Εισαγωγή στην Τεχνολογία Λογισμικού

Γιάννης Σμαραγδάκης

Γιατί μελετάμε την Τεχνολογία Λογισμικού

- Λογισμικό υπάρχει παντού
  - σε όλους τους τεχνικούς τομείς (π.χ. αεροδιαστημική)
  - σε όλες τις επιστήμες
  - στον πολιτισμό και την τέχνη (π.χ. βιβλιοθήκες, μουσική)
  - μεγάλο τμήμα της παγκόσμιας οικονομίας, σημαντικά υποσχόμενος τομέας στην Ελλάδα
- ΑΛΛΑ είναι πολύ δύσκολο να αναπτυχθεί σωστά
  - 5-10 LOC/ανθρωποημέρα;
  - $100Ks/LOC ?
  - προθεσμίες που δεν τηρούνται, bugs με τεράστιο κόστος (π.χ.;)

Ενδιαφέρουσες ερωτήσεις

- **Τι σημαίνει ποιότητα στο λογισμικό;**
- **Τι είναι σφάλμα;**
- **Μπορεί να αποδειχθεί η απουσία σφαλμάτων;**
- **Πώς μπορεί να προσθέτουμε εργαζόμενους και το αποτέλεσμα να χειροτερεύει;**
- **Τι είναι τελικά το λογισμικό; Με τι άλλο μοιάζει;**
- **Βασίζεται το λογισμικό σε κάποια επιστήμη; Έχει νόμους;**

Προσεγγίσεις στα προβλήματα του λογισμικού

- **Περισσότερο προσωπικό**
  -- χάος, πιο πολλά προβλήματα

- **«Καλύτερες» γλώσσες προγραμματισμού**
  -- κακά προγράμματα μπορεί να γραφτούν σε όλες τις γλώσσες

- **Σχεδιασμός πριν τη συγγραφή κώδικα**
  -- πώς εξασφαλίζεται η ποιότητα του σχεδιασμού;

- **Αρχή με προδιαγραφές απαιτήσεων**
  -- αλλά πάντα αλλάζουν, και μετά;

- **Πιο πλήρεις δοκιμασίες**
  -- αν η ποιότητα δεν υπάρχει, δεν προστίθεται με περισσότερες δοκιμασίες

Άλλες προσεγγίσεις

- **Καλύτερα εργαλεία**
  -- ... όπως και δοκιμασίες/γλώσσες

- **Καλύτερες διαδικασίες**
  -- πότε είναι μια διαδικασία «καλή»;

- **Καλύτερη εκπαίδευση**
  -- τι να διδάξουμε;

"There is no Silver Bullet" -- Fred Brooks

Με τι μοιάζει το λογισμικό;

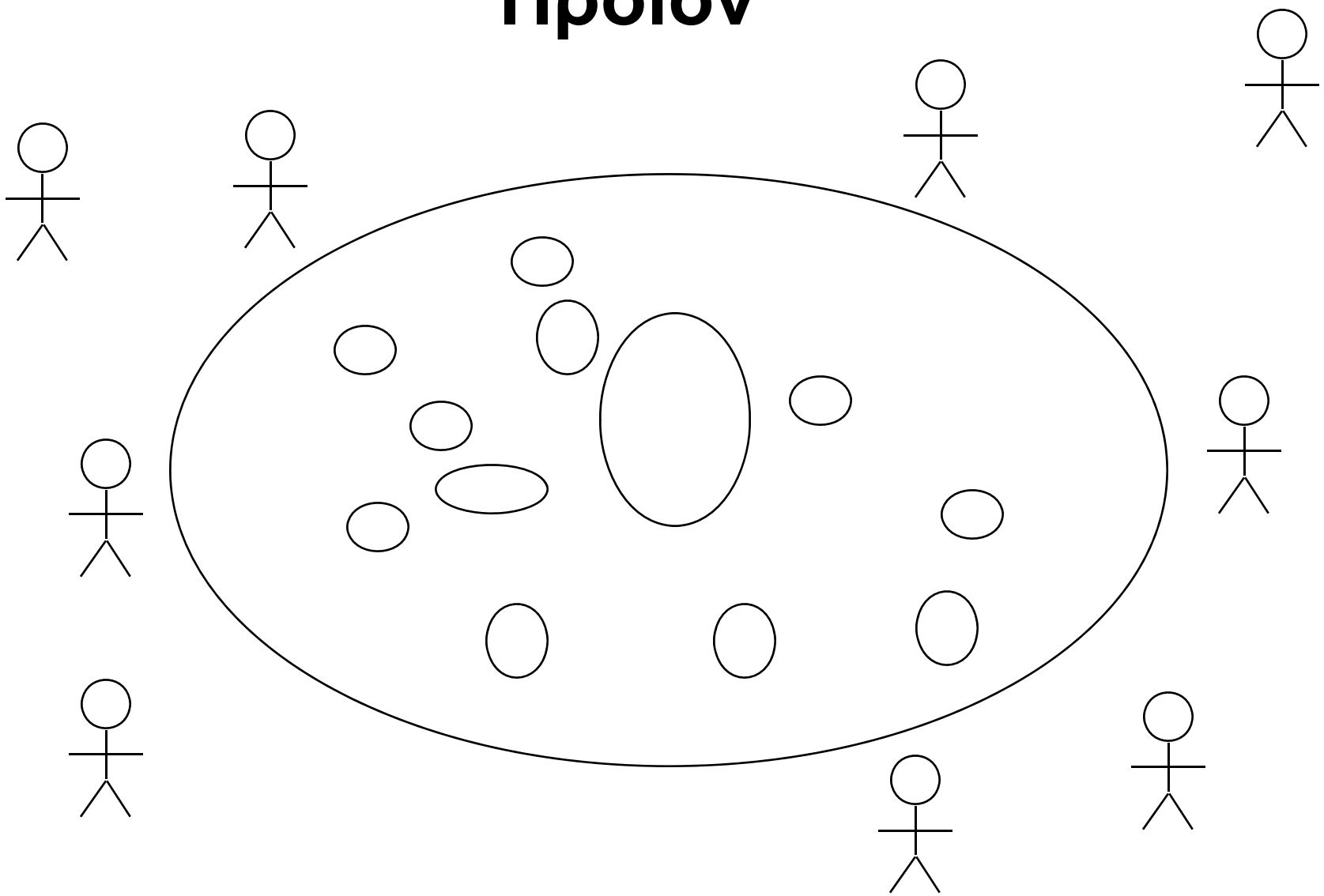
- **Φύση; Πολυπλοκότητα; Μέγεθος; Ευκολία αλλαγής; Ανάγκη συντήρησης;**
- **Παρόμοια άλλα προϊόντα; Ομοιότητες/διαφορές;**
  - **οικίσμα;**
  - **νόμοι/συνθήκες;**
  - **ταινίες;**
  - **συνταγές μαγειρικής;**

