Πανεπιστήμιο Αιγαίου

Εισαγωγή στην Επιστήμη των Υπολογιστών και Επικοινωνιών

Εισαγωγή στην τεχνολογία λογισμικού

Σπύρος Κοκολάκης (sak@aegean.gr)

\

Τμήμα Μηχανικών Πληροφοριακών & Επικοινωνιακών Συστημάτων

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αιγαίου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Κύκλος ζωής λογισμικού

Μια θεμελιώδης έννοια **στην τεχνολογία λογισμικού** είναι ο **κύκλος ζωής λογισμικού**.

Το λογισμικό, όπως και πολλά άλλα προϊόντα, περνά από έναν κύκλο επαναλαμβανόμενων φάσεων.

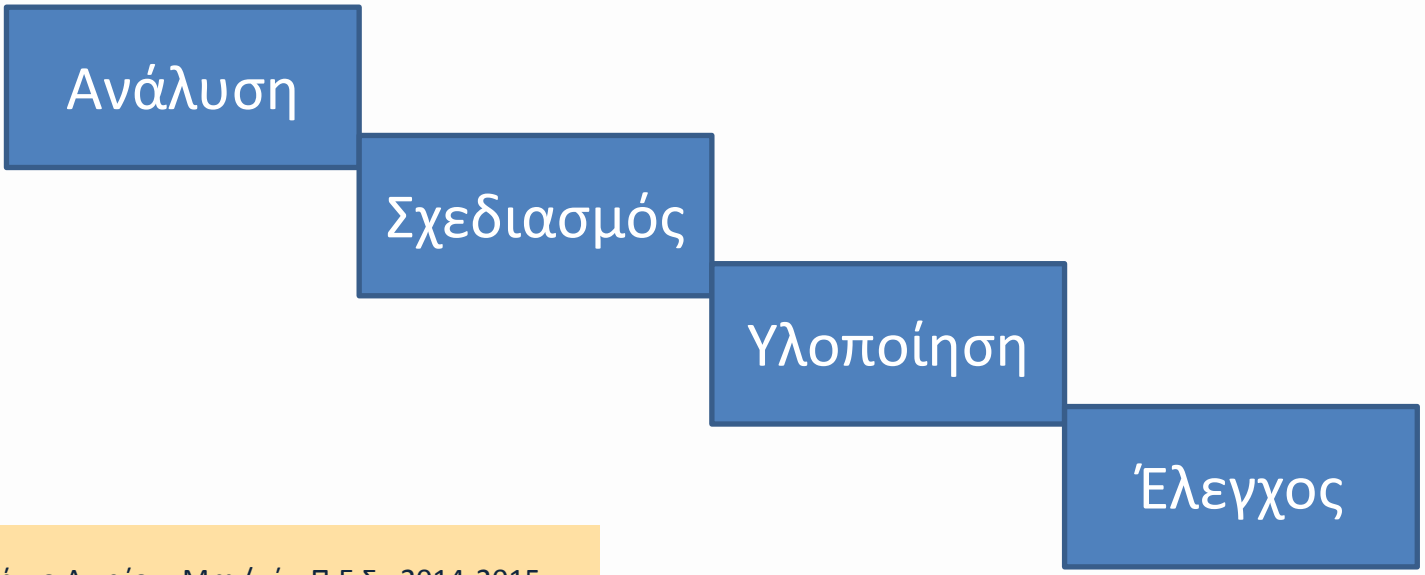
Μοντέλα διαδικασιών ανάπτυξης

Η διαδικασία της ανάπτυξης στον κύκλο ζωής λογισμικού περιλαμβάνει τέσσερις φάσεις: **ανάλυση**, **σχεδιασμό**, **υλοποίηση**, και **έλεγχο**.

Υπάρχουν πολλά μοντέλα για τη διαδικασία της ανάπτυξης. Εμείς θα περιγράψουμε τα δύο πιο συνηθισμένα: το **μοντέλο καταρράκτη** και το **αυξητικό μοντέλο**.

