

# Ανάλυση των απαιτήσεων

## Η ανάλυση

- Αναλύουμε τις απαιτήσεις προκειμένου να τις βελτιώσουμε και να τις δομήσουμε καλύτερα προκειμένου να αναδείξουμε την αρχιτεκτονική του συστήματος
- Η ανάλυση βοηθά να ξεκαθαρίσουμε θέματα που μπορεί να έμειναν αδιευκρίνιστα κατά την καταγραφή των απαιτήσεων

## Γιατί μπορεί να έμειναν αδιευκρίνιστα θέματα

- Γιατί οι περιπτώσεις χρήσης ορίστηκαν ανεξάρτητα η μία από την άλλη
- Γιατί περιγράφηκαν στη γλώσσα του πελάτη
- Κάθε περίπτωση χρήσης δομήθηκε ώστε να αποτελεί μια ολοκληρωμένη και κατανοητή προδιαγραφή λειτουργίας

## Η ανάλυση

- Η ανάλυση επιλύει προβλήματα που προκύπτουν κοιτώντας τις απαιτήσεις σε μεγαλύτερο βάθος
- Η κύρια διαφορά της από την καταγραφή απαιτήσεων είναι ότι τα αποτελέσματα της ανάλυσης περιγράφονται σε τεχνική γλώσσα

## Ποια είναι η διαφορά της ανάλυσης από τη σχεδίαση και την υλοποίηση

- Η ανάλυση βρίσκεται ανάμεσα στην καταγραφή των απαιτήσεων και τη σχεδίαση
- Σκοπός της είναι να περιγράψει τις περιπτώσεις χρήσης σε ένα βαθμό λεπτομέρειας που ο πελάτης μπορεί να βαριόταν να το κάνει
- Στη σχεδίαση πρέπει να απαντηθούν ερωτήματα όπως
  - «πως μπορούμε να κάνουμε αυτή τη συνάρτηση πιο γρήγορη;»
  - «πως μπορούμε να μην υπερφορτώσουμε το δίκτυο;»

## Άλλες περιπτώσεις που η ανάλυση είναι αναγκαία

- Η ανάλυση μπορεί να αποτυπώσει, σχετικά φθηνά, ολόκληρο το σύστημα και να βοηθήσει στην επιλογή των κομματιών που θα σχεδιαστούν ή θα υλοποιηθούν
- Η ανάλυση δίνει μια επισκόπηση του συστήματος που μπορεί να είναι δύσκολο να εξαχθεί από τη σχεδίαση και την υλοποίηση. Νέοι μηχανικοί μπορεί να εκπαιδευτούν πιο εύκολα βλέποντας το μοντέλο ανάλυσης
- Το ίδιο σύστημα μπορεί να σχεδιαστεί διαφορετικά ή να υλοποιηθεί διαφορετικά. Το μοντέλο ανάλυσης παρέχει μια ανεξάρτητη εικόνα
- Ένα παλαιότερο σύστημα για το οποίο υπάρχει μοντέλο ανάλυσης είναι πιο εύκολο να κατανοηθεί και να επανα-σχεδιαστεί

## Σύγκριση του μοντέλου του Πεδίου Εφαρμογής και του Μοντέλου Ανάλυσης

- |  |   |
|--|---|
| <ul style="list-style-type: none"> <li>• Περιγράφεται στη γλώσσα του πελάτη</li> <li>• Εξωτερική άποψη του συστήματος</li> <li>• Δομείται από περιπτώσεις χρήσης - δομεί την εξωτερική άποψη</li> <li>• Χρησιμοποιείται ως «συμβόλαιο» ανάμεσα στους πελάτες και τους κατασκευαστές</li> <li>• Μπορεί να περιέχει πλεονασμούς και ασάφειες</li> <li>• Διατυπώνει τη λειτουργικότητα του συστήματος, περιλαμβανόμενης και της αρχιτεκτονικής που είναι σημαντική</li> <li>• Ορίζει περιπτώσεις χρήσης που αναλύονται παραπάνω στο μοντέλο ανάλυσης</li> </ul> | <ul style="list-style-type: none"> <li>• Περιγράφεται στη γλώσσα του κατασκευαστή</li> <li>• Εσωτερική άποψη του συστήματος</li> <li>• Δομείται από τάξεις και πακέτα - δομεί την εσωτερική άποψη</li> <li>• Χρησιμοποιείται από τους κατασκευαστές για να καταλάβουν πως θα οργανωθεί το σύστημα</li> <li>• Δεν πρέπει να περιέχει πλεονασμούς και ασάφειες</li> <li>• Σκιαγραφεί πως θα υλοποιηθεί η λειτουργικότητα και είναι μια πρώτη προσέγγιση στη σχεδίαση</li> <li>• Ορίζει πραγματώσεις - αναλύσεις περιπτώσεων χρήσης σε αντιστοιχία 1-1 με τις περιπτώσεις χρήσης του πεδίου εφαρμογής</li> </ul> |
|--|---|

