Πανεπιστημιο Δυτικης Αττικης
Τμημα Μηχανικων Πληροφορικης και Υπολογιστων

Τελική Εργασία: Διαχείριση Αποθηκών

Φοιτητές: *Ιάκωβος Μαστρογιαννόπουλος - Νικόλαος Σέργης - Κωνσταντίνος Καμαρόπουλος*

Μάθημα: *Σχεδίαση και Ανάπτυξη Πληροφοριακών Συστημάτων* – Καθηγητής: *Αλέξανδρος Μπουσδέκης*

Abstract

Αυτή είναι η τελική εργασία για το μάθημα «Σχεδίαση και Ανάπτυξη Πληροφοριακών Συστημάτων» των φοιτητών:

- Ιάκωβος Μαστρογιαννόπουλος, 713242017102
- Νικόλαος Σέργης, 18390173
- Κωνσταντίνος Καμαρόπουλος, 71346830

Το θέμα της εργασίας είναι να μελετηθούν οι απαιτήσεις ενός διαγωνισμού και να γίνει η απαραίτητη σχεδίαση και ανάπτυξη του πληροφοριακού συστήματος που θα το συνοδεύει. Το θέμα της ομάδας είναι να φτιάξει την διαχείριση της αποθήκης του συστήματος ΗΔΙΚΑ.

Contents

1. Μεθοδολογία ανάπτυξης εφαρμογής

1.1. Διαθέσιμες μεθοδολογίες.

1.1.1. Μοντέλο build and fix. Το **μοντέλο build and fix** είναι ένα μοντέλο στο οποίο το λογισμικό έχει αναπτυχθεί χωρίς σχεδιασμό. Ουσιαστικά, κατασκευάζεται ένα αρχικό προϊόν και τροποποιείτε μέχρι να ικανοποιήσει τον χρήστη. Το μοντέλο έχει δύο φάσεις:

- Η φάση του **build**: Όπου ο κώδικας κατασκευάζεται και περνάει στην επόμενη φάση.
- Η φάση του **fix**: Όπου ο κώδικας έχει φτάσει σε bug και error free στάδιο και μπορεί να παρουσιαστεί στον χρήστη και να τροποποιηθεί κατάλληλα για να ικανοποιήσει τον τελικό χρήστη.

Πλεονεκτήματα και Μειονεκτήματα του μοντέλου build and fix	
Πλεονεκτήματα	**Μειονεκτήματα**
Χρειάζεται λιγότερη εμπειρία σε οποιονδήποτε άλλο τομέα εκτός του προγραμματισμού	Δεν υπάρχει ένας μετρητής στον οποίο κρίνετε ούτε η πρόοδος, ούτε η ποιότητα του προϊόντος και ούτε ο ρίσκο
Πολύ ταιριαστό για μικρά λογισμικά	Ο κόστος είναι πελώριος επειδή χρειάζεται να γίνονται πάρα πολλές στο λογισμικό μέχρι να ικανοποιήσει τον τελικό χρήστη
Χρειάζεται λιγότερο πλάνο	Είναι πολύ ανεπίσημος τρόπος σχεδίασης ενός λογισμικού
	Η συντήρηση τέτοιων μοντέλων είναι δύσκολη

1.1.2. Μοντέλο καταρράκτη. Το **μοντέλο του καταρράκτη (waterfall model)** είναι ένα από τα πιο κλασσικά παραδείγματα του life cycle. Η ανάπτυξη του λογισμικού είναι γραμμική και πάει από βήμα σε βήμα, η οποία πηγαίνει από βήμα σε βήμα με την ίδια ακριβώς δουλεία, χωρίς να υπάρχει δυνατότητα να μπορεί να γυρίσει πίσω. Κάθε βήμα έχει ξεχωριστό στόχο.

Τα βήματα του μοντέλου καταρράκτη		
Είσοδος στο βήμα	**Βήμα**	**Έξοδος του Βήματος**
Οι απαιτήσεις του λογισμικού πραγματοποιείται μέσω επικοινωνίας	Ανάλυσης	Οι προδιαγραφές του λογισμικού είναι ορισμένες
Οι προδιαγραφές του λογισμικού είναι ορισμένες	Σχεδίασης	Σχεδιασμός του εγγράφου προδιαγραφών
Σχεδιασμός του εγγράφου προδιαγραφών	Ανάπτυξη	Δημιουργία εκτέλεσης προϊόντος
Δημιουργία εκτέλεσης προϊόντος	Δοκιμή	Έτοιμο προϊόν
Έτοιμο προϊόν	Υλοποίηση	Παραδοτέο λογισμικό
Παραδοτέο λογισμικό	Συντήρησης	Αλλαγές στις προδιαγραφές

