Διάλεξη 8-9:
Αντικειμενοστρεφής Σχεδιασμός Ι

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:

- Διαδικασίες Ανάπτυξης Λογισμικού, Μοντέλα
- Παράδειγμα Διαδικασίας Ανάπτυξης Λογισμικού: Βιβλίο Διευθύνσεων
- Φάση 1: Ανάλυση Απαιτήσεων & Προδιαγραφές (Αναφορά Απαιτήσεων, Περιπτώσεις Χρήσης, Ανάλυση Κλάσεων, Analysis Class Diagrams)

Διδάσκων: Παναγιώτης Ανδρέου

Διαδικασία Ανάπτυξης Λογισμικού

- **Διαδικασία Ανάπτυξης Λογισμικού (software process)**
  Ένα δομημένο σύνολο **(framework)** από ενέργειες που χρειάζονται για να αναπτυχθεί ένα λογισμικό
- Υπάρχουν **πολλές διαδικασίες ανάπτυξης λογισμικού (software models/frameworks)** με πολλές ιδιαιτερότητες
- **Μοντέλο Διαδικασίας Ανάπτυξης Λογισμικού (software process model)**
  Είναι μία αφηρημένη/απλοποιημένη αναπαράσταση της διαδικασίας ανάπτυξης λογισμικού.
  - Κάθε στάδιο της διαδικασίας ανάπτυξης λογισμικού αναγνωρίζεται
  - Το **μοντέλο (process model) αναπαριστά τις δραστηριότητες (activities)** που σχετίζονται άμεσα με το κάθε στάδιο
  - Η συλλογή των μοντέλων μπορεί να παράξει την γενική εικόνα που αναπαριστά την διαδικασία ανάπτυξης λογισμικού.

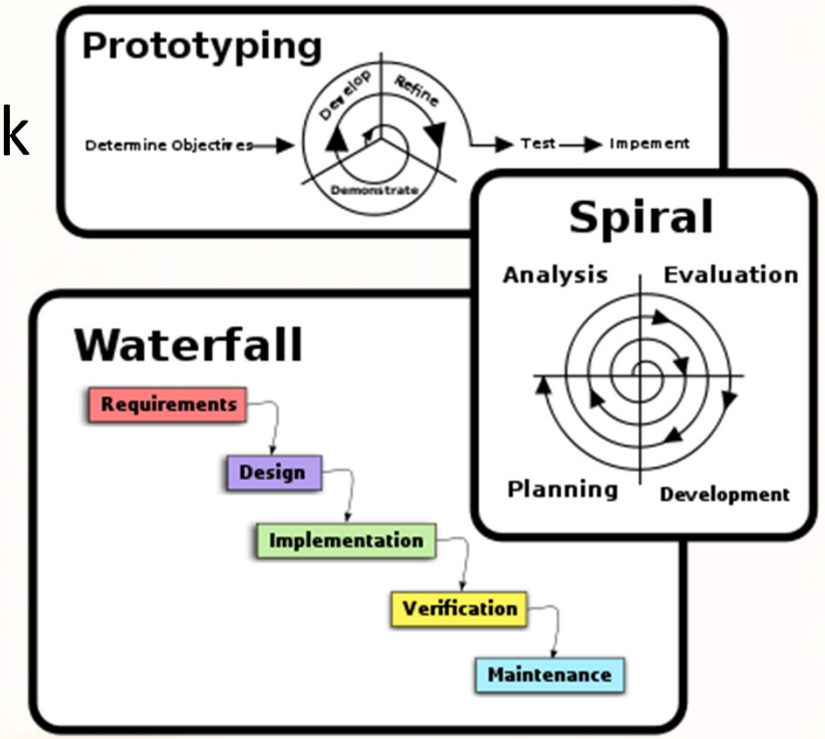
Δραστηριότητες των Μοντέλων Διαδικασίας Ανάπτυξης Λογισμικού

Βασικές Δραστηριότητες των μοντέλων συμπεριλαμβάνουν:

- **Απαιτήσεις:** Συλλογή απαιτήσεων σε διάφορα επίπεδα (επίπεδο οργανισμού, στρατηγικό επίπεδο)
- **Ανάλυση:** κατανόηση της φύσης του λογισμικού που θα αναπτυχθεί, το πεδίο πληροφόρησης, απαραίτητες λειτουργίες, συμπεριφορά, απόδοση και διαπροσωπεία
- **Σχεδίαση:** Αναγνώριση/Σχεδίαση δομών δεδομένων, αλγόριθμων, αρχιτεκτονικής προγράμματος και διαπροσωπείας
- **Υλοποίηση:** μετάφραση των απαιτήσεων και της ανάλυσης σε εκτελέσιμο κώδικα
- **Συντήρηση:** τροποποίηση του λογισμικού μετά την παράδοση του στον πελάτη.

Γενικά Μοντέλα Διαδικασίας Ανάπτυξης Λογισμικού

- **Waterfall:** Γραμμικό framework
- **Prototyping:** Επαναληπτικό framework
- **Αυξητικό:** Συνδυασμός γραμμικού/ επαναληπτικού framework
- **Spiral:** Συνδυασμός γραμμικού/ επαναληπτικού framework, έμφαση στην εκτίμηση του κινδύνου (risk assessment)

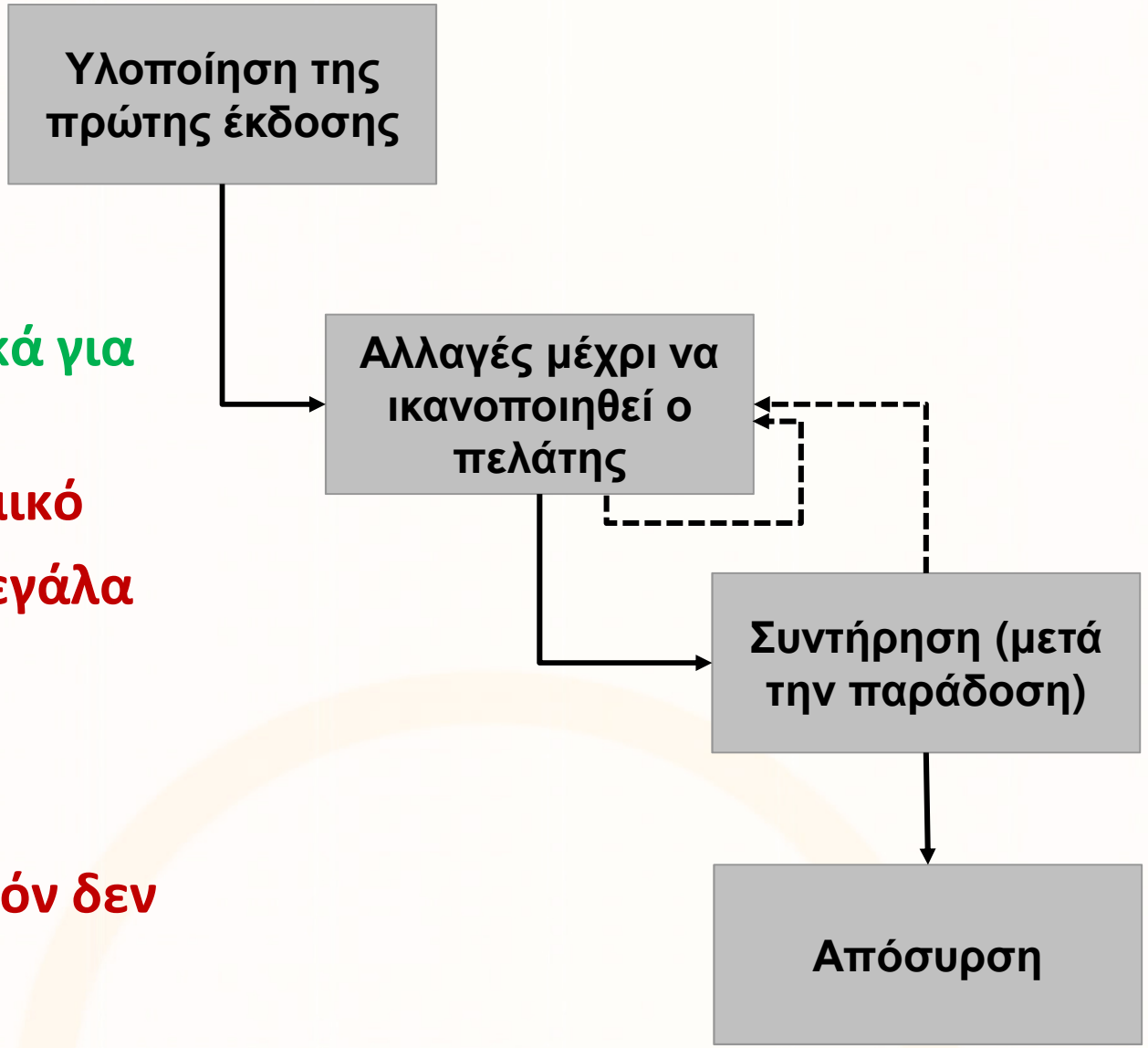


- **Rapid Application Development (RAD):** Επαναληπτικό framework, έμφαση στη γρήγορη ανάπτυξη
- **Extreme Programming:** καινούριες εκδόσεις δημιουργούνται πολύ συχνά, μικροί κύκλοι ανάπτυξης (time-boxing), σχεδιασμένο για συχνές αλλαγές απαιτήσεων

