Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής (Ελληνικά)	**Προσαρμοστικό Σύστημα Διδασκαλίας σε πλατφόρμα java**
Τίτλος Διατριβής (Αγγλικά)	**Adaptive Teaching System in platform java**
Ονοματεπώνυμο Φοιτητή	**Χριστίνα Ρίζου**
Πατρώνυμο	**Σταύρος**
Αριθμός Μητρώου	**ΜΠΣΠ/09001**
Επιβλέπων	**Βίρβου Μαρία, Καθηγήτρια**

Ημερομηνία Παράδοσης **Απρίλιος 2013**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα	Όνομα Επώνυμο Βαθμίδα

Περίληψη

Σήμερα ζούμε στην εποχή της πληροφορίας. Ο Ηλεκτρονικός Υπολογιστής έχει εισχωρήσει σε όλους τους τομείς της επιστήμης και κάθε άλλης παραγωγικής δραστηριότητας συμβάλλοντας έτσι, με έμμεσο και άμεσο τρόπο, στην ραγδαία εξέλιξή τους. Η χρήση του υπολογιστή μας δίνει τη δυνατότητα να έχουμε οποιαδήποτε πληροφορία τη στιγμή που τη θέλουμε και με οποιαδήποτε μορφή, είτε είναι κείμενο, βίντεο ή εικόνα.

Με τη χρήση του υπολογιστή σαν εκπαιδευτικό εργαλείο, ο μαθητής μπορεί να εμπλουτίσει τη γνώση του και να τον χρησιμοποιεί συμπληρωματικά με τα όσα διδάσκεται στο σχολείο ή στο πανεπιστήμιο. Οι υπολογιστές, με άλλα λόγια, μπορούν να χρησιμοποιηθούν σαν υποστηρικτικά μέσα της εκπαιδευτικής διαδικασίας με τη χρήση ενός κατάλληλου εκπαιδευτικού λογισμικού.

Το εκπαιδευτικό λογισμικό μπορεί να χρησιμοποιηθεί είτε επικουρικά από τον καθηγητή σα συμπληρωματικό μέσο διδασκαλίας, είτε να χρησιμοποιηθεί αποκλειστικά από το μαθητή ως μέσο αυτοδιδασκαλίας με σκοπό τον εμπλουτισμό της γνώσης του. Και στις δυο περιπτώσεις ως κύριος γνώμονας πρέπει να είναι η γνώση αλλά και η μάθηση.

Στη μεταπτυχιακή διατριβή, παρουσιάζονται οι βασικές αρχές τεχνολογικής σχεδίασης ενός εκπαιδευτικού λογισμικού πάνω στις οποίες βασίστηκε η σχεδίαση του προσαρμοστικού συστήματος διδασκαλίας μαθητών – καθηγητών στα μαθήματα τα οποία διδάσκουν.

Μέσα από τις φάσεις ζωής του λογισμικού, τις απαιτήσεις για τη σωστή σχεδίαση του συστήματος, της επιλογής του κατάλληλου μοντέλου κύκλου ζωής λογισμικού αλλά και της όσο το δυνατόν καλύτερης κατασκευής του περιβάλλοντος διεπαφής του χρήστη, θα προχωρήσουμε στην κατασκευή του συστήματος μετά από ανάλυση των προδιαγραφών και την ενσωμάτωση των αποτελεσμάτων και των επιλογών που θα προκύψουν από τα ανωτέρω στοιχεία, έτσι ώστε να προχωρήσουμε στη φάση κατασκευής και στην τελική κατασκευή του συστήματος που θα συνάδει με τις απαιτήσεις και τις ανάγκες των χρηστών οι οποίοι θα το χρησιμοποιούν, δηλαδή, οι μαθητές και οι καθηγητές.

Abstract

Nowadays, we live in a society of information. The computer has penetrated in all aspects of science and any other productive activity, contributing indirectly and directly in their rapid evolution. The use of computer gives us the opportunity to have the information we desire, whether it's text, video or image, any time we want and in any form.

Using the computer as an educational tool, the student can enrich his knowledge and use it additionally to what he is taught in school or university. Computers, in other words, can be used as supportive means in the educational process with the use of appropriate educational software.

The educational software can be used by the teacher as a supplementary mean of teaching, or it can be used exclusively by the student as a mean of self-learning in order to enrich his knowledge. In both cases, the main goal should be the knowledge and learning.

In this thesis, the main technological design principals of an educational software are presented on which the design of the adaptive teaching system between students – teachers and the courses they teach are relied.

Through the life phases of the software requirements for the proper system design, the selection of the appropriate software life cycle model and the best possible user interface manufacturing environments, we will proceed in the development of a system, after analyzing the requirements and the integration of the results and choices that will result from the above, in order to proceed to the development phase and the final system development that will be consistent with the requirements and the needs of the users who will use it, ie students and teachers.

Περιεχόμενα

Εισαγωγή – Σύντομη περιγραφή

Οι ραγδαίες εξελίξεις στον οικονομικό, κοινωνικό και τεχνολογικό τομέα, έχουν επιφέρει σημαντικές αλλαγές στον καθημερινό τρόπο ζωής γενικότερα και ειδικότερα στην εκπαίδευση. Πλέον, όταν αναφερόμαστε στον όρο εκπαίδευση, δε μιλάμε για την παραδοσιακή και κλασσική

εκπαίδευση που υπήρχε μέχρι και πριν από κάποια χρόνια, αλλά για ένα νέο είδος εκπαίδευσης που έχει άμεση σχέση με τους υπολογιστές και βασίζεται όπως και η παραδοσιακή εκπαίδευση στη μάθηση και στη γνώση. Η αδυναμία της συνεχούς παρουσίας του καθηγητή καθ' όλη τη διάρκεια της ημέρας, αλλά και οποιαδήποτε στιγμή καταστεί αναγκαίο από την πλευρά του μαθητή, οδήγησε στην ανάπτυξη της εξ' αποστάσεως εκπαίδευσης. Προσφέρει σημαντικά πλεονεκτήματα με κυριότερα τη μείωση του κόστους και την αύξηση της προσβασιμότητας στο μαθησιακό υλικό από την πλευρά των μαθητών οποτεδήποτε το χρειαστούν.

Αντικείμενο της παρούσας διατριβής είναι να εξεταστούν όλες οι φάσεις σχεδίασης ενός εκπαιδευτικού λογισμικού, τόσο σε λειτουργικό, όσο και σε γραφικό επίπεδο, διότι ακόμα και αν ένα σύστημα καλύπτει όλες τις ανάγκες και απαιτήσεις των χρηστών, αν δεν έχει και το κατάλληλο περιβάλλον διεπαφής (user interface) που να μπορεί να το υποστηρίξει, τότε θα είναι δύσκολο και δυσνόητο για τους χρήστες που δεν είναι πλήρως εξοικειωμένοι και έμπειροι με τη χρήση υπολογιστών. Για το λόγο αυτό, λαμβάνονται υπόψη όλες οι παράμετροι ώστε η τελική έκδοση του συστήματος που θα παραδοθεί στους χρήστες να καλύπτει τις ανάγκες και τις απαιτήσεις τους με εύκολο και ευκρινή τρόπο, μόνο τότε το εκπαιδευτικό σύστημα θα θεωρείται επιτυχές.

Για το λόγο αυτό, η παρούσα διατριβή έχει χωριστεί σε κεφάλαια, κάθε ένα από τα οποία εξετάζει όλους τους παράγοντες που προαναφέρθηκαν. Ακολουθεί μια συνοπτική περιγραφή του περιεχομένου του κάθε κεφαλαίου.

Στο πρώτο κεφάλαιο δίνεται ένας ορισμός του εκπαιδευτικού λογισμικού, παρουσιάζονται οι κυριότερες κατηγορίες εκπαιδευτικών λογισμικών, αλλά και ποιοι είναι οι στόχοι του για την επίτευξη των διδακτικών σκοπών.

Στο δεύτερο κεφάλαιο περιγράφονται οι φάσεις ζωής του λογισμικού, από την προδιαγραφή των απαιτήσεων, τη σχεδίαση της αρχιτεκτονικής του συστήματος που θα ακολουθηθεί, μέχρι τη συγγραφή του κώδικα και τον έλεγχο και τη συντήρηση του συστήματος. Όλα τα στάδια του κύκλου ζωής του λογισμικού είναι σημαντικά και κάθε ένα από τα οποία βοηθάει στην παραγωγή και δημιουργία του συστήματος προς παράδοση. Κάποια από τα σημαντικότερα μοντέλα κύκλου ζωής λογισμικού που υπάρχουν, είναι το μοντέλο του καταρράκτη, το μοντέλο πρωτοτυποποίησης, το μοντέλο λειτουργικής επαύξησης, το σπειροειδές μοντέλο, αλλά και το μοντέλο επαναχρησιμοποίησης λογισμικού. Θα πρέπει λοιπόν να συγκριθούν και να διαπιστωθεί ποιο από τα προαναφερθέντα μοντέλα είναι το καταλληλότερο για το σύστημα που θα δημιουργηθεί.

Στο τρίτο κεφάλαιο γίνεται αναφορά στη σχεδίαση του συστήματος, η οποία ουσιαστικά αφορά στην επίτευξη των στόχων που τίθενται. Για να έχουμε λοιπόν μια επιτυχημένη σχεδίαση, θα πρέπει να κατανοούμε τους υπολογιστές αναφορικά με τους περιορισμούς, τις δυνατότητες, τα εργαλεία και τις πλατφόρμες που διαθέτουν, αλλά και να κατανοούμε τους ανθρώπους αναφορικά με τις ψυχολογικές – κοινωνικές απόψεις και το ανθρώπινο σφάλμα. Γι'αυτό το λόγο θα πρέπει να ακολουθηθεί όλη η διαδικασία του βρόγχου επανάληψης στη διαδικασία της σχεδίασης. Οι φάσεις αυτές που εμπλέκονται στο βρόγχο είναι οι απαιτήσεις, η ανάλυση, η σχεδίαση, η επανάληψη και πρωτοτυποποίηση και η υλοποίηση και ανάπτυξη.

Κατά τη φάση σχεδίασης του συστήματος λαμβάνονται πολλές αποφάσεις καθώς το προϊόν εξελίσσεται από ένα σύνολο ασαφών απαιτήσεων του χρήστη σε ένα τελικό σύστημα. Αιτιολόγηση της σχεδίασης είναι ουσιαστικά οι πληροφορίες που εξηγούν γιατί ένα σύστημα υπολογιστή είναι όπως είναι και οι λόγοι που είναι απαραίτητη η αιτιολόγηση της σχεδίασης στο πεδίο της επικοινωνίας ανθρώπου-υπολογιστή αναλύονται στο κεφάλαιο 3.

Στο τέταρτο κεφάλαιο παρουσιάζεται η χρηστικότητα των διαδραστικών μοντέλων κατά τη διάρκεια υλοποίησης ενός συστήματος και αναλύεται το μοντέλο διάδρασης όπως παρουσιάστηκε από τον Norman, αλλά και το πλαίσιο αναφοράς της διάδρασης.

Στο πέμπτο κεφάλαιο γίνεται λόγος για τη διεπαφή του χρήστη, η οποία υλοποιεί την αμφίδρομη επικοινωνία συστήματος – χρήστη. Περιγράφονται δυο διεπαφές χρήστη (το Command-based interface, όπου για να αλληλεπιδράσει ο χρήστης με το σύστημα θα πρέπει να πληκτρολογήσει εντολές και το Graphical User Interface το οποίο βασίζεται σε γραφικά GUI, όπου είναι ο πιο διαδεδομένος τύπος διεπαφής χρήστη και συμφωνεί απόλυτα με τις θεωρήσεις του αντικειμενοστραφούς προγραμματισμού). Για να σχεδιαστεί επιτυχώς μια διεπαφή χρήστη, θα πρέπει interface να χαρακτηρίζεται από συνέπεια, απλότητα, ελαχιστοποίηση των ενεργειών του χρήστη, παροχή άμεσης βοήθειας, ελαχιστοποίηση απομνημόνευσης, εναρμόνισή,

ευκαμψία και να γίνεται χρήση μεταφορών. Στη συνέχεια του κεφαλαίου παρουσιάζονται οι προτάσεις του Sommerville για το σχεδιασμό του περιβάλλοντος της διεπαφής. Στο τέλος του κεφαλαίου, γίνεται λόγος για τις πιο κοινές μορφές διεπαφής, όπως είναι η διεπαφή γραμμής εντολών, το μενού επιλογών, η φυσική γλώσσα, οι ερωταποκρίσεις και διάλογος μέσω ερωτημάτων, η συμπλήρωση φορμών και φύλλων εργασίας, τα παραθυρικά περιβάλλοντα (WIMP), το δείξιμο και κλικ, αλλά και οι τρισδιάστατες διεπαφές.

Στο έκτο κεφάλαιο γίνεται ενσωμάτωση των όσων περιγράφηκαν στα προηγούμενα κεφάλαια στο προσαρμοστικό σύστημα διδασκαλίας που υλοποιήθηκε, και αναλύονται οι σχεδιαστικές αρχές οι οποίες υιοθετήθηκαν, πως προκύπτει η προσαρμοστικότητα του συστήματος, τα χρώματα που χρησιμοποιήθηκαν στη διεπαφή του συστήματος, αλλά και ο τρόπος που είναι κατασκευασμένο το περιβάλλον διεπαφής του συστήματος και κατά πόσο βοηθάει τους χρήστες.

Το έβδομο κεφάλαιο αφορά στη φάση κατασκευής και γίνεται μια περιγραφή των προγραμμάτων που χρησιμοποιήθηκαν.

Στο όγδοο κεφάλαιο γίνεται μια πλήρης περιγραφή του τρόπου λειτουργίας του συστήματος διδασκαλίας μέσω των παραθύρων και των μηνυμάτων που εμφανίζονται στο χρήστη καθ' όλη τη διάρκεια χρήσης του.

Στο ένατο κεφάλαιο γίνεται μια επεξήγηση του τρόπου με τον οποίο είναι δομημένη η βάση δεδομένων αλλά και οι πίνακες που δημιουργήθηκαν για τη σωστή και ευέλικτη λειτουργία του συστήματος και τέλος, γίνεται μια σύντομη και περιληπτική περιγραφή των αρχείων και κλάσεων που δημιουργήθηκαν, αλλά και των συναρτήσεων που χρησιμοποιήθηκαν για την υλοποίηση του συστήματος, τόσο στο προγραμματιστικό τμήμα του, όσο και στο γραφικό κομμάτι του.

Κεφάλαιο 1
Εκπαιδευτικό λογισμικό

Εκπαιδευτικό λογισμικό θεωρείται το λογισμικό που εμπεριέχει διδακτικούς στόχους, ολοκληρωμένα σενάρια, αλληγορίες με παιδαγωγική σημασία και έχει ως κύριο στόχο να συντελέσει σε διδακτικά και μαθησιακά αποτελέσματα. Δηλαδή ως Εκπαιδευτικό Λογισμικό (educational software) νοούνται οι εφαρμογές λογισμικού που χρησιμοποιούνται για την υπολογιστική υποστήριξη της διδασκαλίας και της μάθησης.[25]

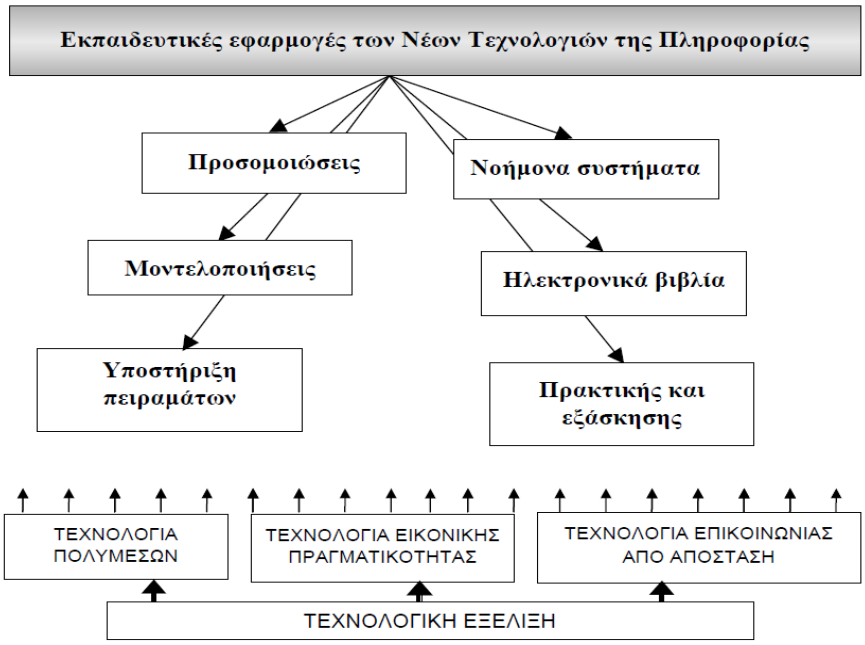
Οι δραματικές αλλαγές που η τεχνολογία έφερε στην κοινωνία και στην αγορά εργασίας έχει πολύ σημαντικές εφαρμογές στον τομέα της εκπαίδευσης. Η εκπαίδευση οφείλει να αλλάξει κατεύθυνση και προτεραιότητες για να συναντήσει την κοινωνία που τεχνολογικά αλλάζει με γρήγορους ρυθμούς. Η εκπαιδευτική τεχνολογία αποτελεί μια διαδοχικών προσεγγίσεων διαδικασία για τον σχεδιασμό αποτελεσματικής διδασκαλίας. Για τον σκοπό αυτό περιλαμβάνει την εφαρμογή γνώσεων, την ανάπτυξη τεχνικών διδακτικής προσέγγισης (ανάλογα με το

διδακτικό μοντέλο που έχει υιοθετηθεί), και κυρίως, την ανάπτυξη, επιλογή και χρήση μέσων για την υποβοήθηση της διδασκαλίας.

Η ανάγκη για μια πιο αποτελεσματική διδασκαλία, ιδιαίτερα στην Επιστήμη και στην Τεχνολογία, καθώς και η δημιουργία νέων μέσων, έχει οδηγήσει τους εκπαιδευτικούς προς την αναζήτηση νέων διδακτικών προσεγγίσεων, κυρίως με τη βοήθεια των δυνατοτήτων της Πληροφορικής. Αυτό έχει ως συνέπεια μια ιδιαίτερα ανθηρή δραστηριότητα στον τομέα της Εκπαιδευτικής Τεχνολογίας που αφορά τη χρήση της Πληροφορικής στην Εκπαίδευση.

Προκειμένου όμως να γίνει η διδακτική προσέγγιση με βάση την Πληροφορική απαιτείται να υπάρχει το σχετικό υπόβαθρο στο συγκεκριμένο κάθε φορά αντικείμενο, καθώς και στη χρήση των αντίστοιχων συσκευών και προγραμμάτων της Πληροφορικής. Για το λόγο αυτό έχουν αναπτυχθεί πολλά είδη λογισμικού, τα οποία αποβλέπουν στο να καταστήσουν ευκολότερη τη σχεδίαση και ανάπτυξη διδακτικών προσεγγίσεων με χρήση των τεχνολογιών της Πληροφορικής από εκπαιδευτικούς, οι οποίοι δεν είναι ειδικοί στον προγραμματισμό Η/Υ.

Στο σχήμα 1.1 που ακολουθεί, παρουσιάζονται οι κατηγορίες των εκπαιδευτικών λογισμικών.