## Ο ρόλος της ανάλυσης στη διάρκεια της ζωής ενός προϊόντος

- Στην ανάλυση εστιάζουμε κατά τις αρχικές φάσεις
- Ο σκοπός και ο στόχος της φάσης της ανάλυσης πρέπει λίγο πολύ να επιτευχθεί σε κάθε έργο. Ο τρόπος όμως που θα το πετύχουμε μπορεί να διαφέρει

# Πως μπορούμε να κάνουμε ανάλυση;

- Να υλοποιηθεί ένα μοντέλο ανάλυσης το οποίο και να συντηρείται μέχρι το τέλος του έργου
- Να υλοποιηθεί ένα μοντέλο ανάλυσης ως ενδιάμεσο βήμα ανάμεσα στις απαιτήσεις και τη σχεδίαση. Μόλις αρχίσουμε τη σχεδίαση σταματά να μας απασχολεί το μοντέλο ανάλυσης
- Δεν φτιάχνουμε μοντέλο ανάλυσης. Η ανάλυση των απαιτήσεων γίνεται είτε κάνοντας καλύτερη καταγραφή των απαιτήσεων είτε θεωρώντας ότι το πρόβλημα που λύνουμε είναι απλό.

# Τι να κάνετε στη διάρκεια της ανάλυσης

- Αναλύστε τις περιπτώσεις χρήσης
- Βρείτε τις τάξεις ανάλυσης (ορίου, ελέγχου και οντότητας)
- Βρείτε τις σχέσεις μεταξύ τους
- Βρείτε τα χαρακτηριστικά κάθε τάξης
- Βρείτε πιθανές σχέσεις περιεκτικότητας και γενικεύσεις

# Βήμα 1

## Αναλύστε τις περιπτώσεις χρήσης

- Για την ανάλυση των περιπτώσεων χρήσης ξεκινήστε προσπαθώντας να κάνετε ένα διάγραμμα συνεργασίας.
- Στο διάγραμμα συνεργασίας χρησιμοποιήστε 3 γενικές κατηγορίες κλάσεων, οι οποίες ονομάζονται «κλάσεις ανάλυσης» (θα τις δούμε παρακάτω)
- Στο διάγραμμα συνεργασίας δείξτε πως θα μπορούσε να πραγματοποιηθεί η περίπτωση χρήσης αν συνεργαζόταν μεταξύ τους «αντικείμενα» αυτών των κλάσεων