Build & Fix

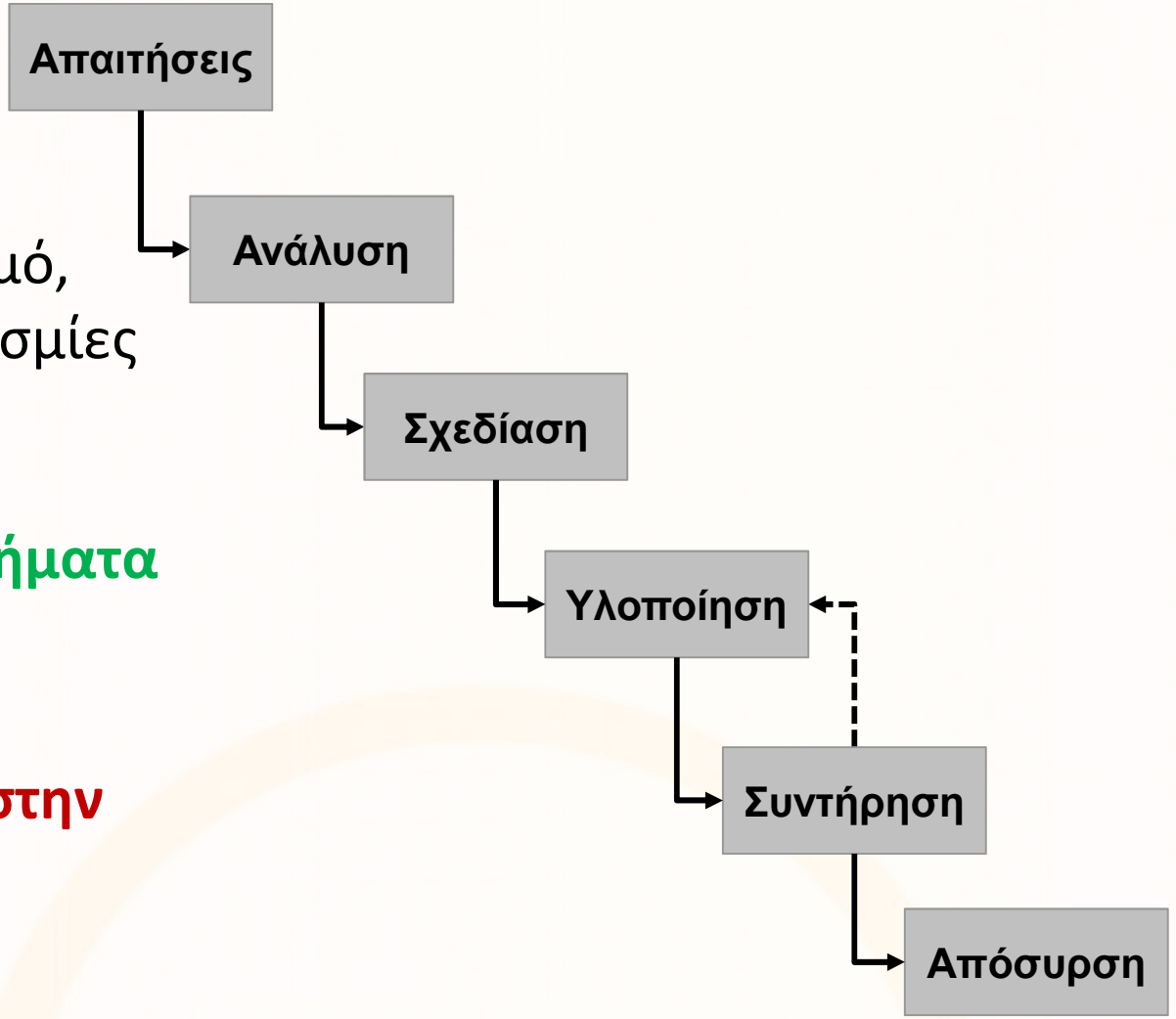
- Καθόλου απαιτήσεις!
- Καθόλου Σχεδίαση!
- Καθόλου Προδιαγραφές!

+ **Πολύ αποδοτικό οικονομικά για μικρά προγράμματα**
- **Χαμηλής ποιότητας λογισμικό**
- **Καθόλου αποδοτικό για μεγάλα προγράμματα**
- **Υψηλό κόστος για μεγάλα προγράμματα**
- **Ο χρόνος παράδοσης σχεδόν δεν τηρείται ποτέ**
- **Εφιάλτης Συντήρησης**



Waterfall Model

- Γραμμικό μοντέλο
- **Διαδοχικές φάσεις** με κάποια μικρή επικάλυψη
- Έμφαση στον προγραμματισμό, προϋπολογισμούς και προθεσμίες



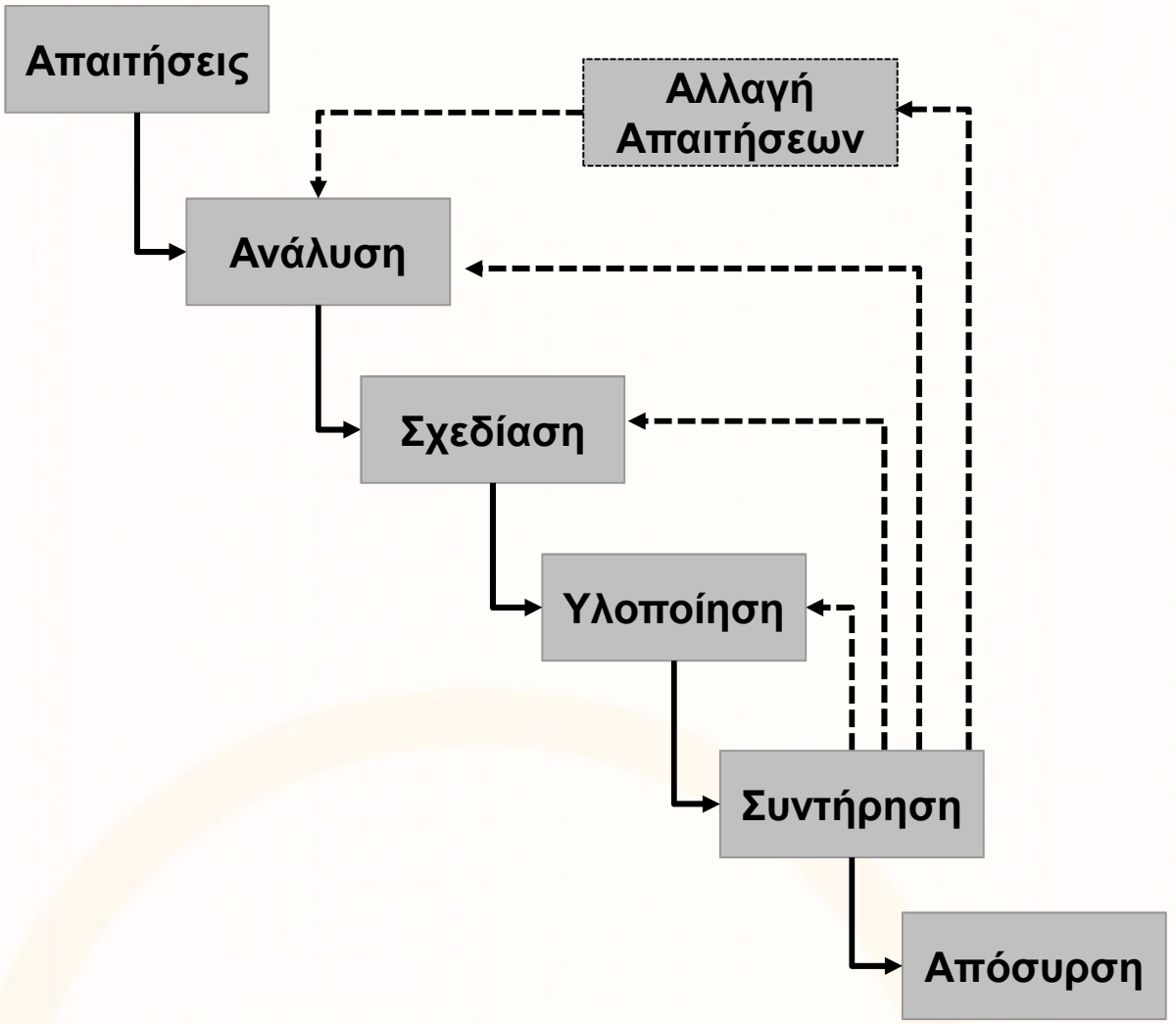
+ **Διακριτές φάσεις**
+ **Αποδοτικό για μεγάλα συστήματα με ξεχωριστά αυτόνομα components**
- **Δύσκολο να προσαρμοστεί στην αλλαγή**
- **Όχι πολύ ευέλικτο**
- **Συνήθως βγαίνεις εκτός προϋπολογισμού**