Σχήμα 1.1. κατηγορίες εκπαιδευτικών λογισμικών

1.1. Τα είδη του εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό μπορεί να ταξινομηθεί ως προς τη χρήση του στη μαθησιακή διαδικασία στις εξής πιο σημαντικές κατηγορίες [27] :

- Λογισμικό εκγύμνασης και εξάσκησης (Drill and Practice) : αυτού του είδους τα λογισμικά δίνουν τη δυνατότητα στους μαθητές να εξασκηθούν στην ύλη την οποία έχουν ήδη διδαχθεί και καλούνται να απαντήσουν σε ερωτήσεις που βασίζονται στην ύλη την οποία έχουν διδαχθεί. Το σύστημα αποθηκεύει την επίδοση του μαθητή, τα λάθη του και επιπλέον του προτείνει να διαβάσει εκ νέου την ύλη που αναφερόταν στις ερωτήσεις που απάντησε λάθος.
- Λογισμικό μοντελοποίησης (modeling) : με αυτού του είδους τα λογισμικά μπορούν να γίνουν αναπαραστάσεις συστημάτων ή διαδικασιών, διότι ένα μοντέλο μπορεί να λειτουργήσει και σαν αναπαράσταση συστήματος.
- Λογισμικό προσομοίωσης (simulation) : αυτού του είδους τα λογισμικά δίνουν τη δυνατότητα στους μαθητές να υλοποίησης καταστάσεων που δεν θα ήταν δυνατές με

κάποιο άλλο τρόπο. Συνήθως, στηρίζονται σε μια σειρά αλγορίθμων και οι μαθητές έχουν τη δυνατότητα να αλλάζουν τις τιμές ορισμένων μεταβλητών και να παρατηρούν τα αποτελέσματα της πράξης τους.

- Λογισμικό επίλυσης προβλήματος (problem solving) : αυτού του είδους τα λογισμικά δίνουν τη δυνατότητα στους μαθητές να επιλύσουν προβλήματα σύμφωνα με γνώσεις που είχε αποκτήσει νωρίτερα και συχνά περιέχουν προσομοίωση ενός φαινομένου του πραγματικού κόσμου.
- Λογισμικό εκπαιδευτικών παιχνιδιών (educational/instructional games) : αυτού του είδους τα λογισμικά βοηθούν στην απόκτηση και ανάπτυξη δεξιοτήτων σε περιβάλλον παιχνιδιού. Το παιχνίδι χρησιμοποιείται ως ένα μέσο για να αποκτήσει ο μαθητής γνώσεις καθώς ολοκληρώνει διαδικασίες μέσα από το παιχνίδι. Αυξάνεται με αυτό τον τρόπο ο ενθουσιασμός τους, αλλά μερικές φορές γίνεται λάθος χρήση τους από τους μαθητές, καθώς τους ενδιαφέρει περισσότερο να φτάσουν στον τελικό στόχο, παρά να χρησιμοποιήσουν τις δυνατότητες που τους προσφέρει να για εμπλουτίσουν τις γνώσεις τους.
- Λογισμικό εκπαίδευσης – φροντιστηρίου (tutorial) : αυτού του είδους τα λογισμικά παρουσιάζουν πρωτότυπη είτε ήδη διδαγμένη ύλη. Εμφανίζουν διαδοχικά σύνολα πληροφοριών και στη συνέχεια θέτουν στο χρήστη ερωτήσεις. Είναι εμπνευσμένα από το ρόλο του δασκάλου και προσαρμόζουν το διδακτικό υλικό στις ιδιαίτερες ανάγκες και ικανότητες του μαθητή.

1.2. Στόχοι εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό πρέπει να καλύπτει τις ανάγκες του χρήστη και να επιτυγχάνει τους εκπαιδευτικούς και διδακτικούς σκοπούς για το οποίο δημιουργήθηκε. Θα πρέπει να μπορεί να ενταχθεί στο κύριο διδακτικό έργο, να συμπληρώνει τη διδακτική διαδικασία και να προκαλεί το ενδιαφέρον του μαθητή. [26]

1.2.1. Λειτουργία του εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό πρέπει να είναι σε θέση να παρέχει καταλληλότητα (suitability), αξιοπιστία (reliability), αποδοτικότητα (efficiency), χρηστικότητα (usability), ασφάλεια (security) και συμμόρφωση (compliance).

1.2.2. Υποστήριξη του εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό θα πρέπει να παρέχει αναλυτικότητα (Analyzability), δυνατότητα αλλαγής (Changeability), σταθερότητα (Stability) και δυνατότητα δοκιμών (testability).

1.2.3. Συμβατότητα του εκπαιδευτικού λογισμικού

Το εκπαιδευτικό λογισμικό θα πρέπει να μπορεί να εγκατασταθεί σε διαφορετικά εργαστηριακά περιβάλλοντα, να έχει δηλαδή δυνατότητα μεταφοράς (Portability), να μπορεί να επαναχρησιμοποιηθεί (Reusability) και να μπορεί να ανταλλάσει πληροφορίες και με άλλες εφαρμογές (Interoperability).

Κεφάλαιο 2
Κύκλος ζωής λογισμικού

Ένα από τα χαρακτηριστικά γνωρίσματα οποιουδήποτε τεχνολογικού τομέα είναι η δομημένη εφαρμογή των επιστημονικών τεχνικών για την ανάπτυξη κάποιου συστήματος. Κατ' επέκταση, ένα θεμελιώδες γνώρισμα της τεχνολογίας λογισμικού είναι ότι παρέχει τη δομή για την εφαρμογή τεχνικών ανάπτυξης συστημάτων λογισμικού. Ο κύκλος ζωής του λογισμικού είναι μια προσπάθεια προσδιορισμού των δραστηριοτήτων που λαμβάνουν χώρα κατά την ανάπτυξη ενός συστήματος λογισμικού. Για οποιοδήποτε έργο ανάπτυξης, οι δραστηριότητες αυτές πρέπει να έχουν συγκεκριμένη χρονική σειρά και να υιοθετούνται οι κατάλληλες τεχνικές για τη διεκπεραίωσή τους. Για την ανάπτυξη ενός συστήματος λογισμικού θεωρούμε ότι δυο είναι τα κύρια συμβαλλόμενα μέρη: ο χρήστης που θα χρησιμοποιήσει το σύστημα και ο σχεδιαστής ο οποίος πρέπει να παρέχει το σύστημα.

2.1. Διαδραστικά συστήματα και ο κύκλος ζωής του λογισμικού

Η παραδοσιακή θεώρηση του κύκλου ζωής του λογισμικού προήλθε από την ανάγκη για μια πιο δομημένη προσέγγιση στην ανάπτυξη λογισμικού για μεγάλα συστήματα στις δεκαετίες του ΄60 και του ΄70. Ο παραδοσιακός κύκλος ανάπτυξης λογισμικού ταιριάζει σε μια βασιζόμενη σε αρχές προσέγγισης της σχεδίασης, δηλαδή, αν γνωρίζουμε από την αρχή τι θέλουμε να παράγουμε, μπορούμε να δομήσουμε την προσέγγισή μας έτσι ώστε να πετύχουμε το στόχο

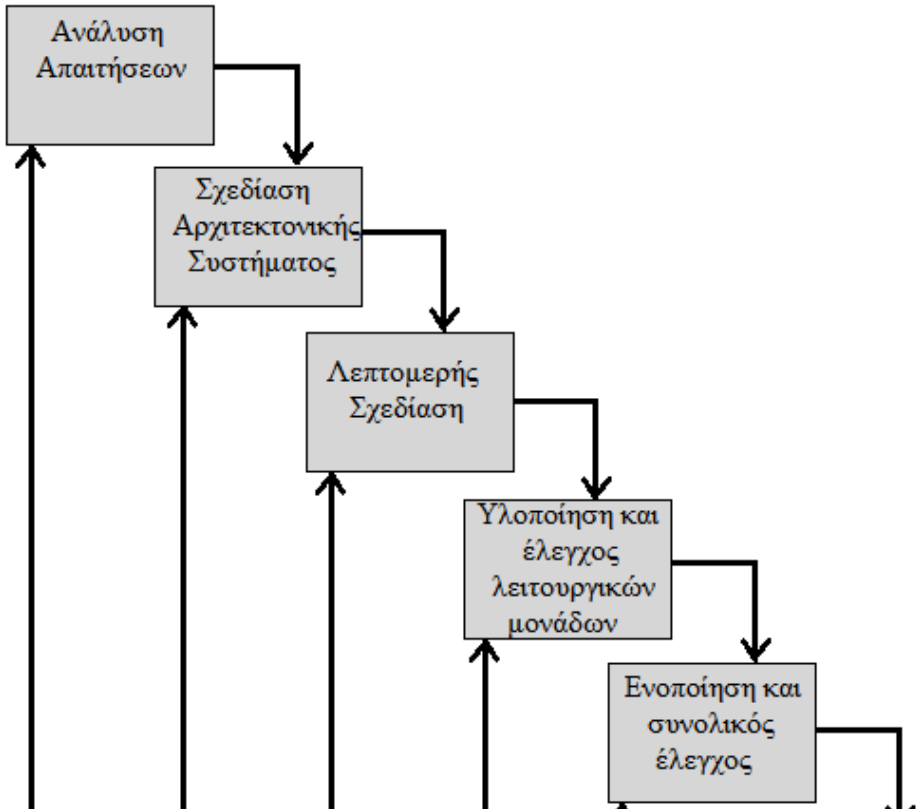
μας. Στην πράξη, οι σχεδιαστές δεν μπορούν να γνωρίζουν όλες τις απαιτήσεις που πρέπει να ικανοποιεί ένα σύστημα πριν ξεκινήσουν την εργασία τους. Πολλές φορές τα ευρήματα μεταγενέστερων δραστηριοτήτων μπορεί να έχουν αντίκτυπο σε προηγούμενα στάδια με αποτέλεσμα να καθίσταται αναγκαία η επιστροφή σε αυτά. Σε πρακτικό επίπεδο αυτό σημαίνει ότι η δραστηριότητα καθορισμού των απαιτήσεων δεν εκτελέστηκε σωστά. Το βασικότερο συμπέρασμα που βγαίνει από όλα αυτά είναι ότι ο εκ των προτέρων προσδιορισμός όλων των απαιτήσεων για ένα διαδραστικό σύστημα είναι αδύνατος και υπάρχουν πολλά πειστικά επιχειρήματα για την υποστήριξη αυτής της άποψης. Το αποτέλεσμα είναι ότι πρέπει πρώτα να δημιουργούμε τα συστήματα και κατόπιν να παρατηρούμε και να αξιολογούμε τη διάδραση των χρηστών με αυτά για να τα κάνουμε πιο εύχρηστα.

Τα μοντέλα από την ψυχολογία και την κοινωνιολογία που περιγράφουν την ανθρώπινη γνώση και τον άνθρωπο δεν είναι πλήρη και δε μας δίνουν τη δυνατότητα να προβλέψουμε πως πρέπει να σχεδιαστεί ένα σύστημα ώστε να μεγιστοποιηθεί η ευχρηστία του. [2] Γίνονται πολλές έρευνες πάνω στα μοντέλα ανθρώπων-χρηστών τα οποία επιτρέπουν την πρόβλεψη απόδοσης των ανθρώπων κατά τη χρήση διαφορετικών διαδραστικών συστημάτων. Αυτή η ανεπάρκεια της ψυχολογίας στον τομέα της πρόβλεψης σημαίνει ότι για τον έλεγχο συγκεκριμένων χαρακτηριστικών ευχρηστίας οι σχεδιαστές ενός προϊόντος είναι υποχρεωμένοι να παρακολουθούν τον τρόπο με τον οποίο οι πραγματικοί χρήστες αλληλεπιδρούν με το προϊόν και να μετρούν την απόδοσή τους. Για να έχουν αξία αυτές οι παρατηρήσεις, οι συνθήκες των πειραμάτων πρέπει να πλησιάζουν όσο το δυνατόν περισσότερο τις συνθήκες πραγματικής χρήσης. Όπως επισήμανε και ο John Carol, οι μικρές λεπτομέρειες του πραγματικού συστήματος μπορεί να επηρεάσουν αποφασιστικά την ευχρηστία του. [5] .

Μια βασιζόμενη σε αρχές προσέγγιση στη σχεδίαση διαδραστικών συστημάτων βασίζεται σε μια ξεκάθαρη κατανόηση των εργασιών που εκτελεί ο κάθε χρήστης από τα αρχικά στάδια της σχεδίασης. Ένα πρόβλημα αυτής της προσέγγισης είναι το γεγονός ότι ο χρήστης συχνά αποσαφηνίζει τις εργασίες που θα εκτελεί αφού εξοικειωθεί με το σύστημα το οποίο τις εκτελεί. Επίσης, ορισμένες εργασίες που εκτελεί ένας χρήστης με ένα σύστημα μπορεί να μην έχουν προβλεφθεί ρητά από το σχεδιαστή του. Μια τελευταία επισήμανση σχετική με τον παραδοσιακό κύκλο ανάπτυξης λογισμικού είναι ότι δεν προωθεί τη χρήση συμβολισμών και τεχνικών που να υποστηρίζουν την άποψη του χρήστη για το διαδραστικό σύστημα.

2.2. Φάσεις κύκλου ζωής λογισμικού

Οι φάσεις του κύκλου ζωής του λογισμικού αποτυπώνονται στο σχήμα 2.1.



Σχήμα 2.1. Πληροφορίες ανάδρασης από τη συντήρηση προς τις άλλες δραστηριότητες.

Προδιαγραφή απαιτήσεων

Η αρχική δραστηριότητα είναι ο καθορισμός των απαιτήσεων και η σύνταξη μιας προδιαγραφής απαιτήσεων (requirements specification): ο σχεδιαστής και ο πελάτης προσπαθούν να δημιουργήσουν μια περιγραφή του τι αναμένεται να παρέχει το υπό ανάπτυξη σύστημα, το οποίο διαφέρει από τον προσδιορισμό του πώς θα παρέχει τις αναμενόμενες υπηρεσίες το σύστημα. [18] Αυτή η δραστηριότητα περιλαμβάνει τη συγκέντρωση πληροφοριών από τον πελάτη σχετικά με το περιβάλλον εργασίας ή τον τομέα εφαρμογής στον οποίο πρόκειται να λειτουργεί το τελικό προϊόν. Η περιγραφή του τομέα εφαρμογής δε θα πρέπει να περιλαμβάνει μόνο τις συγκεκριμένες λειτουργίες που πρέπει να εκτελεί το προϊόν λογισμικού, αλλά και τις λεπτομέρειες για το περιβάλλον στον οποίο πρέπει να λειτουργήσει, όπως οι άνθρωποι που πρόκειται να επηρεάσει και η σχέση του νέου προϊόντος με οποιαδήποτε άλλα προϊόντα τα οποία ενδέχεται να ενημερώνει ή να αντικαθιστά.

Η προδιαγραφή των απαιτήσεων ξεκινά από το σημείο έναρξης της διαδικασίας ανάπτυξης του προϊόντος – συστήματος. Αν και οι απαιτήσεις εκφράζονται από την πλευρά του πελάτη, για να μπορέσει να τις ικανοποιήσει το προϊόν λογισμικού, θα πρέπει να διατυπωθούν σε μια γλώσσα κατάλληλη για την υλοποίησή τους. Αρχικά οι απαιτήσεις εκφράζονται στη φυσική γλώσσα του πελάτη. Οι γλώσσες εκτελέσιμων εντολών που χρησιμοποιούνται για την κατασκευή λογισμικού είναι λιγότερο φυσιολογικές και μοιάζουν περισσότερο με μια μαθητική γλώσσα στην οποία ο κάθε όρος έχει απόλυτα συγκεκριμένη ερμηνεία. Η μετάφραση των απαιτήσεων από την πολύ εκφραστική αλλά σχετικά ασαφή ανθρώπινη γλώσσα στις ακριβέστερες αλλά λιγότερο εκφραστικές γλώσσες εκτελέσιμων εντολών είναι ζωτικής σημασίας για την επιτυχημένη ανάπτυξη λογισμικού.

Σχεδίαση της αρχιτεκτονικής του συστήματος

Αυτή η δραστηριότητα είναι μια υψηλού επιπέδου αποσύνθεση του συστήματος σε επιμέρους συστατικά που είναι δυνατόν να προέλθουν από υπάρχοντα προϊόντα λογισμικού ή να αναπτυχθούν από την αρχή. Η εκτέλεση αυτής της αποσύνθεσης είναι ο στόχος της σχεδίασης της αρχιτεκτονικής του συστήματος, η οποία δεν ασχολείται μόνο με τη λειτουργική αποσύνθεση του συστήματος, αλλά καθορίζει επίσης ποια συστατικά θα παρέχουν ποιες υπηρεσίες. Επιπρόσθετα, πρέπει να περιγράφει τις εξαρτήσεις που υπάρχουν μεταξύ των συστατικών και το διαμοιρασμό των πόρων μεταξύ τους. Υπάρχουν πολλές δομημένες τεχνικές που χρησιμοποιούν οι σχεδιαστές για να παράγουν μια περιγραφή της αρχιτεκτονικής του συστήματος από τις πληροφορίες της προδιαγραφής απαιτήσεων. Η πλειονότητα αυτών των τεχνικών είναι επαρκής για τον καθορισμό των λειτουργικών απαιτήσεων που πρέπει να παρέχει το σύστημα στον τομέα τον οποίο χρησιμοποιείται, αλλά δεν παρέχουν έναν άμεσο τρόπο για τον καθορισμό μη λειτουργικών απαιτήσεων του συστήματος που δε σχετίζονται άμεσα με τις πραγματικές υπηρεσίες που θα παρέχει. Κάποια παραδείγματα τέτοιων μη λειτουργικών απαιτήσεων είναι η αποτελεσματικότητα, η αξιοπιστία, τα χαρακτηριστικά που

σχετίζονται με το χρόνο και την ασφάλεια του συστήματος αλλά και τα διαδραστικά χαρακτηριστικά του συστήματος.

Λεπτομερής σχεδίαση

Η σχεδίαση της αρχιτεκτονικής καταλήγει σε μια αποσύνθεση της περιγραφής του συστήματος η οποία επιτρέπει την ανάπτυξη μεμονωμένων συστατικών ξεχωριστά, τα οποία θα ενοποιηθούν σε επόμενο στάδιο. Για τα συστατικά που δεν είναι διαθέσιμα για άμεση ενοποίηση, ο σχεδιαστής θα πρέπει να παρέχει λεπτομερή και επαρκή περιγραφή τους ώστε να μπορούν να υλοποιηθούν με κάποια γλώσσα προγραμματισμού. Η λεπτομερής σχεδίαση είναι μια βελτίωση της περιγραφής των συστατικών όπως παρέχονται από τη σχεδίαση της αρχιτεκτονικής του συστήματος. Η συμπεριφορά που υποδηλώνει η υψηλότερου επιπέδου περιγραφή, πρέπει να διατηρηθεί στην περισσότερο λεπτομερή περιγραφή.

Συνήθως υπάρχουν περισσότερες από μια πιθανές βελτιώσεις των συστατικών που προκύπτουν από τη σχεδίαση της αρχιτεκτονικής, οι οποίες θα ικανοποιούν τους περιορισμούς που σχετίζονται με τη συμπεριφορά. Η επιλογή της καλύτερης βελτίωσης συχνά σχετίζεται με την προσπάθεια ικανοποίησης όσο το δυνατόν περισσότερων μη λειτουργικών απαιτήσεων του συστήματος. Άρα, η γλώσσα που χρησιμοποιείται για τη λεπτομερή σχεδίαση, πρέπει να επιτρέπει κάποια ανάλυση της σχεδίασης με σκοπό την αξιολόγηση των ιδιοτήτων της. [1] Τέλος, η καταγραφή των σχεδιαστικών επιλογών που εξετάστηκαν, των τελικών αποφάσεων που ελήφθησαν και των λόγων για τους οποίους ελήφθησαν είναι μια σημαντική διαδικασία.