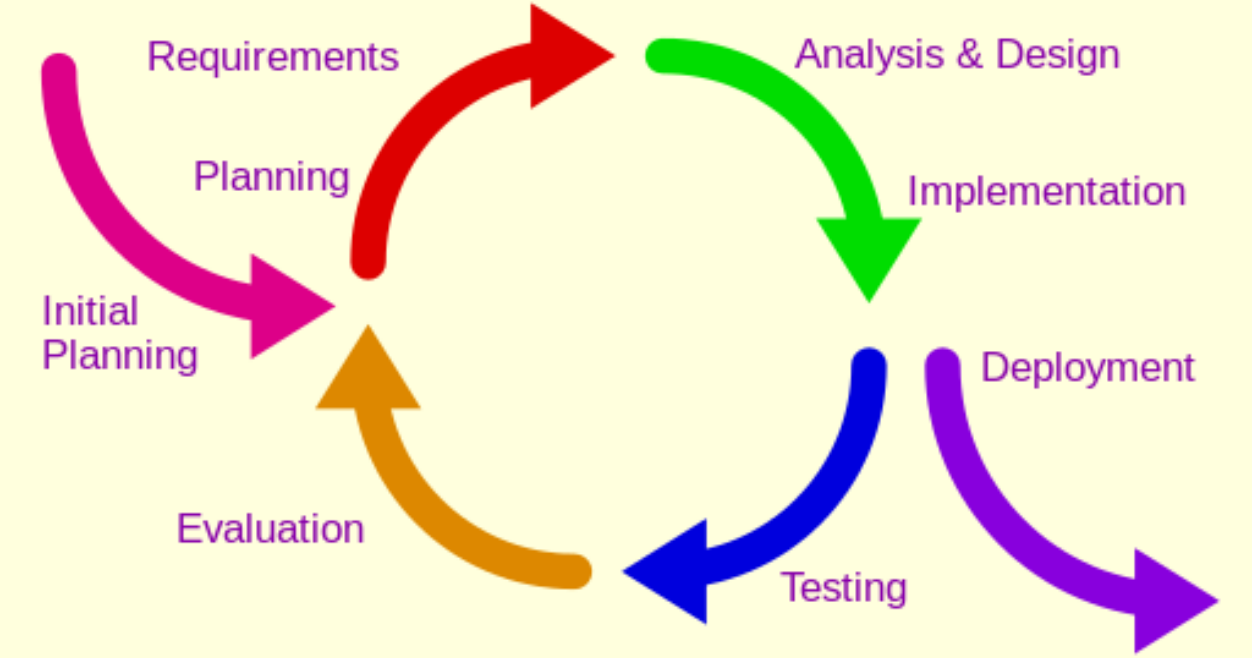
Το μοντέλο καταρράκτη

Στο μοντέλο αυτό η διαδικασία ανάπτυξης ρέει προς μία μόνο κατεύθυνση. Αυτό σημαίνει ότι μια φάση δεν μπορεί να ξεκινήσει αν πρώτα δεν ολοκληρωθεί η προηγούμενή της.



Το επαυξητικό μοντέλο

Στο επαυξητικό μοντέλο το λογισμικό αναπτύσσεται σε κύκλους ανάπτυξης.



Φάση ανάλυσης

Η διαδικασία της ανάπτυξης ξεκινά με τη φάση ανάλυσης. Από τη φάση αυτή προκύπτει ένα έγγραφο προδιαγραφών που παρουσιάζει τι θα κάνει το λογισμικό χωρίς να καθορίζεται το πώς θα το επιτυγχάνει. Στη φάση της ανάλυσης μπορούν να χρησιμοποιηθούν δύο ξεχωριστές προσεγγίσεις, η **διαδικασιακή** και η **αντικειμενοστραφής**.

