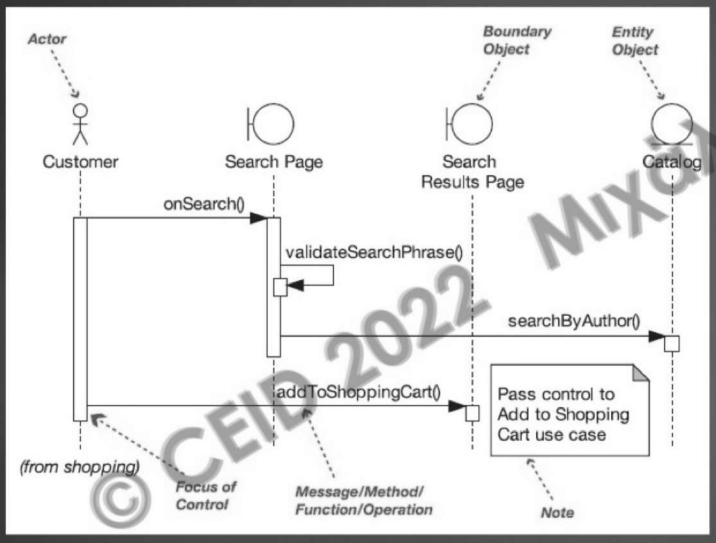
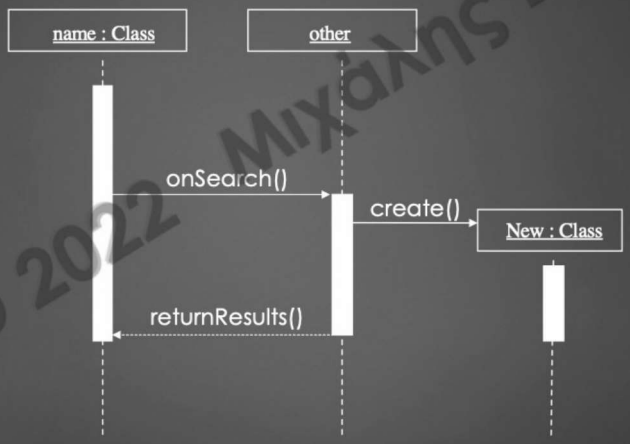


Sequence Diagrams

Εισαγωγή στα Διαγράμματα Ακολουθίας

- Τα διαγράμματα ακολουθίας (sequence diagrams) παρουσιάζουν την αλληλεπίδραση των αντικειμένων μέσω της ανταλλαγής μηνυμάτων
- Δίνουν έμφαση στη χρονική αλληλουχία των γεγονότων
- Ένα διάγραμμα ακολουθίας περιέχει
  - **Χειριστές**
  - **Αντικείμενα** και
  - **Μηνύματα** που ανταλλάσσουν τα αντικείμενα
- Τα αντικείμενα οργανώνονται στον οριζόντιο άξονα και τα γεγονότα στον κάθετο άξονα του χρόνου
- Τα διαγράμματα ακολουθίας χρησιμοποιούνται για να περιγράψουν τη ροή του ελέγχου μέσα στο σύστημα

Δομή Διαγράμματος Ακολουθίας 7/7

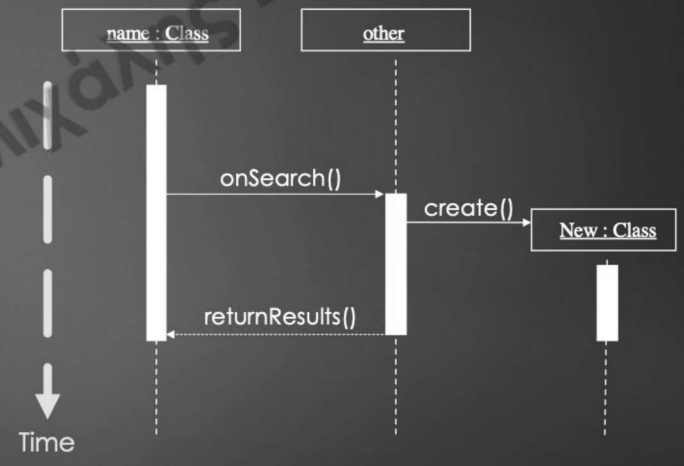


Τα Διαγράμματα Ακολουθίας

Δείχνουν τις αλληλεπιδράσεις ανάμεσα στα αντικείμενα για μια συγκεκριμένη ροή μιας περίπτωσης χρήσης.