Το μοντέλο του καταρράκτη φαίνεται και παρουαστικά στο Σχήμα 1. Υπάρχουν αρκέτες παραλλαγές του μοντέλου καταρράκτη, όπως το παράλληλο μοντέλο όπου αρκετά βήματα γίνονται παράλληλα. Για αρκέτες περιπτώσεις είναι αρκέτα ταιριαστό μοντέλο, αφού έχει χρησιμοποιήθει πάρα πολλές φορές και είναι σίγουρο ότι δουλεύει. Για αυτό τον λόγο, αρκέτες από τις μεθοδολογίες που το ακολούθησαν βαδίζουν στην ίδια λογική.

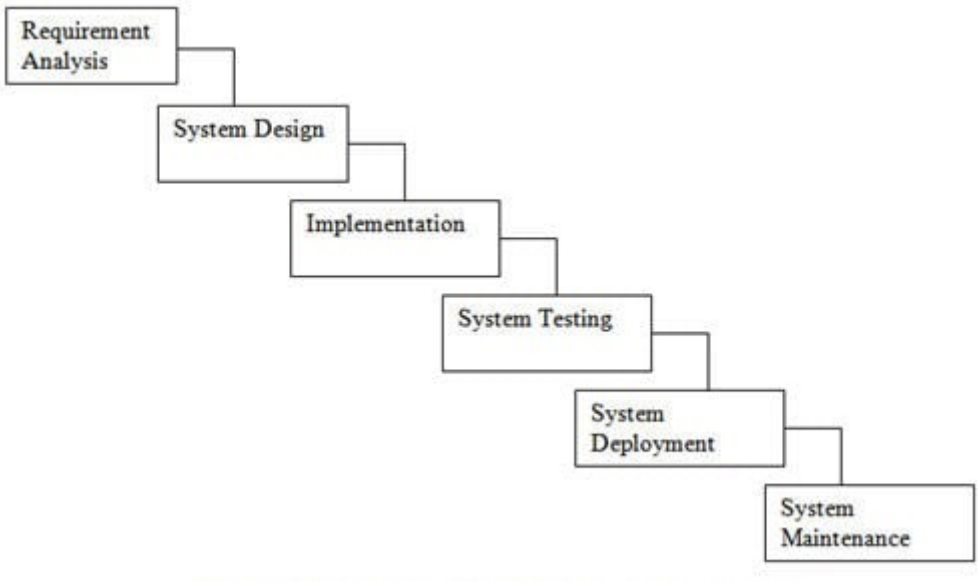

Figure 1: Το μοντέλο του καταρράκτη

Πλεονεκτήματα και Μειονεκτήματα του μοντέλου καταρράκτη	
Πλεονεκτήματα	**Μειονεκτήματα**
Αρκετά απλό στην κατανόηση	Χρειάζεται να είναι οι προδιαγραφές έτοιμες πριν ξεκινήσει η ανάπτυξη
Κάθε βήμα της ανάπτυξης συνεχίζει διαδοχικά	Δεν μπορούν να γίνουν αλλαγές στις προδιαγραφές σε μεταγενέστερα βήματα του μοντέλου. Αυτό σημαίνει ότι ένα λογισμικό μπορεί να μπει στο στάδιο της δοκιμής θα είναι πολύ δύσκολο να γίνουν οι απαραίτητες αλλαγές
Επιτρέπει ελέγχο στην δημιουργία ενός προγράμματος με προθεσμίες σε κάθε βήμα	Δεν υπάρχει καμία επικοινωνία με τον τελικό χρηστή όσο το λογισμικό αναπτύσσετε
Βοηθάει στον έλεγχο των χρονοπρογράμματων, των προϋπολογισμών και του εγγράφου	Δεν παίρνει υπόψιν του τον ρίσκο της διοίκησης
	Θεωρεί ότι οι προδιαγραφές είναι σταθερές και δεν αλλάζουν κατά την διάρκεια του κύκλου ζωής

Μια δεύτερη παραλλαγή του μοντέλου καταρράκτη είναι το **μοντέλο της σπείρας (spiral model)**, το οποίο φαίνεται παρουσιαστικά στο Σχήμα 2.