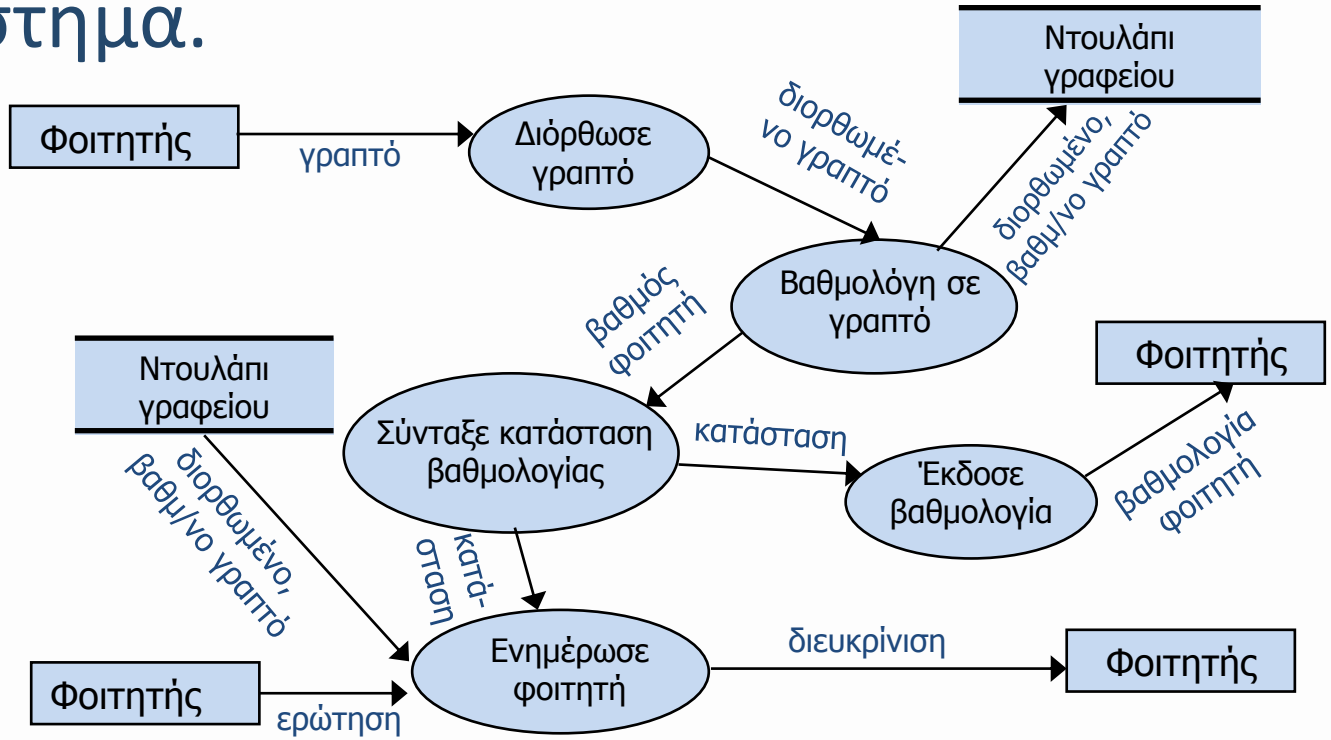
Διαδικασιακή ανάλυση

Η διαδικασιακή ανάλυση, η οποία ονομάζεται και δομημένη ανάλυση ή κλασική ανάλυση, είναι η διαδικασία ανάλυσης που εφαρμόζεται όταν στη φάση υλοποίησης του συστήματος πρόκειται να χρησιμοποιηθεί μια διαδικασιακή γλώσσα προγραμματισμού.

Σε αυτή την περίπτωση, οι προδιαγραφές μπορεί να επιτρέπουν τη χρήση πολλών εργαλείων μοντελοποίησης, όμως εδώ θα περιγράψουμε μόνο μερικά από αυτά.