Συγγραφή κώδικα και έλεγχος σε επίπεδο λειτουργικών μονάδων

Η λεπτομερής σχεδίαση ενός συστατικού του συστήματος πρέπει να γίνει με τέτοια μορφή ώστε να είναι δυνατή η υλοποίησή του με κάποια εκτελέσιμη γλώσσα προγραμματισμού. Μετά τη συγγραφή του κώδικα, μπορεί να επαληθευτεί η ορθή λειτουργία του σύμφωνα με κριτήρια ελέγχου, τα οποία έχουν ήδη προκαθοριστεί. Για το συγκεκριμένο στάδιο του κύκλου ζωής του λογισμικού, η έρευνα έχει επικεντρωθεί σε δυο τομείς. Αρκετές ερευνητικές προσπάθειες προσανατολίζονται στην αυτοματοποίηση της διαδικασίας συγγραφής κώδικα από μια λεπτομερή σχεδίαση χαμηλού επιπέδου. Οι περισσότερες εργασίες που έχουν γίνει με φορμαλιστικές μεθόδους (formal methods), βασίζονται στην υπόθεση ότι η μετάβαση από τη λεπτομερή σχεδίαση στην υλοποίηση είναι ένας μετασχηματισμός από μια μαθηματική αναπαράσταση σε μια άλλη, αλλά ταυτόχρονα θα πρέπει να μπορεί να αυτοματοποιηθεί πλήρως. Σε πιο πρακτικό επίπεδο, άλλες ερευνητικές εργασίες επικεντρώνονται στην αυτοματοποιημένη παραγωγή ελέγχων (tests) που προέρχονται από το αποτέλεσμα προγενέστερων δραστηριοτήτων οι οποίες μπορούν να εκτελούνται σε συγκεκριμένο τμήμα του κώδικα και με αυτό τον τρόπο μπορούν να επαληθεύουν τη σωστή συμπεριφορά του.

Ενοποίηση και έλεγχος

Αφού υλοποιηθεί επαρκής αριθμός συστατικών και ελεγχθούν ένα προς ένα με τη σειρά, αυτά τα συστατικά θα πρέπει να ενοποιηθούν όπως ακριβώς περιγράφεται στη σχεδίαση της αρχιτεκτονικής του συστήματος. Στο στάδιο αυτό εκτελούνται περαιτέρω έλεγχοι για να διασφαλιστεί η σωστή συμπεριφορά των συστατικών και η αποδεκτή χρήση οποιονδήποτε κοινόχρηστων πόρων. Στο συγκεκριμένο στάδιο είναι δυνατόν να εκτελεστούν και έλεγχοι αποδοχής, με τη συμμετοχή χρηστών, για να διασφαλιστεί ότι το σύστημα ικανοποιεί τις απαιτήσεις τους. Το σύστημα είναι έτοιμο για χρήση μόνο μετά την αποδοχή του ενοποιημένου συστήματος. Κάποιες φορές μπορεί να απαιτείται η πιστοποίηση του τελικού συστήματος σύμφωνα με προδιαγραφές που τίθενται από κάποιο ανεξάρτητο φορέα, όπως είναι ο διεθνής οργανισμός προτύπων (International Standards Organization ISO). Με αυτό τον τρόπο οι σχεδιαστές λαμβάνουν υπόψη τις συνέπειες της επικοινωνίας ανθρώπου-υπολογιστή για τη σχεδίαση των προϊόντων τους. [14]

Συντήρηση

Μετά την παράδοση του συστήματος για χρήση, όλες οι εργασίες που γίνονται στο σύστημα εντάσσονται στη διαδικασία της συντήρησης μέχρι τη χρονική στιγμή που θα καταστεί αναγκαία και απαραίτητη η ανάπτυξη μιας νέας έκδοσης του συστήματος ή η απόσυρσή του. Κατά συνέπεια, το μεγαλύτερο μέρος της ζωής ενός συστήματος αντιστοιχεί στη δραστηριότητα της συντήρησης, η οποία περιλαμβάνει τη διόρθωση σφαλμάτων του συστήματος τα οποία ανακαλύπτονται μετά την παράδοσή του στον τελικό χρήστη, καθώς και η προσαρμογή – βελτίωση των υπηρεσιών του συστήματος ώστε να ικανοποιούν τις απαιτήσεις που δεν έγιναν αντιληπτές σε προγενέστερα στάδια της ανάπτυξης. Ουσιαστικά, η συντήρηση παρέχει ανάδραση σε όλες τις άλλες δραστηριότητες του κύκλου ζωής του λογισμικού.

2.3. Μοντέλα κύκλου ζωής λογισμικού

2.3.1. Το μοντέλο του καταρράκτη (waterfall model)

Ήταν το πρώτο μοντέλο που δημιουργήθηκε από τον Royce, το 1970. Χαρακτηρίζεται από σειριακά βήματα (Phases), από ανάδραση ανάμεσα σε δυο γειτονικά βήματα και τη δημιουργία προδιαγραφών σε κάθε βήμα. Η κάθε φάση χρησιμοποιεί ενδιάμεσα προϊόντα τα οποία χρησιμοποιούνται από τις επόμενες φάσεις και για να απαλειφθούν τα σφάλματα τα οποία παράγονται για κάθε προϊόν το οποίο παράγεται ακολουθείται μια διαδικασία επικύρωσης ή επαλήθευσης τους.

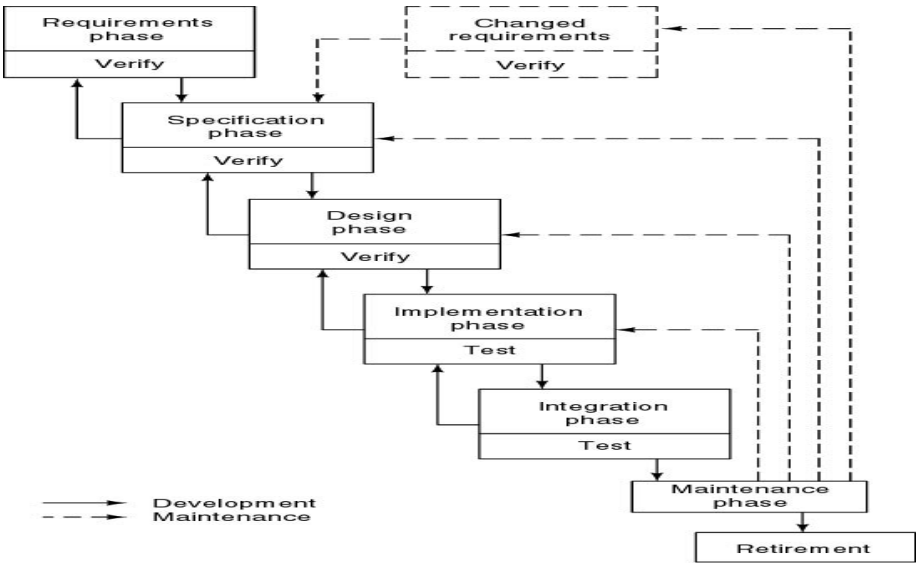
Τα προτερήματα του μοντέλου του καταρράκτη είναι τα εξής:

- Παραγωγή προδιαγραφών
- Διευκόλυνση της συντήρησης

Όσον αφορά στα μειονεκτήματα του μοντέλου επειδή είναι αρκετά υπάρχουν διάφορες παραλλαγές του μοντέλου καταρράκτη με σκοπό να ξεπεραστούν τα εξής:

- Προδιαγραφές δεν μπορούν να αλλάξουν στη πορεία δεν είναι ρεαλιστική παραδοχή
- Ο Χρήστης συμμετέχει μόνο στην αρχή
- Σειριακή και πλήρης ολοκλήρωση κάθε βήματος δεν είναι πάντα ενδεδειγμένη
- Η διαδικασία είναι δύσκολο να ελεγχθεί
- Ό χρήστης βλέπει το προϊόν πολύ αργά στη διάρκεια της διαδικασίας

Ένα τυπικό μοντέλο καταρράκτη παρουσιάζεται στο παρακάτω σχήμα 2.2.



Σχήμα 2.2. τυπικό μοντέλο καταρράκτη

2.3.2. Το μοντέλο πρωτοτυποποίησης (rapid prototyping)

Αποτελεί σχεδίαση πρωτότυπου ακολουθούμενου από το μοντέλο του Καταρράκτη. Στην πρωτοτυποποίηση λογισμικού βασικός σκοπός είναι η ανάπτυξη και ο έλεγχος των πραγματικών προδιαγραφών του συστήματος και είναι ως επί των πλείστων διαφορετικό από το τελικό προϊόν και όχι η επαλύθευση της σχεδίασης, όπως συμβαίνει στην πρωτοτυποποίηση υλικού. Η πρωτοτυποποίηση δεν πρέπει να είναι μέρος της σχεδίασης αλλά μόνο της συλλογής απαιτήσεων.

Διαμέσου του μοντέλου της πρωτοτυποποίησης οι πληροφορίες λαμβάνονται πολύ γρηγορότερα σε σχέση με το μοντέλο του καταρράκτη. Η χρήση της έγκειται τόσο στην εκπαίδευση των χρηστών όσο και στον έλεγχο του συστήματος. Μειώνονται τα λάθη και αποφεύγονται τυχόν παραλήψεις. Ένα πρωτότυπο δεν μπορεί να καλύπτει όλες τις απαιτήσεις του συστήματος και γι' αυτό θα πρέπει κάθε φορά να ορίζονται πλήρως. Γλώσσες προγραμματισμού που μπορούν να χρησιμοποιηθούν για πρωτοτυποποίηση είναι οι Smalltalk, Loops, Prolog και Lisp.

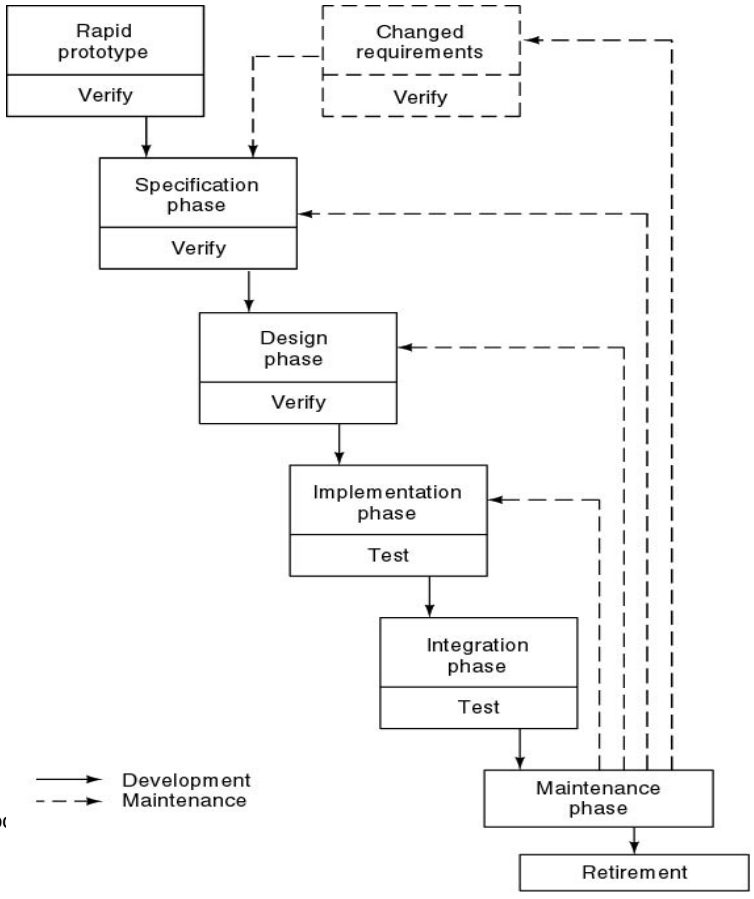
Τα προτερήματα του μοντέλου πρωτοτυποποίησης είναι τα εξής:

- Καλύτερη προδιαγραφή απαιτήσεων
- Καλύτερη μελέτη σκοπιμότητας
- Ο χρήστης συμμετέχει ενεργά στη διαδικασία συλλογής / μοντελοποίησης απαιτήσεων

Τα μειονεκτήματα του μοντέλου είναι:

- Περισσότερη εργασία απαιτείται για την παραγωγή του πρωτότυπου
- Λόγω χρονικών περιορισμών το πρωτότυπο γίνεται μέρος του συστήματος

Το μοντέλο πρωτοτυποποίησης παρουσιάζεται στο ακόλουθο σχήμα 2.3.



Σχήμα 2.3. μοντέλο πρωτοτυποποίησης

2.3.3. Το μοντέλο λειτουργικής επαύξησης (incremental model)

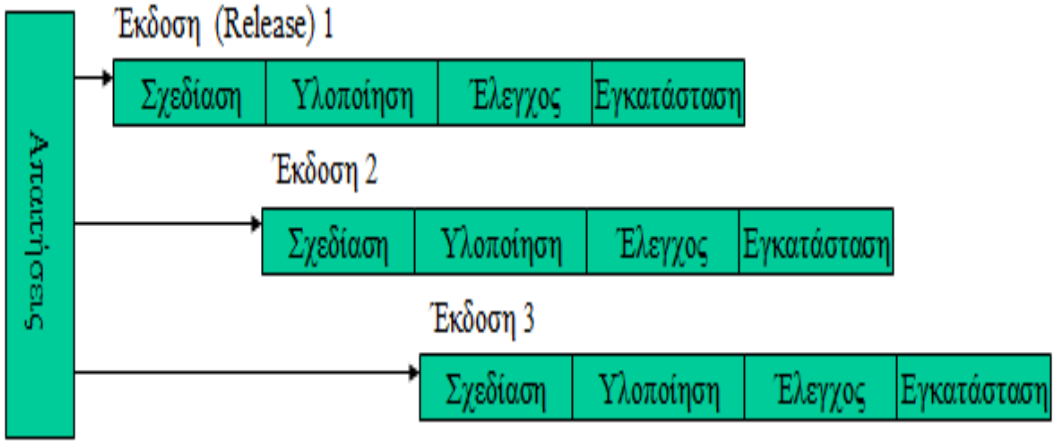
Αποτελεί μια εναλλακτική του μοντέλου του καταρράκτη , όντας μια εναλλακτική διαδικασία η οποία συνδυάζει τα πλεονεκτήματα της εξελικτικής προσέγγισης με τον έλεγχο που απαιτείται για μεγάλα συστήματα. Αναπτύσσονται προδιαγραφές και το μοντέλο παραδίδεται προσαυξημένο. Σε κάθε έκδοση προσθέτουμε και νέες λειτουργίες / ποιοτικά χαρακτηριστικά από ένα προκαθορισμένο σύνολο απαιτήσεων. Τα μέρη που αποτελούν το σύστημα αναπτύσσονται με προσαυξήσεις και παραδίδονται με αυτό τον τρόπο. Τίποτα δεν αλλάζει, εκτός αν βρεθούν λάθη ή παραλείψεις.

Τα προτερήματα του μοντέλου λειτουργικής επαύξησης είναι:

- Σε κάθε έκδοση έχουμε ένα σύστημα σε λειτουργία
- Καλύτερη διανομή κόστους στο χρόνο

Το βασικό μειονέκτημα του μοντέλου είναι ότι οι απαιτήσεις δεν μπορούν να αλλάζουν καθώς αναπτύσσεται το έργο.

Ένα τυπικό μοντέλο λειτουργικής επαύξησης παρουσιάζεται στο παρακάτω σχήμα 2.4.



Σχήμα 2.4. μοντέλο λειτουργικής επαύξησης

2.3.4. Σπειροειδές μοντέλο (spiral model)

Είναι ένα από τα δημοφιλέστερα μοντέλα το οποίο έχει ορισμένα από τα χαρακτηριστικά του μοντέλου του καταρράκτη. Έχει τη μορφή σπείρας και μπορεί να θεωρηθεί ένα γενικό μοντέλο που εμπεριέχει και μέρη άλλων μοντέλων. Κάθε γύρος στη σπείρα αναπαριστά μια φάση. Η διαδοχή των φάσεων δεν γίνεται ούτε σταθερά ούτε γραμμικά, ενώ η εκτέλεση τους μπορεί να γίνει είτε με τη φορά της σπείρας, είτε με την αντίθετη φορά, ανάλογα με το ρίσκο που εμπεριέχεται. Στη σπείρα το εμβαδόν που διαγράφει η ακτίνα στη σπείρα αναπαριστά το

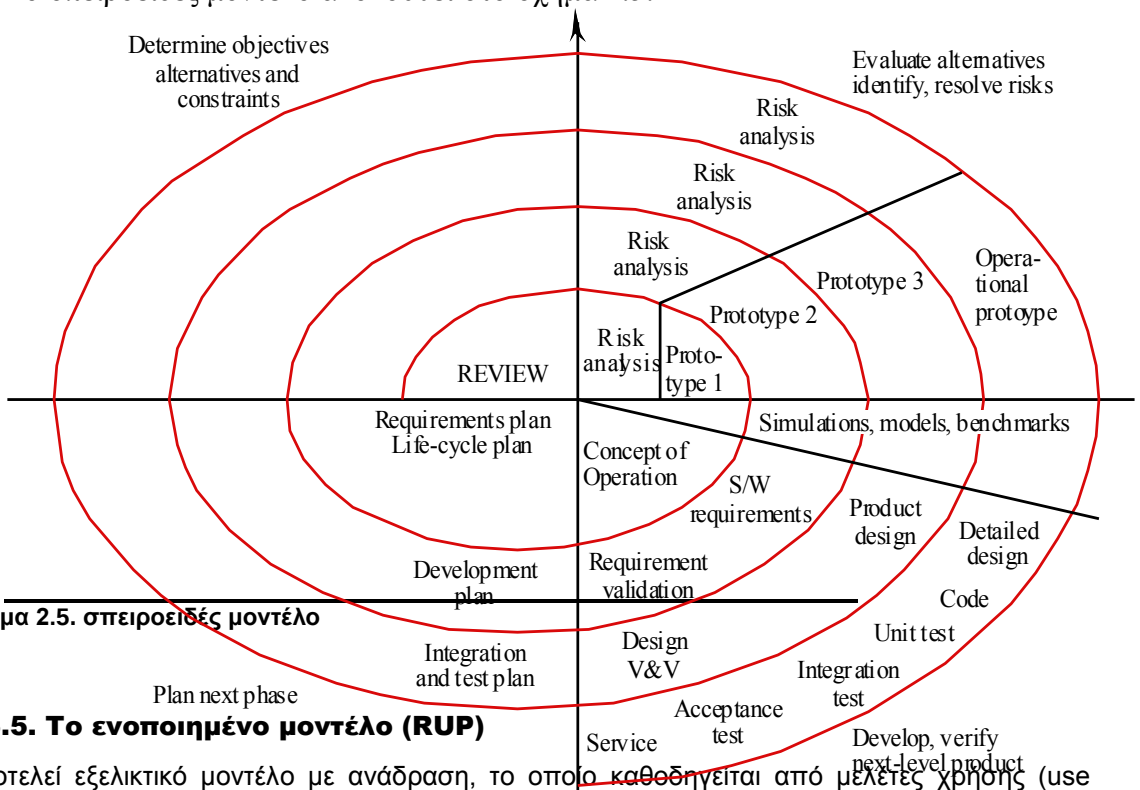
συσσωρευτικό κόστος και η γωνία που σχηματίζει η ακτίνα με τον οριζόντιο άξονα την πρόοδο που έχει σημειωθεί.