Δείχνουν ποια μηνύματα ανταλλάσσονται ανάμεσα σε αντικείμενα αλλά και από και προς εξωτερικούς actors.

Δείχνουν τη χρονική ακολουθία (η έννοια του χρόνου εκφράζεται από πάνω προς τα κάτω στο lifeline κάθε αντικειμένου)



Τα μηνύματα

Σύγχρονα: (γεγονότα, κλήσεις μεθόδων)

- Η διάρκεια αναπαρίσταται με τη γραμμή ενεργοποίησης του αντικειμένου (activation bar)

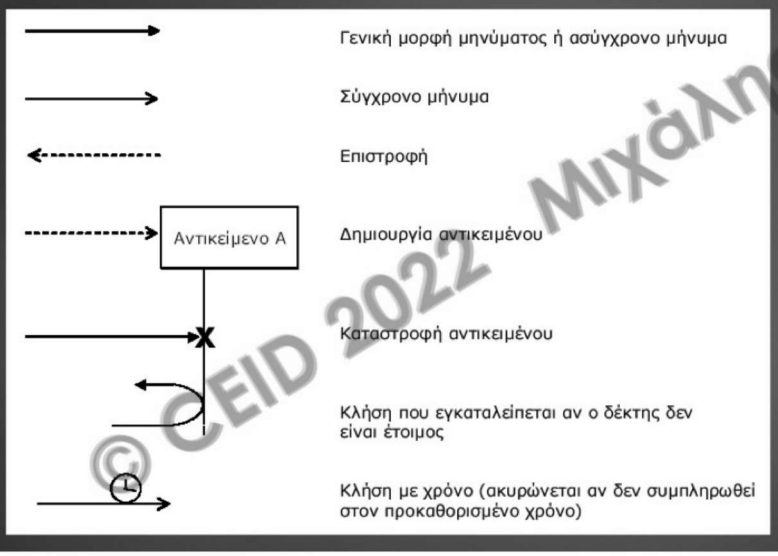
Ασύγχρονα (asynchronous):

- Δεν μπλοκάρουν το αντικείμενο που τα στέλνει (μπορεί να συνεχίσει να κάνει κάτι άλλο)
- Χρησιμοποιούνται συνήθως σε τρεις περιπτώσεις :
  - Να δημιουργήσουν ένα καινούργιο αντικείμενο
  - Να δημιουργήσουν ένα καινούργιο νήμα επεξεργασίας (thread)
  - Να επικοινωνήσουν με ένα υπάρχον νήμα επεξεργασίας που ήδη τρέχει.

Υπάρχουν και άλλα μηνύματα ειδικού τύπου

- «create»: δημιουργία αντικειμένου
- «destroy»: καταστροφή αντικειμένου

Συμβολισμοί για τα μηνύματα

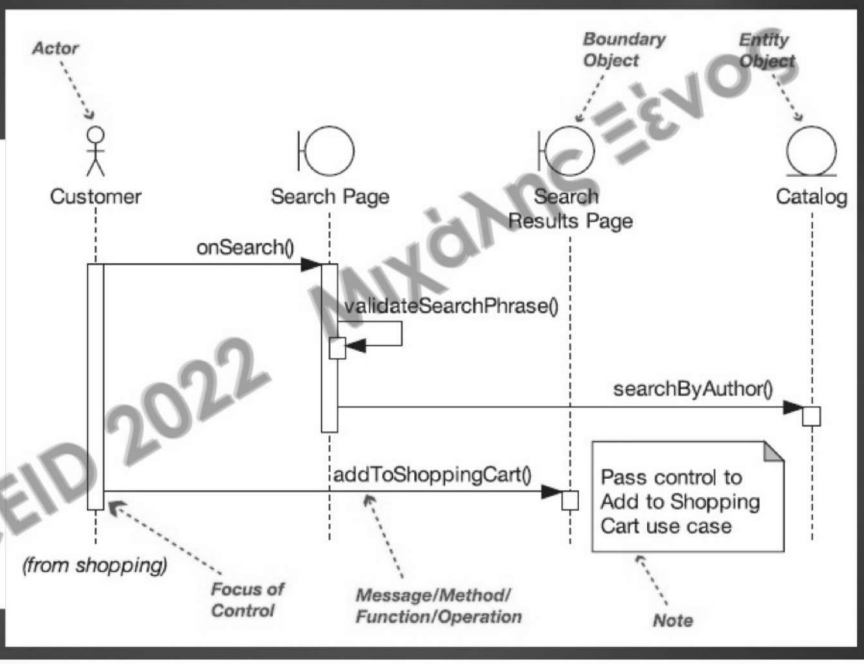


Πώς δουλεύουμε (από RD σε SD)
(επαναλαμβάνουμε για κάθε περίπτωση χρήσης)

