραμμάτων είναι:

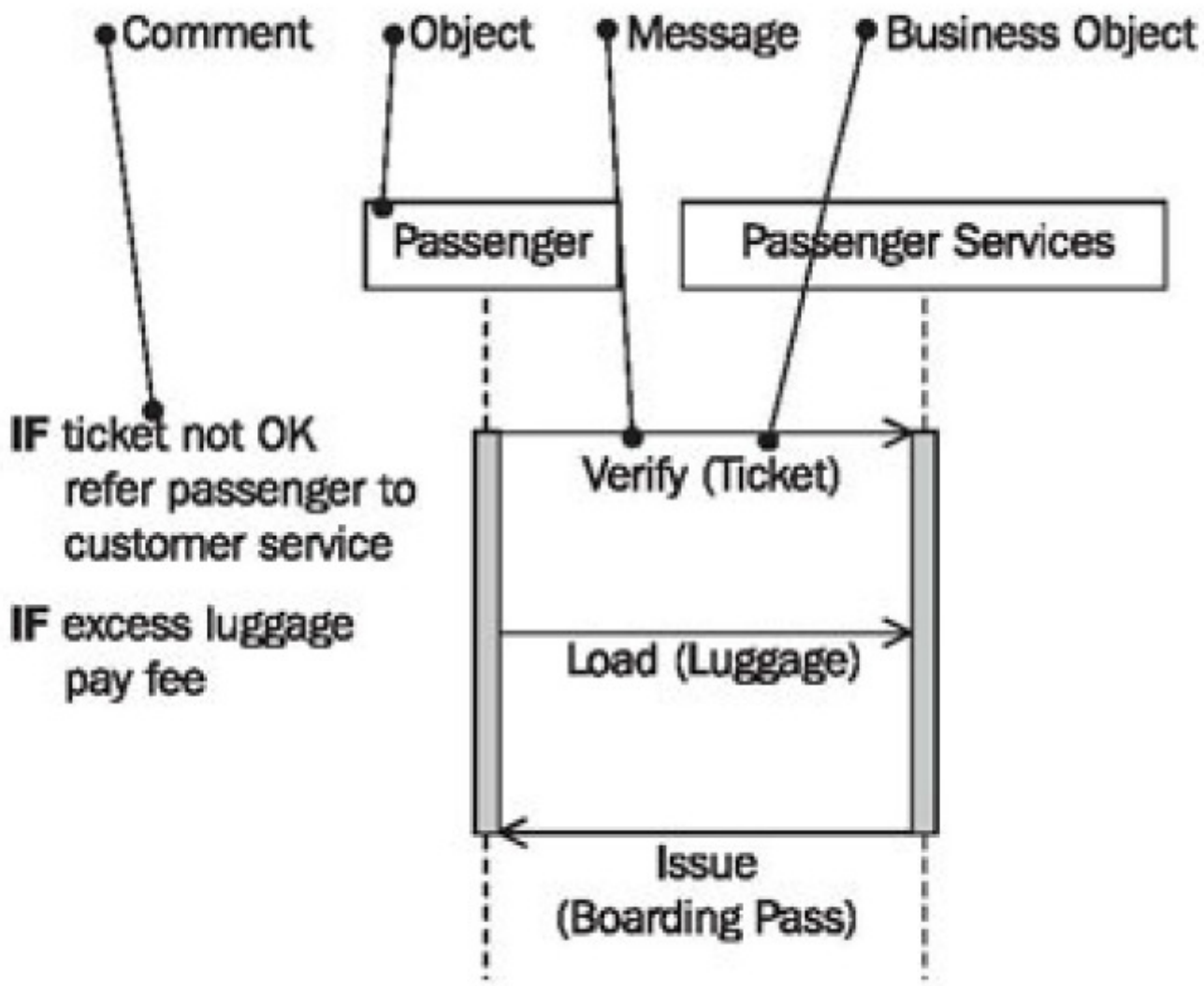
- Οι κλάσεις (δομή και συμπεριφορά τους – μέθοδοι κλάσεων).
- Οι συσχετίσεις μεταξύ των κλάσεων.
- Η πολλαπλότητα (multiplicity) και navigation (ροή μέσα στις κλάσεις).