Σε κάθε γύρο οι φάσεις που ακολουθούνται είναι:

- Καθορισμός στόχων, εναλλακτικών λύσεων και υπολογισμός περιορισμών.
- Ανάλυση και υπολογισμός του κάθε ρίσκου και προσπάθεια μείωσής του.
- Ανάπτυξη και επαλήθευση ενδιάμεσου προϊόντος – εφόσον η προηγούμενη φάση δεν έδειξε
- κάποιο σοβαρό ρίσκο.
- Σχεδιασμός των επόμενων βημάτων.

Ουσιαστικά είναι το μοντέλο πρωτοτυποποίησης όπου στο τέλος κάθε βήματος κάνουμε έλεγχο σκοπιμότητας και ανάλυση ρίσκου. Αποτελεί όμως Πρωτοτυποποίηση για εφαρμογές υψηλού ρίσκου, ενώ εάν η ανάλυση ρίσκου αποτύχει τότε το έργο διακόπτεται. Είναι πιο κατάλληλο για μεγάλα έργα λόγω του μεγαλύτερου κόστους διαχείρισης. Σε περίπτωση που το έργο έχει ήδη προχωρήσει είναι πολύ δύσκολο να τερματιστεί ακόμη και αν η ανάλυση ρίσκου αποτύχει.

Το σπειροειδές μοντέλο ακολουθεί στο σχήμα 2.5.



Σχήμα 2.5. σπειροειδές μοντέλο

2.3.5. Το ενοποιημένο μοντέλο (RUP)

Αποτελεί εξελικτικό μοντέλο με ανάδραση, το οποίο καθοδηγείται από μελέτες χρήσης (use cases). Είναι αρχιτεκτονικο-κεντρικό μοντέλο (4+1 άποψεις – views) που χρησιμοποιεί την UML σαν γλώσσα μοντελοποίησης. Παρέχει πλούσιο πλαίσιο υποστήριξης της διαδικασίας.

Βασικές ροές

- Μοντελοποίηση επιχειρησιακού περιβάλλοντος (Business Modeling)
- Συγγραφή προδιαγραφών (Requirements)
- Ανάλυση και σχεδίαση (Architecture and Design)
- Υλοποίηση (Implementation)
- Ελεγχος (Test)
- Εγκατάσταση (Deployment)

Φάσεις

- Έναρξη (Inception)
- Επεξεργασία (Elaboration)

- Κατασκευή (Construction)
- Μετάβαση (Transition)

Ροές υποστήριξης

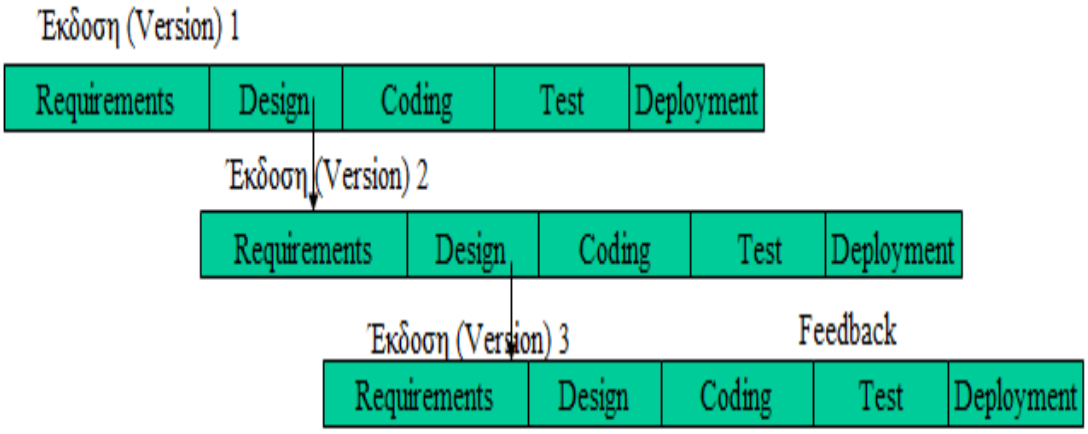
- Διοίκηση σχηματισμών λογισμικού (Configuration Management)
- Διοίκηση έργου (Project Management)
- Διαχείριση περιβάλλοντος ανάπτυξης (Development Environment Management)

Τα προτερήματα του μοντέλου είναι:

- Συνεχής συμμετοχή του χρήστη
- Καλή διαχείριση κρίσης

Το βασικό μειονέκτημα του μοντέλου είναι ότι μπορεί να γίνει ράβε – ξήλωνε τύπος μοντέλου.

Σχηματικά το ενοποιημένο μοντέλο έχει ως εξής στο σχήμα 2.6.



Σχήμα 2.6. το ενοποιημένο μοντέλο

2.3.6. Μοντέλο επαναχρησιμοποίησης λογισμικού (software reusability model)

Με το μοντέλο επαναχρησιμοποίησης λογισμικού, η παραγωγή λογισμικού υποθέτει την υλοποίηση τμημάτων από την αρχή. Η ανάγκη που δημιουργείται για επαναχρησιμοποίηση έτοιμων τμημάτων λογισμικού, τα οποία έχουν ήδη δοκιμαστεί, κερδίζει ολοένα και περισσότερο έδαφος, καθώς παρέχεται υψηλότερη ποιότητα στο σύστημα, αλλά και το κόστος παραγωγής και συντήρησης μειώνεται. Εκτός των άλλων, προσφέρει αξιοπιστία, καθώς, όπως αναφέραμε και προηγουμένως, τα τμήματα του λογισμικού που θα χρησιμοποιηθούν έχουν προγενέστερα ελεγχθεί και είναι σαφές ότι θα περιέχουν λιγότερα λάθη από τα καινούρια τμήματα του λογισμικού.

Τα επίπεδα που θα επαναχρησιμοποιηθούν μπορούν να είναι διαφορετικών μεγεθών, όπως :

- Επαναχρησιμοποίηση ολόκληρης της εφαρμογής (Application System reuse).
- Επαναχρησιμοποίηση μεγάλο μέρος μιας εφαρμογής (Sub-system reuse).
- Επαναχρησιμοποίηση τμημάτων του συστήματος που περιέχουν συλλογή συναρτήσεων (Module/object reuse).
- Επαναχρησιμοποίηση τμημάτων λογισμικού που υλοποιούν μια και μόνο συνάρτηση (Function reuse).

Η διαδικασία αυτή δεν είναι εύκολη, αφού παρουσιάζονται δυσκολίες, λόγω της ανυπαρξίας εργαλείων και κατάλληλων τεχνικών για τη συγκεκριμένη δουλειά, αλλά και λόγω της έλλειψης

προτύπων κατασκευής ενοτήτων (modules) λογισμικού που να μπορούν να επαναχρησιμοποιηθούν.

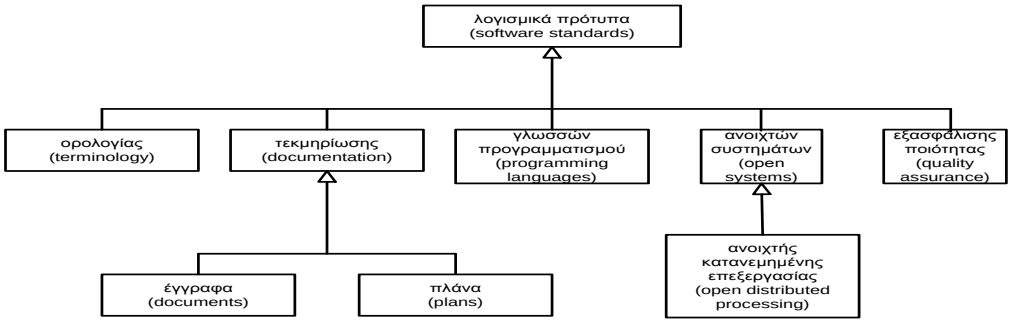
Από την άλλη, λόγω του ότι λιγότερα συστατικά του συστήματος χρειάζεται να σχεδιαστούν και εν συνεχεία να υλοποιηθούν, μειώνεται το κόστος της ανάπτυξης.

2.3.7. Άλλα μοντέλα

Άλλα Μοντέλα που σχετίζονται με τα Μοντέλα Κύκλου Ζωής και την Αναμενόμενη Ποιότητα Λογισμικού είναι:

2.3.7.1. ISO 9000

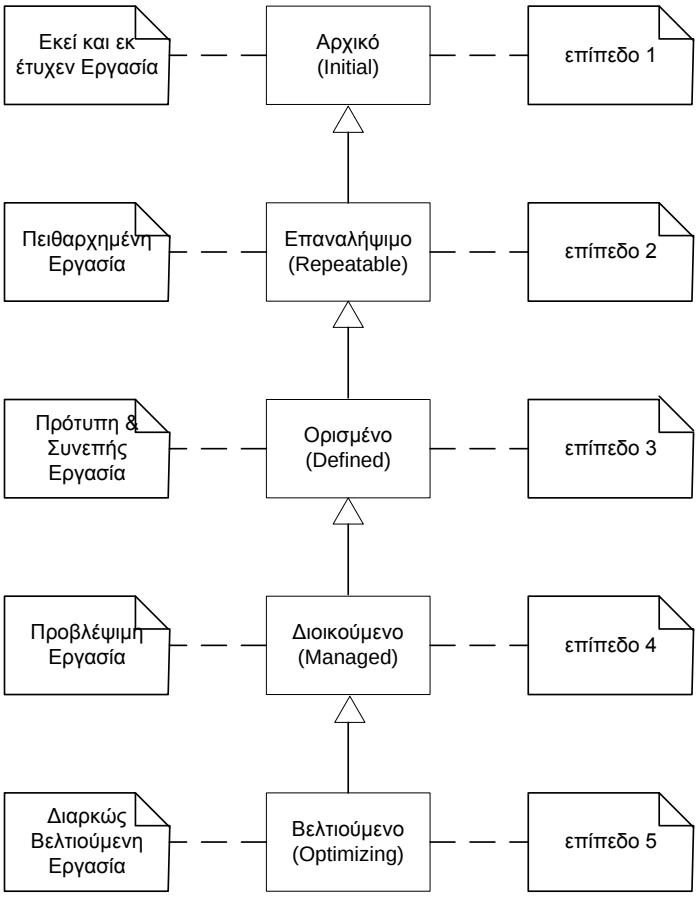
Αφορά στην εξασφάλιση της ποιότητας προϊόντων και υπηρεσιών, όπως φαίνεται και στο σχήμα 2.7.



Σχήμα 2.7. το μοντέλο ISO 9000

2.3.7.2. Capability Maturity Model

Αφορά στη σύνδεση της διαδικασίας με την αναμενόμενη ποιότητα του προϊόντος, όπως φαίνεται και στο σχήμα 2.8 η περιγραφή του.



Σχήμα 2.8. το μοντέλο Capability Maturity

2.4. Επιλογή κατάλληλου μοντέλου για τη δημιουργία ενός εκπαιδευτικού λογισμικού

Λαμβάνοντας υπόψη τα μοντέλα που αναλύσαμε παραπάνω και λαμβάνοντας υπόψη ότι η ανάπτυξη ενός εκπαιδευτικού λογισμικού δεν είναι μια αυτόματη διαδικασία, αλλά περιλαμβάνει τη συμμετοχή πολλών παραγόντων και διαφορετικών ειδικοτήτων ανθρώπους, αποκλείουμε τα μοντέλα αυτόματου προγραμματισμού.

Το μοντέλο του καταρράκτη παρόλο που είναι μια από τις πιο δημοφιλείς επιλογές, το οποίο απορρίπτουμε για το λόγο ότι η ανάλυση του συστήματος και ο εντοπισμός των απαιτήσεων σπανίως μπορούν να ολοκληρωθούν στην αρχή ενός έργου και επίσης, είμαστε σε θέση να γνωρίζουμε αν έχουμε κατασκευάσει το σύστημα που επιθυμούμε μόνο αφού έχει

ολοκληρωθεί η ανάπτυξη του συστήματος. Οπότε, το συγκεκριμένο μοντέλο και παρεμφερή σε αυτό μοντέλα αποκλείονται.

Το μοντέλο επαναχρησιμοποίησης, παρόλο αυξάνει την αξιοπιστία του συστήματος, λόγω του γεγονότος ότι τα επαναχρησιμοποιούμενα τμήματα είναι πιο αξιόπιστα από τα νέα διότι έχουν ελεγχθεί στην πράξη σε άλλα συστήματα, κάτω από ρεαλιστικές συνθήκες χρήσης και ο χρόνος παραγωγής λογισμικού μειώνεται, δε θα μπορούσε να συντελέσει σε ολοκληρωμένη υλοποίηση ενός εκπαιδευτικού λογισμικού, καθώς η παραγωγή λογισμικού υποθέτει την υλοποίηση τμημάτων από την αρχή και προσπαθεί να αξιοποιήσει την επαναχρησιμοποίηση υφιστάμενων τμημάτων λογισμικού.

Το μοντέλο πρωτοτυποποίησης παρόλο που ανιχνεύει γρήγορα τις ανάγκες και τα προβλήματα πριν την ανάπτυξη μεγάλου μέρους του εκπαιδευτικού λογισμικού και είναι κατάλληλο για συστήματα με αβεβαιότητα στις απαιτήσεις, δεν υπάρχει διακριτότητα στα στάδια ανάπτυξης, αφού κάθε φορά μπορεί να αλλάζει οποιαδήποτε παράμετρος ανάπτυξης. Το βασικό μειονέκτημα του μοντέλου της πρωτοτυποποίησης είναι ότι το κόστος ανάπτυξής του αποτελεί ένα μεγάλο μέρος του συνολικού κόστους του συστήματος που αναπτύσσεται και πολλές φορές είναι οικονομικά πιο συμφέρον να μεταβληθεί το τελικό προϊόν από το να δημιουργηθεί ένα πρωτότυπο.

Το σπειροειδές μοντέλο διαφέρει από τα προαναφερθέντα μοντέλα, καθώς σε αυτό υπολογίζεται πριν την έναρξη κάθε φάσης ο κίνδυνος, γεγονός που ουσιαστικά αποτελεί και το βασικό του πλεονέκτημα, αν και, από πρακτική άποψη, ο υπολογισμός και η ανάλυση του κινδύνου δεν είναι εύκολη υπόθεση. Σε κάθε γύρο που διανύεται πραγματοποιείται καθορισμός των στόχων, των εναλλακτικών λύσεων, υπολογίζονται οι περιορισμοί και γίνεται σχεδιασμός των επόμενων βημάτων. Λόγω του σχεδιασμού του κάθε βήματος, δεν υπάρχει ο κίνδυνος να παραμείνει μια φάση ανάπτυξης του λογισμικού χωρίς να υλοποιηθεί και συγχρόνως μετά από κάθε φάση έχουμε ένα ενδιάμεσο πρωτότυπο που μπορεί σταδιακά να αξιολογείται.

Λαμβάνοντας υπόψη τα ανωτέρω, αλλά και το γεγονός ότι η ανάπτυξη εκπαιδευτικού λογισμικού χρησιμοποιεί κάποιες γενικές τάσεις και κατευθύνσεις από ήδη υπάρχοντα μοντέλα, θα επιλέξουμε το σπειροειδές μοντέλο για την ανάπτυξη του συστήματός μας, αφού πρώτα το προσαρμόσουμε στις δικές μας ανάγκες οι οποίες αφορούν στο περιβάλλον ανάπτυξης αλλά και το σύστημα που υλοποιούμε.

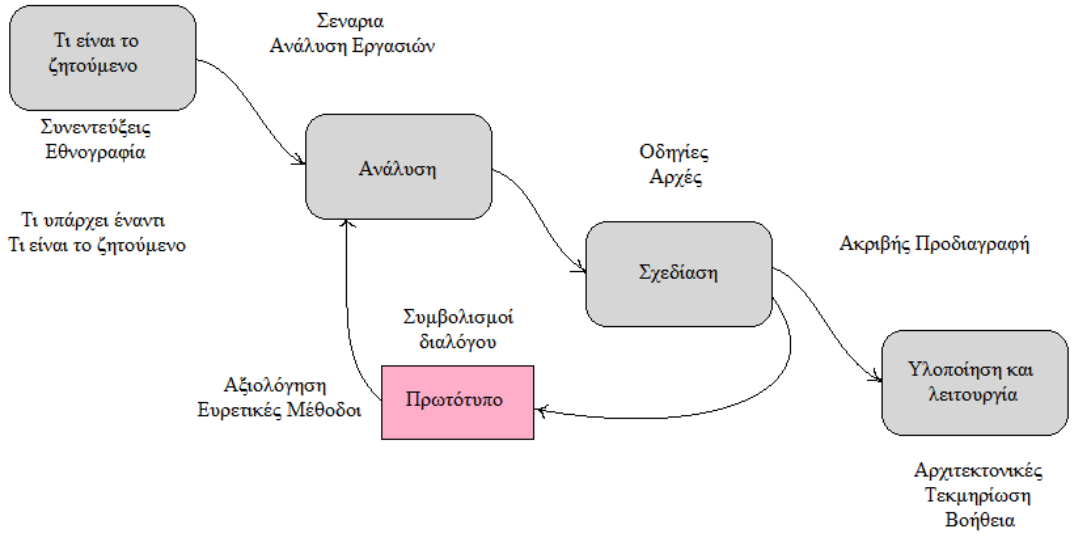
Κεφάλαιο 3
Σχεδίαση λογισμικού και αιτιολόγηση σχεδίασης

3.1. Σχεδίαση λογισμικού

Σχεδίαση είναι η επίτευξη στόχων εντός συγκεκριμένων περιορισμών. Σε ένα βαθμό, η κατανόηση που χρειαζόμαστε για τη σχεδίαση ενός συστήματος σχετίζεται με τις καταστάσεις και το ευρύτερο νοηματικό πλαίσιο στο οποίο εντάσσεται το συγκεκριμένο πρόβλημα σχεδίασης που αντιμετωπίζουμε. Ο χρυσός κανόνας της σχεδίασης είναι να κατανοούμε τους υπολογιστές αναφορικά με τους περιορισμούς, τις δυνατότητες, τα εργαλεία και τις πλατφόρμες που διαθέτουν, αλλά και να κατανοούμε τους ανθρώπους αναφορικά με τις ψυχολογικές – κοινωνικές απόψεις και το ανθρώπινο σφάλμα. [15]

3.1.1. Διαδικασία – Στάδια της σχεδίασης

Συχνά, όσοι ασχολούνται με την επικοινωνία ανθρώπου – υπολογιστή, καλούνται να αναπτύξουν ένα σύστημα σε πολύ σύντομο χρονικό διάστημα. Στο σχήμα 3.1 που ακολουθεί, παρουσιάζεται μια θεώρηση των βασικών φάσεων και ενός βρόγχου επανάληψης αναφορικά με τη σχεδίαση, εστιάζοντας στη σχεδίαση της διάδρασης [21].



Σχήμα 3.1. Η διαδικασία σχεδίασης της διάδρασης

- Απαιτήσεις – ποιο είναι το ζητούμενο : το πρώτο στάδιο είναι ο καθορισμός του τι ακριβώς χρειάζεται. Πριν γίνει αυτό, είναι αναγκαίο να εξακριβωθεί τι συμβαίνει επί του παρόντος. Υπάρχουν αρκετές τεχνικές που χρησιμοποιούνται για το σκοπό αυτό στην επικοινωνία ανθρώπου-υπολογιστή, όπως είναι

συνεντεύξεις με ανθρώπους και βιντεοσκόπησή τους, εξέταση των εγγράφων και των αντικειμένων με τα οποία δουλεύουν, αλλά και η άμεση παρατήρησή τους.