Τι σημαίνει για ένα προϊόν να είναι «καλό»;

Ικανοποιεί τις ανάγκες όλων των συμμετόχων

- **Τύποι συμμετόχων (stakeholders):**
  - **χρήστες**
  - **προγραμματιστές, δοκιμαστές**
  - **πελάτες (τι διαφορά από το χρήστη;)**
  - **τελικοί πελάτες, κοινωνία**
  - **επενδυτές**
  - **κλπ.**

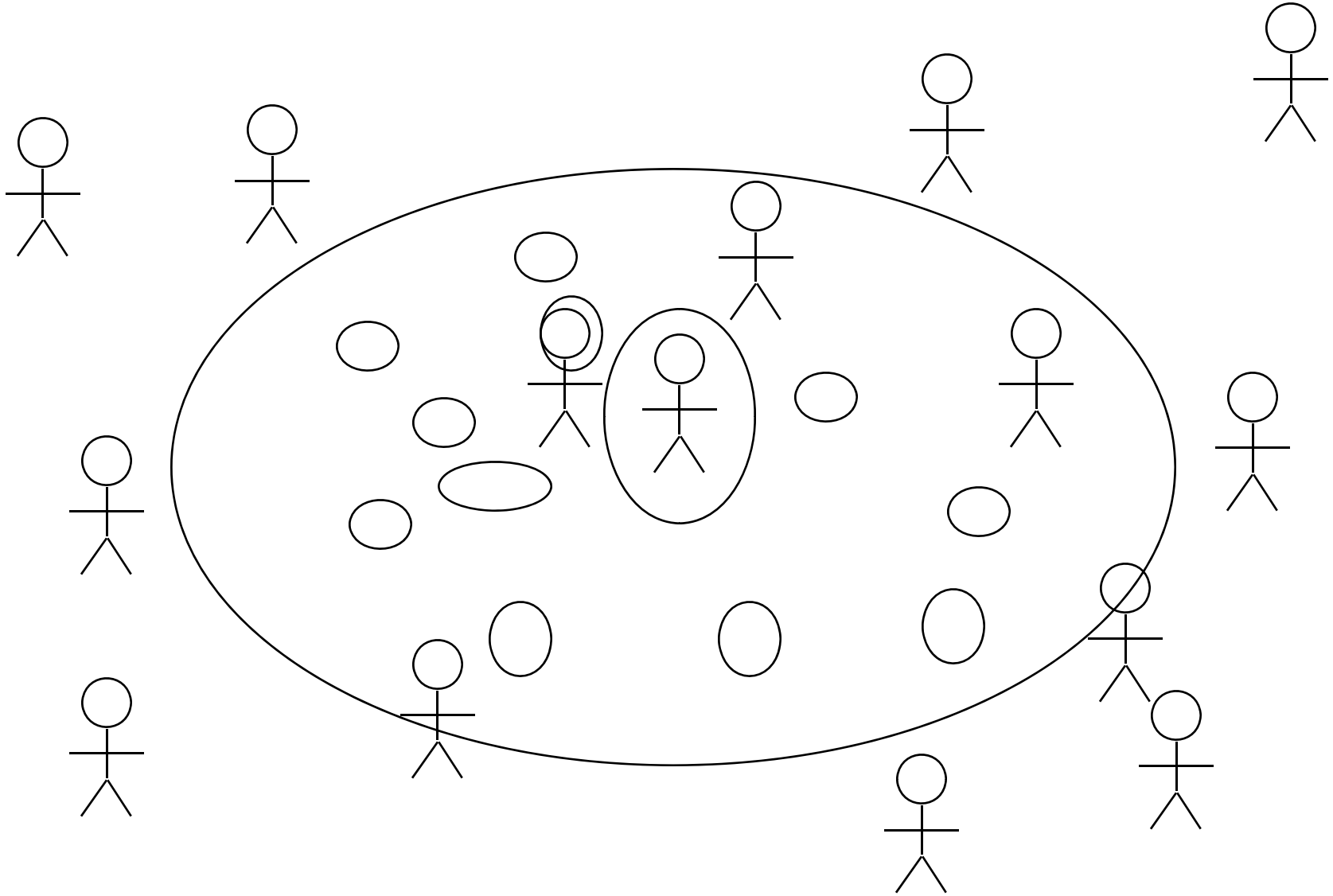
Προϊόν