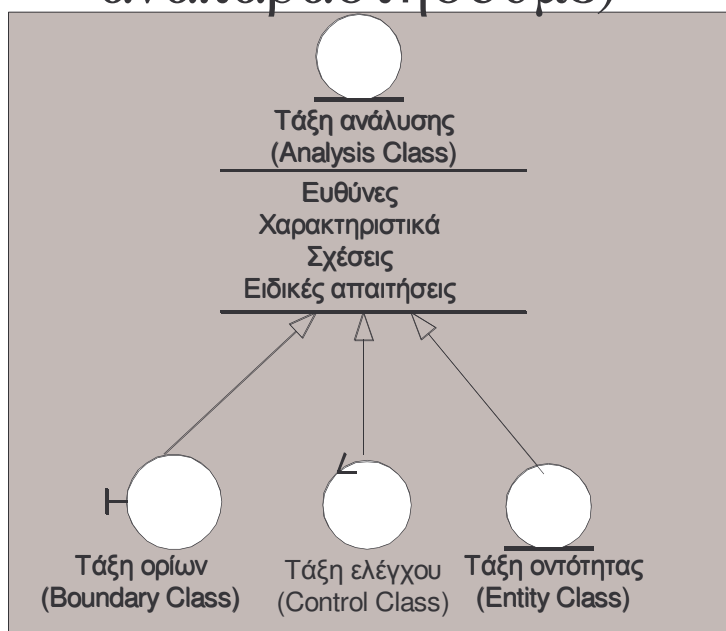
## Κλάση ανάλυσης

- Αποτελεί μια γενικευμένη κλάση η οποία κατά τη φάση της σχεδίασης θα αντιστοιχηθεί σε περισσότερες από μια τάξεις
- Επικεντρώνεται στις λειτουργικές απαιτήσεις και αναβάλλει το χειρισμό των μη λειτουργικών απαιτήσεων στη φάση σχεδιασμού και υλοποίησης
- Δεν περιέχει μεθόδους. Το πως συμπεριφέρεται ορίζεται ουσιαστικά από τις υποχρεώσεις της
- Έχει χαρακτηριστικά (τα οποία αργότερα μπορεί να γίνουν κλάσεις)
- Σχετίζεται με άλλες κλάσεις

# Για να βρείτε τις κλάσεις ανάλυσης

- Δείτε κάθε περίπτωση χρήσης
- Φανταστείτε ότι έχετε μόνο τρία είδη κλάσεων στη διάθεσή σας
  - Κλάσεις που είναι υπεύθυνες για να στέλνουν δεδομένα σε χαρακτήρες ή να παίρνουν δεδομένα από αυτούς. Αυτές ονομάζονται κλάσεις ορίου
  - Κλάσεις που είναι υπεύθυνες για να αποθηκεύουν δεδομένα. Αυτές ονομάζονται κλάσεις οντότητας ή κλάσεις δεδομένων
  - Και κλάσεις που κάνουν ελέγχους, ορίζουν με ποια σειρά πρέπει να γίνουν διάφορες ενέργειες και «συνδέουν» τις ενέργειες των κλάσεων ορίου με τις κλάσεις οντότητας. Αυτές ονομάζονται κλάσεις ελέγχου

## Στερεότυπα κλάσεων ανάλυσης (πώς μπορούμε να τις αναπαραστήσουμε)



# Στερεότυπα

- Εναλλακτικά μπορείτε να τις αναπαραστήσετε ως εξής:

<code>&lt;&lt;boundary&gt;&gt;</code> Name
---

<code>&lt;&lt;control&gt;&gt;</code> Name
--

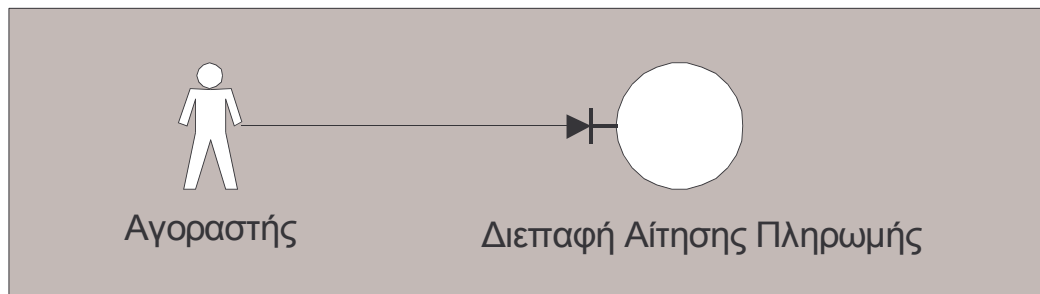
<code>&lt;&lt;entity&gt;&gt;</code> Name
---