Πλεονεκτήματα και Μειονεκτήματα του μοντέλου της σπειράς	
Πλεονεκτήματα	**Μειονεκτήματα**
Παρέχει ένα εργατικό μοντέλο στον χρήστη νωρίς στην διεργασία και επιτρέπει μία πρόωρη εκτίμηση και αυξάνει την αυτοπεποίθηση του χρήστη	Άμα ο χρήστης δεν είναι ικανοποιημένος με το τελικό πρωτότυπο, τότε ένα νέο πρωτότυπο κατασκευάζεται. Έτσι επιτρέπει να δημιουργηθεί το τέλειο πρωτότυπο
Ο developer αποκτάει εμπειρίες και γνώση από την δημιουργία του πρωτότυπου και έτσι καταφέρνει να δημιουργήσει καλύτερες προδιαγραφές για την υλοποίηση	Ο developer χάνει το σωστό επίκεντρο του πρωτότυπου και έτσι χάνετε η ποιότητα της εφαρμογής
Το μοντέλο του πρωτότυπου πρέπει να εξυπηρετεί τις ανάγκες των προδιαγραφών οι οποίες δεν είναι καθαρές και έτσι μειώνει την ασάφεια και βελτιώνει την επικοινωνία μεταξύ των developers και των χρηστών	Τα πρωτότυπα μπορούν να οδηγήσουν σε λανθασμένες προσδοκίες
Βοηθάει στην μείωση των ρίσκων τα οποία συνδέονται με το λογισμικό	Ο κύριος στόχος είναι να γίνονται γρήγορα η ανάπτυξη και έτσι το παρακάτω μοντέλο είναι αρκετά αργό