Waterfall Model + feedback loops

- Ανατροφοδότηση από τη συντήρηση σε προηγούμενες φάσεις

\+ Πιο ευέλικτο από το απλό waterfall model

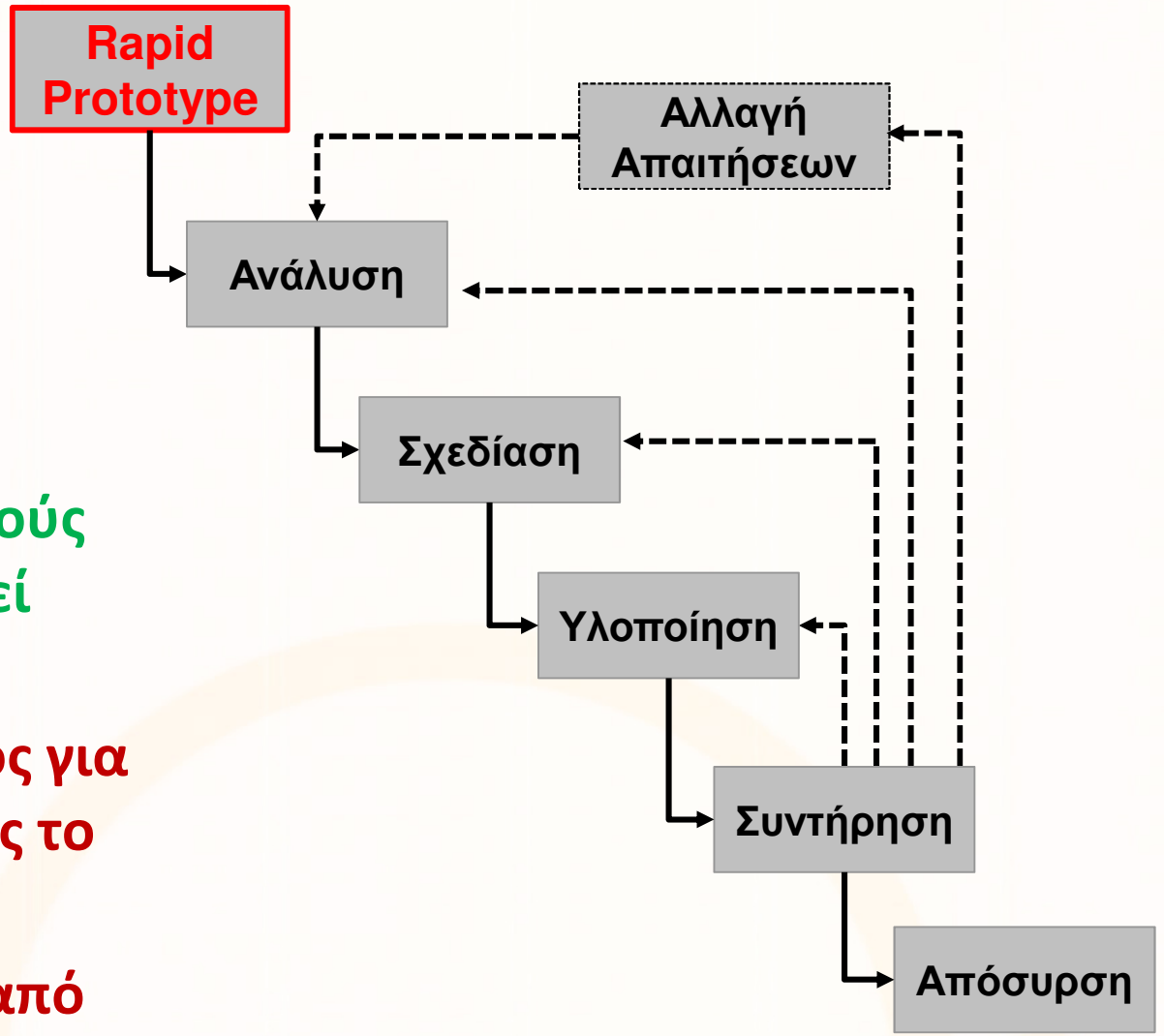
\- Όχι κατάλληλο για μικρά προγράμματα



Rapid Prototyping

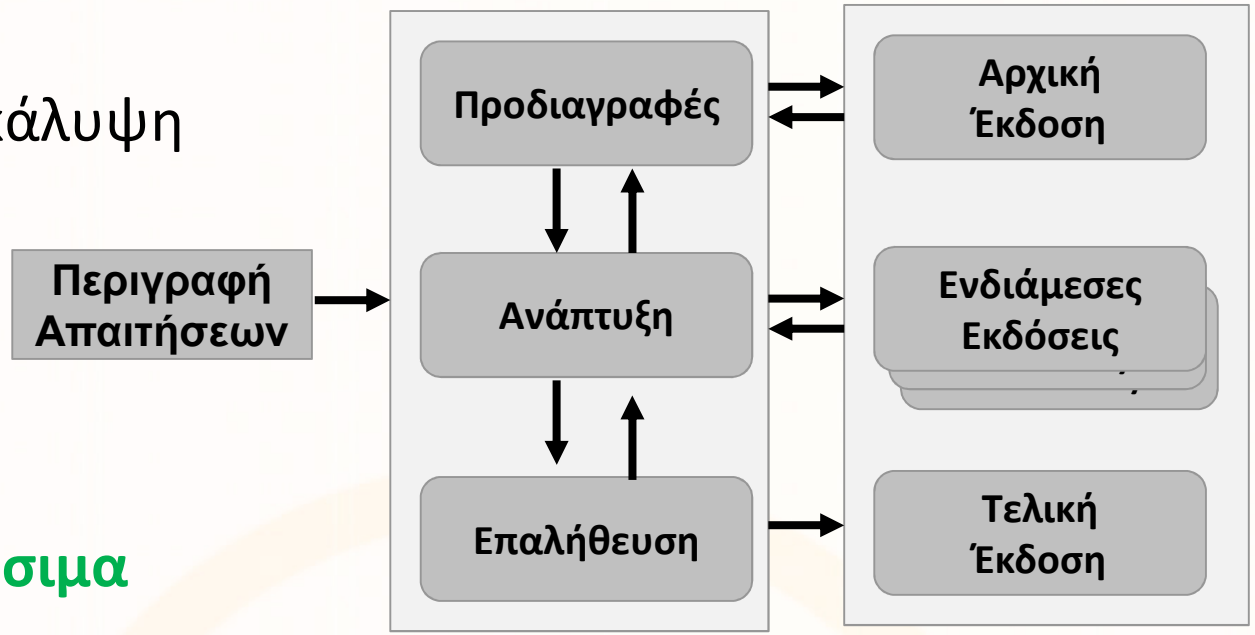
- **Βασισμένο σε Πρωτότυπα:** ατελείς εκδόσεις του λογισμικού
- Διατίθενται από τα αρχικά στάδια του προγράμματος

+ **Οι χρήστες πειραματίζονται γρήγορα με το σύστημα ➔ πιο ποιοτικές απαιτήσεις**
+ **Πιο αποδεκτό από τους τελικούς χρήστες αφού έχουν εμπλακεί στην υλοποίηση**
- **Συνήθως γίνεται συμβιβασμός για να εξελιχθεί το πρωτότυπο ως το τελικό προϊόν πιο γρήγορα**
- **Χρειάζεται αρκετή εμπειρία από τους προγραμματιστές**



Incremental Development

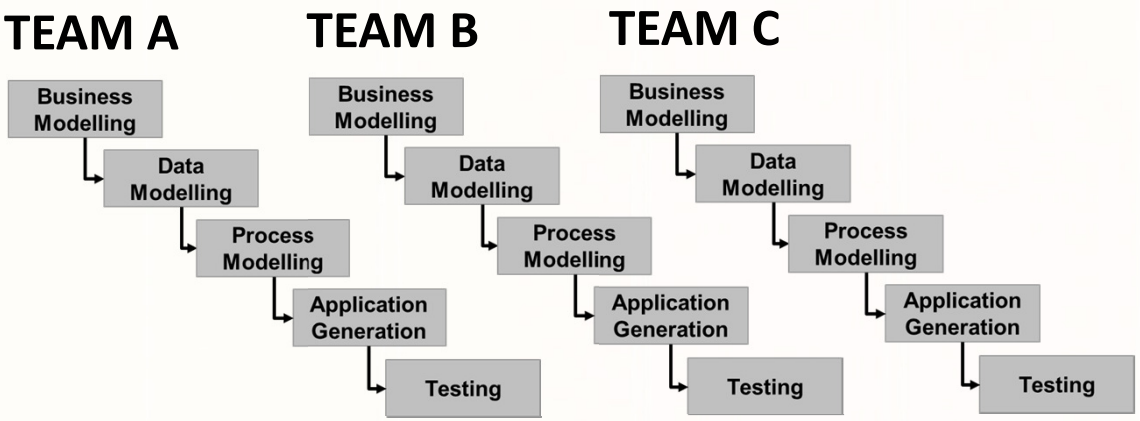
- **Συνεχής αναπληροφόρηση**
- **Συνεχής βελτίωση**
- Το πρόγραμμα διασπάται σε μικρότερες εκδόσεις
- Κάθε φάση έχει μερική επικάλυψη με τις διπλανές της



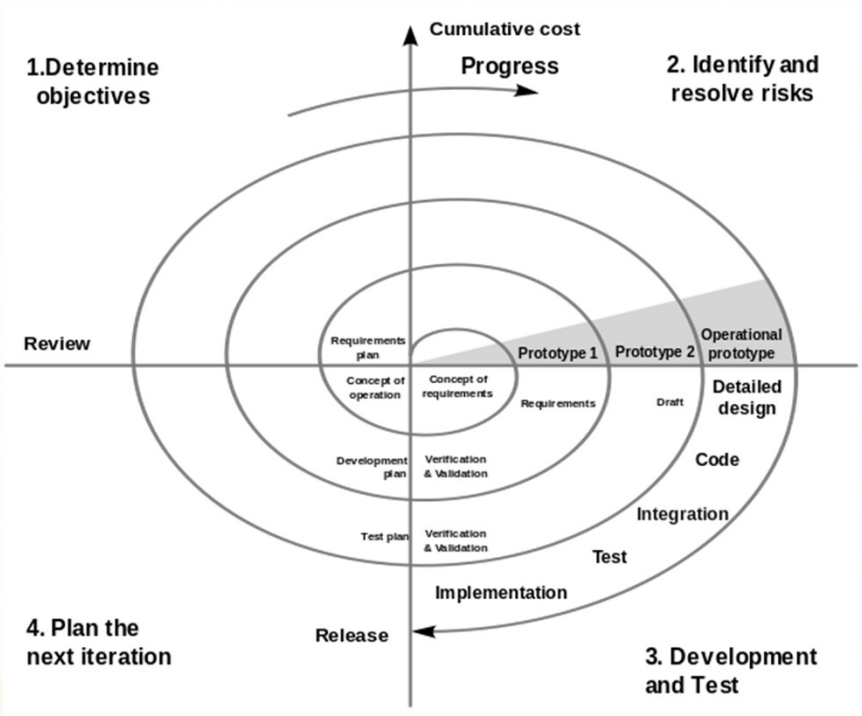
+ **Μειώνει το ρίσκο**
+ **Πολύ ευέλικτο**
+ **Μικρά, επαναχρησιμοποιήσιμα κομμάτια κώδικα**
- **Κατάλληλο για Μικρά ως Μεσαία προγράμματα**

Άλλα γνωστά μοντέλα

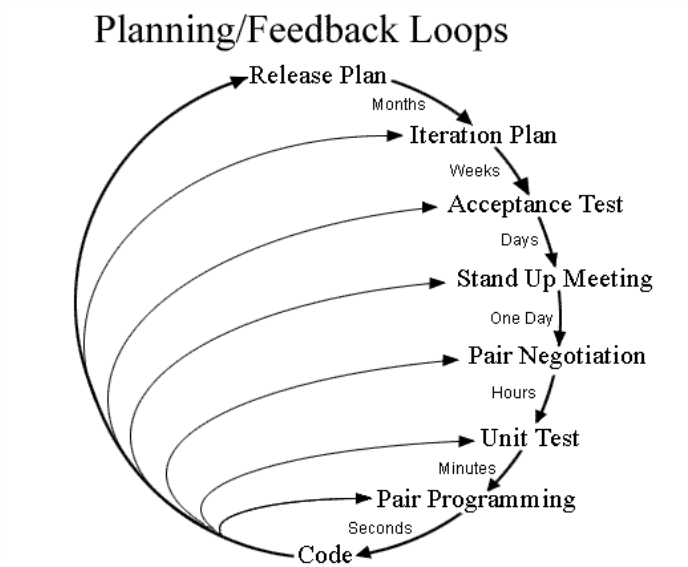
Rapid Application Development RAD



Spiral Development Model



Extreme Programming

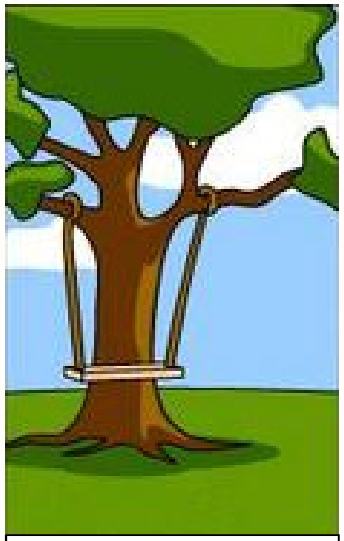


ΥΒΡΙΔΙΚΑ

...και πολλά άλλα!

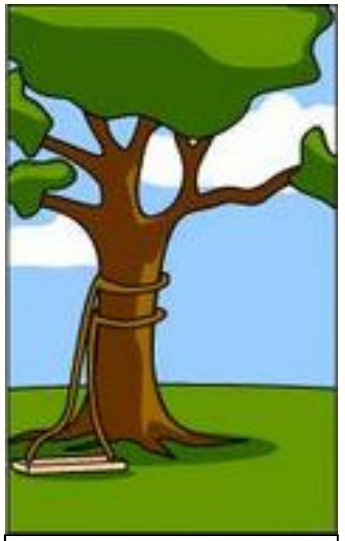
Το Συμπέρασμα!