## Τάξεις ορίου (Boundary Classes)

- Χρησιμοποιούνται για να περιγράψουν τη διασύνδεση μεταξύ του συστήματος και των χαρακτήρων
- Αυτή η διασύνδεση περιλαμβάνει συνήθως λήψη και παρουσίαση πληροφορίας και αιτήσεις από και προς το σύστημα
- Είναι συνήθως γενικεύσεις τερματικών, εκτυπωτών, παραθύρων κτλ
- Οι τάξεις αυτές περιγράφουν το ΤΙ επιτυγχάνεται από τη διασύνδεση και όχι το ΠΩΣ
- Κάθε τέτοια τάξη συνδέεται με τουλάχιστον ένα χαρακτήρα



# Παράδειγμα



- Η διεπαφή επιτρέπει στο χρήστη να βλέπει τιμολόγια με μεγαλύτερη λεπτομέρεια και στη συνέχεια να ζητά από το σύστημα να πληρώνει

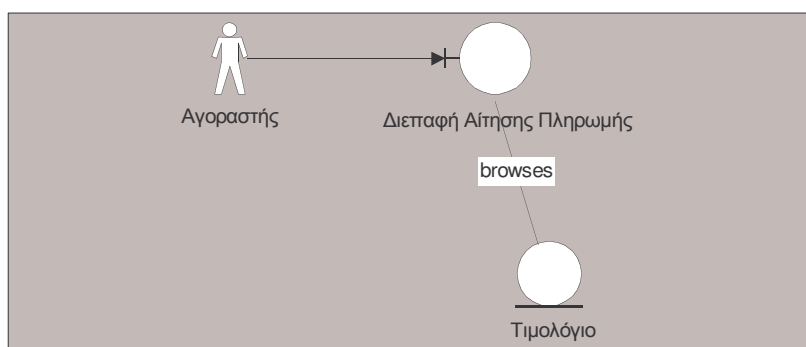
## Χαρακτηριστικά (attributes) τάξεων ορίου

- Οι τάξεις ορίου που επικοινωνούν με ανθρώπους έχουν συνήθως χαρακτηριστικά που αφορούν πληροφορίες (πχ text fields, labels etc)
- Οι τάξεις ορίου που επικοινωνούν με άλλα συστήματα έχουν χαρακτηριστικά κάποιου πρωτοκόλλου επικοινωνίας

# Τάξεις οντότητας (Entity Class)

- Αναπαραστούν πληροφορία η οποία διατηρείται για κάποιο χρονικό διάστημα
- Στις περισσότερες περιπτώσεις προέρχονται κατευθείαν από το πεδίο εφαρμογής (domain model)

## Παράδειγμα



- Η τάξη οντότητας «Τιμολόγιο» σχετίζεται με τη «Διεπαφή Αίτησης Πληρωμής» με την οποία ο χρήστης βλέπει τιμολόγια

## Χαρακτηριστικά τάξεων οντοτήτων

- Τα χαρακτηριστικά μιας τάξης οντοτήτων είναι συνήθως αρκετά φανερά
- Συνήθως μπορείτε να τα βρείτε από τα χαρακτηριστικά των εννοιών που έχετε αναπαραστήσει στο domain model

## Ανακαλύπτοντας τάξεις οντοτήτων

- Αρχικά μπορούν να σκιαγραφηθούν οι βασικές τάξεις οντοτήτων από το domain model.
- Οι σχέσεις που υπάρχουν στο domain model μπορούν να χρησιμοποιηθούν για να αποκαλύπτουν και τις σχέσεις ανάμεσα στις τάξεις οντοτήτων