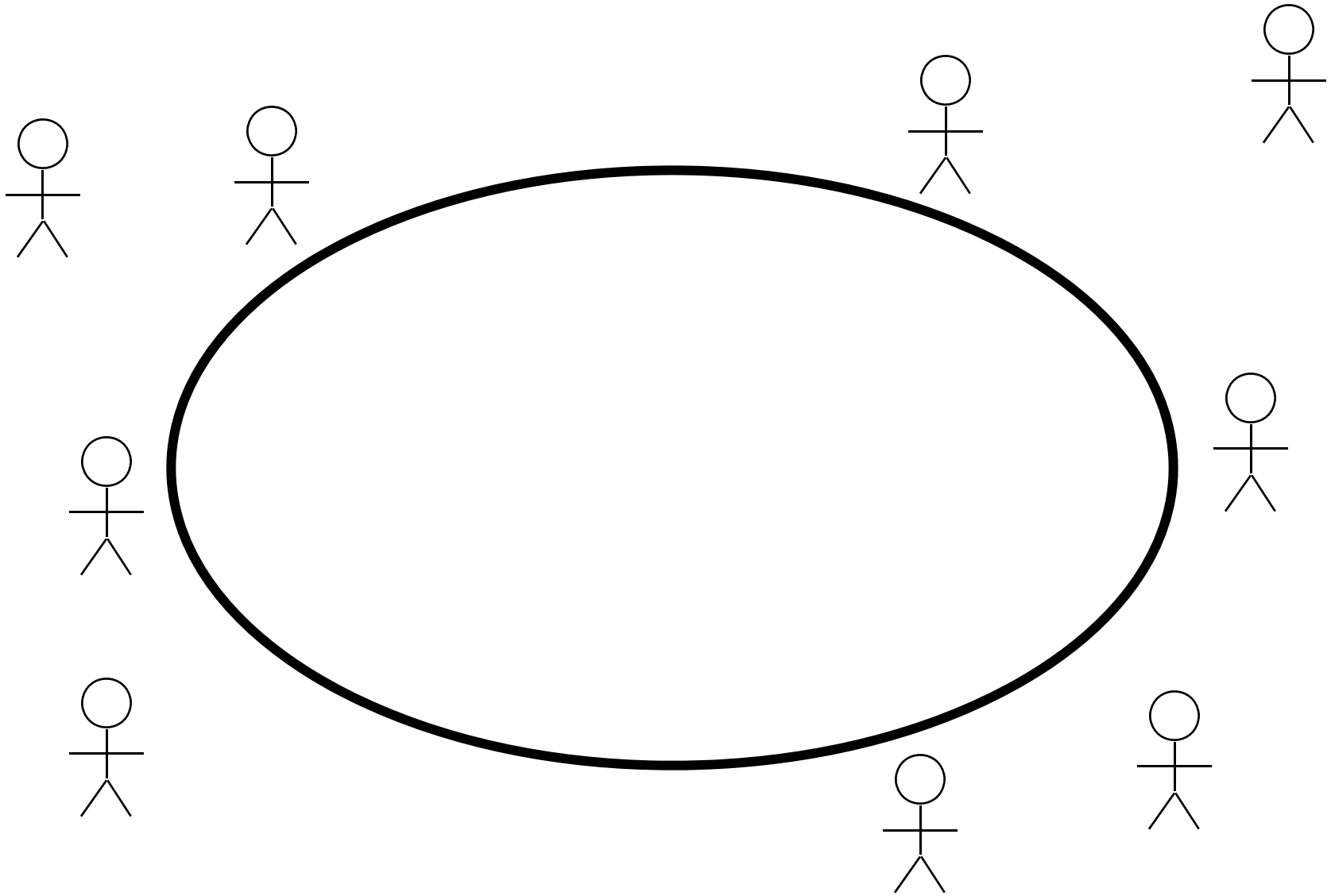
Διαφορετικές απαιτήσεις

- **Χρήστης:**
  - **Θέλει πολλές λειτουργίες**
  - **Θέλει να είναι εύκολη η δουλειά του**
- **Αρχιτέκτονας:**
  - **Θέλει όλα τα κομμάτια να ταιριάζουν**
  - **Θέλει να είναι εύκολη η αλλαγή**
  - **Θέλει να είναι εύκολο να δείξει ότι κάνει αυτό που πρέπει**
- **Πελάτης:**
  - **Θέλει μικρό κόστος**