How the customer explained it



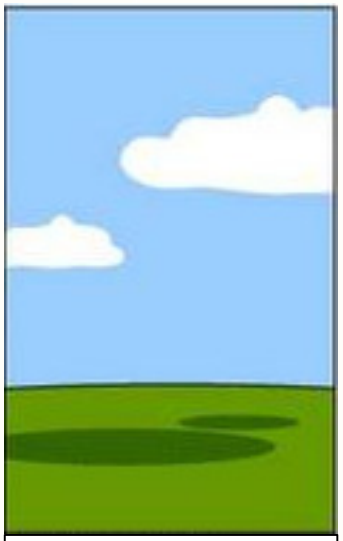
How the project leader understood it

How the analyst designed it

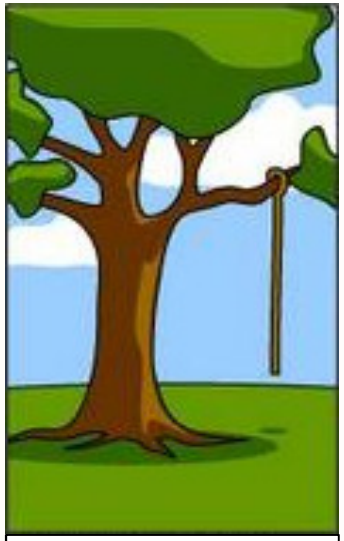


How the programmer coded it

How the business consultant described it

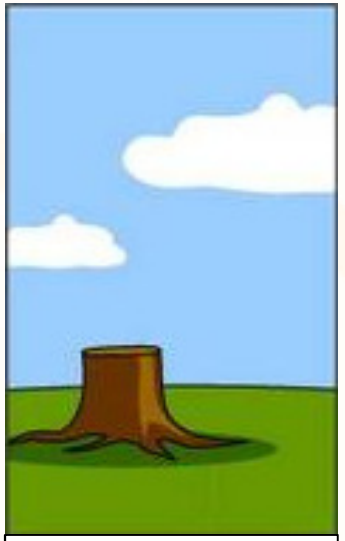


How the project was documented



What operations were installed

What the customer was billed

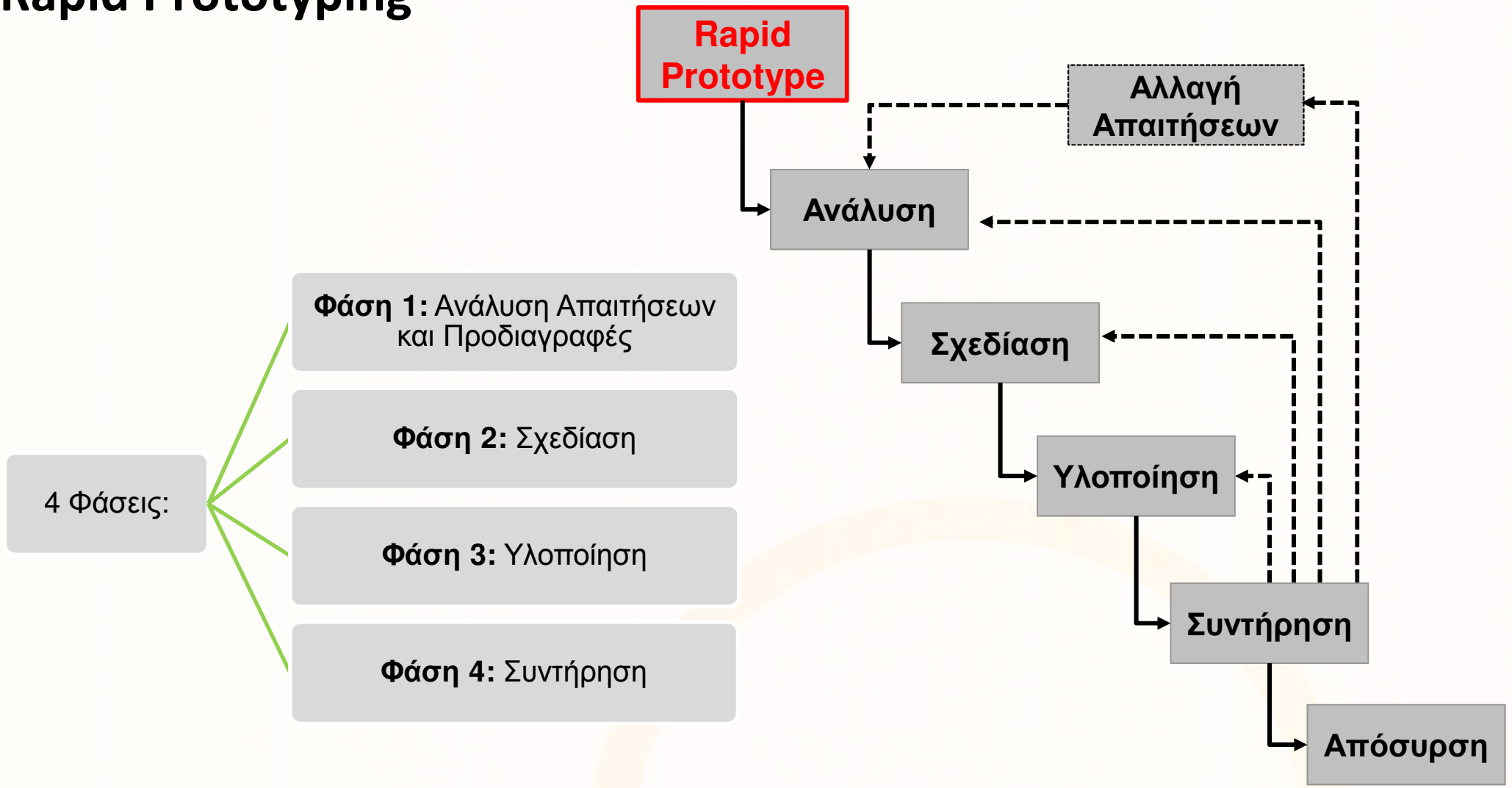


How it was supported

What the customer really needed

Τι θα χρησιμοποιήσουμε;

Rapid Prototyping



Φάση 1: Ανάλυση Απαιτήσεων και Προδιαγραφές

1Α. Ανάλυση Απαιτήσεων (Requirements Analysis)

- Περιγραφή των αναγκών του πελάτη **(requirements statement)**
- Αναγνώριση ενδιαφερομένων
- Συνεντεύξεις χρηστών
- Ανάπτυξη περιπτώσεων χρήσης **(use cases)**. Αποκαλύπτουν βασικά χαρακτηριστικά του προγράμματος. Αποτελούν περιγραφικές απαντήσεις σε ερωτήσεις όπως:
  - Ποιός θα χρησιμοποιήσει το σύστημα;
  - Τι ενέργειες μπορούν να κάνουν οι χρήστες;
  - Πώς αλλιώς θα μπορούσε αυτό να λειτουργήσει αν κάποιος άλλος έκανε την ενέργεια ή ο ίδιος χρήστης είχε διαφορετικό στόχο;
  - Τι προβλήματα μπορεί να προκύψουν με το σύστημα;
- Ανάπτυξη Γρήγορων Πρωτοτύπων **(rapid prototyping)**

Φάση 1: Ανάλυση Απαιτήσεων και Προδιαγραφές (συν.)

1Β. Προδιαγραφές Συστήματος (System Specification)

- Προκαταρκτική αναγνώριση κλάσεων **(Analysis Class Diagram)** Συγκεκριμένη περιγραφή του τι θα πρέπει κάνει το πρόγραμμα (όχι πως θα το κάνει). Παράδειγμα του τι μπορεί να περιλαμβάνουν οι προδιαγραφές:

- Overall description
  - Product perspective
    - System Interfaces
    - User Interfaces
    - Hardware interfaces
    - Software interfaces
    - Communication Interfaces
    - Memory Constraints
    - Operations
    - Site Adaptation Requirements
  - Product functions
  - User characteristics
  - Constraints, assumptions and dependencies

- Specific requirements
  - External interface requirements
  - Functional requirements
  - Performance requirements
  - Design constraints
    - Standards Compliance
  - Logical database requirement
  - Software System attributes
    - Reliability, Availability, Security, Maintainability, Portability

- Other requirements

Φάση 2: Σχεδίαση

Σχεδιασμός των αναγνωρισμένων κλάσεων και της αλληλεπίδρασης των κλάσεων

- Χρήση της τεχνικής των Καρτών **Class-Responsibilities-Collaborators (CRC)**
- Χρήση Διαγραμμάτων Ακολουθίας **(sequence diagrams)**
- Χρήση Διαγραμμάτων Κλάσεων **(class diagrams)**
- Πέντε στάδια σχεδιασμού αντικειμένων:
  - Ανακάλυψη Αντικειμένου (Object discovery)
  - Συναρμολόγηση Αντικειμένου (Object assembly)
  - Κατασκευή Συστήματος
  - Επέκταση Συστήματος
  - Επαναχρησιμοποίηση κλάσεων

Φάση 2: Σχεδίαση

Κατευθυντήριες γραμμές για σχεδιασμό των αντικειμένων:

- Οι κλάσεις προκύπτουν από το πρόβλημα – επεκτείνονται και ωριμάζουν καθώς χρησιμοποιούνται για την επίλυση και άλλων προβλημάτων.
- Το μεγαλύτερο τμήμα του σχεδιασμού σας είναι να ανακαλύψετε τις κλάσεις που σας χρειάζονται.
- **Δεν είναι ανάγκη να προβλέψετε τα πάντα από την αρχή** – μαθαίνετε καθώς προχωράτε.
- Ξεκινήστε τον προγραμματισμό νωρίς, ώστε να έχετε κάτι που θα αποδεικνύει την αξία ή προχειρότητα του σχεδιασμού σας.
- **KIS: keep-it-simple.** Οι απλές λύσεις είναι συνήθως οι καλύτερες.

Φάση 3: Υλοποίηση

Φάση της μετατροπής του αρχικού σχεδιασμού σε πρόγραμμα προς μετάφραση και εκτέλεση

3Α. Αναγνώριση Τμήματος Πυρήνα

- Πυρήνας: Στόχος εδώ είναι να εντοπίσετε το τμήμα που αποτελεί τον πυρήνα του προγράμματός σας, από τον οποίο θα προκύψει το τελικό σύστημα.
- Ο πυρήνας μπορεί να μην περιέχει τα πάντα, είναι όμως η βάση για το χτίσιμο των λειτουργιών του συστήματός σας.

Φάση 3: Υλοποίηση (συν.)

3Β. Πρόσθεση Λειτουργιών

- Μετά το κτίσιμο του πυρήνα, γίνεται η υλοποίηση των λειτουργιών του συστήματος.
- Κάθε ξεχωριστή λειτουργία αντιπροσωπεύεται στη φάση τού σχεδιασμού από μια αντίστοιχη περίπτωση χρήσης (use case).
- Η υλοποίηση των διαφορετικών περιπτώσεων χρήσης γίνεται διαδοχικά – σε διαφορετικές επαναλήψεις – και μέχρι να ολοκληρωθεί η υλοποίηση όλων των συστατικών του συστήματος.
- Πως δικαιολογείται η «επαναληπτική» προσέγγιση (iteration over the use cases);

Φάση 4: Συντήρηση

- Η φάση της συντήρησης ή εξέλιξης του κώδικα:
  - Διόρθωση σφαλμάτων.
  - Αλλαγές στον κώδικα με βάση την εμπειρία χρήσης του.
  - Πρόσθεση νέων λειτουργιών.
  - Καλύτερη ικανοποίηση των απαιτήσεων.