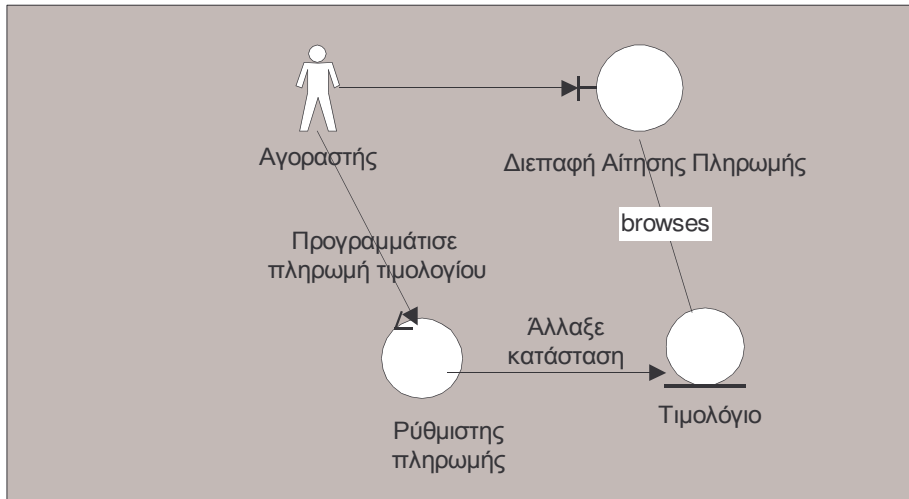
## Παράδειγμα

- Η τάξη «Τιμολόγιο» μπορεί να προέλθει από την αντίστοιχη τάξη στο domain model.
- Τα αρχικά χαρακτηριστικά μπορεί να είναι τα ίδια που είχαμε ορίσει και τότε (ποσό, ημ. Πληρωμής κτλ)
- Ανάλογα μπορούμε να ορίσουμε και τις σχέσεις με την τάξη παραγγελία όπως είχε οριστεί και στο domain model

## Τάξεις ελέγχου

- Αναπαριστούν συνεργασίες, μεταφορές, έλεγχο άλλων αντικειμένων.  
Αναπαραστούν υπολογισμούς
- Σε γενικές γραμμές η «δυναμική» του συστήματος αναπαραστάται από τις τάξεις ελέγχου

# Παράδειγμα



- Ο ρυθμιστής πληρωμής δέχεται μια αίτηση πληρωμής για ένα τιμολόγιο για μια συγκεκριμένη ημέρα, και όταν έρθει η μέρα κάνει την πληρωμή και αλλάζει την κατάσταση στο τιμολόγιο

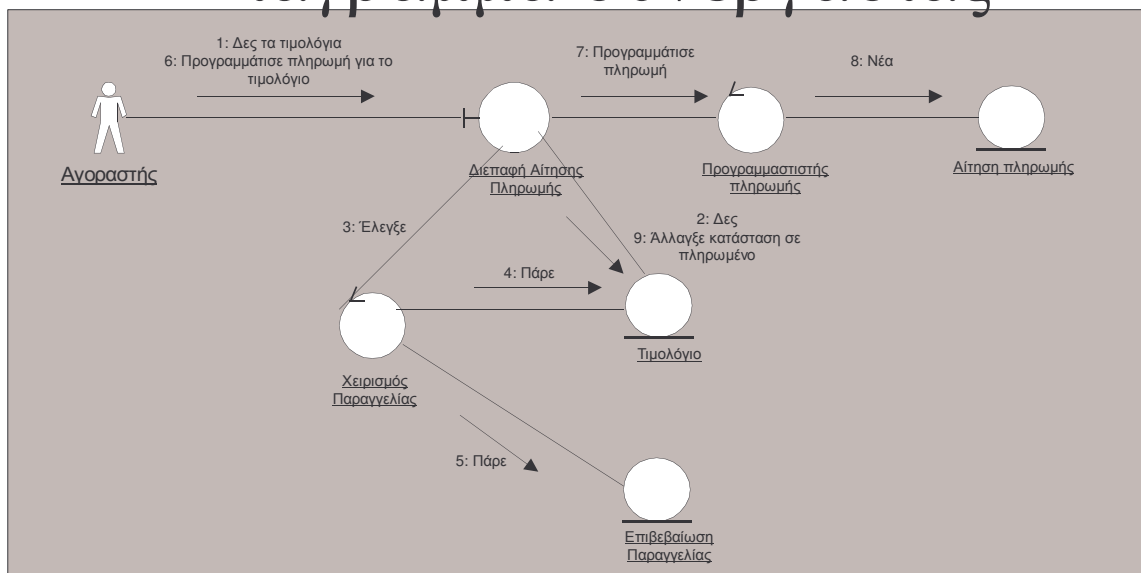
## Χαρακτηριστικά τάξεων ελέγχου

- Συνήθως οι τάξεις ελέγχου δεν έχουν κάποια χαρακτηριστικά γιατί συνήθως έχουν και σύντομη διάρκεια ζωής.
- Πολλές φορές όμως μπορεί να έχουν χαρακτηριστικά που συσσωρεύονται, πχ μετρητές, που χρησιμοποιούνται σε μια περίπτωση χρήσης