- Αντιγράφουμε στο αριστερό μέρος του διαγράμματος το κείμενο της περίπτωσης χρήσης
- Προσθέτουμε σε μια γραμμή στην κορυφή τα αντικείμενα οντοτήτων (από το μοντέλο robustness) και ενημερώνουμε το domain model με τις κλάσεις που λείπουν
- Προσθέτουμε επίσης στην κορυφή τους χειριστές και τα συνοριακά αντικείμενα
- **Καταχωρούμε μεθόδους σε κλάσεις**
  - Μετατρέπουμε κάθε αντικείμενο ελέγχου σε ένα σύνολο μεθόδων και μηνυμάτων που υλοποιούν τη συμπεριφορά του
- Εμπλουτίζουμε το διάγραμμα κλάσεων (domain model) με τις νέες μεθόδους
- Μορφοποιούμε το διάγραμμα ακολουθίας (και ίσως σβήνουμε τη λεκτική περιγραφή)

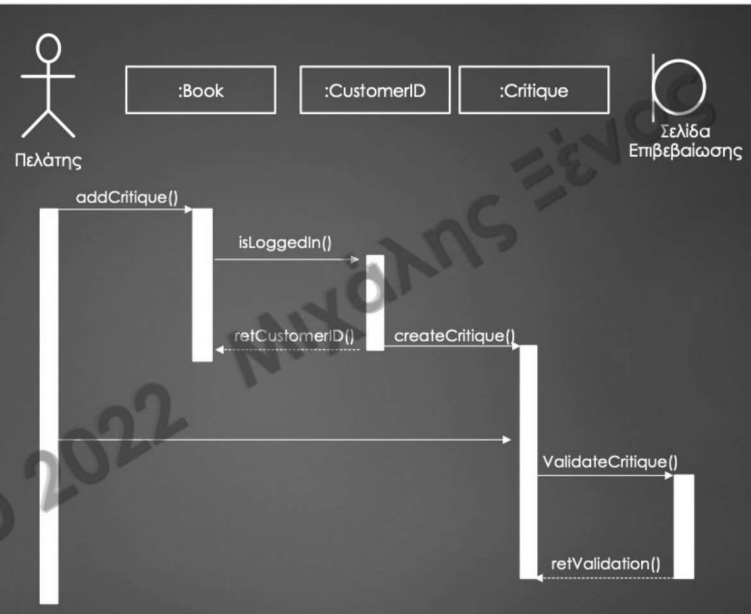
Λεκτική Περιγραφή της ροής

Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο
Κείμενο



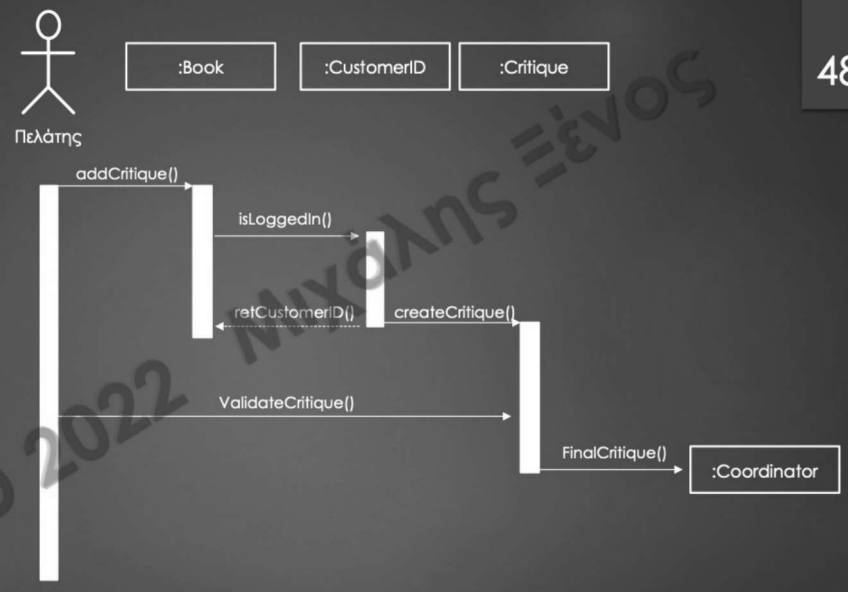
47

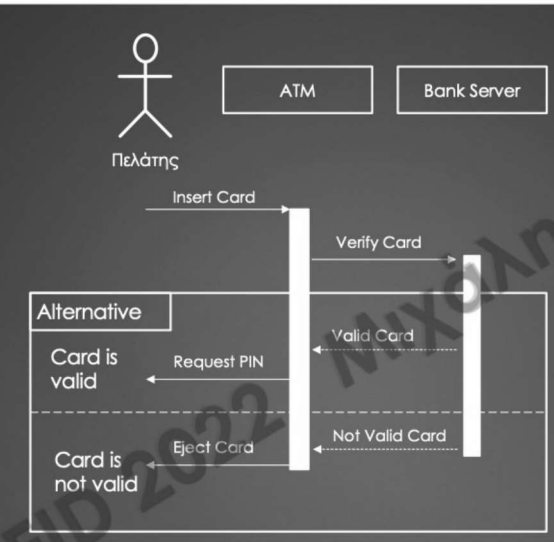
Στη σελίδα «Λεπτομέρειες Βιβλίου» ο Πελάτης επιλέγει το πλήκτρο «Συγγραφή Κριτικής». Το σύστημα ελέγχει τη συνεδρία πελάτη για να επιβεβαιώσει ότι ο πελάτης έχει κάνει login και στη συνέχεια εμφανίζει τη σελίδα «Συγγραφή Κριτικής».
Ο Πελάτης εισάγει μια κριτική βιβλίου, δίνει μια βαθμολογία από 0 έως 5 άστρα και επιλέγει το πλήκτρο «Αποστολή».