ΟΟ Σχεδιασμός και Ευθύνες

Πως σχεδιάζουμε αντικειμενοστρεφή προγράμματα;

- Ο ΟΟ σχεδιασμός εστιάζει στην αναγνώριση των **κλάσεων** ενός προβλήματος και στην ανάθεση **ευθυνών** στις κλάσεις (λειτουργίες που η κλάση είναι υπεύθυνη να εκτελέσει).
- Η ανάθεση ευθυνών συνεπάγεται ανάθεση ανεξαρτησίας.
- Δημιουργία **ανεξάρτητων υποσυστημάτων**, τα οποία μπορούν εύκολα να **επαναχρησιμοποιηθούν**.

Παράδειγμα Διαδικασίας Ανάπτυξης Λογισμικού

Στις ακόλουθες διαφάνειες θα δούμε σαν παράδειγμα τη διαδικασία ανάπτυξης του συστήματος **«Βιβλίο Διευθύνσεων»*** αναλύοντας όλες τις φάσεις που αναφέραμε.

- **Φάση 1Α: Ανάλυση Απαιτήσεων – Σενάριο, Use Cases**
- **Φάση 1Β: Προδιαγραφές – Ανάλυση Κλάσεων, Αναφορά**
- **Φάση 2: Σχεδιασμός – CRC, Sequence Diagrams, Class Diagram**
- **Φάση 3: Υλοποίηση – Λεπτομερής Σχεδ. Κλάσεων, Κώδικας**
- **Φάση 4: Συντήρηση: Νέες Απαιτήσεις, μετάφραση/διάδοση στις άλλες φάσεις**

Όπου υπάρχουν επιπρόσθετα εργαλεία από αυτά που θα χρησιμοποιήσουμε, θα αναφέρονται περιληπτικά

*** Το παράδειγμα έχει προσαρμοστεί από το παράδειγμα που βρίσκεται στη ιστοσελίδα http://www.cs.gordon.edu/courses/cs211/AddressBookExample/**

Παράδειγμα: Βιβλίο Διευθύνσεων
Φάση 1Α: Ανάλυση Απαιτήσεων

Αναφορά Απαιτήσεων

- **Βιβλίο Διευθύνσεων: το πρόγραμμα αυτό διατηρεί και συντηρεί ένα βιβλίο διευθύνσεων.**
- Το βιβλίο διευθύνσεων θα είναι μία συλλογή από οντότητες οι οποίες αναπαριστούν άτομα με πληροφορίες για όνομα, επίθετο, διεύθυνση, πόλη, επαρχία, ταχυδρομικό κώδικα και τηλέφωνο
- Το πρόγραμμα πρέπει να υποστηρίζει προσθήκη ενός ατόμου, ενημέρωση υφιστάμενου ατόμου (εκτός από το όνομα) και διαγραφή ατόμου
- <u>Επιπρόσθετες λειτουργίες:</u>
  - Ταξινόμηση των ατόμων αλφαβητικά με επίθετο, όνομα ή ταχ. κώδικα
  - Εκτύπωση όλων των ατόμων σε μορφή “mailing label”
  - Δημιουργία ενός καινούριου και άνοιγμα ενός υφιστάμενου βιβλίου διευθύνσεων από τον δίσκο με τη χρήση διαπροσωπείας με τις συνηθισμένες επιλογές κάτω από το File (New, Open, Close, Save and Save As)

Αναφορά Απαιτήσεων (συν.)

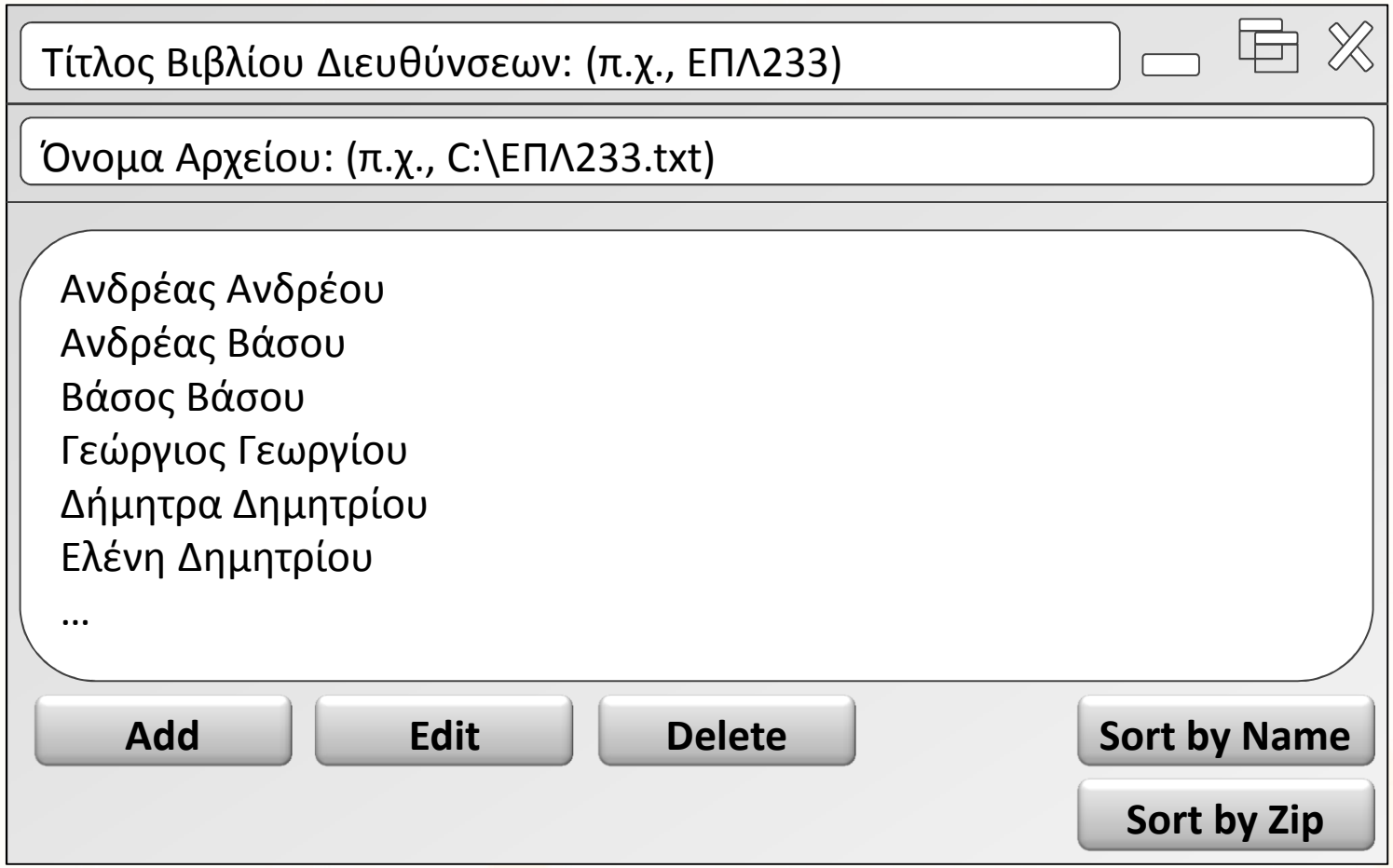
- Επιπρόσθετες λειτουργίες (συν.):
  - Το πρόγραμμα θα πρέπει να επεξεργάζεται, ανά πάσα στιγμή, μόνο ένα βιβλίο διευθύνσεων. Στην περίπτωση που ο χρήστης επιλέξει «Νέο Βιβλίο Διευθύνσεων» ή «Άνοιγμα Υφιστάμενου Βιβλίου Διευθύνσεων» τότε το πρόγραμμα θα σταματάει/ κλείνει το υφιστάμενο/τρέχον βιβλίο διευθύνσεων
  - Σε κάποια φάση στο μέλλον, το πρόγραμμα πιθανόν να μπορεί να υποστηρίζει να τρέχουν πολλά γραμματοκιβώτια ταυτόχρονα, το καθένα στο δικό του παράθυρο. Σε αυτή την περίπτωση, όταν ο χρήστης επιλέξει «Νέο Βιβλίο Διευθύνσεων» ή «Άνοιγμα Υφιστάμενου Βιβλίου Διευθύνσεων» τότε το πρόγραμμα δεν θα επηρεάζει τα τρέχοντα βιβλία διευθύνσεων
  - Το πρόγραμμα θα παρακολουθεί τις αλλαγές που γίνονται στο βιβλίο διευθύνσεων και θα παρέχει στο χρήστη την δυνατότητα να αποθηκεύσει τις αλλαγές στο υφιστάμενο ή σε νέο βιβλίο διευθύνσεων

Αναφορά Απαιτήσεων (συν.)

- Επιπρόσθετες λειτουργίες (συν.):
  - Το πρόγραμμα θα παρακολουθεί το αρχείο από το οποίο έχει διαβαστεί ή που έχει αποθηκευτεί πιο πρόσφατα, θα παρουσιάζει το όνομα του αρχείου στον τίτλο του παραθύρου και θα το χρησιμοποιεί όταν ο χρήστης επιλέξει το «Save»
  - Όταν ένα καινούριο παράθυρο δημιουργηθεί, τότε το παράθυρο θα παρουσιάζει τον τίτλο «Untitled» και η επιλογή «Save» μετατρέπεται σε «Save As» και ο χρήστης θα πρέπει να δώσει ένα όνομα αρχείου

Αναφορά Απαιτήσεων (συν.)