## Βήμα 2

- Κάντε ένα διάγραμμα συνεργασίας για κάθε περίπτωση χρήσης.
- Στο διάγραμμα συνεργασίας δείξτε με ποιο τρόπο κλάσεις ορίου, ελέγχου και οντότητας μπορούν να συνεργαστούν για να πραγματοποιήσουν την περίπτωση χρήσης

### Διάγραμμα συνεργασίας



- ... ακολουθεί και μια περιγραφή

## Βήμα 3

- Προσθέστε, αν θεωρήσετε αναγκαίο, και μια περιγραφή

## Ροή γεγονότων που επεξηγεί το διάγραμμα συνεργασίας

- *Ο αγοραστής βλέπει τα τιμολόγια προκειμένου να βρεί αυτά που θέλει (1,2) διαμέσου της διεπαφής αίτησης πληρωμής. Η διεπαφή χρησιμοποιεί τον Χειρισμό Παραγγελίας για να δει αν τα τιμολόγια αναφέρονται σε επιβεβαιωμένες παραγγελίας (3,4,5) προτού τα εμφανίσει στον πελάτη. Πως γίνεται αυτός ο έλεγχος εξαρτάται από τους κανόνες του οργανισμού και μπορεί να περιλαμβάνουν ημ. Παράδοσης, τιμές κτλ. ...*

# Πως διαφέρει από την περίπτωση χρήσης;

Η περίπτωση χρήσης περιγράφει το σύστημα «από έξω»  
Η πραγμάτωση βλέπει το σύστημα «από μέσα»

## Σύντομη περιγραφή

Η περίπτωση χρήσης «Πλήρωσε το τιμολόγιο» χρησιμοποιείται από τον Αγοραστή για να προγραμματίσει τις πληρωμές. Στη συνέχεια η περίπτωση χρήσης ενεργοποιεί την πληρωμή τη συγκεκριμένη μέρα

## Αρχική περιγραφή βήμα-βήμα

Πριν αρχίσει αυτή η περίπτωση χρήσης ο Αγοραστής έχει ήδη πάρει ένα τιμολόγιο (περιγραφή στην περίπτωση χρήσης «Τιμολόγησε τον Αγοραστή») και επίσης έχει πάρει τα αγαθά που παρήγγειλε.

1. Ο Αγοραστής εξετάζει ότι το τιμολόγιο είναι σωστό
2. Ο Αγοραστής προγραμματίζει την πληρωμή
3. Τη συγκεκριμένη μέρα το σύστημα ελέγχει αν υπάρχουν αρκετά χρήματα στο λογαριασμό, και αν υπάρχουν υλοποιεί τη συναλλαγή

## Βήμα 4

- Αφού ολοκληρώσετε τα διαγράμματα συνεργασίας για κάθε περίπτωση χρήσης δείτε από τις κλάσεις που έχετε ορίσει αν μπορείτε να βρείτε σχέσεις γενίκευσης ή εξάρτησης. Δείτε αν μπορείτε να επαναχρησιμοποιήσετε κάποια κλάση.
- Ίσως σας βοηθήσει για κάθε κλάση να φτιάξετε μια καρτέλα στην οποία φαίνεται το όνομα της κλάσης, τι κάνει και με ποιους συνεργάζεται



# Παράδειγμα

<b>Class</b>	ApplyForm	
<b>Superclass(es)</b>		
<b>Subclasses</b>		
	<b>Ευθύνη</b>	<b>Συνεργάτες</b>
	<p>Ο χρήστης μπορεί να δώσει OK, να επιλέξει προφιλ και όνομα χώρου</p> <p>Εμφάνισε όλα τα προφιλ για να επιλέξει ο χρήστης</p> <p>Μόλις ο χρήστης πατήσει OK στέλνονται το όνομα χώρου και το επιλεγμένο προφιλ</p>	<p>User</p> <p>Profile</p> <p>ProcessApply</p>