- Ανάλυση : τα αποτελέσματα της παρατήρησης και των συνεντεύξεων θα πρέπει να οργανώνονται με κάποιο τρόπο ώστε να αναδεικνύονται τα σημαντικά ζητήματα και να επικοινωνούν με επόμενες φάσεις της σχεδίασης. Γίνεται εξέταση των σεναρίων και των ιστοριών διάδρασης, τα οποία μπορούν να χρησιμοποιούνται σε συνδυασμό με μεθόδους όπως η ανάλυση εργασιών αλλά και μόνα τους για την καταγραφή της πραγματικής διάδρασης. Αυτές οι τεχνικές μπορούν να χρησιμοποιούνται για την αναπαράσταση τόσο μιας συγκεκριμένης κατάστασης, όσο και της επιθυμητής κατάστασης.
- Σχεδίαση : υπάρχει μια κεντρική φάση όταν μεταβαίνουμε από αυτό που επιθυμούμε να κάνουμε και στον τρόπο με τον οποίο θα το υλοποιήσουμε. Υπάρχουν πολλοί κανόνες, οδηγίες, βασικές αρχές σχεδίασης που μπορούν να χρησιμοποιούνται σα βοηθήματα σε αυτή τη φάση. [13] Επίσης, πρέπει να καταγράφουμε της σχεδιαστικές επιλογές μας με κάποιο τρόπο, υπάρχουν διάφοροι συμβολισμοί και μέθοδοι για να γίνει αυτό, συμπεριλαμβανομένων των συμβολισμών και μεθόδων που χρησιμοποιούνται για την καταγραφή της υφιστάμενης κατάστασης.
- Επανάληψη και πρωτοτυποποίηση : οι άνθρωποι είναι πολύπλοκα όντα και για το λόγο αυτό δεν μπορεί κανείς να περιμένει ότι μια σχεδίαση θα είναι επιτυχημένη από την πρώτη κιόλας στιγμή. Συνεπώς, χρειάζεται να αξιολογήσουμε μια σχεδίαση για να εξακριβώσουμε πόσο καλά λειτουργεί και σε ποια σημεία μπορούν να γίνουν βελτιώσεις. Επομένως, η αξιολόγηση είναι απαραίτητη. Ορισμένες μορφές αξιολόγησης της σχεδίασης μπορούν να γίνονται “επί χάρτου”, αλλά είναι δύσκολο να αποκτήσουμε πραγματική γνώση της λειτουργικότητας της σχεδίασης χωρίς να το δοκιμάσουμε στην πράξη. Για το λόγο αυτό. Στις περισσότερες περιπτώσεις, η σχεδίαση διεπιφανειών χρήστη προϋποθέτει τη δημιουργία κάποιας μορφής πρωτοτύπου, δηλαδή, την παραγωγή πρώιμων εκδόσεων του συστήματος που θα δοκιμάζονται στην πράξη από πραγματικούς χρήστες.
- Υλοποίηση και ανάπτυξη : τέλος, όταν είμαστε πλέον ικανοποιημένοι με τη σχεδίασή μας, θα πρέπει να την υλοποιήσουμε, δηλαδή να δημιουργήσουμε ένα σύστημα και να το αναπτύξουμε. Το στάδιο αυτό περιλαμβάνει τη συγγραφή κώδικα, πιθανώς την κατασκευή υλικού εξοπλισμού, τη συγγραφή πληροφοριών τεκμηρίωσης και εγχειριδίων, δηλαδή όλα όσα χρειάζεται το πραγματικό σύστημα που επρόκειτο να παραδοθεί στους χρήστες.

3.2. Αιτιολόγηση της σχεδίασης

Κατά τη σχεδίαση οποιουδήποτε συστήματος υπολογιστή λαμβάνονται πολλές αποφάσεις καθώς το προϊόν εξελίσσεται από ένα σύνολο ασαφών απαιτήσεων του χρήστη σε ένα τελικό σύστημα. Πολλές φορές είναι δύσκολο να αναδημιουργήσει κανείς τους λόγους που κρύβονται πίσω από διάφορες σχεδιαστικές αποφάσεις [21].

Αιτιολόγηση της σχεδίασης (design rationale) είναι οι πληροφορίες που εξηγούν γιατί ένα σύστημα υπολογιστή είναι όπως είναι και περιλαμβάνει τη δομική ή αρχιτεκτονική περιγραφή του συστήματος και την περιγραφή των λειτουργιών ή της συμπεριφοράς του. Σχεδιάζεται με μια δραστηριότητα αναλογισμού και τεκμηρίωσης η οποία λαμβάνει χώρα καθ’ όλη τη διάρκεια του κύκλου ζωής του λογισμικού. Αρκετά οφέλη προκύπτουν από την πρόσβαση στην αιτιολόγηση της σχεδίασης και αναλύονται παρακάτω:

- όταν έχει ρητή μορφή, η αιτιολόγηση της σχεδίασης παρέχει έναν μηχανισμό για την επικοινωνία μεταξύ των μελών της ομάδας σχεδίασης έτσι ώστε τα επόμενα στάδια της σχεδίασης και συντήρησης του συστήματος να είναι δυνατή η κατανόηση κρίσιμων αποφάσεων που ελήφθησαν, των εναλλακτικών λύσεων που μελετήθηκαν και του λόγου για τον οποίο έγινε μια συγκεκριμένη εναλλακτική επιλογή έναντι των υπολοίπων. Αυτό μας βοηθάει στην αποφυγή λανθασμένων υποθέσεων στο μέλλον.

- η γνώση που συσσωρεύεται με τη μορφή των αιτιολογήσεων της σχεδίασης για ένα σύνολο συστημάτων μπορεί να επαναχρησιμοποιηθεί για τη μεταφορά μιας λύσης σε μια άλλη περίπτωση με παρόμοιες ανάγκες, η οποία δούλεψε σε μια περίπτωση. Το σκεπτικό αιτιολόγησης της σχεδίασης μπορεί να καταγράψει το γενικότερο πλαίσιο στο οποίο βασίστηκε μια σχεδιαστική απόφαση. Με αυτό τον τρόπο μια διαφορετική ομάδα σχεδίασης μπορεί να εξακριβώσει αν μια παρόμοια αιτιολόγηση είναι κατάλληλη για το δικό της σύστημα.
- η προσπάθεια που απαιτείται για την παραγωγή μιας αιτιολόγησης της σχεδίασης υποχρεώνει το σχεδιαστή να μελετά και να σταθμίζει πιο προσεκτικά τις αποφάσεις που λαμβάνει. Η τεχνική που χρησιμοποιείται για την αιτιολόγηση της σχεδίασης μπορεί να τον βοηθήσει σε αυτό το θέμα, υποδεικνύοντας το πώς διατυπώνονται τα επιχειρήματα που αιτιολογούν ή απορρίπτουν μια συγκεκριμένη σχεδιαστική επιλογή.

Στο πεδίο της επικοινωνίας ανθρώπου-υπολογιστή, η αιτιολόγηση της σχεδίασης ήταν και είναι ιδιαίτερα σημαντική για αρκετούς λόγους:

- συνήθως δεν υπάρχει μόνο μια βέλτιστη εναλλακτική λύση για τη σχεδίαση. Στις περισσότερες περιπτώσεις ο σχεδιαστής αντιμετωπίζει ένα σύνολο συμβιβασμών μεταξύ διαφορετικών εναλλακτικών λύσεων.
- ακόμη και να υπήρχε μια βέλτιστη λύση για μια συγκεκριμένη σχεδιαστική απόφαση, το σύνολο των εναλλακτικών λύσεων είναι τόσο μεγάλο, που είναι απίθανο να μπορέσει να τις ανακαλύψει όλες ο σχεδιαστής. Σε αυτή την περίπτωση είναι σημαντικό ο σχεδιαστής να υποδείξει όλες τις εναλλακτικές λύσεις που έχουν μελετηθεί. Κατόπιν, σε ένα επόμενο στάδιο, είναι δυνατό να εξακριβωθεί αν δεν επέλεξε τη βέλτιστη λύση ή αν προβληματίστηκε πάνω σε αυτή και αν την απέρριψε για κάποιο λόγο.
- η ευχρηστία ενός διαδραστικού συστήματος εξαρτάται σε μεγάλο βαθμό από το περιβάλλον χρήσης του. Ακόμα και μια διεπιφάνεια με τα πιο εντυπωσιακά γραφικά του κόσμου είναι άχρηστη, αν ο τελικός χρήστης δεν έχει πρόσβαση σε μιας υψηλής ποιότητας οθόνη γραφικών ή σε μια δεικτική συσκευή. Η διατύπωση και καταγραφή του πλαισίου του οποίου λαμβάνεται μια σχεδιαστική απόφαση μπορεί να βοηθήσει μελλοντικά στη σχεδίαση νέων συστημάτων. Αν το πλαίσιο παραμένει ίδιο, μπορεί να υιοθετηθεί η παλιά αιτιολόγηση χωρίς αναθεώρηση.

Υπάρχουν διάφορες τεχνικές αιτιολόγησης της σχεδίασης σύμφωνα με την κατηγοριοποίηση που έχει γίνει από τους Lee και Lai [6].

- προσανατολισμένη σε διεργασίες αιτιολόγηση της σχεδίασης : η πρώτη κατηγορία τεχνικών επικεντρώνεται στη δημιουργία ενός ιστορικού των σχεδιαστικών αποφάσεων που χρησιμοποιούνται κατά τη διάρκεια των πραγματικών συζητήσεων που γίνονται για τη σχεδίαση ενός συστήματος. Αυτές οι τεχνικές αναφέρονται με τον όρο “προσανατολισμένη σε διεργασίες αιτιολόγηση της σχεδίασης” (process oriented design rationale), επειδή ενοποιούνται με την πραγματική διαδικασία σχεδίασης. Το σύστημα IBIS (Issue Based Information System) του Rittel βασίζεται σε σύστημα πληροφοριών και είναι μια μορφή αναπαράστασης και σχεδίασης και προγραμματισμού διαλόγων. [9] Στο IBIS δημιουργείται μια ιεραρχική δομή. Ορίζεται το ριζικό ζήτημα που αντιπροσωπεύει το κύριο πρόβλημα ή ερώτημα με το οποίο σχετίζεται ένα επιχείρημα και διατυπώνονται διάφορες γνώμες σαν πιθανές λύσεις για το ριζικό ζήτημα και, τέλος, κάθε γνώμη υποστηρίζεται ή αντικρούεται από επιχειρήματα τα οποία τροποποιούν τη σχέση ζητήματος και γνώμης. Χρησιμοποιείται κατά τη διάρκεια των συσκέψεων που πραγματοποιούνται για τη σχεδίαση ενός συστήματος και διατηρεί τη σειρά μελέτης των ζητημάτων και λήψης αποφάσεων για ένα συγκεκριμένο σύστημα.
- ανάλυση χώρου σχεδίασης : Οι τεχνικές αυτής της κατηγορίας δεν αφορούν τόσο τις ιστορικές ή τις προσανατολισμένες στις διεργασίες πληροφορίες, αλλά τη δομή του πεδίου ορισμού όλων των εναλλακτικών επιλογών σχεδίασης, το οποίο μπορεί να αναδημιουργηθεί με την “εκ των υστέρων” εξέταση της δραστηριότητας σχεδίασης. Αυτή η προσέγγιση προτάθηκε από τον MacLean και τους συνεργάτες του. [8] Η προσανατολισμένη στη δομή προσέγγιση δεν καταγράφει ιστορικές πληροφορίες, αλλά καταγράφει το πλήρες ιστορικό της στιγμής (story of the moment) σαν μια ανάλυση του

πεδίου ορισμού των εναλλακτικών επιλογών σχεδίασης. Αρχικά, ο χώρος σχεδίασης δομείται από ένα σύνολο ερωτήσεων οι οποίες αντιπροσωπεύουν τα βασικά ζητήματα της σχεδίασης. Επειδή η ανάλυση του χώρου σχεδίασης είναι προσανατολισμένη στη δομή, δεν είναι τόσο σημαντικό οι ερωτήσεις που καταγράφει να είναι οι πραγματικές που γίνονται κατά τις συσκέψεις για τη σχεδίαση, μπορούν να αντιπροσωπεύουν έναν προσυμφωνημένο χαρακτηρισμό των ζητημάτων που εγείρονται, βάσει του αναλογισμού και της κατανόησης των πραγματικών δραστηριοτήτων της σχεδίασης.

- αιτιολόγηση της σχεδίασης στο επίπεδο ψυχολογίας: Η τελευταία κατηγορία τεχνικών αιτιολόγησης της σχεδίασης επικεντρώνεται στην καταγραφή των επιχειρημάτων γύρω από την ψυχολογία του χρήστη, όπως διαφαίνονται σε ένα διαδραστικό σύστημα και των εργασιών που εκτελούνται επ' αυτών. Αυτή η βασιζόμενη στην ψυχολογία αιτιολόγηση της σχεδίασης παρουσιάστηκε από τους Carroll και Rosson. [7]Συγκεκριμενοποιεί τα ψυχολογικά ερεθίσματα για την ευχρηστία που υπάρχει σε οποιοδήποτε διαδραστικό σύστημα, με στόχο την καλύτερη προσαρμογή ενός συστήματος στις εργασίες του κάθε χρήστη. Ο σκοπός της ψυχολογικής αιτιολόγησης της σχεδίασης είναι να υποστηρίζει το φυσιολογικό κύκλο εργασίας – κατασκευής στη δραστηριότητα της σχεδίασης.

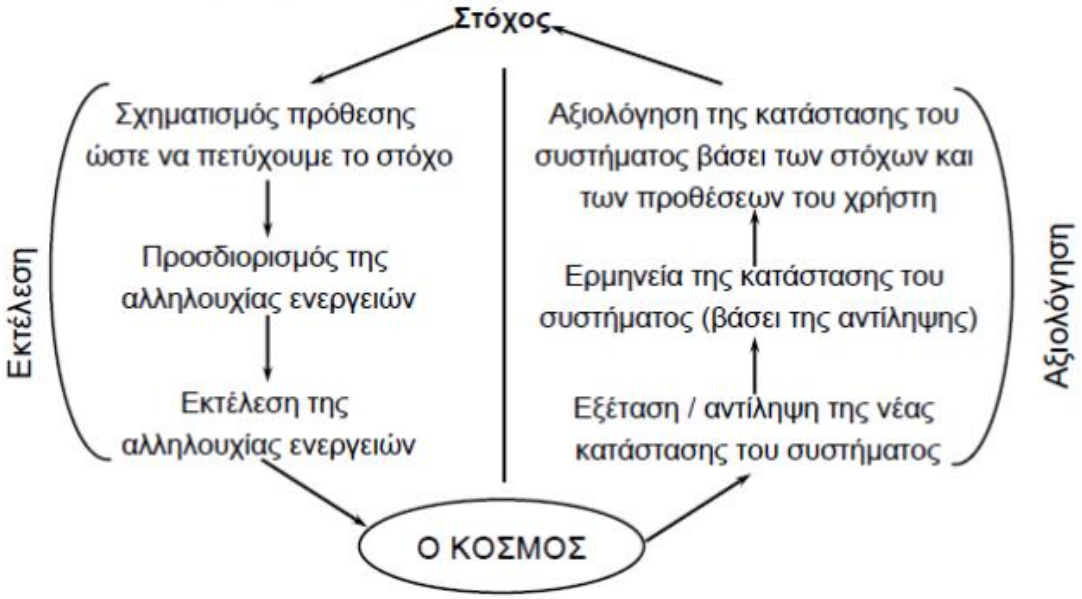
Κεφάλαιο 4
Μοντέλα διάδρασης

Τα διαδραστικά συστήματα πρέπει να είναι WYSIWYG (what you see is what you get, παίρνετε αυτό που βλέπετε), δηλαδή θα πρέπει να είναι συνεπή, να διαθέτουν τη λειτουργία αναίρεσης. Ο σκοπός ενός διαδραστικού συστήματος είναι η υποβοήθηση του χρήστη για την επίτευξη στόχων σε κάποιο πεδίο εφαρμογής.

4.1. Ο κύκλος εκτέλεσης – αξιολόγησης

Το μοντέλο διάδρασης όπως παρουσιάστηκε από τον Norman, είναι εκείνο με τη μεγαλύτερη επιρροή στην επιστήμη της επικοινωνίας ανθρώπου-υπολογιστή. [10] Ο χρήστης διατυπώνει ένα πλάνο ενεργειών. Αφού εκτελεστεί το πλάνο ή κάποιο μέρος αυτού, ο χρήστης παρατηρεί τη διεπιφάνεια του υπολογιστή για να αξιολογήσει το αποτέλεσμα και να αποφασίσει τις επόμενες ενέργειες και βήματά του. Ο κύκλος διάδρασης μπορεί να διαχωριστεί σε δυο φάσεις:

την εκτέλεση και την αξιολόγηση. Οι ανωτέρω φάσεις χωρίζονται σε επτά συνολικά στάδια, όπως φαίνονται και στο σχήμα 4.1 που ακολουθεί:



Σχήμα 4.1. τα στάδια του μοντέλου διάδρασης του Norman

1. ορισμός του στόχου
2. σχηματισμός πρόθεσης
3. προσδιορισμός αλληλουχίας ενεργειών
4. εκτέλεση των ενεργειών
5. αντίληψη της κατάστασης του συστήματος
6. ερμηνεία της κατάστασης του συστήματος
7. αξιολόγηση της κατάστασης του συστήματος βάσει των στόχων και των προθέσεών του

Κάθε ένα από τα στάδια αφορούν σε μια δραστηριότητα του χρήστη. Αρχικά, ο χρήστης ορίζει ένα στόχο, ο οποίος αντιστοιχεί στην άποψη του χρήστη αναφορικά με το τι χρειάζεται να γίνει και προσδιορίζεται μέσα στο πλαίσιο που ορίζει το πεδίο εφαρμογής. Ο ορισμός του στόχου από το χρήστη μπορεί να είναι ανακριβής, γι' αυτό θα πρέπει να μεταφραστεί σε μια πιο συγκεκριμένη πρόθεση και στις πραγματικές ενέργειες που πρέπει να γίνουν για να επιτευχθεί ο συγκεκριμένος στόχος πριν εκτελεστεί από το χρήστη. Ο χρήστης εξετάζει τη νέα κατάσταση του συστήματος μετά την εκτέλεση όλων των ενεργειών και την ερμηνεύει με βάσει τις προσδοκίες του και αυτό που πραγματικά θέλει. Αν η κατάσταση του συστήματος καλύπτει το χρήστη και το στόχο που είχε, τότε ο υπολογιστής με επιτυχία εκτέλεσε αυτό που ήθελε ο χρήστης και έτσι υπήρξε μια επιτυχής διάδραση, αντίθετα, ο χρήστης θα πρέπει να ορίσει ένα νέο στόχο και να επαναλάβει την όλη διαδικασία.

Ο Norman χρησιμοποιεί το μοντέλο διάδρασης για να δείξει ότι ορισμένες διεπιφάνειες προκαλούν προβλήματα στους χρήστες και δεν έχουμε επιτυχή διάδραση. Αυτά τα προβλήματα περιγράφονται με τους όρους χάσμα εκτέλεσης και χάσμα αξιολόγησης.