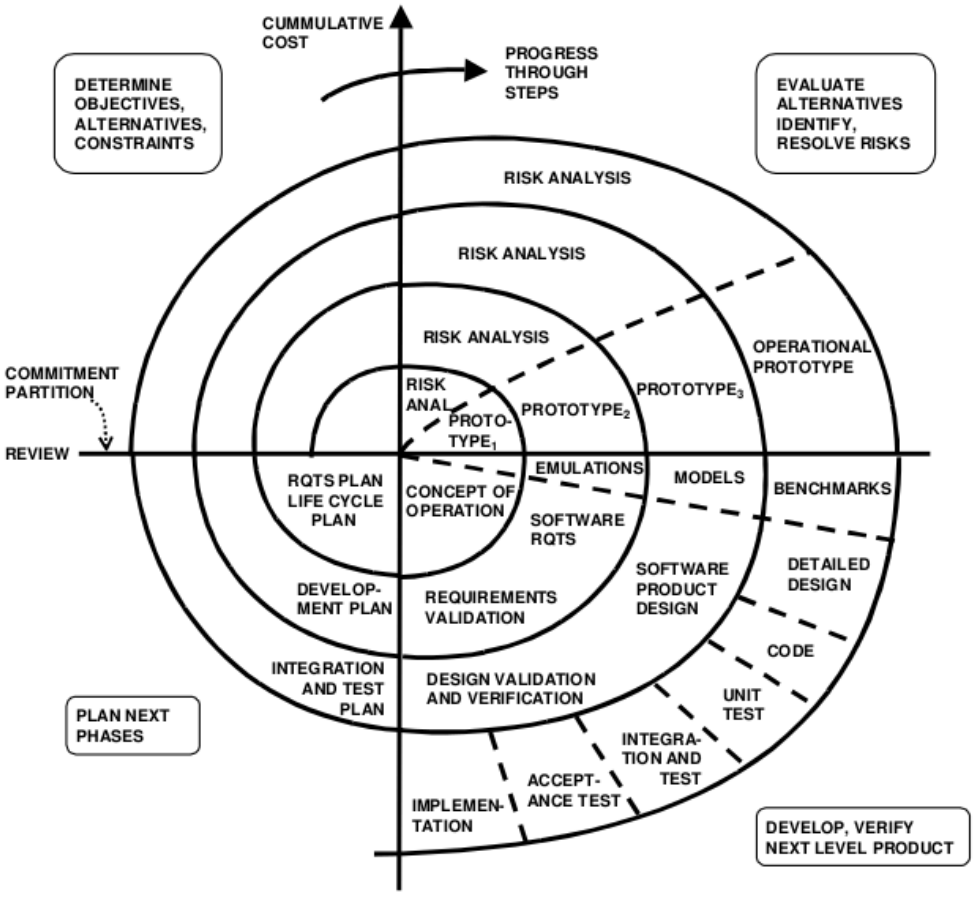

Figure 2: Το μοντέλο της σπειράς

1.1.3. Εξελεκτικές (Ταχείας Ανάπτυξης) Μεθοδολογίες. Το **μοντέλο εξελικτικής ή ταχείας ανάπτυξης (Rapid Application Development, RAD)** είναι ένα μοντέλο το οποίο στηρίζετε στο να σπάει το μεγάλο πρότζεκτ σε μικρότερα πρότζεκτ και να δημιουργεί διάφορα πρωτότυπα ώστε να μπορούν να διακριθούν τα προβλήματα που υπάρχουν και να διορθωθούν. Ένα μεγάλο χαρακτηριστικό των RAD μοντέλων είναι ότι γίνετε επαναχρησιμοποίηση του ίδιου κώδικα, διεργασιών, templates και εργαλείων. Οι φάσεις του RAD είναι οι εξής:

- Σχεδιασμού
- Πρωτοτύπων
- Επανάληψη των βημάτων της ανάλυσης και των πρωτοτύπων όπου χρειάζεται
- Ολοκλήρωση των πρωτοτύπων
- Υλοποίηση

Στο Σχήμα 4 φαίνεται παρουαστικά πως μία τέτοια μεθοδολογία πρέπει να εφαρμοστεί. Μερικές rad μεθοδολογίες είναι το Rapid Programming και το Throwaway Prototyping.

Πλεονεκτήματα και Μειονεκτήματα των RAD	
Πλεονεκτήματα	**Μειονεκτήματα**
Για τα παραδοτέα είναι πολύ εύκολο στο να μπορούν να μεταφερθούν σε πιο υψηλού επίπεδου αφηρημένου κώδικα	Είναι χρήσιμο μόνο για μεγάλα project
Παρέχει πολύ μεγάλη ευελιξία στον επανασχεδιασμό σε περίπτωση που θεωρείτε απαραίτητο	Τα RAD project αποτυγχάνουν όταν δεν υπάρχει η απαραίτητη δέσμευση από τους developers ή τους χρήστες όταν το λογισμικό τελείωσει στην ώρα του
Μείωση της ανάγκης εγγραφής νέου κώδικα λόγο της χρήσης γεννήτριας κώδικα και επαναχρησιμοποίηση ήδη υπάρχων κώδικα	Δεν είναι αποδεκτό σε περίπτωση μεγάλου κινδύνου τεχνικών προβλημάτων
Ενθαρρύνει τους χρήστες στο να συμμετέχουν στην ανάπτυξη του project	Τα ενδιαφέροντα των χρηστών και των developers τείνουν στο να διαφέρουν με αποτέλεσμα να μην μπορούν να πραγματοποιηθούν οι απαιτήσεις του project με την χρήση του RAD μοντέλου
Πιθανότητα να υπάρχουν ελαττωματικά προϊόντα λόγω των πρωτοτύπων	