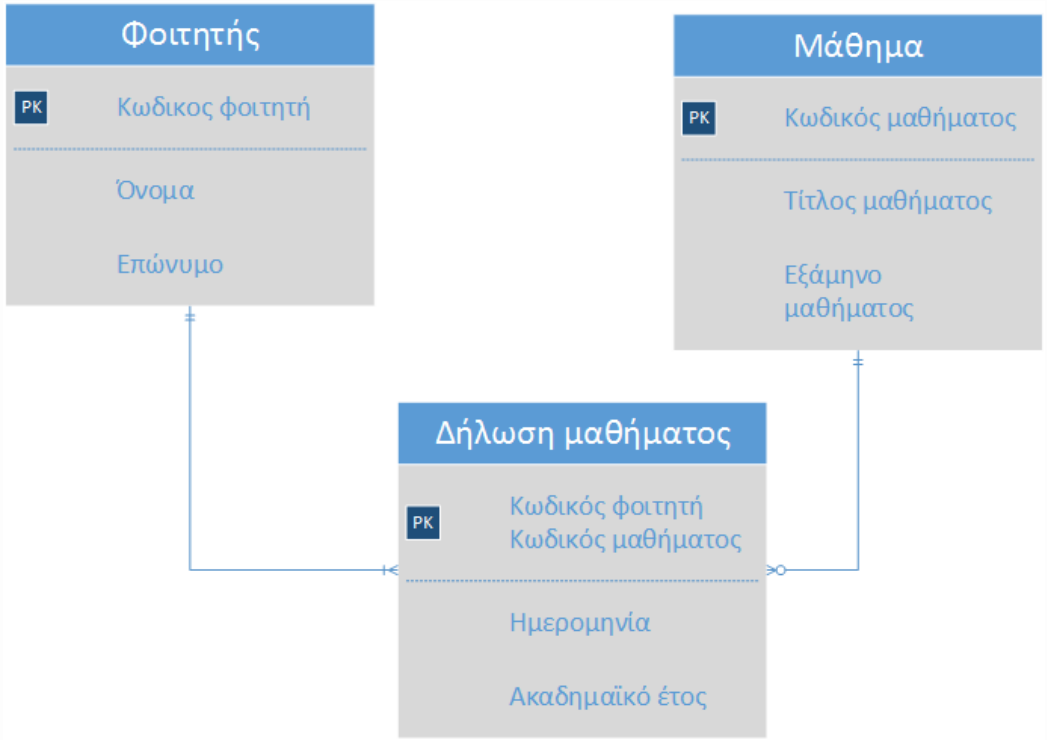
Διαγράμματα ροής δεδομένων

Τα διαγράμματα ροής δεδομένων παρουσιάζουν την κίνηση των δεδομένων μέσα στο σύστημα.



Διαγράμματα οντοτήτων-σχέσεων

Ένα άλλο εργαλείο μοντελοποίησης που χρησιμοποιείται κατά τη φάση της ανάλυσης είναι το διάγραμμα οντοτήτων-σχέσεων.



Αντικειμενοστρεφής ανάλυση

Η αντικειμενοστρεφής ανάλυση είναι η διαδικασία ανάλυσης που χρησιμοποιείται όταν κατά την υλοποίηση πρόκειται να χρησιμοποιηθεί μια αντικειμενοστρεφής γλώσσα.

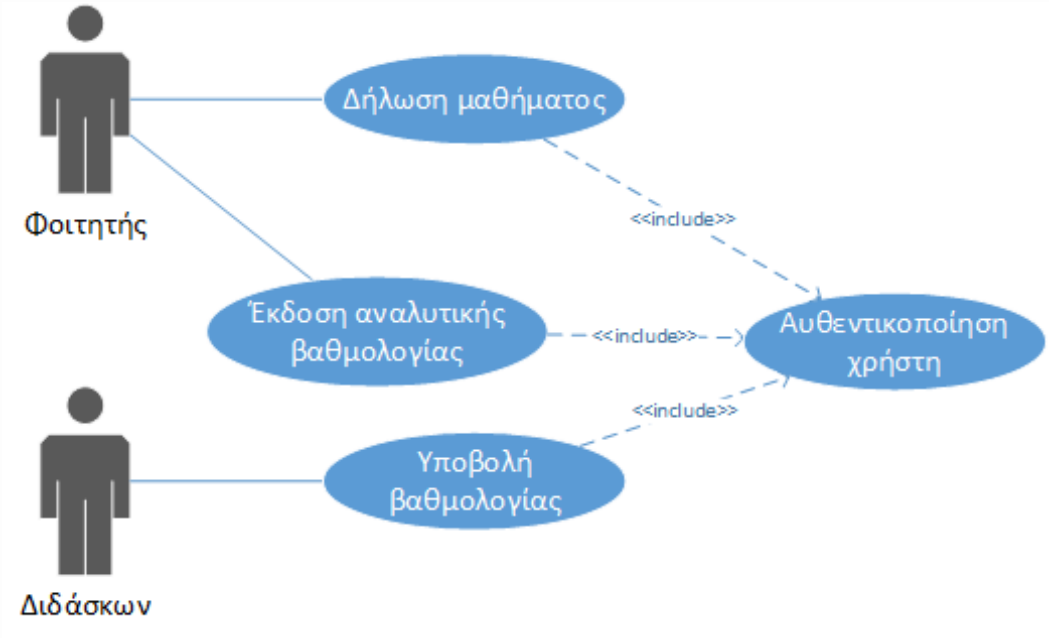
Σε αυτή την περίπτωση, το έγγραφο προδιαγραφών μπορεί να επιτρέπει τη χρήση πολλών εργαλείων, όμως εδώ θα περιγράψουμε μόνο μερικά από αυτά.