Director
1
oversees
1
Agency
1..*
Unit
Division
Department

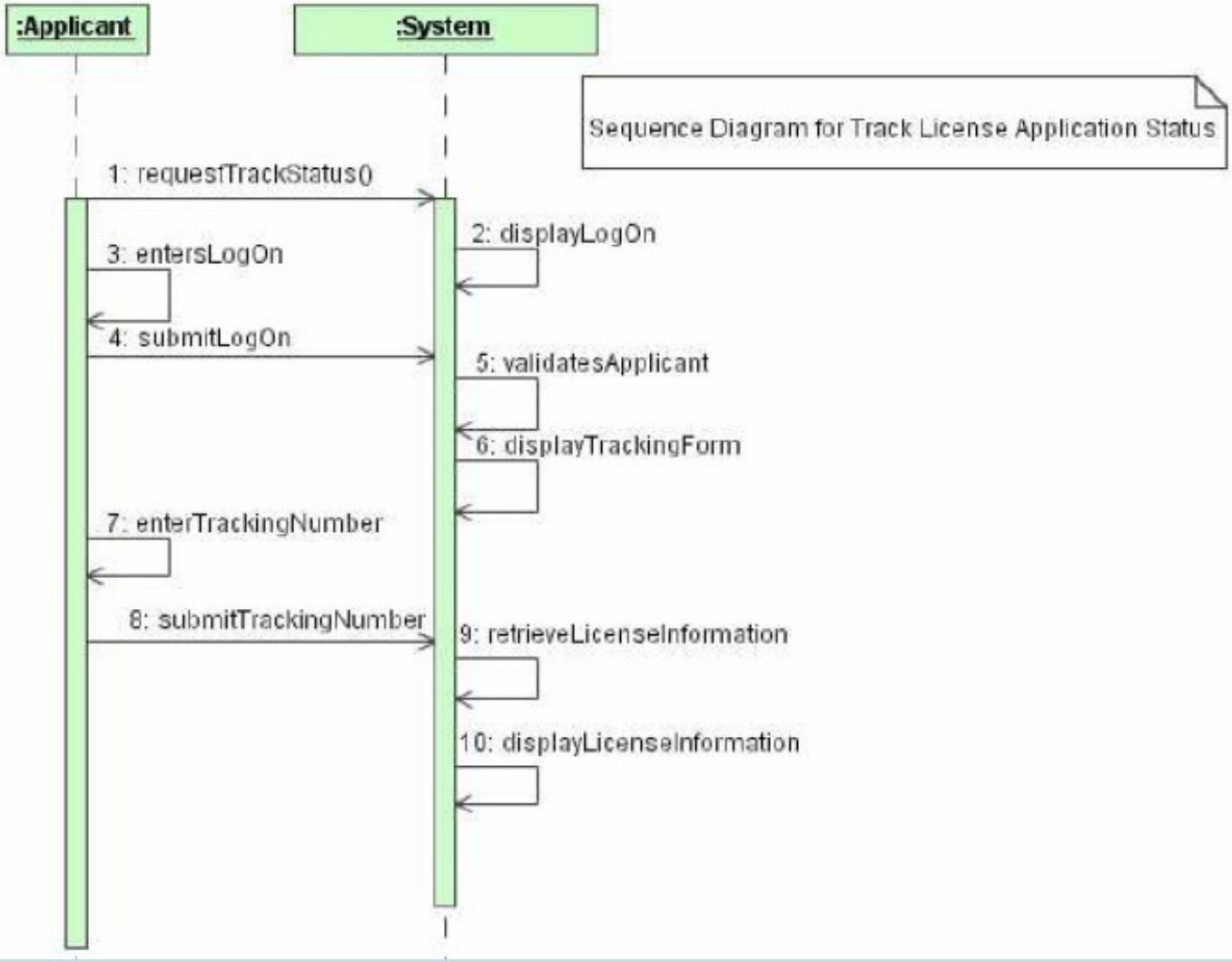
Sequence Diagrams

- Προβολή αλληλεπιδράσεων μεταξύ των αντικειμένων στη διάρκεια του χρόνου.
- Προβολή ανταλλαγής μηνυμάτων μεταξύ των αντικειμένων.
- Περιγραφή συγκεκριμένων λειτουργιών του συστήματος.
- Τα αντικείμενα εντοπίζονται από τα class διαγράμματα.