Το σύστημα επιβεβαιώνει ότι η κριτική δεν είναι μεγάλη μικρή και ότι η βαθμολογία είναι μεταξύ 0 και 5 άστρων. Στη συνέχεια εμφανίζει μια σελίδα επιβεβαίωσης και σε περίπτωση επιβεβαίωσης η κριτική αποστέλλεται στο Συντονιστή.



48

Στη σελίδα «Λεπτομέρειες Βιβλίου» ο Πελάτης επιλέγει το πλήκτρο «Συγγραφή Κριτικής». Το σύστημα ελέγχει τη συνεδρία πελάτη για να επιβεβαιώσει ότι ο πελάτης έχει κάνει login και στη συνέχεια εμφανίζει τη σελίδα «Συγγραφή Κριτικής».
Ο Πελάτης εισάγει μια κριτική βιβλίου, δίνει μια βαθμολογία από 0 έως 5 άστρα και επιλέγει το πλήκτρο «Αποστολή».
Το σύστημα επιβεβαιώνει ότι η κριτική δεν είναι μεγάλη μικρή και ότι η βαθμολογία είναι μεταξύ 0 και 5 άστρων. Στη συνέχεια εμφανίζει μια σελίδα επιβεβαίωσης και σε περίπτωση επιβεβαίωσης η κριτική αποστέλλεται στο Συντονιστή.

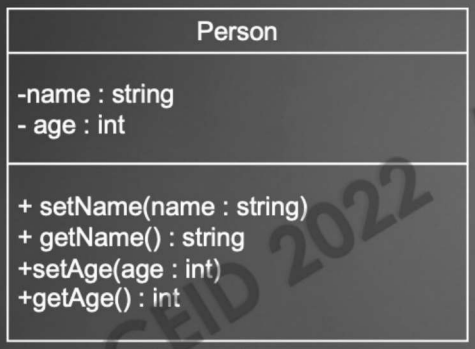




Με τον ίδιο τρόπο περιγράφουμε και επαναλήψεις

Συχνά λάθη σε αυτή τη φάση…

- Ξεχνάμε το use case! (για αυτό και προτείνω να έχετε την περιγραφή αρχικά δίπλα στο sequence diagram)
- Δεν έχουμε εντοπίσει όλα τα σχετικά αντικείμενα από το robustness diagram
- Ασχολείστε με getters και setters και όχι με την πραγματική συμπεριφορά του συστήματος
- Δεν έχετε τη σωστή φορά των βέλων
- Ξεχνάτε την έννοια της «προσωπικότητας» κάθε κλάσης (αντικειμένου) όταν σχεδιάζετε μηνύματα (άρα και μεθόδους)
- Ξεχνάτε να περάσετε όλη την πληροφορία στο class diagram