Διαγράμματα περιπτώσεων χρήσης

Το διάγραμμα περιπτώσεων χρήσης παρουσιάζει ένα σύστημα από την οπτική γωνία των χρηστών: δηλαδή, δείχνει τον τρόπο με τον οποίο οι χρήστες επικοινωνούν με το σύστημα.

Διαγράμματα περιπτώσεων χρήσης

Ένα διάγραμμα περιπτώσεων χρήσης αποτελείται από τέσσερα συστατικά στοιχεία: το σύστημα, τις περιπτώσεις χρήσης, τους δρώντες, και τις σχέσεις.



Διαγράμματα κλάσεων

Στην αντικειμενοστραφή ανάλυση και σχεδίαση χρησιμοποιούμε εκτενώς τα διαγράμματα κλάσεων.

Φάση σχεδιασμού

Η φάση σχεδιασμού δείχνει πώς θα πραγματοποιήσει το σύστημα αυτά που έχουν καθοριστεί στη φάση της ανάλυσης.

Στη φάση του σχεδιασμού καθορίζονται όλα τα στοιχεία του συστήματος.

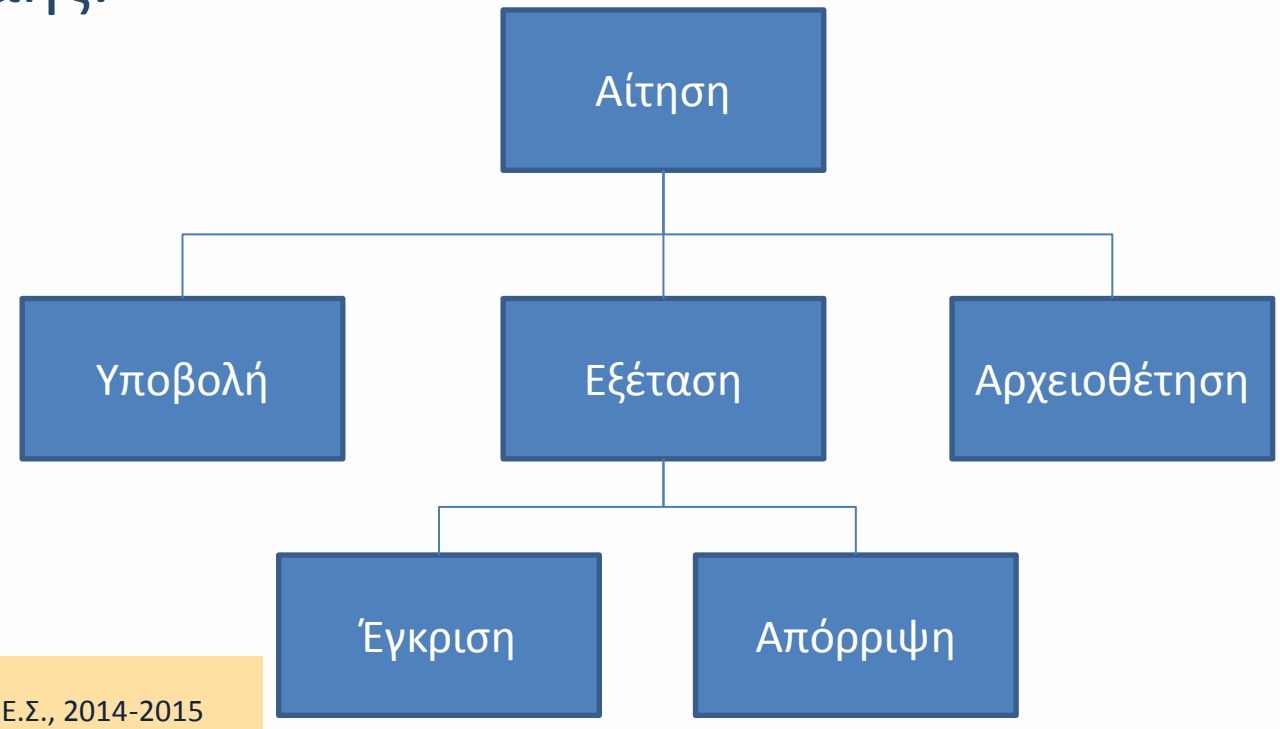
Διαδικασιακός σχεδιασμός

Στο **διαδικασιακό σχεδιασμό** πρέπει να σχεδιαστούν και οι διαδικασίες και τα δεδομένα. Εμείς θα περιγράψουμε μια κατηγορία μεθόδων σχεδιασμού που επικεντρώνονται στις διαδικασίες.

Στο διαδικασιακό σχεδιασμό, ολόκληρο το σύστημα χωρίζεται σε ένα σύνολο διαδικασιών ή υπομονάδων.

Διαγράμματα δομής

Ένα συνηθισμένο εργαλείο που χρησιμοποιείται για την απεικόνιση των σχέσεων μεταξύ των υπομονάδων στον διαδικασιακό σχεδιασμό είναι το διάγραμμα δομής.



Τμηματικότητα