Sequence Diagram: Check-In

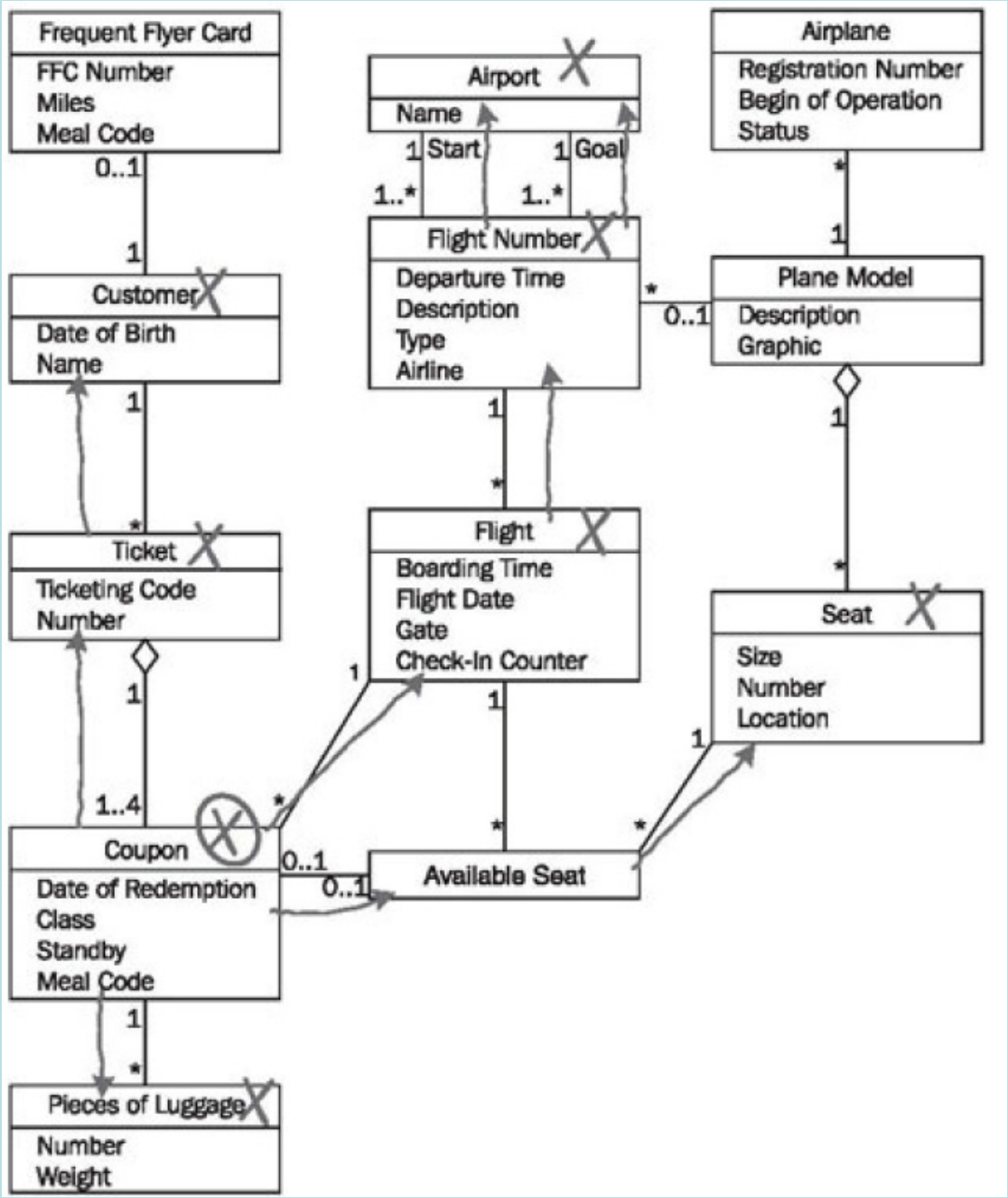


Sequence Diagram: Applicant

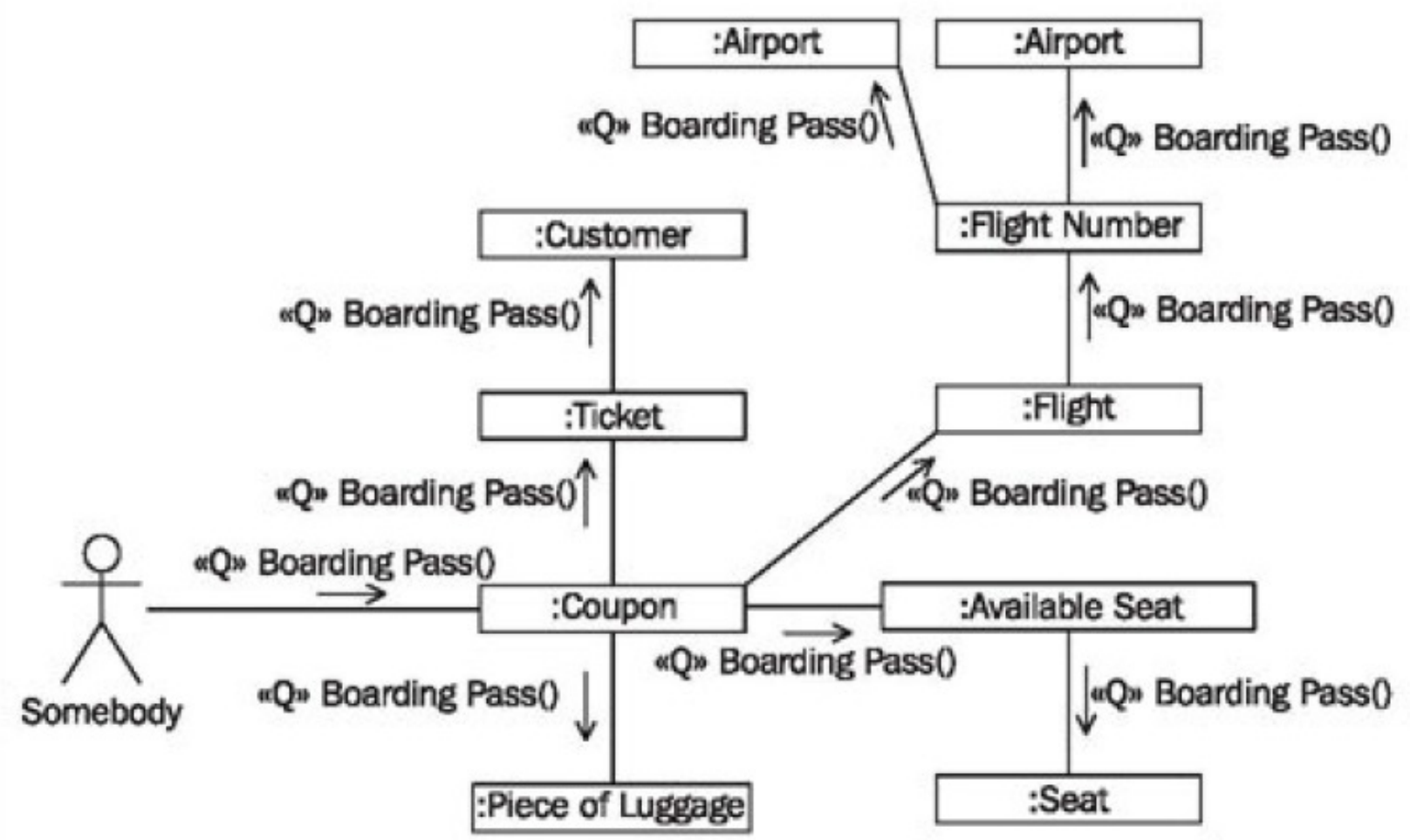


Communication (ή Collaboration) Diagrams

- Παρόμοια χρήση με τα Sequence διαγράμματα.
- Αναπαριστά σειρά ανταλλαγής μηνυμάτων.


Frequent Flyer Card
FFC Number
Miles
Meal Code
0..1
1
Customer
Date of Birth
Name
1
*
Ticket
Ticketing Code
Number
1
1..4
Coupon
Date of Redemption
Class
Standby
Meal Code
1
*
Pieces of Luggage
Number
Weight
Airport
Name
1 Start
1 Goal
1..*
1..*
Flight Number
Departure Time
Description
Type
Airline
1
*
Flight
Boarding Time
Flight Date
Gate
Check-In Counter
1
*
1
*
0..1
0..1
Available Seat
Airplane
Registration Number
Begin of Operation
Status
*
1
Plane Model
Description
Graphic
*
0..1
1
*
Seat
Size
Number
Location
1
*