- Το χάσμα εκτέλεσης
    - Μας επιτρέπει η διεπαφή να εκτελέσουμε τις ενέργειες που απαιτούνται από την αντίστοιχη πρόθεση;

  είναι η διαφορά που υπάρχει μεταξύ της άποψης του χρήστη σχετικά με τις ενέργειες που απαιτούνται για την επίτευξη του στόχου και των ενεργειών που επιτρέπονται από το σύστημα. Έχουμε αποτελεσματική και επιτυχή διάδραση όταν το σύστημα μπορεί να ικανοποιήσει τους στόχους του χρήστη.
- Το χάσμα αξιολόγησης

- πόσο εύκολα μπορεί κανείς να προσδιορίσει τη λειτουργία της;
- να προσδιορίσει τις εφικτές δράσεις;
- να προσδιορίσει την αντιστοίχιση από την πρόθεση στη φυσική κίνηση;
- να πραγματοποιήσει την ενέργεια;
- να προσδιορίσει αν το σύστημα βρίσκεται στην επιθυμητή κατάσταση;
- να ερμηνεύσει την κατάσταση του συστήματος;
- να προσδιορίσει σε ποια κατάσταση είναι το σύστημα;

είναι η απόσταση μεταξύ της φυσικής παρουσίασης της κατάστασης του συστήματος και αυτού που περιμένει ο χρήστης. Αν ο χρήστης είναι σε θέση να αξιολογήσει άμεσα την παρουσίαση του συστήματος με βάση αυτό που έχει ως στόχο, τότε το χάσμα αξιολόγησης είναι μικρό και επομένως υπάρχει επιτυχής και αποτελεσματική διάδραση.

Πλεονεκτήματα του μοντέλου

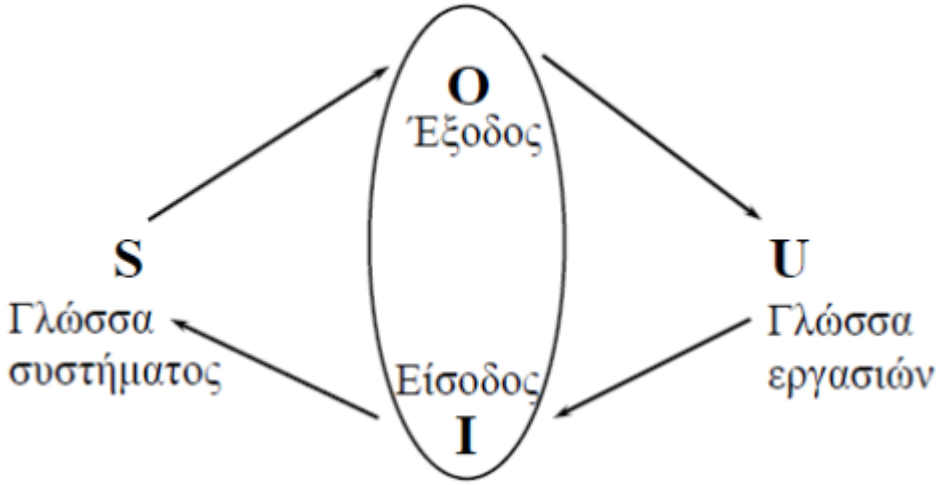
- Παρουσιάζει τη διάδραση με τρόπο σαφή και διαισθητικό.
- Επιτρέπει την ένταξη λεπτομερέστερων, εμπειρικών και αναλυτικών εργασιών σε ένα κοινό πλαίσιο.

Μειονεκτήματα του μοντέλου

- Εξετάζει το σύστημα μόνο μέχρι τη διεπαφή.
- Επικεντρώνεται εξ' ολοκλήρου στην άποψη του χρήστη για τη διάδραση. Δεν επιχειρεί να ασχοληθεί με την επικοινωνία του συστήματος μέσω της διεπαφής.

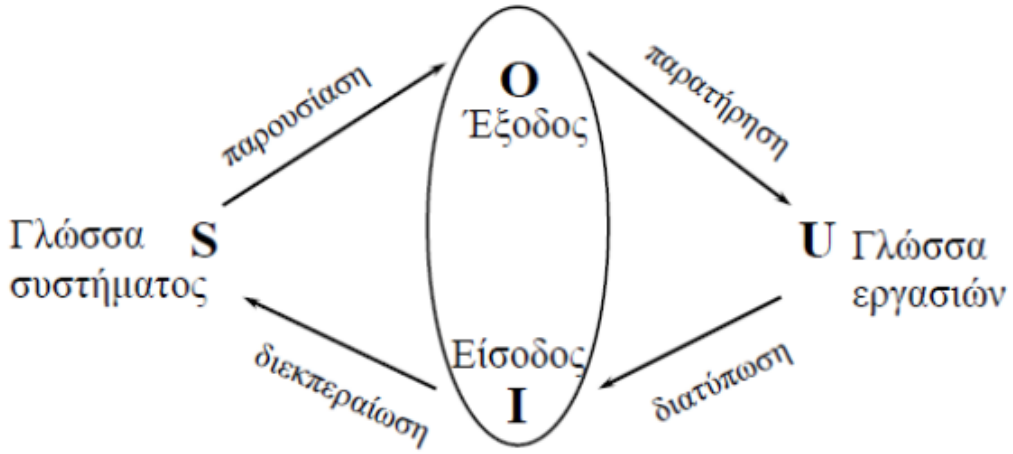
4.2. Το πλαίσιο της αναφοράς

Το πλαίσιο αναφοράς της διάδρασης (interaction framework) επιχειρεί μια πιο ρεαλιστική περιγραφή της διάδρασης και τη διακρίνει σε τέσσερα βασικά συστατικά τα οποία παρουσιάζονται στο σχήμα 4.2.



Σχήμα 4.2. το γενικό πλαίσιο αναφοράς

Οι κόμβοι αντιπροσωπεύουν τα τέσσερα βασικά συστατικά ενός διαδραστικού συστήματος, το σύστημα S (System), το χρήστη U (User), την είσοδο I (Input) και την έξοδο O (Output), όπου καθένα από τα οποία έχει τη δική του γλώσσα. Η είσοδος και η έξοδος αποτελούν τη διεπιφάνεια (Interface). Όπως παρατηρούμε, η διεπιφάνεια βρίσκεται μεταξύ του χρήστη και του συστήματος. Στο σχήμα 4.3 που ακολουθεί φαίνονται τα τέσσερα βήματα στον κύκλο διάδρασης.



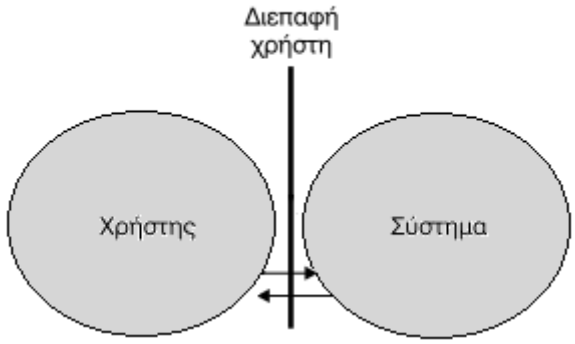
Σχήμα 4.3. μεταφράσεις μεταξύ των συστατικών

Όπως φαίνεται και στο σχήμα, ο χρήστης ξεκινάει τον κύκλο διάδρασης με τον ορισμό ενός στόχου και μιας εργασίας για την επίτευξη αυτού του στόχου. Ο μοναδικός τρόπος όπου ο χρήστης μπορεί να χειριστεί το σύστημα είναι μέσω της εισόδου, οπότε η εργασία πρέπει να διατυπωθεί στη γλώσσα εισόδου, η οποία μεταφράζεται σε γλώσσα συστήματος, δηλαδή στις λειτουργίες τις οποίες θα εκτελέσει το σύστημα. Στη συνέχεια, το σύστημα μετασχηματίζει τον εαυτό του όπως του υπαγορεύουν οι λειτουργίες που μεταφράστηκαν από την είσοδο. Έτσι, ολοκληρώνεται η φάση της εκτέλεσης και ξεκινάει η φάση της αξιολόγησης. Το σύστημα βρίσκεται σε μια νέα κατάσταση, η οποία μπορεί να κοινοποιηθεί στο χρήστη. Οι τρέχουσες τιμές των χαρακτηριστικών του συστήματος αποδίδονται με τα χαρακτηριστικά της εξόδου. Άρα, είναι ευθύνη του χρήστη να αξιολογήσει το αποτέλεσμα της διάδρασης σε σχέση με το στόχο που είχε θέσει αρχικά. Κατά τη φάση της διάδρασης λαμβάνουν χώρα τέσσερις βασικές μεταφράσεις, διατύπωση, διεκπεραίωση, παρουσίαση και παρατήρηση. Η εργασία που καθορίζει ο χρήστης για την επίτευξη κάποιου στόχου πρέπει να διατυπωθεί στη γλώσσα εισόδου. Οι εργασίες είναι αντιδράσεις του χρήστη και πρέπει να μεταφραστούν σε ερεθίσματα για την είσοδο. Η εργασία διατυπώνεται με βάση συγκεκριμένα ψυχολογικά χαρακτηριστικά, τα οποία αν αντιστοιχίζονται με σαφήνεια στη γλώσσα εισόδου, τότε η διατύπωση της εργασίας γίνεται πολύ πιο απλή.

Συμπερασματικά, το πλαίσιο αναφοράς της διάδρασης δεν περιορίζεται στους υπολογιστές, αναγνωρίζει όλα τα βασικά χαρακτηριστικά που εμπλέκονται στην αλληλεπίδραση και επιτρέπει τη συγκριτική αξιολόγηση των συστημάτων.

Κεφάλαιο 5
Ανάπτυξη διεπαφής χρήστη

Η διεπαφή χρήστη (User Interface, UI) υλοποιεί την αμφίδρομη επικοινωνία συστήματος - χρήστη (υπολογιστή - ανθρώπου). Ένα από τα πιο σημαντικά κομμάτια ανάπτυξης ενός σωστού λογισμικού είναι η υλοποίηση ενός ικανοποιητικού συστήματος διεπαφής με το χρήστη. Ένα σύστημα διεπαφής επιτρέπει στο χρήστη ενός συστήματος ή μιας εφαρμογής να αντιληφθεί κάποιες πληροφορίες, να τις αποθηκεύσει στην μνήμη του και να τις επεξεργαστεί. Η διεπαφή του χρήστη έχει σχέση με το ίδιο το σύστημα, με το χρήστη του συστήματος αλλά και τον τρόπο που αλληλεπιδρούν μεταξύ τους, όπως φαίνεται και στο σχήμα 5.1 που ακολουθεί [17]



Σχήμα 5.1. Σχηματική αναπαράσταση διεπαφής χρήστη

5.1. Τύποι διεπαφής χρήστη

5.1.1. Command-based interface

Αρχικά, είχε παρουσιαστεί ο τύπος διεπαφής που ήταν βασισμένος σε εντολή (command-based), όπου ο χρήστης για να έχει αλληλεπίδραση με το σύστημα, θα έπρεπε να πληκτρολογήσει μια εντολή ή εντολές, με πιο χαρακτηριστικό παράδειγμα το DOS. Ο χρήστης θα έπρεπε να είναι ειδικός στη γλώσσα που υποστήριζε το σύστημα, ώστε να έχει τη μεγαλύτερη δυνατή απόδοση. Ο κύκλος λειτουργίας του προγράμματος ήταν: είσοδος – επεξεργασία - έξοδος και το μεγαλύτερο τμήμα του κώδικα αφιερωνόταν στην επεξεργασία. Η είσοδος γινόταν αποκλειστικά από το πληκτρολόγιο. Ουσιαστικά ο σχεδιαστής επινοούσε μια

«γλώσσα» εισόδου που έπρεπε να κατέχει και να καταλαβαίνει ο κάθε χρήστης που θα χρησιμοποιούσε το σύστημα και την οποία να αποδέχεται το σύστημα, ώστε να την επεξεργάζεται και να παράγει το αποτέλεσμα στη γλώσσα εξόδου που ήταν συνήθως γραμμογραφημένες εκτυπώσεις. Η διεπαφή χρήστη σε τέτοια συστήματα ήταν ρυθμού χαρακτήρων (character-mode UI).

5.1.2. Graphical User Interface

Όσο περνάει ο καιρός και οι απαιτήσεις αυξάνονται, αλλά και για λόγους ευχρηστίας η διεπαφή βασίζεται σε γραφικά (Graphical User Interfaces, GUI) με χρήση εικονιδίων και γραφημάτων στις διαδικασίες επικοινωνίας του χρήστη με τον υπολογιστή. [19] Η υλοποίηση GUI απαιτεί περισσότερο κώδικα και η εκτέλεσή τους καταναλώνει περισσότερους πόρους. Σε αρκετές περιπτώσεις για το σχεδιασμό της διεπαφής του συστήματος απαιτείται η συγγραφή μεγαλύτερου όγκου κώδικα, από ότι χρειάζεται σε πλήρως “παραγωγικό” κώδικα που να υπολογίζει κάποιο αποτέλεσμα. Οι ενέργειες του χρήστη πλέον καταδεικνύονται και δεν πληκτρολογούνται. Τα δεδομένα πληκτρολογούνται, όταν αυτό είναι πιο πρακτικό από να επιλέγονται. Κάποιες φορές, σε εξειδικευμένες εφαρμογές η χρήση GUI μπορεί να επιβραδύνει τη χρήση της εφαρμογής, γι’αυτό και πολλές φορές, όπου είναι απαραίτητο, χρησιμοποιείται ο παραδοσιακός τρόπος που αναφέραμε παραπάνω. Αυτή η μορφή διεπαφής αυξάνει τη λειτουργικότητα, την αποτελεσματικότητα και την ταχύτητα διαλόγου μεταξύ χρήστη και συστήματος. Το GUI, σήμερα, είναι ο πιο διαδεδομένος τύπος διεπαφής χρήστη και συμφωνεί απόλυτα με τις θεωρήσεις του αντικειμενοστραφούς προγραμματισμού.

5.2. Βασικές αρχές εργονομίας λογισμικού που θα πρέπει να λαμβάνονται υπόψη κατά τη σχεδίαση μιας επιτυχούς και λειτουργικής διεπαφής χρήστη

1. Συνέπεια : αν για μια συγκεκριμένη ενέργεια ή διαδικασία χρησιμοποιείται ένας τρόπος λειτουργίας, σε οποιοδήποτε τμήμα του συστήματος και να συναντηθεί, θα πρέπει να λειτουργεί με τον ίδιο ακριβώς τρόπο και αυτό πρέπει να γίνεται διότι ο χρήστης έχει μάθει να την εκτελεί με ένα συγκεκριμένο τρόπο, οπότε κάθε φορά που θα κληθεί να την επανεκτελέσει, θα πρέπει να χρησιμοποιήσει τον ίδιο τρόπο.
2. Απλότητα : για την παρουσίαση μιας συγκεκριμένης διεργασίας, θα πρέπει να επιλέγεται από το σχεδιαστή του συστήματος η απλούστερη. Ακόμα και αν υπάρχουν σύνθετες διεργασίες, θα πρέπει να λειτουργούν όσο το δυνατόν απλούστερα και να καταβάλλεται κάθε δυνατή προσπάθεια για την επίτευξη αυτή.
3. Χρήση μεταφορών : πρέπει να χρησιμοποιούνται προσεγγίσεις που είναι γνωστές και οικείες στον κάθε χρήστη.
4. Ελαχιστοποίηση ενεργειών χρήστη : ο κάθε χρήστης θα πρέπει να επιτυγχάνει αυτό που θέλει με τις λιγότερες δυνατές ενέργειες, δηλαδή, θα πρέπει να περιορίζονται στις απολύτως αναγκαίες και απαραίτητες για την ορθή λειτουργία του συστήματος.
5. Παροχή άμεσης ανάδρασης : σε κάθε περίπτωση, πρέπει ο χρήστης να ενημερώνεται με μηνύματα από το σύστημα, ακόμα και στις περιπτώσεις που κάποια από τα αποτελέσματα επιλογών του χρήστη δεν έχουν ολοκληρωθεί. Αυτό πρέπει να λαμβάνεται σοβαρά υπόψη όταν κάποια από τις επιλογές του χρήστη απαιτεί χρόνο για να ολοκληρωθεί, έτσι, θα πρέπει να ενημερώνεται με αντίστοιχο μήνυμα.
6. Παροχή βοήθειας : ακόμα και αν ο χρήστης γνωρίζει κάποιο σύστημα και το έχει χρησιμοποιήσει πολλές φορές, θα πρέπει κάθε στιγμή να γνωρίζει τι του προσφέρει και να ανατρέξει σε κάποιου είδους βοήθεια, όπως on line βοήθεια [11], άμεση βοήθεια, έμμεση βοήθεια.
7. Ελαχιστοποίηση απομνημόνευσης : ο χρήστης να είναι σε θέση να θυμάται μόνο τα απολύτως απαραίτητα για να μπορεί να αλληλεπιδρά σωστά με το σύστημα. [12]

8. Εναρμόνιση : θα πρέπει να συνυπολογίζεται η προηγούμενη εμπειρία του χρήστη από άλλα συστήματα λογισμικού. Δεδομένου ότι όλο και περισσότερο τυποποιούνται ενέργειες και διαδικασίες, θα πρέπει τα συστήματα να εναρμονίζονται με αυτές.
9. Ευκαμψία : το σύστημα θα πρέπει να παρουσιάζει ευκαμψία στις ενέργειες και τις πληκτρολογήσεις του χρήστη.

5.3. Προτάσεις Sommerville (2000) για σχεδιασμό περιβάλλοντος διεπαφής

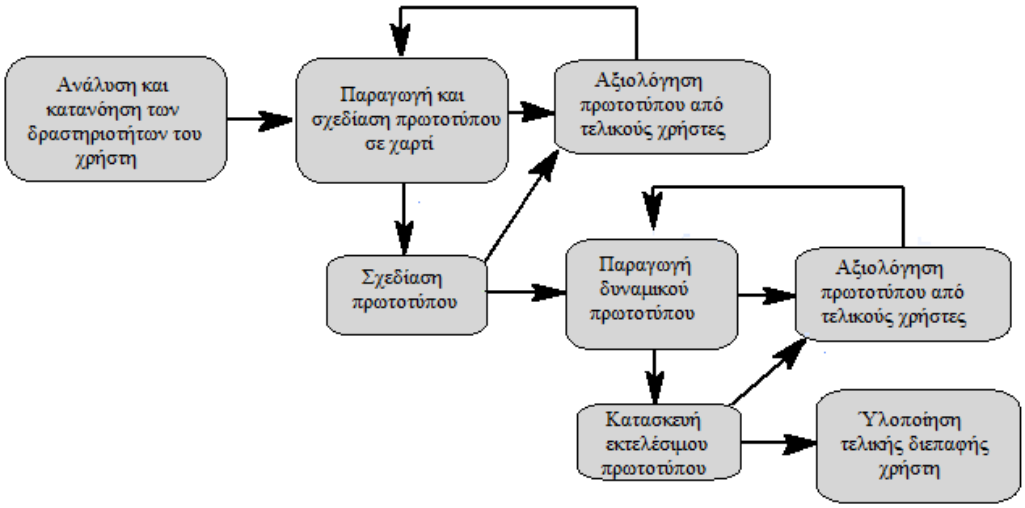
5.3.1. Βασικά χαρακτηριστικά του GUI

Τα βασικά χαρακτηριστικά για τη σχεδίαση ενός γραφικού περιβάλλοντος διεπαφής φαίνονται στον πίνακα 5.2 που ακολουθεί.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ	ΠΕΡΙΓΡΑΦΗ
Παράθυρα	Τα πολλαπλά παράθυρα επιτρέπουν τη σύγχρονη εμφάνιση πολλών διαφορετικών πληροφοριών στην οθόνη.
Εικονίδια	Τα εικονίδια αντιπροσωπεύουν διαφορετικού τύπου πληροφορίες. Σε άλλα συστήματα αντιπροσωπεύουν αρχεία ενώ σε άλλα διαδικασίες.
Μενού	Οι διαταγές μπορούν να επιλεχθούν από ένα menu αντί να πληκτρολογηθούν από τον χρήστη.
Δείξιμο (Pointing)	Pointing εργαλεία, όπως το ποντίκι, χρησιμοποιούνται για την επιλογή ενός μενού ή αντικειμένων μέσα στα παράθυρα.
Γραφικά	Τα γραφικά μπορούν να «ανακατευτούν» με τα κείμενα σε μια ενιαία εμφάνιση.