- Ανάλυση Διαπροσωπείας:

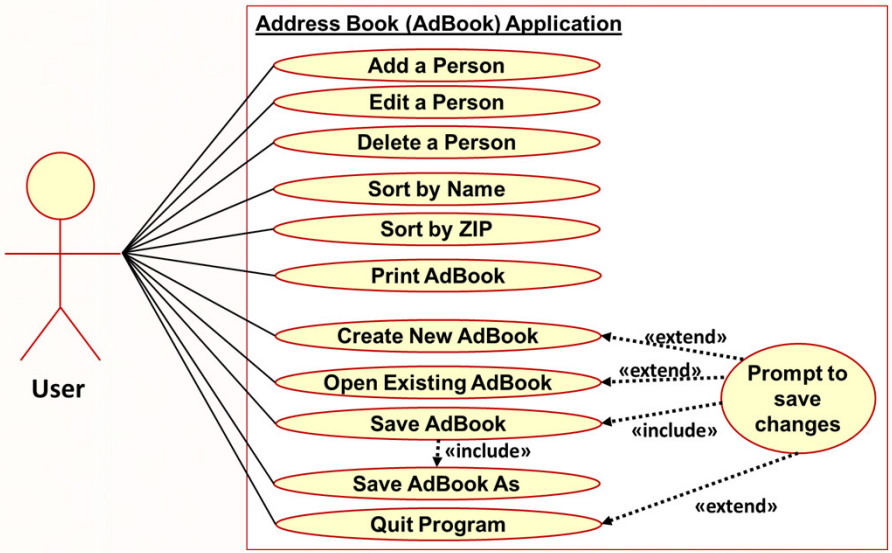


Περιπτώσεις Χρήσης (Use Cases)

- Τεχνική ανάλυσης με την οποία περιγράφουμε, με τρόπο τυπικό **(use case)**, πώς πρέπει να λειτουργήσει ένα πρόγραμμα.
- **Μετάφραση της Αναφοράς Απαιτήσεων σε Use Cases**
- Κάθε **use case εστιάζει σε ένα συγκεκριμένο σενάριο**
- **Κάθε σενάριο περιγράφει** τα **βήματα** που πρέπει να πραγματοποιηθούν για να φέρουν το σενάριο σε μία επιτυχημένη ολοκλήρωση.
- **Η ολοκλήρωση του σεναρίου** πρέπει να αντιστοιχεί στην **επίτευξη ενός σημαντικού στόχου**
- Κάθε βήμα αναπαριστά μια αλληλεπίδραση του προγράμματος με ανθρώπους ή εξωτερικές οντότητες του προγραμματιστικού συστήματος

Διαγράμματα Περίπτωσης Χρήσης (Use Case Diagrams)

- **Use Case Diagram:** γραφική αναπαράσταση υψηλού επιπέδου του συστήματος



- Βλέποντας ένα use case diagram πρέπει να μπορούμε να πούμε εύκολα **τι κάνει το σύστημα** και **ποιος αλληλεπιδρά μαζί του**
- Περιλαμβάνει τα **use cases**, που αντιπροσωπεύουν τις διάφορες λειτουργίες του συστήματος και **actors**, που αντιπροσωπεύουν τους χρήστες/ρόλους του συστήματος

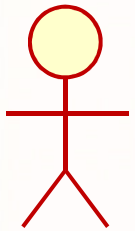
Διαγράμματα Περίπτωσης Χρήσης (Use Case Diagrams)

- Μπορούμε να δημιουργήσουμε **περισσότερο του ένα use case diagrams για ένα σύστημα**:
  - Το **κάθε use case** θα δείχνει ένα **υποσύνολο του συστήματος**
  - Μπορούν να **οργανωθούν σε πακέτα** για καλύτερη οργάνωση του μοντέλου
- Ένα use case diagram μπορεί να βοηθήσει στην επικοινωνία με τους τελικούς χρήστες του συστήματος.
  - Είναι απλό, μη τεχνικό ώστε ο καθένας στην ομάδα (**προγραμματιστές και τελικοί χρήστες**) να καταλήξουν σε μία **κοινή κατανόηση του συστήματος**
  - Συνήθως δημιουργείται από την τεχνική ομάδα σε συνδυασμό με αντιπρόσωπους των τελικών χρηστών

Περιπτώσεις Χρήσης (Use Cases) (συν.)

Χρήση των πιο κάτω διαγραμμάτων

- **Ρόλος/Χρήστης (actor):** ένα άτομο ή ένα σύνολο από άτομα που ανήκουν στον ίδιο ρόλο και αλληλεπιδρούν με το σύστημα ή μεταξύ τους



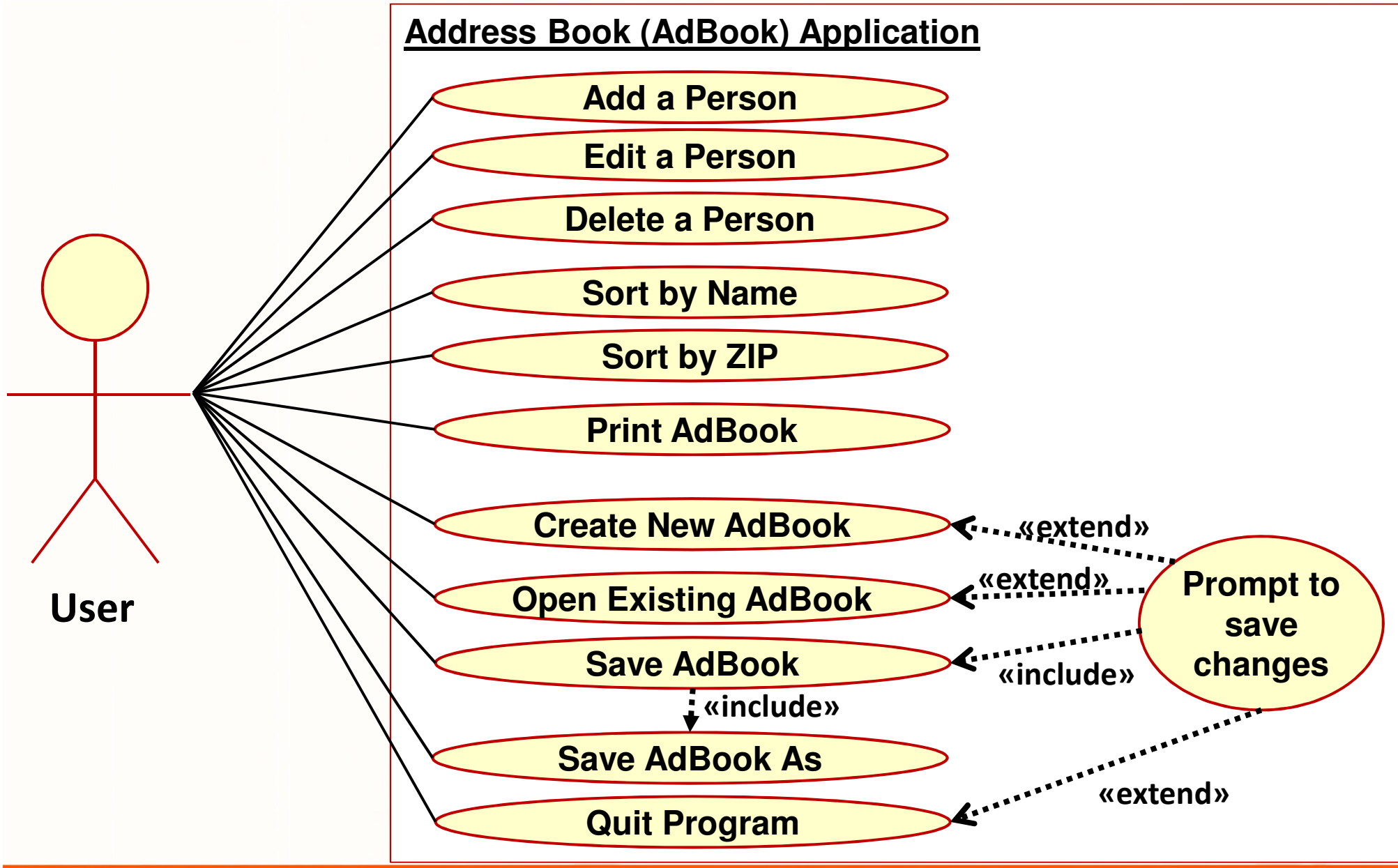
- **Περιπτώσεις Χρήσης (use case):** μία υψηλού επιπέδου λειτουργία που παρέχει το σύστημα

- **Σχέσεις (relationships):** αλληλεπίδραση actor με use case
  - **Σχέση «include»:** υποδηλώνει ότι ένα use case ΠΡΕΠΕΙ να χρησιμοποιήσει κάποιο άλλο

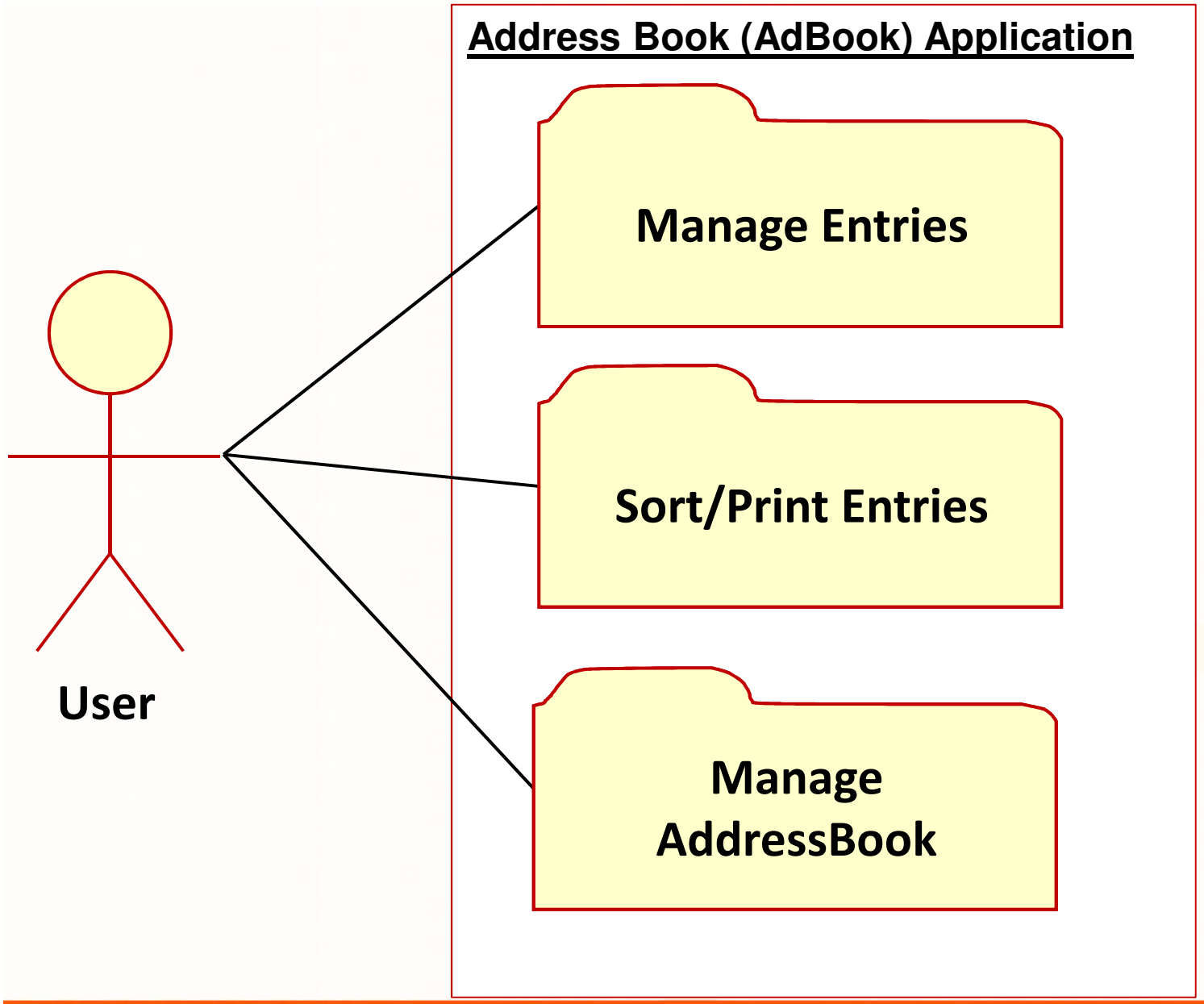
  - **Σχέση «extend»:** υποδηλώνει ότι ένα use case επεκτείνει την βασική λειτουργία κάποιου άλλου use case