Η τμηματικότητα αναφέρεται στη διάσπαση ενός μεγάλου έργου σε μικρότερα τμήματα τα οποία είναι πιο και πιο εύκολα στον χειρισμό τους.

Η τμηματικότητα είναι ο χωρισμός ενός συστήματος σε τμήματα που αλληλεπιδρούν μεταξύ τους.

Το διάγραμμα δομής που περιγράψαμε στην προηγούμενη ενότητα παρουσιάζει την έννοια της τμηματικότητας στο σύστημα διαχείρισης αιτήσεων.

Όταν ένα σύστημα χωρίζεται σε υπομονάδες μας απασχολούν δύο ζητήματα: η **σύζευξη** και η **συνεκτικότητα**.

Σύζευξη και συνεκτικότητα

Σύζευξη είναι ένα μέτρο που δείχνει το πόσο στενά δεσμευμένες είναι δύο υπομονάδες μεταξύ τους.

- Η σύζευξη μεταξύ των υπομονάδων σε ένα σύστημα λογισμικού πρέπει να περιορίζεται στο ελάχιστο.

Η συνεκτικότητα είναι ένα μέτρο του πόσο στενά είναι συνδεδεμένες μεταξύ τους οι υπομονάδες ενός συστήματος.

- Σε ένα σύστημα λογισμικού, η συνεκτικότητα μεταξύ των υπομονάδων πρέπει να είναι η μέγιστη δυνατή.

Αντικειμενοστρεφής σχεδιασμός

Στον αντικειμενοστρεφή σχεδιασμό, η φάση του σχεδιασμού συνεχίζεται με την ανάλυση των λεπτομερειών των κλάσεων.

Μια κλάση αποτελείται από ένα σύνολο μεταβλητών (ιδιοτήτων) και ένα σύνολο μεθόδων. Στη φάση του αντικειμενοστρεφούς σχεδιασμού καταγράφονται οι λεπτομέρειες αυτών των ιδιοτήτων και των μεθόδων.