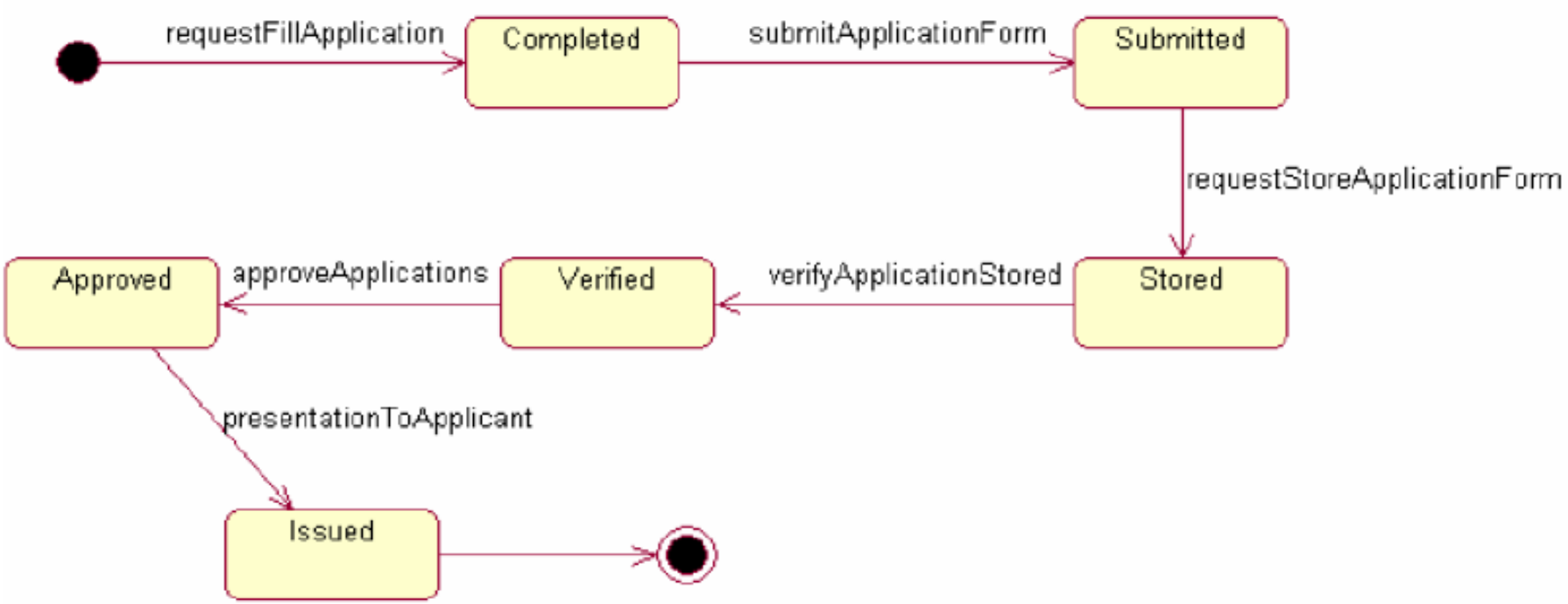
Communication Diagram: Boarding



State chart diagrams

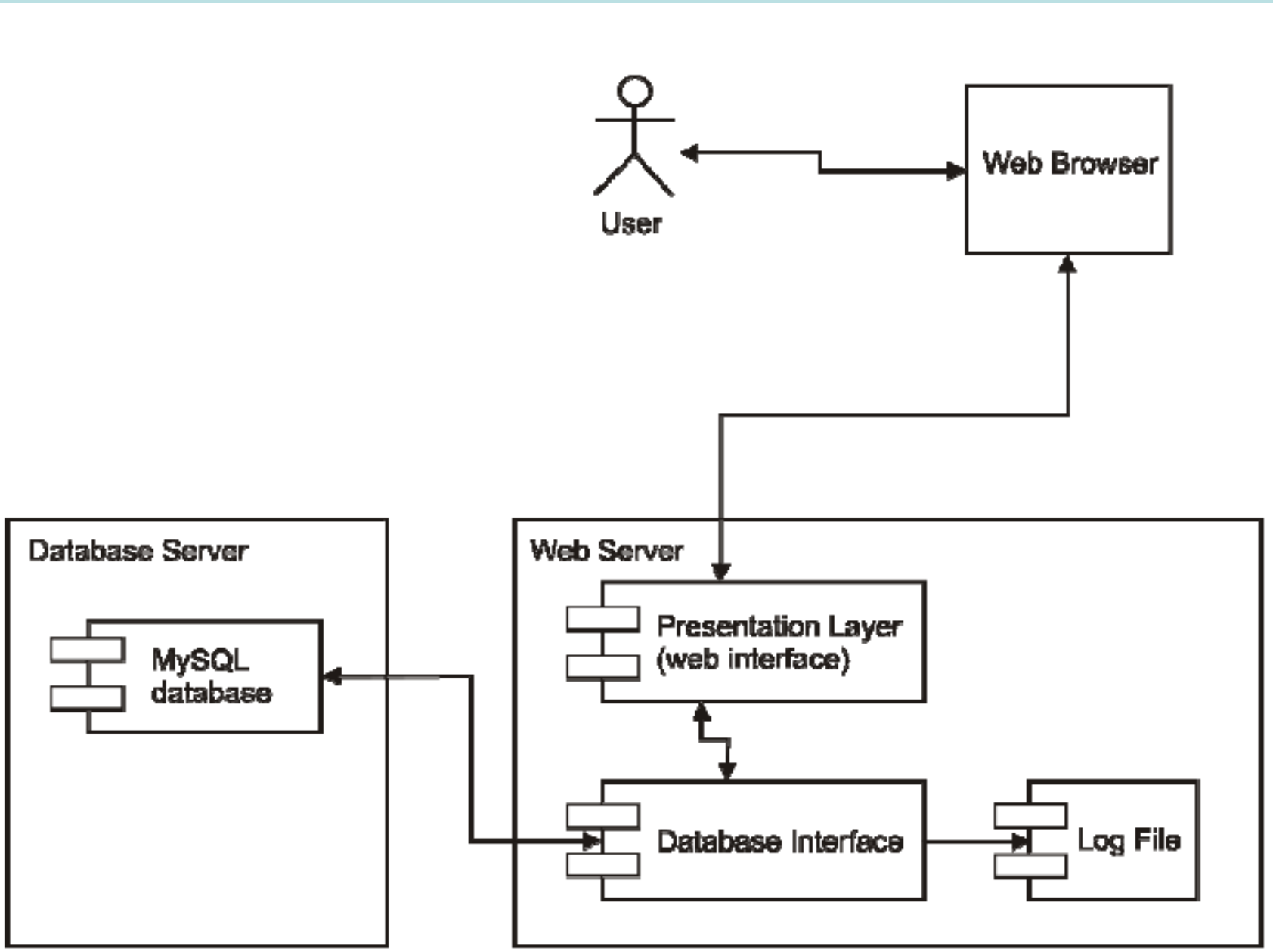
- Αναπαριστούν την διάρκεια ζωής των αντικειμένων.
- Παρουσιάζονται οι καταστάσεις των βασικών αντικειμένων της εφαρμογής.
- Παρουσιάζονται τα γεγονότα που αλλάζουν μια κατάσταση.
- Παρουσιάζουν τις εσωτερικές λειτουργίες ενός αντικειμένου σε μία κατάσταση
- Τα αντικείμενα προκύπτουν από τα use case, sequence και class διαγράμματα