Πίνακας 5.2. βασικά χαρακτηριστικά του GUI

Ο Sommerville προτείνει, επίσης, μια επαναληπτική διαδικασία για το σχεδιασμό του περιβάλλοντος διεπαφής όπως φαίνεται και στο σχήμα 5.3.



Σχήμα 5.3. σχεδιασμός του περιβάλλοντος διεπαφής

Όπως φαίνεται και στο σχήμα, ο σχεδιαστής αφού πρώτα αναλύσει και κατανοήσει τις δραστηριότητες του χρήστη καθ' όλη τη διάρκεια που θα χρησιμοποιεί το σύστημα, θα σχεδιάσει το πρωτότυπο αρχικά σε χαρτί πριν προχωρήσει στη σχεδίαση του πρωτοτύπου. Μετά τη σχεδίασή του, θα γίνει αξιολόγησή του από τελικούς χρήστες και ανάλογα την αξιολόγηση, μπορεί να χρειαστεί να σχεδιάσει ξανά το πρωτότυπο στο χαρτί, διαφορετικά θα προχωρήσει στην παραγωγή δυναμικού πρωτοτύπου. Μετά από αυτή τη φάση, το πρωτότυπο θα αξιολογηθεί εκ νέου από τελικούς χρήστες και ανάλογα την αξιολόγηση θα προχωρήσει στην παραγωγή εκ νέου δυναμικού πρωτοτύπου ή στην κατασκευή εκτελέσιμου πρωτοτύπου. Τέλος, θα υλοποιήσει την τελική διεπαφή του χρήστη.

5.4. Διάδραση χρήστη – λογισμικού στο σχεδιασμό διεπαφής

Ο σχεδιαστής της διεπαφής του χρήστη θα πρέπει να λάβει υπόψη τη διάδραση που θα προκύψει, δηλαδή το διάλογο που θα λαμβάνει χώρα μεταξύ του υπολογιστή και του χρήστη. Η επιλογή της μορφής διεπιφάνειας μπορεί να έχει σημαντική επίδραση στη φύση του διαλόγου αυτού. [3] Ακολουθεί μια περιγραφή των πιο κοινών μορφών διεπιφάνειας όπως είναι :

- διεπαφή γραμμής εντολών (command line interface) : είναι το πρώτο διαδραστικό στυλ διαλόγου που χρησιμοποιήθηκε ευρέως και συνεχίζει να χρησιμοποιείται ευρέως. Παρέχει ένα μέσο για την απευθείας διατύπωση οδηγιών προς τον υπολογιστή χρησιμοποιώντας λειτουργικά πλήκτρα, μεμονωμένους χαρακτήρες ή εντολές. Σε ορισμένα συστήματα η γραμμή εντολής είναι ο μοναδικός τρόπος επικοινωνίας με το σύστημα, κυρίως για την προσπέλαση του συστήματος εξ αποστάσεως. Οι διεπιφάνειες γραμμής εντολής είναι ισχυρές επειδή παρέχουν άμεση πρόσβαση σε όλες τις λειτουργίες του συστήματος και μπορούν να χρησιμοποιούνται για την εφαρμογή πολλών εργαλείων στα ίδια δεδομένα. Είναι ευέλικτες και συχνά μια εντολή έχει πολλαπλές επιλογές ή παραμέτρους, οι οποίες προσαρμόζουν τη συμπεριφορά της με κάποιο τρόπο και μπορεί επίσης να εφαρμόζεται σε πολλά αντικείμενα ταυτόχρονα, το οποίο είναι ιδιαίτερα χρήσιμο για την εκτέλεση επαναλαμβανόμενων εργασιών.
- μενού επιλογών (menu) : σε μια διεπιφάνεια η οποία βασίζεται σε μενού, το σύνολο των επιλογών που είναι διαθέσιμες στο χρήστη παρουσιάζεται στην οθόνη σε μορφή μενού. [16]Ο χρήστης μπορεί να επιλέγει τις διαθέσιμες επιλογές χρησιμοποιώντας το ποντίκι και τα αριθμητικά ή αλφαβητικά πλήκτρα. Επειδή οι επιλογές είναι ορατές, η διεπιφάνεια αυτή θέτει λιγότερες απαιτήσεις στο χρήστη, βασιζόμενη πιο πολύ στην αναγνώριση

παρά στην απομνημόνευση. Πολλές φορές τα μενού οργανώνονται με ιεραρχικό τρόπο, που σημαίνει ότι η επιλογή που θέλει ο χρήστης δε θα είναι πάντα διαθέσιμη στο κορυφαίο επίπεδο της ιεραρχίας.

- φυσική γλώσσα (natural language) : σε πρώτη εξέταση, το ελκυστικότερο μέσο επικοινωνίας με τους υπολογιστές δείχνεινα είναι η φυσική γλώσσα. Αρκετοί χρήστες που δεν μπορούν να θυμούνται εντολές ή ιεραρχία μενού, θα επιθυμούσαν οι υπολογιστές να έχουν τη δυνατότητα κατανόησης οδηγιών εκφρασμένες με λέξεις καθημερινής χρήσης. Αλλά ακόμα και μια πρόταση να είναι σαφής, μπορεί να υπάρχει ασάφεια στη σημασία των λέξεων που χρησιμοποιήθηκαν. Σε αντίθεση με τον άνθρωπο, μια μηχανή δεν έχει την ίδια δυνατότητα στην κατανόηση λέξεων που έχουν διαφορετική ερμηνεία. Με δεδομένα αυτά τα προβλήματα, είναι μάλλον απίθανο να υπάρξει σύντομα μια διεπιφάνεια γενικής χρήσης η οποία θα βασίζεται στη φυσική γλώσσα.
- ερωταποκρίσεις και διάλογος μέσω ερωτημάτων (question/answer and query dialog) : ο διάλογος μέσω ερωτήσεων και απαντήσεων είναι ένας απλός μηχανισμός που μπορεί να παρέχει είσοδο σε μια εφαρμογή, σε έναν συγκεκριμένο τομέα εφαρμογής. Ο χρήστης καλείται να απαντήσει σε μια σειρά ερωτήσεων και καθοδηγείται βήμα προς βήμα καθ' όλη τη διάρκεια της διάδρασής του με το σύστημα. Οι διεπιφάνειες που βασίζονται στο μηχανισμό των ερωταποκρίσεων είναι εύκολες στην εκμάθηση και στη χρήση, αλλά έχουν περιορισμένη λειτουργικότητα και ισχύ.
- συμπλήρωση φορμών και φύλλων εργασίας (form-fills and spreadsheets) : οι διεπιφάνειες που βασίζονται στη συμπλήρωση φορμών χρησιμοποιούνται κυρίως για την εισαγωγή δεδομένων, αλλά μπορούν να φανούν χρήσιμες σε εφαρμογές ανάκτησης δεδομένων. Στο χρήστη παρουσιάζεται μια οθόνη που μοιάζει με έντυπη φόρμα και περιλαμβάνει ειδικές θέσεις – πεδία για τη συμπλήρωση στοιχείων. Συνήθως, η εμφάνιση της φόρμας στην οθόνη βασίζεται στην εμφάνιση μιας πραγματικής έντυπης φόρμας η οποία είναι αρκετά οικεία στο χρήστη, γεγονός που το κάνει ακόμα πιο εύχρηστο. Τα φύλλα εργασίας είναι μια πιο εξελιγμένη παραλλαγή της συμπλήρωσης φορμών. Το φύλλο εργασίας απαρτίζεται από ένα πλέγμα κελιών, κάθε ένα από τα οποία μπορεί να περιέχει μια τιμή ή έναν τύπο υπολογισμού. Ο χρήστης μπορεί να εισάγει και να αλλάζει τιμές και τύπους με οποιαδήποτε σειρά και το σύστημα διατηρεί τη συνέπεια μεταξύ των εμφανιζόμενων τιμών, διασφαλίζοντας ότι όλοι οι τύποι δίνουν σωστά αποτελέσματα. Τα φύλλα εργασίας είναι ένα ελκυστικό μέσο διάδρασης, όπου ο χρήστης είναι ελεύθερος να χειρίζεται τιμές κατά βούληση και η απόσταση μεταξύ της εισόδου και της εξόδου μειώνεται. Κατά συνέπεια, η διεπιφάνεια είναι πιο ευέλικτη και δείχνει πιο φυσική.
- παραθυρικά περιβάλλοντα (WIMP) : πολλά από τα κοινά περιβάλλοντα διάδρασης με τον υπολογιστή βασίζονται σε μορφή διεπιφάνειας WIMP, τα οποία αποκαλούνται παραθυρικά συστήματα. Το σύστημα αυτό χρησιμοποιείται στην πλειονότητα των διαδραστικών υπολογιστικών συστημάτων, κυρίως στο χώρο των προσωπικών υπολογιστών και σταθμών εργασίας.
- δείξιμο και κλικ (point and click) : στις περισσότερες εφαρμογές πολυμέσων και περιήγησης στο Διαδίκτυο, απαιτείται μόνο ένα κλικ με το πλήκτρο του ποντικιού για τις περισσότερες ενέργειες. Αυτό το στυλ διεπιφάνειας σχετίζεται με το στυλ WIMP. Εκτός από τα κουμπιά, μπορεί να περιλαμβάνει και να χρησιμοποιεί άλλα στοιχεία του στυλ WIMP. Χρησιμοποιείται ευρύτατα στις ιστοσελίδες, οι οποίες ενσωματώνουν όλους τους προαναφερθέντες μηχανισμούς πλοήγησης που λειτουργούν με "δείξιμο και κλικ", λέξεις που εμφανίζονται με έμφαση στην οθόνη, χάρτες με ενεργείς περιοχές και κουμπιά με εικονίδια στην πρόσοψή τους.
- τρισδιάστατες διεπαφές (three-dimensional interfaces) : η απλούστερη τεχνική είναι η χρήση των γνωστών συστατικών του στυλ WIMP, αλλά με τρισδιάστατη εμφάνιση, η οποία επιτυγχάνεται με κατάλληλη χρήση σκίασης, δίνοντας την εντύπωση ότι είναι ανάγλυφα. Η διάδραση με τρισδιάστατες διεπιφάνειες μας ενθαρρύνει να χρησιμοποιήσουμε ικανότητες οικείες από τον πραγματικό κόσμο και να τις μεταφέρουμε στον ηλεκτρονικό κόσμο.

5.5. Παρουσίαση της πληροφορίας

Όλα τα διαδραστικά συστήματα, πρέπει να είναι σε θέση να παρέχουν έναν τρόπο που να μπορούν να παρουσιάζουν την πληροφορία στους χρήστες, η οποία μπορεί να είναι σε μορφή κειμένου ή να παρουσιάζεται γραφικά. Προκειμένου να βρεθεί ο καλύτερος τρόπος για να παρουσιαστεί η πληροφορία στο χρήστη είναι να υπάρχει γνώση του υποβάθρου των χρηστών που θα χρησιμοποιούν το σύστημα. [20] Για να μπορεί ο σχεδιαστής να αποφασίσει τον τρόπο που θα παρουσιάσει την πληροφορία, πρέπει να απαντήσει σε βασικές ερωτήσεις που να καλύπτουν ένα ευρύ φάσμα των χαρακτηριστικών που αφορούν την ποιότητα ενός συστήματος διαχείρισης και αυτές είναι οι εξής:

- Ποιος είναι ο χρήστης;
- Ενδιαφέρεται ο χρήστης για τη συγκεκριμένη πληροφορία;
- Ερμηνεύει τις πληροφορίες του συστήματος σωστά;
- Πόσο γρήγορα αλλάζει η πληροφορία που παρουσιάζεται στην οθόνη; Μετά από αλλαγή, μετά από πόση ώρα γίνεται εμφανής στο χρήστη;
- Η πληροφορία θα εμφανίζεται με τη μορφή κειμένου ή με τη μορφή αριθμού;

Λαμβάνοντας υπόψη τις βασικές ερωτήσεις που καλύπτουν τα βασικά χαρακτηριστικά που πρέπει να έχει η διεπαφή ενός συστήματος, θα προχωρήσουμε στα χαρακτηριστικά που πρέπει να έχει η διεπαφή μας απαντώντας στα συγκεκριμένα ερωτήματα.

5.6. Σχεδιασμός του προσαρμοστικού συστήματος διδασκαλίας

Πριν περάσουμε στη σχεδίαση του συστήματος διεπαφής θα πρέπει να απαντήσουμε στις σημαντικές ερωτήσεις που τέθηκαν πιο πάνω. Ποιος είναι, λοιπόν, ο χρήστης του συστήματος; Οι χρήστες στο σύστημα μας είναι είτε άνθρωποι με λίγες γνώσεις πάνω σε θέματα υπολογιστών είτε πιο εξειδικευμένοι χρήστες. Οι κατηγορίες χρηστών θα είναι μαθητές και καθηγητές. Έχουμε, λοιπόν, να κάνουμε με χρήστες που κατά κύριο λόγο χρησιμοποιούν το ποντίκι σε μια εφαρμογή, απλά πληκτρολογούν δεδομένα και περιμένουν να λάβουν απλά και περιεκτικά μηνύματα πληροφοριών από τον υπολογιστών. Θέλουν, επίσης, να κάνουν γρήγορα την δουλειά τους χωρίς άσκοπες και επαναλαμβανόμενες διαδικασίες.

Λαμβάνοντας υπόψη τα ανωτέρω, εξάγεται το αποτέλεσμα της σχεδίασης ενός συστήματος που θα πρέπει να δίνει πολλές βοήθειες για την χρήση του, να περιέχει περιεκτικά μηνύματα διαλόγου ή λάθους, να παρέχει συντομεύσεις και γενικότερα να ακολουθεί τους σωστούς κανόνες σχεδιασμού.

Τα μοντέλα διάδρασης (interaction models) είναι γενικευμένα, τυπικά μοντέλα διαδραστικών συστημάτων. Η διάδραση απαιτεί τουλάχιστον δυο συμμετέχοντα μέρη, το χρήστη και το σύστημα, τα οποία είναι πολύπλοκα και διαφορετικά μεταξύ τους. Η χρήση μοντέλων διάδρασης μας βοηθάει να κατανοήσουμε τι λαμβάνει χώρα κατά τη διαδικασία της διάδρασης, αλλά και να προσδιορίσουμε πιθανές δυσκολίες που αντιμετωπίζονται. Παρέχουν, επίσης, ένα γενικότερο πλαίσιο για τη σύγκριση διαφορετικών μορφών διάδρασης για την εξέταση του κάθε προβλήματος.

Η διεπαφή του εκπαιδευτικού συστήματος θα πρέπει να υποστηρίζει τη στάση και τις δράσεις του μαθητή. Η επικοινωνία συστήματος – μαθητή μπορεί να θεωρηθεί ανάλογη με την επικοινωνία μαθητή – καθηγητή. Κατά την επικοινωνία αυτή, ο καθηγητής έχει τον κυρίαρχο ρόλο ενώ ελέγχει και τη μαθησιακή διαδικασία, ενώ σε μια εφαρμογή εκπαιδευτικού λογισμικού ο μαθητής έχει τον έλεγχο της μαθησιακής διαδικασίας και επιλέγει τη διαδρομή που θα ακολουθήσει αναζητώντας και συνδυάζοντας πληροφορίες. Η λειτουργικότητα του συστήματος δεν είναι ο μόνος παράγοντας επιτυχίας του, αλλά θα πρέπει να είναι και εύχρηστη και φιλική προς το χρήστη, που στη δική μας περίπτωση είναι οι μαθητές και οι καθηγητές.

Κεφάλαιο 6
Αρχές και μοντέλα ανάπτυξης του συστήματος

Οι χρήστες του συστήματος θα είναι μαθητές αλλά και καθηγητές. Για να καταστεί δυνατή κάλυψη των διαφορετικών απαιτήσεων της κάθε ομάδας χρηστών, αλλά και του περιβάλλοντος χρήσης, υιοθετήθηκαν οι κάτωθι σχεδιαστικές αρχές :

- ανοιχτή αρχιτεκτονική : το σύστημα που δημιούργησα έχει τη δυνατότητα να γίνει δυναμικό, διότι ο κάθε καθηγητής μπορεί κάθε φορά που κάνει είσοδο στο σύστημα να προσθέτει μαθήματα, ύλη για το κάθε μάθημα, αλλά και ερωτήσεις που αφορούν συγκεκριμένη ύλη.
- ευχρηστία : όσοι θα χρησιμοποιήσουν το σύστημα διδασκαλίας δεν είναι απαραίτητο να είναι άριστοι γνώστες των υπολογιστών, αλλά μπορεί να χρησιμοποιηθεί τόσο από αρχάριους χρήστες (με τη χρήση WIMP διεπαφής που έχει αναλυθεί παραπάνω, εύκολων και εύχρηστων φορμών, μηνυμάτων διαλόγου και λαθών, αλλά και κουμπιών με σαφή λειτουργία), αλλά και από έμπειρους χρήστες χωρίς να γίνεται κουραστικό. Έχει κατασκευαστεί με τις αρχές επικοινωνίας ανθρώπου – υπολογιστή και έχουν χρησιμοποιηθεί μόνο λειτουργικά χαρακτηριστικά που έχουν εκπαιδευτική αξία. Τέλος, δίνεται έμφαση στο user interface, καθιστώντας το έτσι σε ένα εύχρηστο και εύκολο σύστημα στη χρήση.
- μοντέλο πρωτοτυποποίησης : διορθώθηκαν παρεξηγήσεις μεταξύ των δημιουργών και των χρηστών, δυσκολίες που προέκυψαν στη χρήση του και κενά στις προδιαγραφές του. Με αυτό τον τρόπο έγινε απόλυτα κατανοητό πόσο εφικτό ήταν να δημιουργηθεί το εν λόγω σύστημα με τις απαιτούμενες προδιαγραφές, παρουσιάστηκε ένα μοντέλο συστήματος το οποίο είναι πλήρως λειτουργικό και το πρώτυπο συντέλεσε στην εκ των προτέρων καταγραφή των προδιαγραφών για την κατασκευή ενός ποιοτικού συστήματος.

6.1. Διδακτικοί στόχοι του συστήματος διδασκαλίας & προσαρμοστικότητα

Το προσαρμοστικό σύστημα διδασκαλίας που δημιούργησα, διαθέτει όλους τους διδακτικούς στόχους που είχαν τεθεί και αφορούσαν στη γνώση που μπορεί να παραχθεί μέσω αυτού κατά τη διάρκεια της χρήσης του από την πλευρά των μαθητών και χωρίστηκε σε δυο κατηγορίες, τους βασικούς στόχους και τους ειδικούς. Οι βασικοί στόχοι αφορούν στην ύλη του κάθε μαθήματος που έχει τη δυνατότητα να διαβάσει ο μαθητής με ένα από κλικ του ποντικιού του.