  - **Θέλει ορατή και σταθερή πρόοδο**
- **Τελικοί πελάτες/κοινωνία:**
  - **Θέλουν να είναι βέβαιοι ότι δεν θα προκληθεί ζημιά**
  - **Θέλουν να δουν ότι παρέχονται οι επιθυμητές υπηρεσίες**
- **Επενδυτής**
  - **Θέλει απόδειξη ότι θα έχει κέρδος**

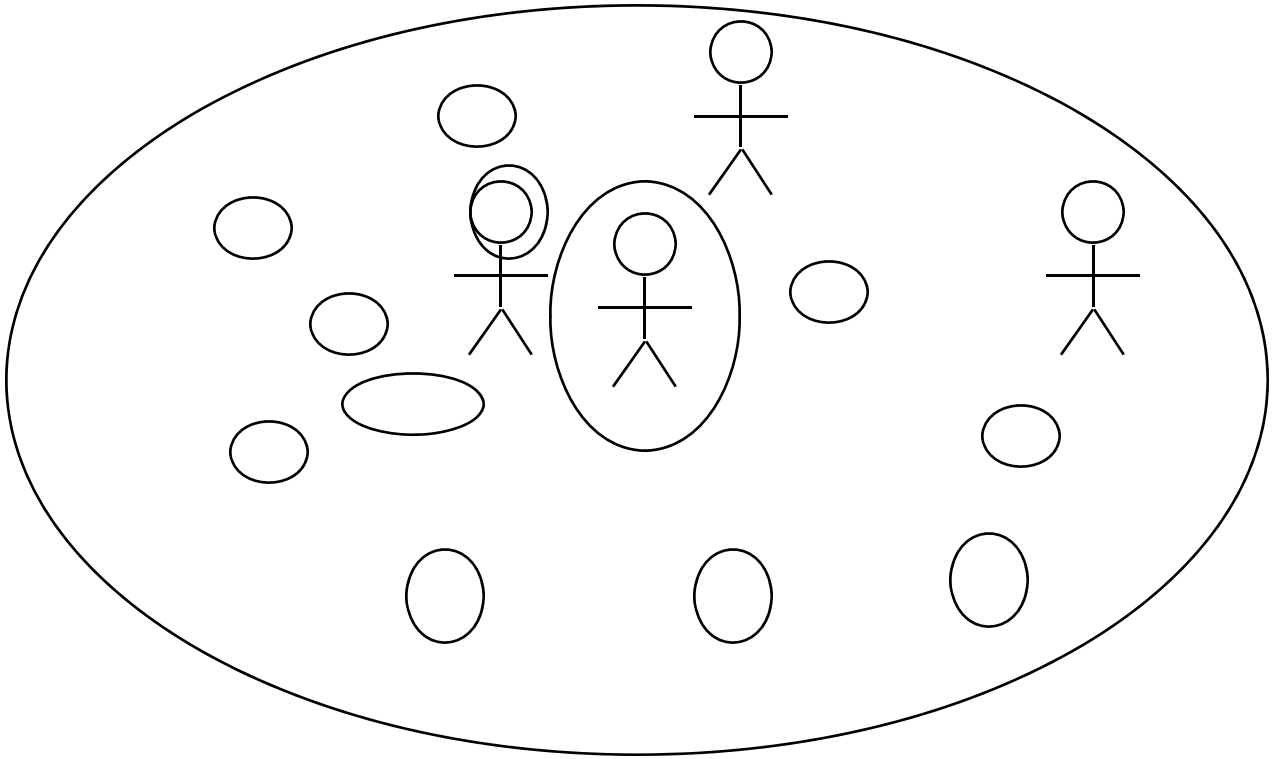
Διαφορετικοί συμμετέχοντες έχουν διαφορετικές όψεις