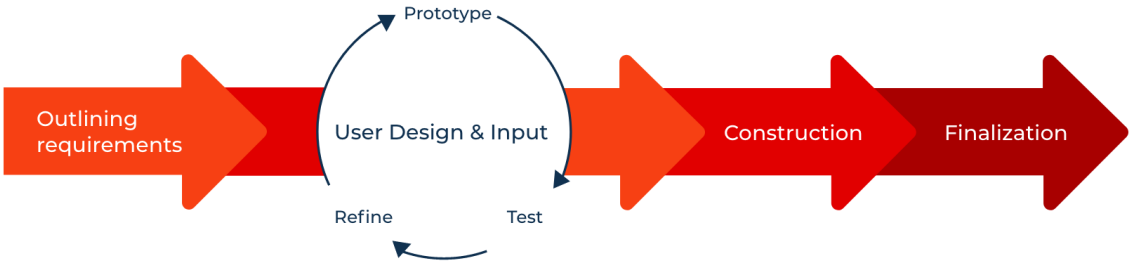

Figure 3: Οι εξελικτές μεθοδολογίες

1.1.4. Εύκαμπτες/ευέλικτες μεθοδολογίες ανάπτυξης (Agile methodologies). Οι **εύκαμπτες μεθοδολογίες ανάπτυξης (Agile methodologies)** έχουν την ικανότητα να μπορούν να δημιουργούν και να αλληλεπιδρούν με την αλλαγή. Η διαφορά του Agile με άλλες μεθοδολογίες είναι ότι εστιάζει στους ανθρώπους της ομάδας και πως δουλεύουν μεταξύ τους. Οι λύσεις αναπτύσσονται μέσω συνεργασίας μεταξύ της αυτοδιοικούμενη ομάδα χρησιμοποιώντας πρακτικές για το context που βολεύει την κάθε ομάδα. Αυτό σημαίνει ότι δεν υπάρχουν διοικητές (managers) στην ομάδα, αλλά κάθε ομάδα έχει την ικανότητα να μπορεί να οργανωθεί μόνη της. Τα μέλη αυτής της ομάδας είναι ίσα και δεν έχουν συγκεκριμένους ρόλους μέσα στην ομάδα. Το Agile, σύμφωνα με το Agile Manifesto έχει 12 βασικά θεμελιώδες ιδεολογίες:

1. Η μεγαλύτερη προτεραιότητα είναι να ικανοποιηθεί ο πελάτης νωρίς και να υπάρχει συνέχει παράδοση καλού λογισμικού
2. Να υπάρχουν αλλαγές στις προδιαγραφές μέχρι και όταν είναι αργά στο στάδιο της ανάπτυξης
3. Να παραδίδετε ένα λειτουργικό λογισμικό συχνά, σε οποιονδήποτε χρονικό πεδίο
4. Οι επιχειρηματίες και οι developers πρέπει να δουλεύουν μαζί καθημερινά κατά την διάρκεια όλου του project
5. Να κτίζονται projects γύρο από άτομα που έχουν ισχυρό κίνητρο
6. Η πιο αποτελεσματική μέθοδος στο να μπορεί να μεταφερθεί πληροφορία από και μέσω της developing ομάδας πρόσωπο προς πρόσωπο
7. Ένα λειτουργικό λογισμικό είναι ο κύριος μετρητής της προόδου
8. Οι διεργασίες της Agile προωθούν στο να μπορεί να υπάρξει ένα σταθερό development περιβάλλον από σπόνσορες, developers και χρήστες που θα μπορούν να συντηρήσουν μία σταθερή πρόοδο για μεγάλο χρονικό διάστημα

9. Συνεχές προσοχή σε τεχνική τελειότητα και η καλή σχεδίαση βελτιώνει την Agile μεθοδολογία
10. Η απλότητα και η τέχνη του περιορίζεται το μέγεθος της δουλείας που δεν γίνετε είναι ουσιώδες χαρακτηριστικό του Agile
11. Οι καλύτερες αρχιτεκτονικές, προδιαγραφές και σχεδιάσεις έρχονται από αυτοδιοικούμενες ομάδες
12. Σε συχνές συναντήσεις, η ομάδα ψάχνει πως μπορεί να γίνει όλο και πιο αποτελεσματική, κοιτάει τα λάθη της και συνεχίζει

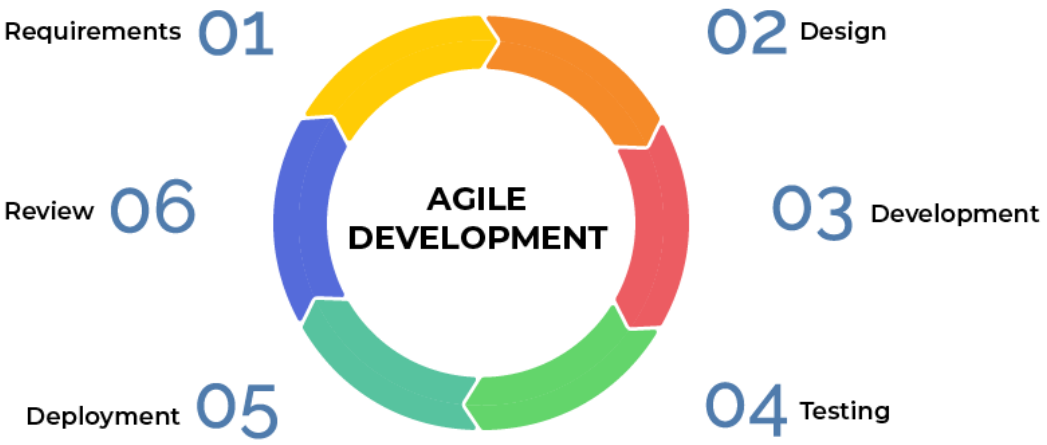

Figure 4: Οι agile μεθοδολογίες

Υπάρχουν αρκετές agile μεθοδολογίες, δύο από τις οποίες είναι το **extreme programming** και η **μεθοδολογία SCRUM**.