  - **Σχέση «generalization»**: ένας actor ή use case κληρονομεί τα χαρακτηριστικά κάποιου άλλου actor ή use case

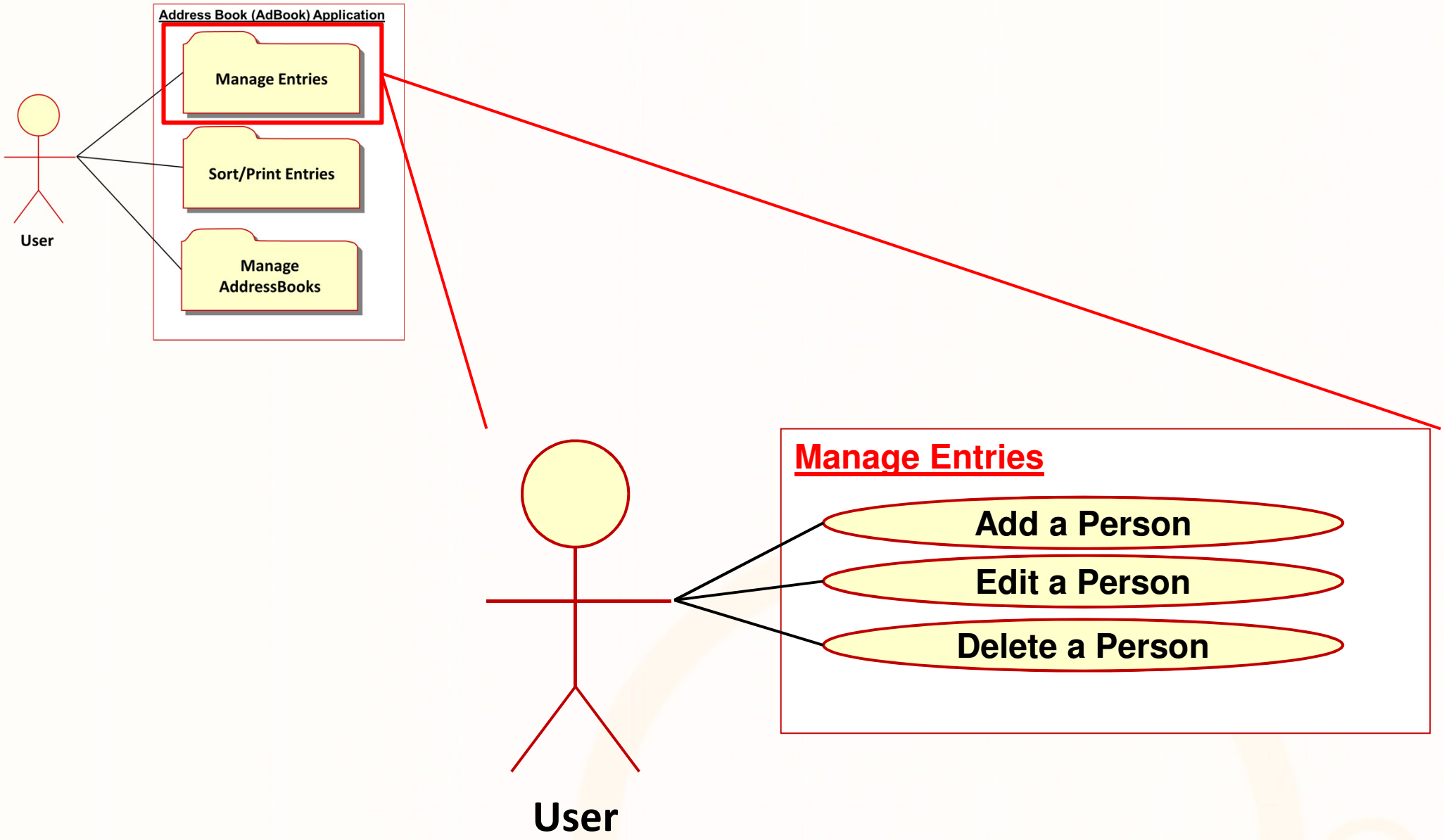
Βιβλίο Διευθύνσεων: Use Case Diagram



Βιβλίο Διευθύνσεων: Use Case Package Diagram



Βιβλίο Διευθύνσεων: Use Case Diagram (packets)



Βιβλίο Διευθύνσεων: Use Cases

- **Add Person**

  Το εν λόγω use case ξεκινά όταν ο χρήστης κάνει κλικ στο κουμπί "Προσθήκη" στο κύριο παράθυρο. Εμφανίζεται ένα παράθυρο διαλόγου, με τίτλο "Νέο Άτομο", το οποίο περιέχει πεδία στα οποία πρέπει να συμπληρωθεί το όνομα και το επώνυμο του νέου ατόμου και άλλες πληροφορίες. Το παράθυρο διαλόγου μπορεί να κλείσει είτε κάνοντας κλικ στο κουμπί "ΟΚ" ή "Άκυρο". Εάν πατηθεί το κουμπί "ΟΚ", ένα νέο άτομο προστίθεται στο τέλος του βιβλίου διευθύνσεων, και το όνομα του ατόμου, προστίθεται στο τέλος της λίστας των ονομάτων στο κύριο παράθυρο. Εάν πατηθεί το κουμπί "Άκυρο", δεν γίνονται αλλαγές ούτε στο βιβλίο διευθύνσεων, ούτε στο κύριο παράθυρο.

Βιβλίο Διευθύνσεων: Use Cases

- **Edit Person**

  Το εν λόγω use case ξεκινά όταν ο χρήστης επιλέξει είτε ένα όνομα από τη λίστα των ονομάτων στο κύριο παράθυρο και, στη συνέχεια κάνει κλικ στο κουμπί "Επεξεργασία", είτε κάνει διπλό κλικ σε ένα όνομα. Σε κάθε περίπτωση, ένα παράθυρο διαλόγου, με τίτλο "Επεξεργασία ονόματος του ατόμου", εμφανίζεται περιέχοντας πληροφορίες σχετικές με το επιλεγμένο πρόσωπο (εκτός από το όνομα του ατόμου, το οποίο εμφανίζεται μόνο στον τίτλο). Ο χρήστης μπορεί στη συνέχεια να επεξεργαστεί τα πεδία με τα χαρακτηριστικά του ατόμου. Το παράθυρο διαλόγου μπορεί να κλείσει είτε κάνοντας κλικ στο "ΟΚ" ή "Άκυρο". Εάν πατηθεί το κουμπί "ΟΚ", η καταχώρηση στο βιβλίο διευθύνσεων για το επιλεγμένο πρόσωπο ενημερώνεται με τις αλλαγές που έγιναν από το χρήστη. Εάν πατηθεί το κουμπί "Άκυρο", δεν γίνονται αλλαγές στο βιβλίο διευθύνσεων.

Βιβλίο Διευθύνσεων: Use Cases

- **Sort by Name**

  Το εν λόγω use case ξεκινά όταν ο χρήστης κάνει κλικ στο κουμπί “Ταξινόμηση κατά Όνομα” στο κύριο παράθυρο. Οι καταχωρήσεις στο βιβλίο διευθύνσεων ταξινομούνται αλφαβητικά κατά όνομα, και ο κατάλογος στο κύριο παράθυρο ενημερώνεται ώστε να αντικατοπτρίζει τη ταξινόμηση.

Βιβλίο Διευθύνσεων: Use Cases

- **Prompt to Save Changes Extension**

  Το εν λόγω use case ξεκινά μέσα από τη Δημιουργία νέου βιβλίου διευθύνσεων, Άνοιγμα υπάρχον βιβλίο διευθύνσεων, ή Έξοδος προγράμματος, εφόσον το τρέχον βιβλίο διευθύνσεων έχει αλλάξει από τη τελευταία επιτυχημένη λειτουργία New, Open, Save, ή Save As. Ένα παράθυρο διαλόγου εμφανίζεται, ενημερώνοντας τον χρήστη ότι υπάρχουν μη αποθηκευμένες αλλαγές, και ζητώντας από το χρήστη είτε να αποθηκεύσει τις αλλαγές, να μην αποθηκεύσει τις αλλαγές, ή να ακυρώσει τη λειτουργία. Αν ο χρήστης επιλέξει να αποθηκεύσει τις αλλαγές, το use case Αποθήκευση Βιβλίου Διευθύνσεων εκτελείται (το οποίο μπορεί να οδηγήσει στην εκτέλεση του use case Αποθήκευση Βιβλίου Διευθύνσεων ως, σε περίπτωση που δεν υπάρχει τρέχον αρχείο). Εάν ο χρήστης δεν επιλέξει να αποθηκεύσει τις αλλαγές, η αρχική λειτουργία απλά επαναλαμβάνεται. Εάν ο χρήστης επιλέξει να ακυρώσει (ή ακυρώσει το διάλογο Αποθήκευση αρχείου), η αρχική λειτουργία ακυρώνεται.

Πλήρης Περιγραφή Use Cases

1. Title
2. Primary Actor
3. Goal in Context
4. Scope
5. Level
6. Stakeholders and Interests
7. Precondition
8. Minimal Guarantees
9. Success Guarantees
10. Trigger
11. Main Success Scenario
12. Extensions
13. Technology & Data Variations List
14. Related Information.

Περιγραφή Αποκλίσεων (Variations List)

- Κατά την εκτέλεσή του, ένα σενάριο μπορεί σε κάποιο στάδιο να αποτύχει.
- Για παράδειγμα, στις Περιπτώσεις Χρήσης του Βιβλίου Διευθύνσεων:
  - η αποθήκευση σε ένα αρχείο μπορεί να αποτύχει γιατί το αρχείο ήδη υπάρχει και επιτρέπεται μόνο η ανάγνωση
  - η δημιουργία ενός καινούριου ατόμου μπορεί να αποτύχει γιατί δεν υπάρχει άλλη μνήμη
- Για την κάλυψη των περιπτώσεων αποτυχίας ενός σεναρίου, στην περιγραφή της Περίπτωσης Χρήσης συμπεριλαμβάνεται και η περιγραφή Αποκλίσεων (variations)
- Σε αυτό το κείμενο αναφέρονται και εναλλακτικές λύσεις για επίλυση των προβλημάτων

Παράδειγμα: Βιβλίο Διευθύνσεων
Επόμενη Φάση 1Β: Προδιαγραφές

Φάση 1: Ανάλυση Απαιτήσεων και Προδιαγραφές (συν.)