Φάση υλοποίησης

Στο μοντέλο καταρράκτη, όταν ολοκληρωθεί η φάση του σχεδιασμού ξεκινά η φάση υλοποίησης. Σε αυτή τη φάση οι προγραμματιστές γράφουν τον κώδικα για τις υπομονάδες στον διαδικασιακό σχεδιασμό, ή γράφουν τις μονάδες του προγράμματος για την υλοποίηση των κλάσεων στον αντικειμενοστρεφή σχεδιασμό. Υπάρχουν αρκετά θέματα που πρέπει να αναφέρουμε για κάθε μία από αυτές τις περιπτώσεις.

Επιλογή γλώσσας

Στη διαδικασιακή ανάπτυξη, η ομάδα που ασχολείται με το έργο πρέπει να επιλέξει μια γλώσσα ή ένα σύνολο γλωσσών μεταξύ των διαδικασιακών γλωσσών προγραμματισμού.

Παρόλο που μερικές γλώσσες, όπως η C++, θεωρούνται και διαδικασιακές και αντικειμενοστρεφείς, συνήθως για την υλοποίηση χρησιμοποιείται μια καθαρά διαδικασιακή γλώσσα όπως η C.

Στην αντικειμενοστρεφή ανάπτυξη, δύο συνηθισμένες γλώσσες είναι η C++ και η Java.

Ποιότητα λογισμικού

Η ποιότητα του λογισμικού που δημιουργείται κατά τη φάση της υλοποίησης είναι πολύ σημαντικό θέμα. Ποιοτικό είναι το σύστημα λογισμικού που ικανοποιεί τις απαιτήσεις του χρήστη, καλύπτει τα λειτουργικά πρότυπα της εταιρείας, και εκτελείται αποτελεσματικά στο υλικό για το οποίο έχει αναπτυχθεί. Ωστόσο, αν θέλουμε να αναπτύξουμε ένα σύστημα λογισμικού υψηλής ποιότητας, θα πρέπει να είμαστε σε θέση να ορίσουμε ορισμένα χαρακτηριστικά ποιότητας.

Παράγοντες ποιότητας

Οι παράγοντες της ποιότητας λογισμικού χωρίζονται σε τρεις μεγάλες κατηγορίες: τη **λειτουργικότητα**, τη **συντηρησιμότητα**, και τη **μεταφερσιμότητα**.

Φάση ελέγχου

Ο σκοπός της φάσης ελέγχου είναι ο εντοπισμός σφαλμάτων, το οποίο επιτυγχάνεται με μια καλή στρατηγική ελέγχου. Υπάρχουν δύο τύποι ελέγχου: ο **έλεγχος γυάλινου κουτιού** και ο **έλεγχος μαύρου κουτιού**.

Έλεγχος γυάλινου κουτιού

Ο έλεγχος "γυάλινου κουτιού" (ή έλεγχος "λευκού κουτιού") προϋποθέτει γνώση της εσωτερικής δομής του λογισμικού. Ο έλεγχος γυάλινου κουτιού υποθέτει ότι ο ελεγκτής γνωρίζει τα πάντα σχετικά με το λογισμικό. Σε αυτή την περίπτωση, το πρόγραμμα είναι σαν ένα γυάλινο κουτί με όλα τα περιεχόμενα του εσωτερικού του ορατά. Ο έλεγχος γυάλινου κουτιού πραγματοποιείται από τον μηχανικό λογισμικού ή από μια ομάδα που ασχολείται αποκλειστικά με αυτή την εργασία.

Έλεγχος μαύρου κουτιού

Ο όρος έλεγχος μαύρου κουτιού (black box testing) αναφέρεται στον έλεγχο του λογισμικού χωρίς να γνωρίζουμε τι υπάρχει μέσα σε αυτό και πώς ακριβώς λειτουργεί. Με άλλα λόγια, το λογισμικό μοιάζει με ένα μαύρο κουτί στο εσωτερικό του οποίου δεν μπορεί να δει ο ελεγκτής. Με τη διαδικασία αυτή ελέγχεται η λειτουργικότητα του λογισμικού σε σχέση με αυτό που υποτίθεται πως πρέπει να επιτυγχάνει το λογισμικό, όπως οι είσοδοι και οι έξοδοί του. Στον έλεγχο μαύρου κουτιού χρησιμοποιούνται αρκετές μέθοδοι, οι οποίες περιγράφονται παρακάτω.