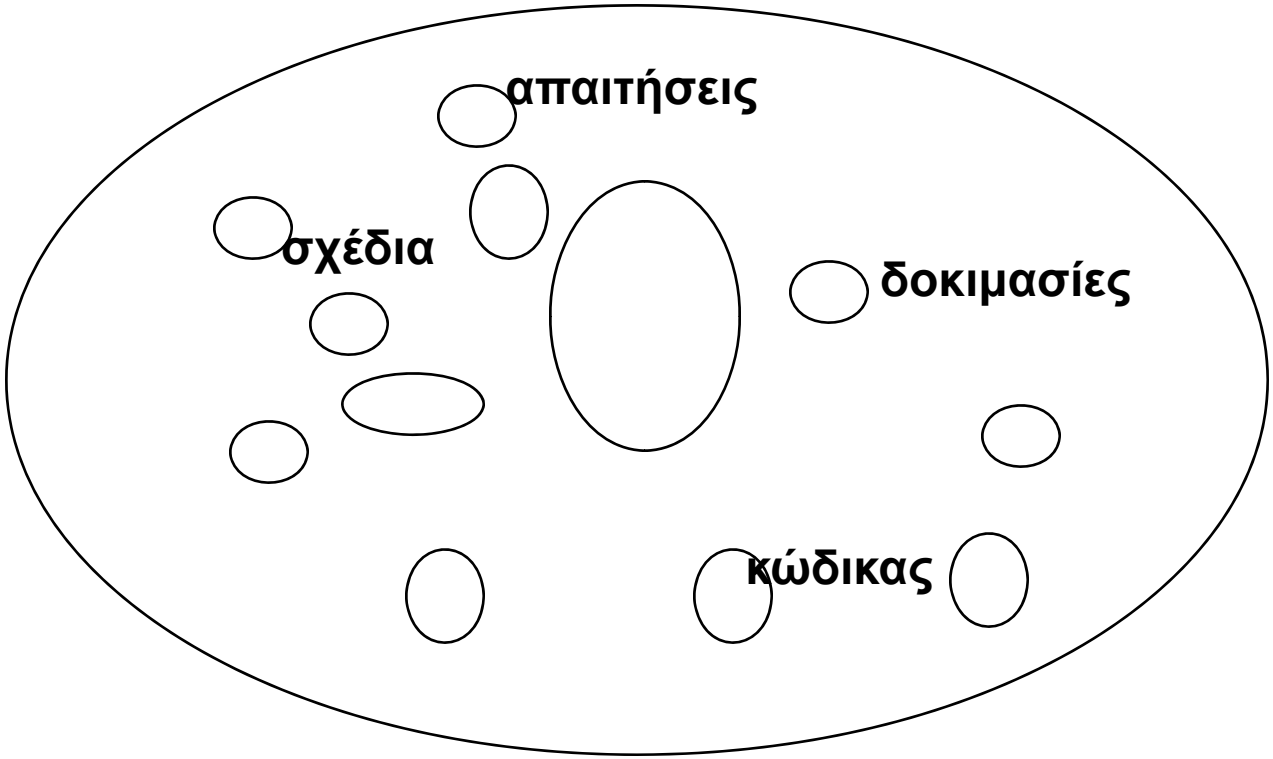
Εξωτερική όψη: πελάτες, επενδυτές, κοινωνία