Περισσότερα για τις κλάσεις

Κλάσεις

- Οι κλάσεις ενσωματώνουν τα δεδομένα (τιμές σε ιδιότητες) καθώς και τις λειτουργίες (μεθόδους) που επενεργούν στα δεδομένα
- Συνήθως οι ιδιότητες μια κλάσης είναι ιδιωτικές (private) και οι μέθοδοι δημόσιες (public)
  - *Σημειολογία:* τα αντικείμενα άλλων κλάσεων δεν έχουν τη δυνατότητα να προσπελάσουν απευθείας τις ιδιότητες μια κλάσης, παρά μόνο καλώντας τις μεθόδους της κλάσης
- Αρχή απόκρυψης των δεδομένων (information hiding)
  - Κέρδος: η εσωτερική αναπαράσταση των δεδομένων μπορεί να αλλάξει κατά βούληση χωρίς αυτό να επηρεάσει τους χρήστες των αντικειμένων της κλάσης εφόσον η δημόσια διασύνδεση παραμένει σταθερή

Συντακτικό UML: Δήλωση ιδιοτήτων κλάσης

Προσδιοριστής-πρόσβασης όνομα-ιδιότητας : Τύπος [πολλαπλότητα διάταξη] = Αρχική-τιμή {συμβολοσειρά ιδιοτήτων}

Προσδιοριστής-πρόσβασης
+(public), -(private), #(εμβέλεια κλάσης), ~(εμβέλεια πακέτου)
Τύπος
String, boolean, int, κτλ.
πολλαπλότητα
Διάστημα τιμών (1..5) ή (1..*) ή και αριθμός
διάταξη
Ordered/unordered
{συμβολοσειρά ιδιοτήτων}
Π.χ. frozen (αφού πάρει μια αρχική τιμή μετά δε μεταβάλλεται)

- students : string[* ordered]

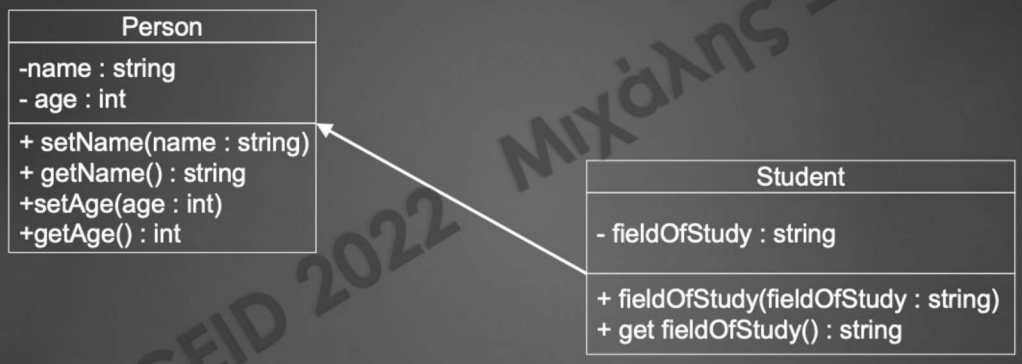
- interestRate : double = 0.035% {frozen}

Συντακτικό UML: Δήλωση μεθόδων κλάσης

Προσδιοριστής-πρόσβασης όνομα-μεθόδου (λίστα-παραμέτρων) : Τύπος-επιστροφής {συμβολοσειρά ιδιοτήτων}

Προσδιοριστής-πρόσβασης
+(public), -(private), #(εμβέλεια κλάσης), ~(εμβέλεια πακέτου)
Λίστα-παραμέτρων
[in/out/inout] όνομα-παραμέτρου : τύπος
withdraw(in amount:Money)
Τύπος
String, boolean, int, κτλ.
{συμβολοσειρά ιδιοτήτων}
Π.χ. query (πρόκειται για ερώτημα και δε μεταβάλει την κατάσταση του αντικειμένου όταν καλείται)