Διεξοδικός και τυχαίος έλεγχος

Διεξοδικός έλεγχος. Η καλύτερη μέθοδος ελέγχου μαύρου κουτιού είναι η δοκιμή του λογισμικού με κάθε πιθανή τιμή εισόδου. Ωστόσο, στα περίπλοκα προγράμματα το εύρος των πιθανών δεδομένων εισόδου είναι τόσο μεγάλο που συχνά είναι ανεφάρμοστο κάτι τέτοιο.

Τυχαίος έλεγχος. Στον τυχαίο έλεγχο επιλέγεται μόνο ένα υποσύνολο των τιμών εισόδου. Είναι πολύ σημαντικό το υποσύνολο να επιλέγεται με τέτοιον τρόπο ώστε να λαμβάνονται τιμές από ολόκληρο το εύρος τιμών εισόδου.

Έλεγχος οριακών τιμών

Σφάλματα προκύπτουν συχνά όταν χρησιμοποιούνται οριακές τιμές. Για παράδειγμα, αν σε μια υπομονάδα ορίζεται ότι η μία από τις εισόδους της πρέπει να είναι μεγαλύτερη ή ίση με 100, είναι πολύ σημαντικό η υπομονάδα να ελεγχθεί με την οριακή τιμή της 100. Αν η υπομονάδα αστοχήσει σε αυτή την οριακή τιμή, είναι πιθανό κάποια συνθήκη στον κώδικα της υπομονάδας να είναι λάθος, όπως το $x \geq 100$ να έχει γραφεί ως $x > 100$.

Τεκμηρίωση

Για να μπορεί να χρησιμοποιείται σωστά και να συντηρείται με αποδοτικότητα ένα πακέτο λογισμικού χρειάζεται τεκμηρίωση (documentation). Συνήθως δημιουργούνται τρία διαφορετικά σετ τεκμηρίωσης για ένα πακέτο λογισμικού: η **τεκμηρίωση χρήστη**, η **τεκμηρίωση συστήματος**, και η **τεχνική τεκμηρίωση**.

- Η τεκμηρίωση είναι μια συνεχής διαδικασία.

Τεκμηρίωση χρήστη

Για να εκτελεί σωστά το σύστημα λογισμικού, ο χρήστης χρειάζεται τεκμηρίωση, η οποία παραδοσιακά ονομάζεται εγχειρίδιο ή manual, όπου περιλαμβάνονται βήμα προς βήμα οδηγίες χρήσης τού συστήματος. Συνήθως το εγχειρίδιο περιλαμβάνει μια εκπαιδευτική ενότητα για να καθοδηγεί τον χρήστη στην κάθε λειτουργία του πακέτου.

Τεκμηρίωση συστήματος

Η τεκμηρίωση συστήματος αναφέρεται στο ίδιο το λογισμικό. Θα πρέπει να γραφτεί με τέτοιον τρόπο ώστε το πακέτο να μπορεί να συντηρηθεί και να τροποποιηθεί από άτομα διαφορετικά από τους αρχικούς προγραμματιστές. Η τεκμηρίωση συστήματος πρέπει να καλύπτει και τις τέσσερις φάσεις της ανάπτυξης.

Τεχνική τεκμηρίωση

Η τεχνική τεκμηρίωση περιγράφει τη διαδικασία εγκατάστασης και συντήρησης του συστήματος λογισμικού. Η τεκμηρίωση εγκατάστασης ορίζει τον τρόπο με τον οποίο πρέπει να εγκατασταθεί το λογισμικό σε κάθε υπολογιστή όπως, για παράδειγμα, σε διακομιστές και πελάτες. Η τεκμηρίωση συντήρησης καθορίζει τον τρόπο με τον οποίο πρέπει να συντηρείται και να ενημερώνεται το σύστημα.

Σύνοψη

- Φάσεις ανάπτυξης λογισμικού
  - Ανάλυση
  - Σχεδιασμός
  - Υλοποίηση
  - Έλεγχος
- Διαδικασιακή και αντικειμενοστρεφής ανάπτυξη λογισμικού