Εσωτερικές όψεις: αρχιτέκτονες, διοικητικοί, προγραμματιστές



Ένα προϊόν λογισμικού έχει πολλών τύπων εξαρτήματα/τμήματα



Πρόκληση Τεχνολογίας Λογισμικού

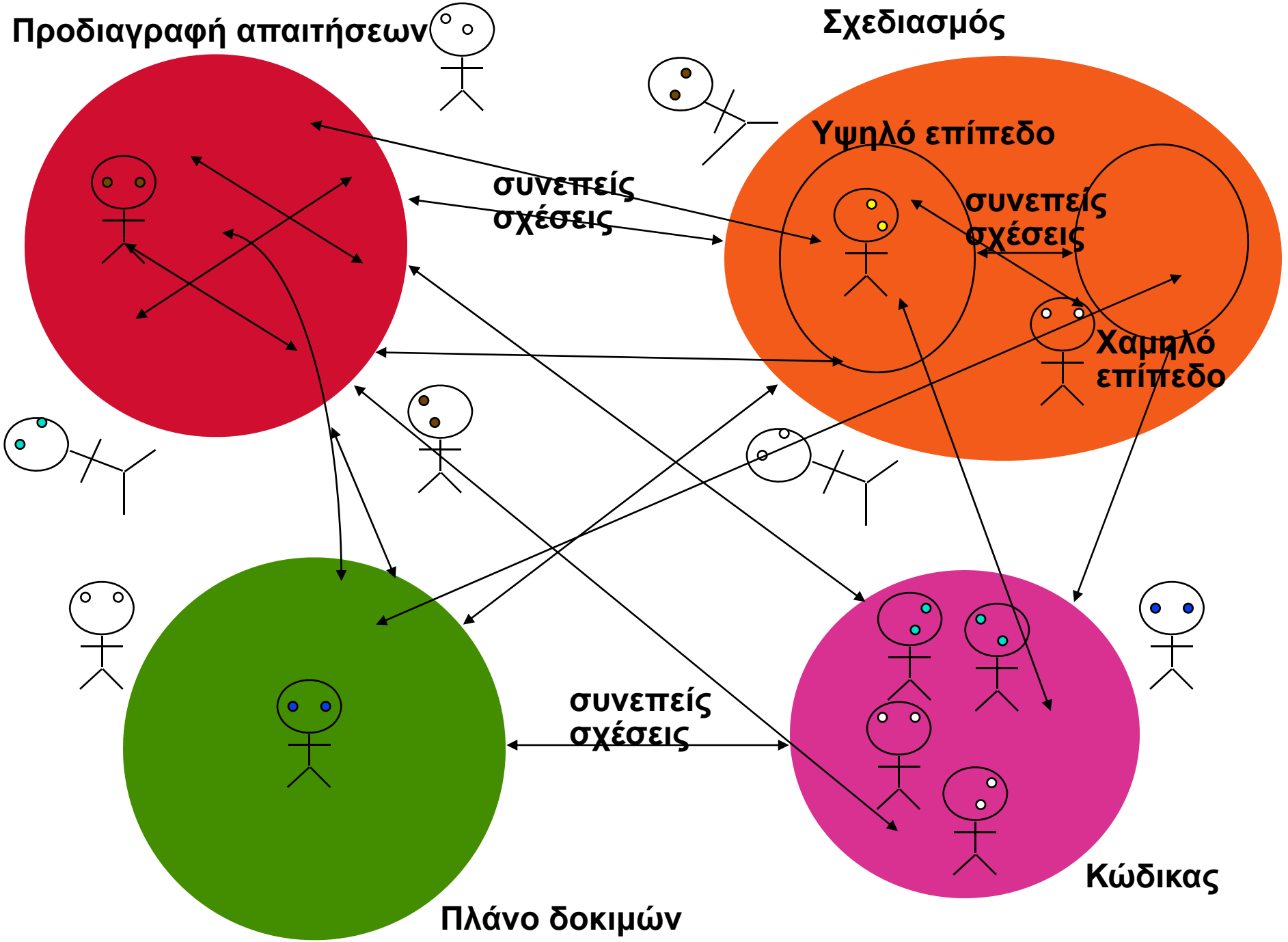
- Πώς να ικανοποιήσουμε όλες τις απαιτήσεις
- Με μικρό κόστος
- Σε αποδεκτό χρονικό διάστημα
- ... και να είμαστε σίγουροι ότι το κάναμε

Τα κυριότερα παραδοτέα/προϊόντα μιας διεργασίας ανάπτυξης λογισμικού

- Άρθρωση του προβλήματος, κατανόηση (προδιαγραφές απαιτήσεων)
  - Ποιο πρόβλημα λύνουμε;
- Μορφοποίηση της λύσης (αρχιτεκτονική)
  - Πώς ίσως λύνεται το πρόβλημα
- Αναγωγή της λύσης στην πράξη (σχεδίαση)
  - Πώς θα λύσουμε πράγματι το πρόβλημα;
- Υλοποίηση της λύσης (συγγραφή κώδικα)
- Σχέσεις μεταξύ των παραπάνω
- Αποδείξεις συνέπειας των παραπάνω (ανάλυση/δοκιμασία - testing)

Σχέσεις μεταξύ τμημάτων/εξαρτημάτων

- Π.χ.
  - Τα αποτελέσματα δοκιμασιών είναι αυτά που περιμένουμε
  - Ο κώδικας υλοποιεί το σχεδιασμό
  - Οι δοκιμασίες καλύπτουν τις απαιτήσεις χρήσης
- Αυτές οι σχέσεις ορίζονται νωρίς σε ένα έργο λογισμικού
  - αυτές και μόνο ορίζουν πότε ένα προϊόν είναι «σωστό»
- Οι συμμετέχοντες βλέπουν την επιβεβαίωση των σχέσεων μεταξύ τμημάτων του λογισμικού