Συμβολισμοί UML για κλάσεις



Κριτήρια κατανομής μεθόδων σε κλάσεις

- Επαναχρησιμοποίηση
  - Όσο πιο γενική είναι μια κλάση τόσο που εύκολα μπορούμε να την επαναχρησιμοποιήσουμε. Κατά πόσο η νέα μέθοδος θα επηρεάσει την επαναχρησιμοποίηση της κλάσης;
- Εφαρμοσιμότητα
  - Ταιριάζει η μέθοδος με τα δεδομένα της κλάσης και με τις υπόλοιπες μεθόδους της;
- Πολυπλοκότητα
  - Τι βαθμό δυσκολίας έχει η υλοποίηση της μεθόδου στη συγκεκριμένη κλάση;
- Στοχεύουμε σε
  - υψηλή επαναχρησιμοποίηση και εφαρμοσιμότητα
  - χαμηλή πολυπλοκότητα

Πότε μια κλάση είναι καλή;

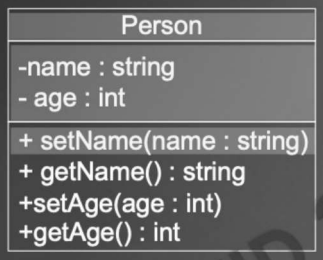
- Η ανάθεση λειτουργιών σε μια κλάση ισοδυναμεί με ανάθεση "προσωπικότητας"
- Dan Rawsthorne:
- *"Αν θεωρήσετε τα αντικείμενά σας (κλάσεις) ως ανθρώπους, τότε το σύνολο των λειτουργιών που τους έχουν ανατεθεί αποτελούν ένα είδος προσωπικότητας.*

  Πρέπει να προσέχουμε για την ύπαρξη σχιζοφρενικών αντικειμένων (αντικειμένων με πολλαπλές προσωπικότητες), καθώς ένα αντικείμενο πρέπει να εστιάζει σε ένα συνεκτικό σύνολο λειτουργιών"

Κριτήρια ποιότητας των κλάσεων

- **Σύζευξη (coupling)**
  - Στοχεύουμε χαλαρή σύζευξη, δηλ. σε κλάσεις κατά το δυνατό ανεξάρτητες μεταξύ τους
- **Συνοχή (cohesion)**
  - Στοχεύουμε σε υψηλό βαθμό λειτουργικής συνοχής ανάμεσα στα κατηγορήματα και τις μεθόδους μιας κλάσης. Αποφεύγουμε να σχεδιάσουμε κλάσεις με «διχασμένη προσωπικότητα».
- Επάρκεια (sufficiency)
  - Κάθε κλάση πρέπει να ενθυλακώνει αρκετά αντικείμενα του πεδίου εφαρμογής ώστε να αποτελεί ένα αυτόνομο και λογικά πλήρες τμήμα του συστήματος
- Πληρότητα (completeness)
  - Κάθε κλάση θα πρέπει αν έχει υψηλό βαθμό πληρότητας ώστε να είναι επαναχρησιμοποιήσιμη σε παρόμοια πεδία εφαρμογής.
- Πρωτογένεια (primitiveness)
  - Σχεδιάζουμε ορισμένες βασικές μεθόδους και τις χρησιμοποιούμε για να χτίσουμε τις πιο πολύπλοκες

Από το class diagram στον κώδικα




```
public class person {

    private string name;
    private int age;

    public void setName (string name) {

    }

    κτλ...


}
```


Το Git του έργου σας πρέπει…

- Να έχει ήδη μέσα τις κλάσεις που έχει εντοπίσει ο καθένας σας
- Να εμπλουτίζεται σε κάθε βήμα (**όχι μόνο σε κάθε παράδοση**)
- Να απεικονίζει τη συνεργατικότητα (π.χ. Ο Χ έβαλε κάτι, ο Υ έκανε μια διόρθωση και δημιούργησε ένα branch, ο Χ επέλεξε το branch ως βασικό, ο Ζ έβαλε κάτι άλλο, κτλ.)
- Αν στην τελική παράδοση κάποιος μπει μια μέρα πριν και βάλει τα πάντα, δεν είναι συνεργατικό project!