# Κάρτες CRC

- Οι κάρτες αυτές είναι γνωστές ως CRC από τα αρχικά
  - Class
  - Responsibility
  - Collaboration

# Βρίσκοντας τις σχέσεις ανάμεσα στις τάξεις

- Χρησιμοποιήστε διαγράμματα συνεργασίας
- Αν σε μια περίπτωση χρήσης υπάρχουν πάνω από μια ροές μπορείτε να φτιάξετε ένα διάγραμμα συνεργασίας για κάθε ροή
- Αρχίστε κοιτώντας τη ροή των γεγονότων ένα βήμα τη φορά αποφασίζοντας ποιο αντικείμενο ανάλυσης θα χρησιμοποιήσετε

## Περισσότερα...

- Η περίπτωση χρήσης ξεκινά με ένα μήνυμα από ένα χαρακτήρα
- Κάθε τάξη ανάλυσης που ορίστηκε προηγουμένως πρέπει να έχει τουλάχιστον ένα αντικείμενο σε μια περίπτωση χρήσης
- Τα μηνύματα δεν έχουν σχέση με τις μεθόδους μιας τάξης απλά σημειώνουν το σκοπό που καλούμε το αντικείμενο.
- Οι σύνδεσμοι στο διάγραμμα είναι οι σχέσεις ανάμεσα στις τάξεις
- Η ακολουθία στο διάγραμμα δεν είναι ο βασικός στόχος

# Βρίσκοντας την ευθύνη μιας τάξης

- Οι ευθύνες μιας τάξης μπορούν να καταγραφούν αν συνδυαστούν όλοι οι ρόλοι που έχει αναλάβει η τάξη στις διάφορες πραγματώσεις-αναλύσεις μιας περίπτωσης χρήσης.

# Βρίσκοντας τα χαρακτηριστικά (attributes) μιας τάξης

- Ένα χαρακτηριστικό ορίζει μια ιδιότητα μιας τάξης ανάλυσης και συχνά εννοείται από το ρόλο που έχει η κλάση
- Συνήθως τα χαρακτηριστικά μιας τάξης είναι ουσιαστικά (π.χ. Τηλέφωνο, ημερομηνία)
- Φροντίστε αυτά να ορίζονται γενικά και όχι με βάση κάποιο συγκεκριμένο τύπο γλώσσας προγραμματισμού
- Αν κάποιο χαρακτηριστικό πρέπει να είναι το ίδιο για πολλά αντικείμενα ανάλυσης, ορίστε το σε χωριστή τάξη
- Αν μια τάξη έχει πολλά χαρακτηριστικά, σκεφθείτε να τα ορίσετε σε χωριστή τάξη

# Μη-λειτουργικές απαιτήσεις

- Σε ένα κείμενο μπορείτε να ορίσετε και τις μη-λειτουργικές απαιτήσεις που αφορούν την πραγματοποίηση μιας περίπτωσης χρήσης. Παράδειγμα
  - *Δεν πρέπει να περνάν πάνω από 5 δευτερόλεπτα από τη στιγμή που θα ζητήσει ένας αγοραστής να δει τα τιμολόγια μέχρι τη στιγμή που θα εμφανιστούν*

# Βρείτε τις μη-λειτουργικές απαιτήσεις

- Οι μη-λειτουργικές απαιτήσεις είναι ειδικές απαιτήσεις που αποκαλύπτονται στη φάση της ανάλυσης (ή και πιο μπροστά)
- Παραδείγματα ειδικών απαιτήσεων είναι
  - ασφάλεια
  - Ανοχή σε λάθη
  - Χειρισμό συναλλαγών κτλ
- Υπεύθυνος για να τις ορίσει είναι ο αρχιτέκτονας ώστε να τις γνωρίζουν και οι υπόλοιποι

# Βιβλιογραφία

- Οι σημερινές σημειώσεις προέρχονται από
- Jacobson, Rumbaugh, Booch, *The Unified Software Development Process*, Addison Wesley