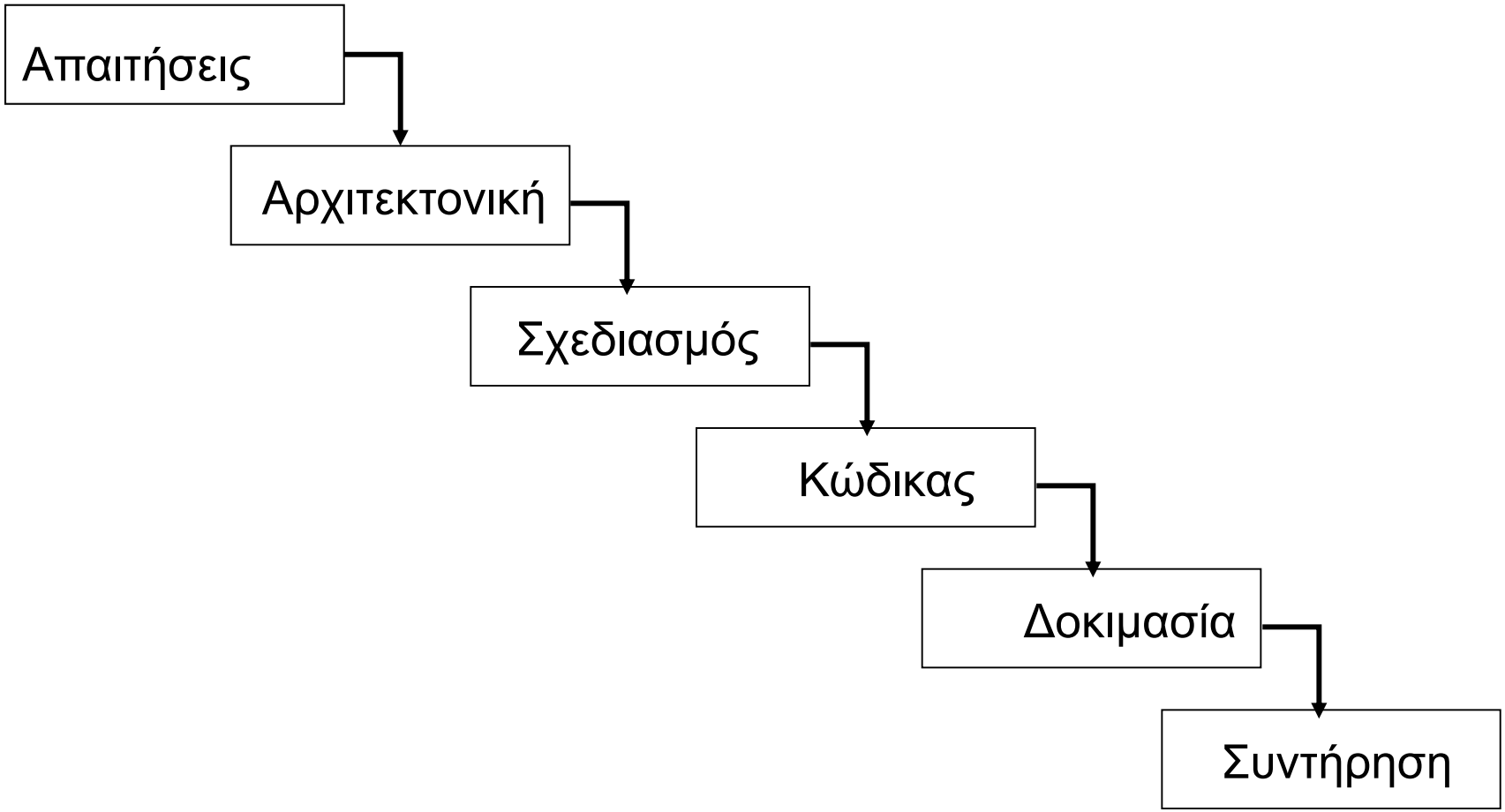

Προδιαγραφή απαιτήσεων
Σχεδιασμός
Υψηλό επίπεδο
συνεπείς σχέσεις
συνεπείς σχέσεις
Χαμηλό επίπεδο
συνεπείς σχέσεις
Πλάνο δοκιμών
Κώδικας