Statechart Diagram: Application form



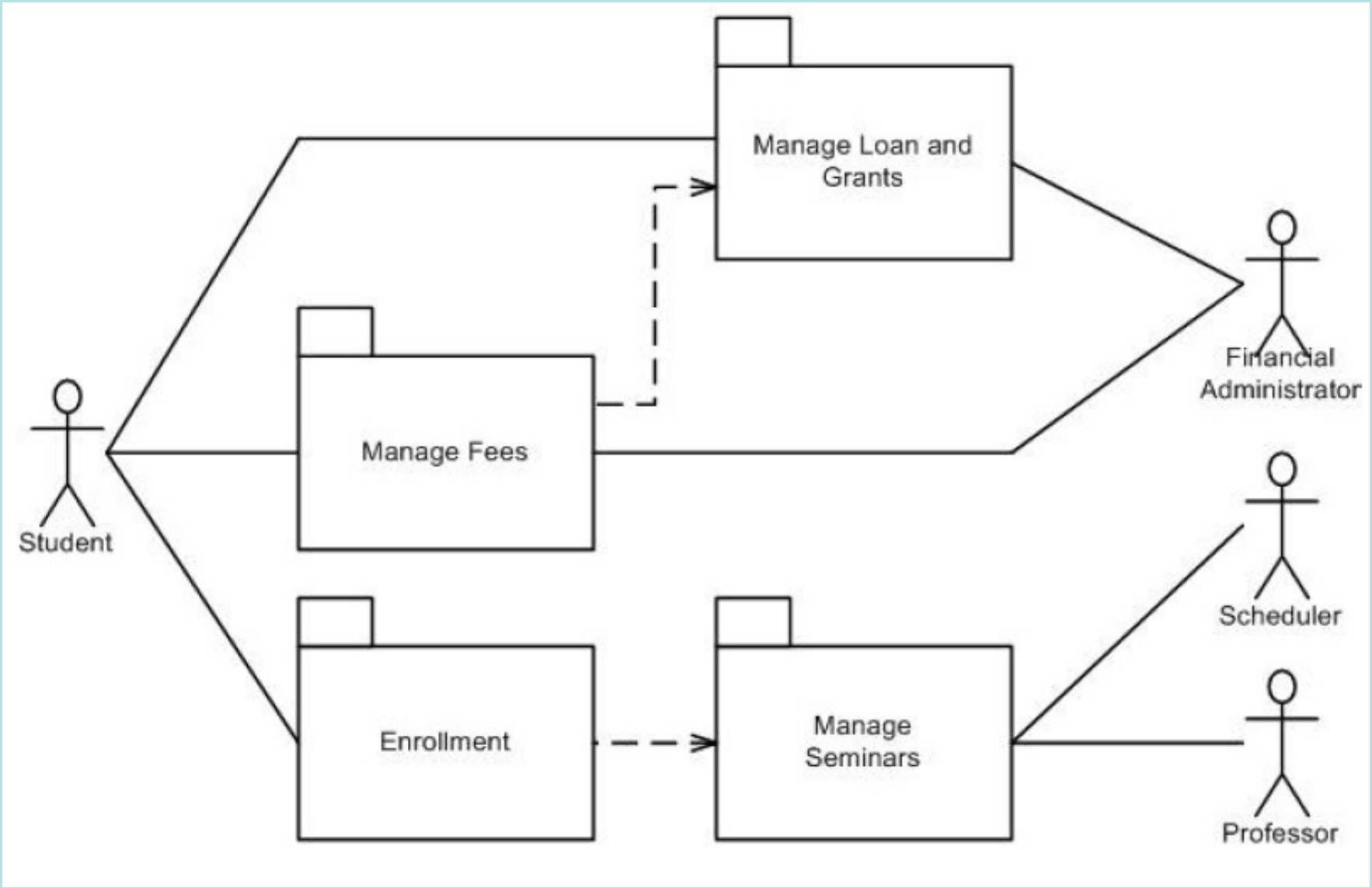
Component diagrams

- Αναπαριστούν την φυσική δομή των τμημάτων λογισμικού της εφαρμογής.
- Παρουσιάζονται οι συσχετίσεις μεταξύ τους.
- Δείχνει την οργάνωση και τις συσχετίσεις μεταξύ των τμημάτων του λογισμικού.
- Παρουσιάζει μια σχετική άποψη του συστήματος.
- Κάθε τμήμα του λογισμικού απεικονίζει μία ή περισσότερες κλάσεις.


User
Web Browser
Database Server
MySQL database
Web Server
Presentation Layer (web interface)
Database Interface
Log File

Package diagrams

- Αποτελεί οργάνωση υψηλού επιπέδου αφαίρεσης.
- Ένα package διάγραμμα απεικονίζει τις εξαρτήσεις μεταξύ των packages που αποτελούν το σύστημα.
- Όλα τα στοιχεία στη UML ομαδοποιούνται σε packages. Έτσι classes, objects, use cases, components οργανώνονται σε packages.


Manage Loan and Grants
Financial Administrator
Manage Fees
Student
Scheduler
Enrollment
Manage Seminars
Professor

Deployment diagrams

- Αναπαριστούν την φυσική μορφή του hardware που θα χρησιμοποιείται από την εφαρμογή.

- Σε συνδυασμό με τα component διαγράμματα δημιουργούν μια ολοκληρωμένη εικόνα της εφαρμογής

Deployment diagram