1Β. Προδιαγραφές Συστήματος (System Specification)

- Προκαταρκτική αναγνώριση κλάσεων **(Analysis Class Diagram)** Συγκεκριμένη περιγραφή του τι θα πρέπει κάνει το πρόγραμμα (όχι πως θα το κάνει). Παράδειγμα του τι μπορεί να περιλαμβάνουν οι προδιαγραφές:

- Overall description
  - Product perspective
    - System Interfaces
    - User Interfaces
    - Hardware interfaces
    - Software interfaces
    - Communication Interfaces
    - Memory Constraints
    - Operations
    - Site Adaptation Requirements
  - Product functions
  - User characteristics
  - Constraints, assumptions and dependencies

- Specific requirements
  - External interface requirements
  - Functional requirements
  - Performance requirements
  - Design constraints
    - Standards Compliance
  - Logical database requirement
  - Software System attributes
    - Reliability, Availability, Security, Maintainability, Portability

- Other requirements

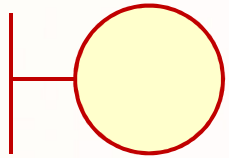
Ανάλυσης Κλάσεων

Στα αρχικά στάδια του προγράμματος, είναι καλό να διεξαχθεί μία προκαταρκτική αναγνώριση των κλάσεων/αντικειμένων του συστήματος (Class Analysis) καθώς και οι αλληλεπιδράσεις μεταξύ τους.

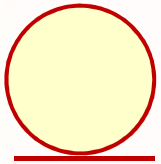
- Οι κλάσεις σε αυτό το προκαταρκτικό στάδιο παίρνουν την μορφή κλάσεων **συνόρων**, κλάσεων **οντοτήτων** και κλάσεων **ελέγχου**.
- Τα **use cases** συνήθως **αποκαλύπτουν το μεγαλύτερο ποσοστό των κλάσεων**
- Οι έμπειροι αναλυτές/προγραμματιστές μπορεί να αναγνωρίσουν όλες τις κλάσεις → προς ολοταχώς στην υλοποίηση, χωρίς σχεδίαση ☺

Διαγράμματα Ανάλυσης Κλάσεων

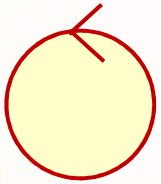
- **Κλάσεις Σύνορα (Boundary Classes):** χειρίζονται την επικοινωνία μεταξύ actors και components του συστήματος, π.χ., user interface, device interface. Συνήθως μόνο ένα boundary class για κάθε use case.



- **Κλάσεις Οντότητες (Entity Classes):** μοντελοποιούν την πληροφορία που χειρίζεται το σύστημα



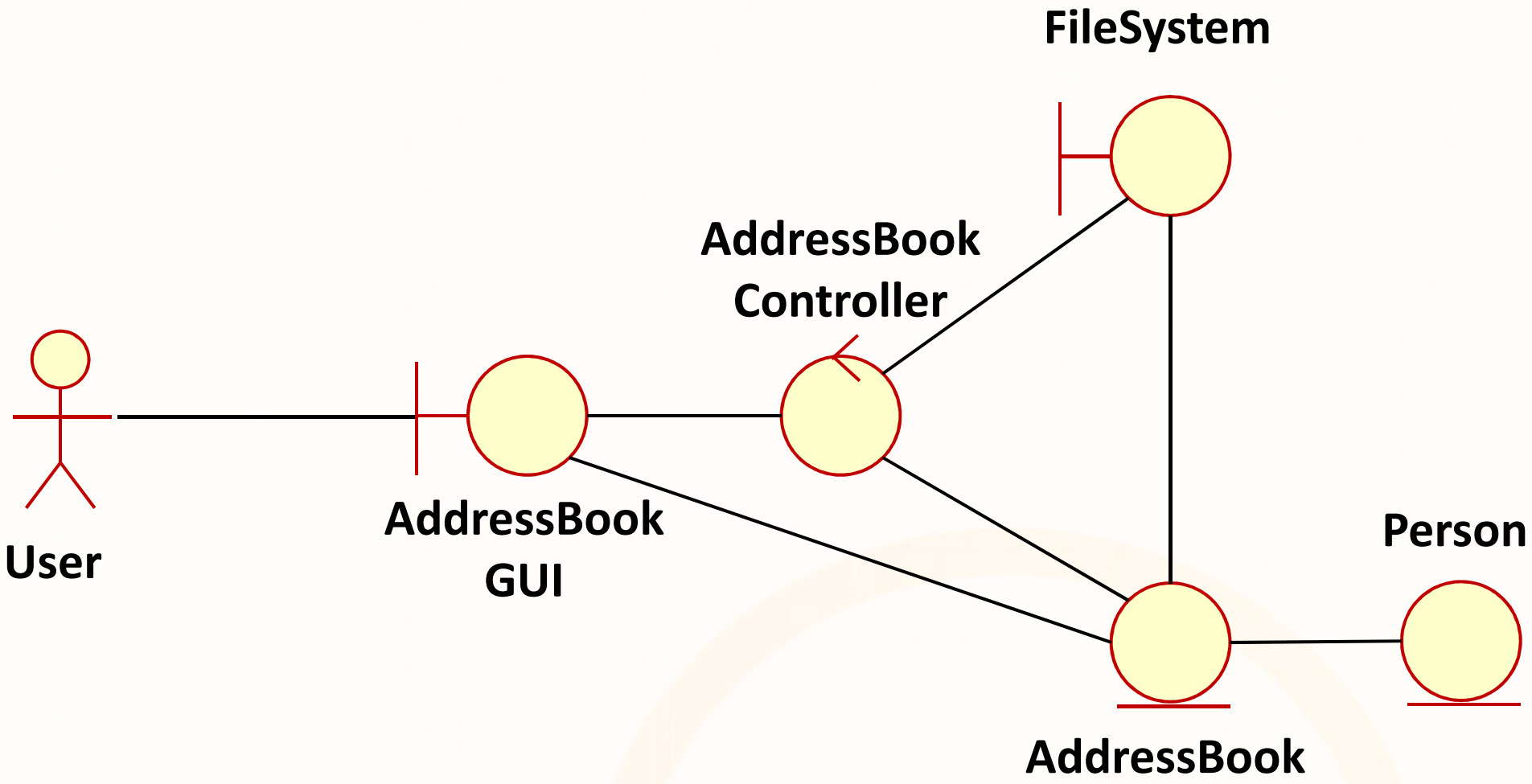
- **Κλάσεις Ελέγχου (Control Classes):** αντιπροσωπεύουν το συντονισμό/ροή ελέγχου για ένα use case.



Βιβλίο Διευθύνσεων: Class Analysis

- Το πρόγραμμα, **ανά πάσα στιγμή** διαχειρίζεται **ένα βιβλίο διευθύνσεων (AddressBook)**. → οντότητα
- Κάθε άτομο **(Person)** μπορεί να αναπαρασταθεί σαν μία **οντότητα** με τα χαρακτηριστικά της → οντότητα
- Μία **διαπροσωπεία (AddressBookGUI)** θα υπάρχει **μεταξύ χρήστη και συστήματος**. Ο χρήστης δεν θα μπορεί να έχει πρόσβαση διαφορετικά με το σύστημα → **σύνορο**
- Μία **διαπροσωπεία/αντικείμενο (FileSystem)** θα υπάρχει **μεταξύ του συστήματος και του λειτουργικού** ώστε το σύστημα να μπορεί να διαχειρίζεται αρχεία → **σύνορο**
- Ένα αντικείμενο διαχειριστής **(AddressBookController)** χρειάζεται για να αναλάβει την **εκτέλεση των use cases** σε ανταπόκριση των διαφόρων λειτουργιών που θα επιλέξει ο χρήστης από το μενού. → controller
*(Για ένα τέτοιο μικρό πρόβλημα, ένας controller είναι αρκετός.)

Βιβλίο Διευθύνσεων: Analysis Class Diagram



Βιβλίο Διευθύνσεων: Class Analysis (συν.)

- Το use case “Add a Person” έχει σχέση με την εισαγωγή νέας πληροφορίας (δηλ. νέο άτομο) από τον χρήστη. Τέλος, θα καλεστεί το αντικείμενο AddressBook ώστε να εισάξει το νέο άτομο στη συλλογή.
- Το use case “Edit a Person” έχει σχέση με την παρουσίαση της τρέχουσας πληροφορίες κάποιου ατόμου μέσω του αντικειμένου AddressBook και παροχή της ικανότητας αλλαγή των στοιχείων αυτού. Στο τέλος, θα καλεστεί το αντικείμενο AddressBook για να αποθηκεύει τις αλλαγές.
- Το use case “Delete a Person” έχει σχέση με την διαγραφή ενός υπάρχον ατόμου. παρουσίαση της τρέχουσας. Στο τέλος, θα καλεστεί το αντικείμενο AddressBook για να αφαιρέσει το άτομο από την συλλογή.
- Το use case “Sort by Name” θα καλέσει το αντικείμενο AddressBook για να ταξινομήσει τη συλλογή των ατόμων με το όνομα και επίθετο τους.
- Το use case “Sort by ZIP” θα καλέσει το αντικείμενο AddressBook για να ταξινομήσει τη συλλογή των ατόμων με τον ταχυδρομικό κώδικα τους.

Βιβλίο Διευθύνσεων: Class Analysis (συν.)

- Το use case “Create New Address Book” δημιουργεί ένα καινούριο αντικείμενο τύπου AddressBook.
- Το use case “Open Existing Address Book” θα ζητήσει από τον χρήστη ένα αρχείο και μετά θα χρησιμοποιήσει το αντικείμενο FileSystem για να διαβάσει ένα αντικείμενο τύπου AddressBook που έχει αποθηκευτεί στο αρχείο.
- Το use case “Save Address Book” θα χρησιμοποιήσει το αντικείμενο FileSystem για να αποθηκεύσει ένα αντικείμενο τύπου AddressBook στο πιο πρόσφατο αρχείο που έχει διαβαστεί ή αποθηκευτεί). Αν δεν υπάρχει κάποιο αρχείο τότε θα εκτελέσει το use case “Save Address Book As”
- Το use case “Print Address Book” θα ζητήσει από το αντικείμενο AddressBook να τυπώσει την συλλογή της σύμφωνα με την τρέχουσα ταξινόμηση.
- Το use case “Quit Program” δεν έχει σχέση με τα υπόλοιπα αντικείμενα.

Παράδειγμα: Βιβλίο Διευθύνσεων
Επόμενη Φάση 2: Σχεδίαση