Βασική έμφαση της τεχνολογίας λογισμικού

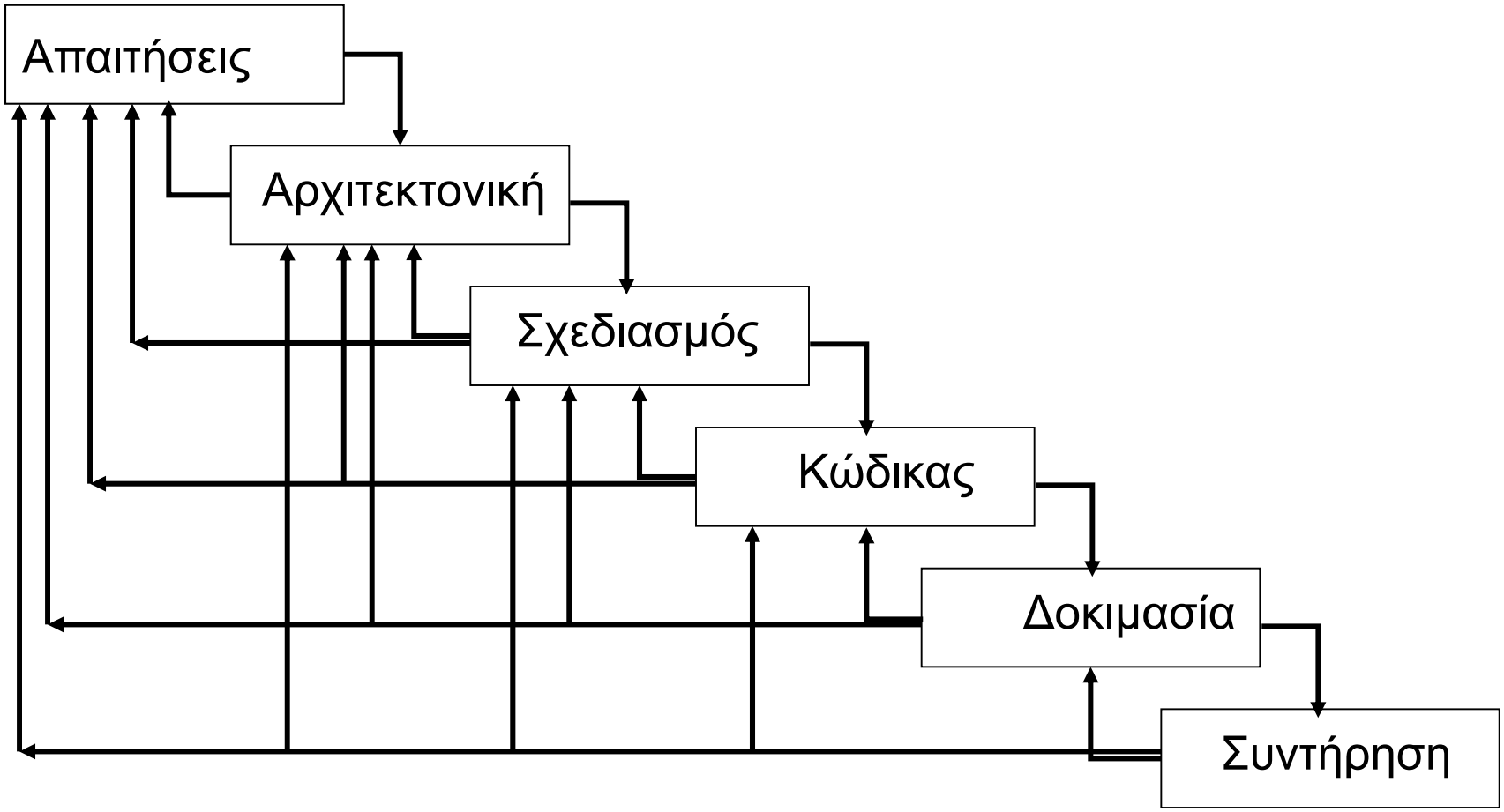
- **Πώς να περιγράψουμε προϊόντα;**
  - **τα συστατικά τους**
  - **τη σχέση και αλληλεπίδρασή τους**
- **Ποιες διαδικασίες πρέπει να ακολουθηθούν για να αναπτύξουμε τέτοια προϊόντα;**
  - **και να εξασφαλίσουμε την «ποιότητά» τους εν τέλει;**
  - **(παραδείγματα διαδικασίας;)**
- **Πώς να αναπτύξουμε και να εξελίξουμε προϊόντα με:**
  - **αποδεκτό κόστος**
  - **βελτιωμένη ποιότητα**

Τεχνολογία λογισμικού = Προϊόντα + Διαδικασίες

Η διαδικασία «καταράκτη»



Πιο ακριβής διαδικασία καταράκτη



Βασική αρχή του μαθήματος

- Όλες οι στρατηγικές ανάπτυξης πρέπει να επιλεγούν με γνώμονα
  - τη δοκιμασία και επαλήθευση
  - τη δυνατότητα συντήρησης του λογισμικού

Προδιαγραφές Απαιτήσεων

Γιατί προδιαγράφουμε απαιτήσεις;

- Αποσαφηνίζονται οι ανάγκες πριν αρχίσει η σχεδίαση
  - ο πελάτης ξέρει τι θέλει αλλά συνήθως δεν ξέρει τι είναι εφικτό
- Αποσαφηνίζεται η λειτουργικότητα που θα παρέχεται
- Αποσαφηνίζεται ο χειρισμός ανεπιθύμητων περιστάσεων
- Προδιαγράφονται μη-λειτουργικοί και περιβαλλοντικοί περιορισμοί που πρέπει να ικανοποιηθούν
- Οι προδιαγραφές καθοδηγούν προγραμματιστές, δοκιμαστές, συντηρητές, κλπ.
- Τεκμηριώνεται η πιθανή χρήση και οι χρήστες
- ...

Μια επιτυχημένη προδιαγραφή απαιτήσεων είναι ένα ΣΥΜΒΟΛΑΙΟ που περιγράφει αποτελεσματικά ΤΙ θέλουν όλοι οι συμμέτοχοι του έργου λογισμικού

Διάφορα στοιχεία προδιαγραφών απαιτήσεων

Παράδειγμα:

- **Εισαγωγή**
- **Λειτουργικές απαιτήσεις**
- **Περιβάλλον**
- **Επίδοση**
- **Ακρίβεια**
- **Χειρισμός λαθών**
- **Εξωτερική ασφάλεια (security)**
- **Εσωτερική ασφάλεια (safety)**