Να βλέπει τα διαθέσιμα μαθήματα που υπάρχουν στο σύστημα, αλλά και την ύλη που έχει προσθέσει ο κάθε καθηγητής. Επειδή είναι ένα δυναμικό σύστημα, διότι ο κάθε καθηγητής μπορεί να εισάγει αρχεία με την ύλη που έχει διδάξει και δεν αφορά ένα συγκεκριμένο αντικείμενο το σύστημα αυτό, δίνεται η δυνατότητα στο μαθητή να ανακαλύπτει κάθε φορά νέα θέματα που θα μπορούν να του διευρύνουν τους ορίζοντες και να τον βοηθήσουν να αποκτήσει γνώση για μια ευρεία κατηγορία θεμάτων.

Αναφορικά με την προσαρμοστικότητα, το σύστημα διδασκαλίας, είναι πλήρως προσαρμοστικό, διότι κάθε φορά που ο μαθητής επιλέγει να συμπληρώσει ένα τεστ, του δίνεται η δυνατότητα να επιλέξει τον αριθμό των ερωτήσεων οι οποίες αφορούν μόνο σε ύλη την οποία έχει διαβάσει. Κάθε φορά που τελειώνει και υποβάλει ένα τεστ για αξιολόγηση, άμεσα του εμφανίζεται στην οθόνη του υπολογιστή του το ποσοστό των σωστών απαντήσεων, αλλά και του προτείνεται να διαβάσει την ανάλογη ύλη σύμφωνα με τις απαντήσεις που απάντησε λανθασμένα. Κάθε φορά που ο χρήστης πραγματοποιεί είσοδο στο σύστημα, στην κεντρική οθόνη του συστήματος, του εμφανίζεται η ύλη που πρέπει να διαβάσει, ανάλογα με τα λάθη που είχε κάνει σε τεστ που είχε συμπληρώσει. Με αυτό τον τρόπο, ο μαθητής μπορεί να παρακολουθεί την πορεία του για κάθε ένα από τα μαθήματα που υπάρχουν στο σύστημα και με αυτό τον τρόπο να βελτιώνει το γνωστικό του επίπεδο κάθε φορά που εισέρχεται σε αυτό.

6.2. Μοντέλο ανάπτυξης κύκλου ζωής που χρησιμοποιήθηκε

Το μοντέλο στο οποίο βασίστηκε η κατασκευή του συστήματος διδασκαλίας είναι το σπειροειδές, όπως είχαμε αναφέρει σε προηγούμενο κεφάλαιο. Ακολουθήθηκαν, ωστόσο, κάποιες από τις αρχές του μοντέλου πρωτοτυποποίησης. Ο συνδυασμός αυτός των μοντέλων έδωσε τη δυνατότητα στους χρήστες του συστήματος να αξιολογούν κάθε ενδιάμεσο προϊόν έτσι ώστε το τελικό σύστημα να καλύπτει όσο το δυνατόν περισσότερο τις απαιτήσεις τους.

6.3. Το περιβάλλον διεπαφής του συστήματος διδασκαλίας

Όπως αναλύσαμε και στο αντίστοιχο κεφάλαιο, η σωστή σχεδίαση του περιβάλλοντος διεπαφής είναι σημαντική για την επιτυχή λειτουργία του λογισμικού. Η αλληλεπίδραση του χρήστη με το σύστημα γίνεται κυρίως με απευθείας χειρισμό, δηλαδή, ο χρήστης αλληλεπιδρά άμεσα με αντικείμενα/εικόνες/κουμπί που βρίσκονται στην οθόνη του συστήματος. [4] Τα πλεονεκτήματα αυτού του τρόπου αλληλεπίδρασης είναι ότι η αλληλεπίδραση είναι γρήγορη και διαισθητική και το περιβάλλον είναι εύκολο στην εκμάθηση. Το σύστημα διδασκαλίας που δημιούργησα έχει κατασκευαστεί βασισμένο στις κυριότερες και σημαντικότερες σχεδιαστικές αρχές δεδομένου ότι:

- Το περιβάλλον διεπαφής έχει συνέπεια, διότι όλες οι ίδιες διαδικασίες λειτουργούν με τον ίδιο τρόπο, χωρίς να μπερδεύουν το χρήστη.
- Είναι ένα απλό σύστημα, διότι ακόμα και για σύνθετες διαδικασίες έχει χρησιμοποιηθεί ο απλούστερος τρόπος λειτουργίας, όπως είναι η συμπλήρωση των ερωτηματολογίων – τεστ.
- Έχει επιτευχθεί ελαχιστοποίηση των ενεργειών που απαιτούνται από το χρήστη και έχουν περιοριστεί στις απολύτως αναγκαίες.
- Παρέχεται άμεση ανάδραση του συστήματος προς το χρήστη, καθώς σε κάθε επιλογή του, εμφανίζεται αντίστοιχο μήνυμα.
- Ο χρήστης δε χρειάζεται να απομνημονεύει πληροφορίες για να μπορεί να αλληλεπιδρά σωστά με το σύστημα.
- Το σύστημα εναρμονίζεται πλήρως με παρόμοια συστήματα διδασκαλίας, γεγονός που βοηθάει το χρήστη να προσαρμοστεί ευκολότερα με το εν λόγω σύστημα.

6.4. Το χρώμα στη σχεδίαση του περιβάλλοντος διεπαφής

Η επιλογή του σωστού χρώματος βοηθάει τους χρήστες να διαχειριστούν την πολυπλοκότητα του συστήματος. Στο σύστημα διδασκαλίας που δημιούργησα έχουν χρησιμοποιηθεί μόνο δυο

χρώματα για να μην μπερδεύονται οι χρήστες, αλλά και για να είναι το σύστημα όσο το δυνατόν πιο απλό και ελκυστικό. Το φόντο που έχει χρησιμοποιηθεί σε όλα τα παράθυρα είναι το μαύρο, δίνοντας της αίσθηση γραμμής εντολών και το χρώμα της γραμματοσειράς που επιλέχθηκε είναι το πράσινο. Το πράσινο χρώμα είναι ουδέτερο και εκφράζει την ενέργεια, αλλά ταυτόχρονα είναι κατευναστικό και βοηθάει άτομα που είναι κουρασμένα. Η αντίθεση αυτή των δυο χρωμάτων επιλέχθηκε για να είναι ευδιάκριτα τα γράμματα σε σχέση με το χρώμα του παραθύρου και να μπορεί ο κάθε χρήστης να διαβάσει με ευκολία αυτά που του επιστρέφει το σύστημα.

6.5. Το είδος εκπαιδευτικού λογισμικού που βασίστηκε το σύστημα

Το σύστημα που δημιούργησα βασίστηκε στο εκπαιδευτικό λογισμικό εξάσκησης-εκγύμνασης, καθώς οι μαθητές έχουν τη δυνατότητα να διαβάσουν μια συγκεκριμένη ύλη – θεματική ενότητα που είναι αποθηκευμένη στο σύστημα από συγκεκριμένο καθηγητή και στη συνέχεια μπορεί να απαντήσει σε ερωτήσεις που αφορούν στη συγκεκριμένη ύλη. Έχει άμεση ανάδραση αναφορικά στην εκπαίδευση του χρήστη σε συγκεκριμένα θέματα και αυτό επιτυγχάνεται μέσω των τεστ-ερωτήσεων που απαντάει. Το συγκεκριμένο σύστημα έχει τη δυνατότητα αποθήκευσης της επίδοσης του μαθητή, να του δείξει τα λάθη του και να του προτείνει να διαβάσει τις θεματικές ενότητες που αφορούσαν στα λάθη που έκανε κατά τη διάρκεια απαντήσεων στις ερωτήσεις του τεστ. Ο κάθε μαθητής έχει τη δυνατότητα για απεριόριστη εξάσκηση χωρίς κάποιο κόστος και έχει τη δυνατότητα να ελέγχει την πρόοδό του.

Κεφάλαιο 7
Φάση κατασκευής

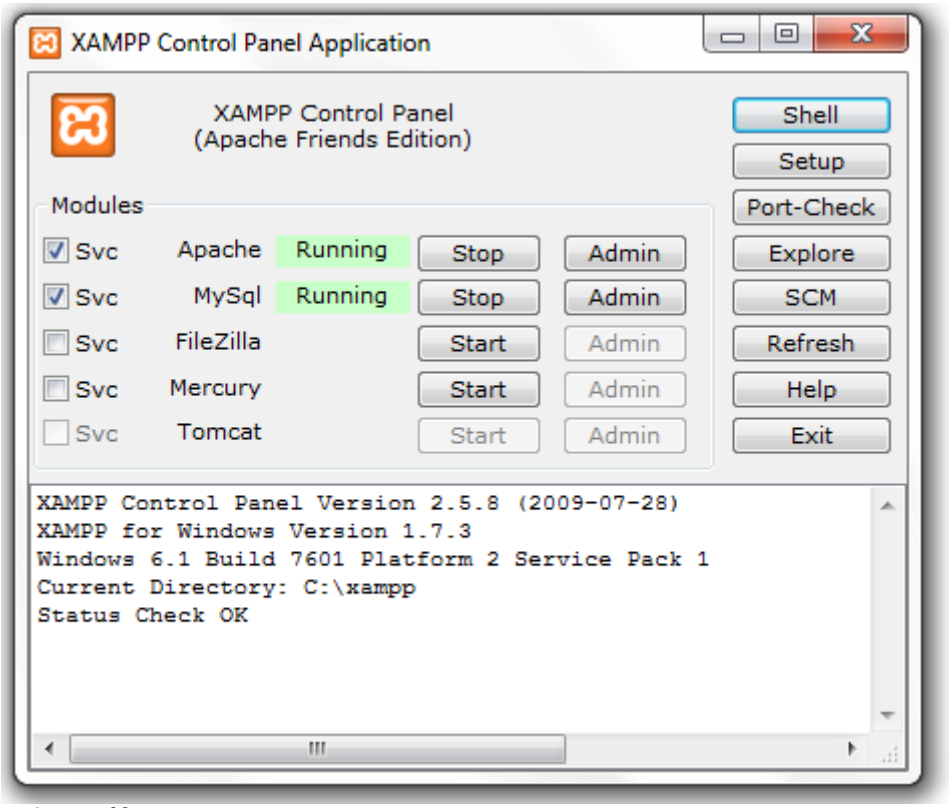
7.1. Προγράμματα που χρησιμοποιήθηκαν

Τα προγράμματα που χρησιμοποιήθηκαν και συνδυάστηκαν είναι το NetBeans IDE και το Xampp. Ακολουθεί περιγραφή τους:

- Το NetBeans είναι περιβάλλον ανάπτυξης εφαρμογών γραμμένο σε Java και υποστηρίζει εύρος γλωσσών προγραμματισμού όπως είναι η Java, η C/C++. Παρέχει πρώτης κατηγορίας ολοκληρωμένη υποστήριξη για τις νεότερες τεχνολογίες Java.[23] Είναι ένα εργαλείο συγγραφής κώδικα και διαθέτει μια μεγάλη ποικιλία από εργαλεία και φόρμες. Παρέχει διαφορετικές προβολές των δεδομένων, μέσα από πολλαπλά παράθυρα και χρήσιμα εργαλεία για τη δημιουργία εφαρμογών και για τη διαχείρισή τους αποδοτικά, επιτρέποντάς στην εστίαση των δεδομένων γρήγορα και εύκολα. Ο κώδικας είναι καλά οργανωμένος και παρέχει αναλυτικά εργαλεία για αυτόματη εύρεση των λαθών και προτάσεις για τη διόρθωσή τους. [24] Τέλος, παρέχει την ευκολία δημιουργίας GUI. Το μοναδικό μειονέκτημά του είναι η ταχύτητά του, καθότι επηρεάζεται επειδή “τρέχει” σε εικονική μηχανή.
- Το XAMPP αποτελεί στην ουσία ένα πακέτο, το οποίο περιλαμβάνει τις τελευταίες εκδόσεις του Apache, της PHP και της MySQL, ενώ περιλαμβάνει επίσης και αλλα τρία χρήσιμα εργαλεία, PhpMyAdmin, Filezilla Server, Mercury Mail. Το XAMPP είναι κατάλληλο για διάφορα λειτουργικά συστήματα (Linux, Windows ,Solaris ,Mac). [22] Η εφαρμογή που αναπτύχθηκε έχει σχεδιαστεί έχοντας ως βάση την μέγιστη δυνατή λειτουργικότητα. Η εφαρμογή αυτή μπορεί να εγκατασταθεί σε οποιοδήποτε λειτουργικό σύστημα, είναι προσπελάσιμη μέσω οποιουδήποτε φυλλομετρητή (browser) και είναι προσιτή στη διαχείριση. Αποτελείται από μια βάση δεδομένων που περιέχει όλες τις απαραίτητες πληροφορίες για τη λειτουργία του συστήματος, από αποθηκευτικά μέσα και από το περιβάλλον εργασίας, δηλαδή το λογισμικό διεπαφής που επεξεργάζεται τις πληροφορίες και κάνει δυνατή την αλληλεπίδραση των χρηστών με το εκπαιδευτικό υλικό. Τέλος, η εφαρμογή είναι βασισμένη στο πρότυπο τύπου «πελάτη-εξυπηρετητή» (clientserver). Το XAMPP χρησιμοποιεί το MySql διακομιστή ως σύστημα διαχείρισης βάσης δεδομένων. Η χρήση του MySql παρέχει πολλές δυνατότητες, όπως είναι υψηλή απόδοση, χαμηλό κόστος διαθεσιμότητα του κώδικα προέλευσης και έχει την ευχέρεια να μας επιτρέπει να αποθηκεύουμε, να αναζητάμε, να ανακαλούμε τα δεδομένα μας με εύκολο και γρήγορο τρόπο.

7.2. Χρήση προγραμμάτων

Αρχικά, έγινε εγκατάσταση του xampp για να μπορώ να λειτουργώ τον server τοπικά στον υπολογιστή μου. Αφού το εγκατέστησα, εμφανίστηκε η ακόλουθη εικόνα 7.1.



Εικόνα 7.1. παράθυρο Xampp

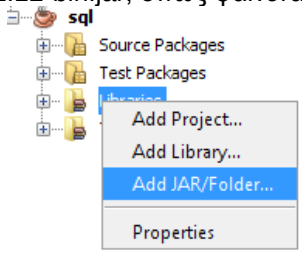
Για να συνεχίζει να λειτουργεί, επέλεξα Apache και MySql και μόλις έδειξε δίπλα τους τη λειτουργία Running, για να μπορώ να έχω πρόσβαση στο http://localhost/xampp/ την αρχική σελίδα του xampp. Πατώντας στο κάτω αριστερό μέρος της οθόνης το phpmyadmin, μεταφέρθηκα στη σελίδα http://localhost/phpmyadmin/ όπου μπορώ να φτιάξω τη δική μου βάση δεδομένων με πίνακες και πεδία.

Για να μπορώ να έχω πρόσβαση στο server μέσω του προγράμματος NetBeans, χρησιμοποίησα τις ακόλουθες γραμμές στον κώδικά μου στη Java :

```
Class.forName("com.mysql.jdbc.Driver");
Connection con  = DriverManager.getConnection("jdbc:mysql://localhost:3306/test","root","");
```

Όπου test είναι η βάση δεδομένων που δημιούργησα με όλα τα δεδομένα του προγράμματος. Αφού έχω πλέον δημιουργήσει μια σύνδεση μεταξύ του Netbeans και του xampp, μπορώ να χρησιμοποιήσω όλες τις δυνατότητες που έχω και στην php, δηλαδή να δημιουργήσω ερωτήματα, να δημιουργήσω/να διαγράψω πίνακες και να εισάγω δεδομένα.

Για να μπορέσω, όμως, να έχω πρόσβαση στη βάση δεδομένων που βρίσκεται και δημιουργήθηκε στο Xampp, πέρασα στο πρόγραμμα NetBeans το αρχείο mysql-connector-java-5.1.12-bin.jar, όπως φαίνεται στην εικόνα 7.2.



Εικόνα 7.2. Σύνδεση με βάση δεδομένων MySql

Κεφάλαιο 8
Τεκμηρίωση του λογισμικού και ανάπτυξη του συστήματος διδασκαλίας

Το προσαρμοστικό σύστημα διδασκαλίας που δημιούργησα προορίζεται για μαθητές και καθηγητές οποιασδήποτε βαθμίδας και οποιουδήποτε επιπέδου. Αποτελεί ένα σημαντικό εργαλείο για όλους τους καθηγητές για οποιοδήποτε μάθημα και αν αυτοί διδάσκουν και τους δίνει τη δυνατότητα να προσθέσουν μαθήματα, ύλη για το κάθε μάθημα το οποίο διδάσκουν, αλλά και να θέτουν ερωτήσεις που σχετίζεται με συγκεκριμένη ύλη μαθημάτων. Από την πλευρά των μαθητών, μπορούν να εισέρχονται στο σύστημα και να διαβάζουν τη διαθέσιμη ύλη των καθηγητών, αλλά και να απαντούν σε ερωτήσεις που αφορούν στην ύλη την οποία έχουν διαβάσει. Αναλυτικότερα, ακολουθούν οι λειτουργίες που έχει ο κάθε μαθητής που χρησιμοποιεί το σύστημα, αλλά και οι καθηγητές.

8.1. Περιγραφή του τρόπου λειτουργίας του συστήματος διδασκαλίας

Αρχικά, όταν κάποιος ανοίγει το πρόγραμμα του εμφανίζονται οι επιλογές login, register (student, teacher), about και exit. Αν επιλέξει να κάνει login, τότε θα εισάγει το username και το password που είχε δώσει κατά την εγγραφή του και να κάνει είσοδο στο πρόγραμμα και να χρησιμοποιήσει τις λειτουργίες του. Σχηματικά φαίνεται στην εικόνα 8.1.

Εικόνα 8.1. Παράθυρο διαλόγου για είσοδο χρήστη

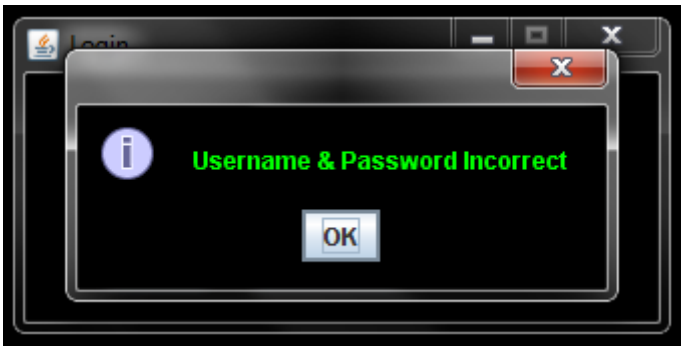
Όταν θα εισάγει το όνομα χρήστη και το συνθηματικό, θα γίνει έλεγχος αν το username, το password ή και τα δύο είναι λάθος και θα εμφανίσει αντίστοιχο μήνυμα λάθους στην οθόνη, όπως φαίνεται στην εικόνα 8.2.

Εικόνα 8.2. Παράθυρο μηνύματος λάθος πληκτρολόγηση κωδικού

Όταν θα εισάγει το όνομα χρήστη και το συνθηματικό, θα γίνει έλεγχος αν το username, το password ή και τα δύο είναι λάθος και θα εμφανίσει αντίστοιχο μήνυμα λάθους στην οθόνη, όπως φαίνεται στην εικόνα 8.3.

Εικόνα 8.3. Παράθυρο μηνύματος λάθος πληκτρολόγηση ονόματος χρήστη

Σε περίπτωση που το όνομα και το συνθηματικό χρήστη είναι λανθασμένα, θα εμφανίσει αντίστοιχο μήνυμα λάθους στην οθόνη, όπως φαίνεται στην εικόνα 8.4.



Εικόνα 8.4. Παράθυρο μηνύματος λάθος πληκτρολόγηση ονόματος χρήστη και συνθηματικού

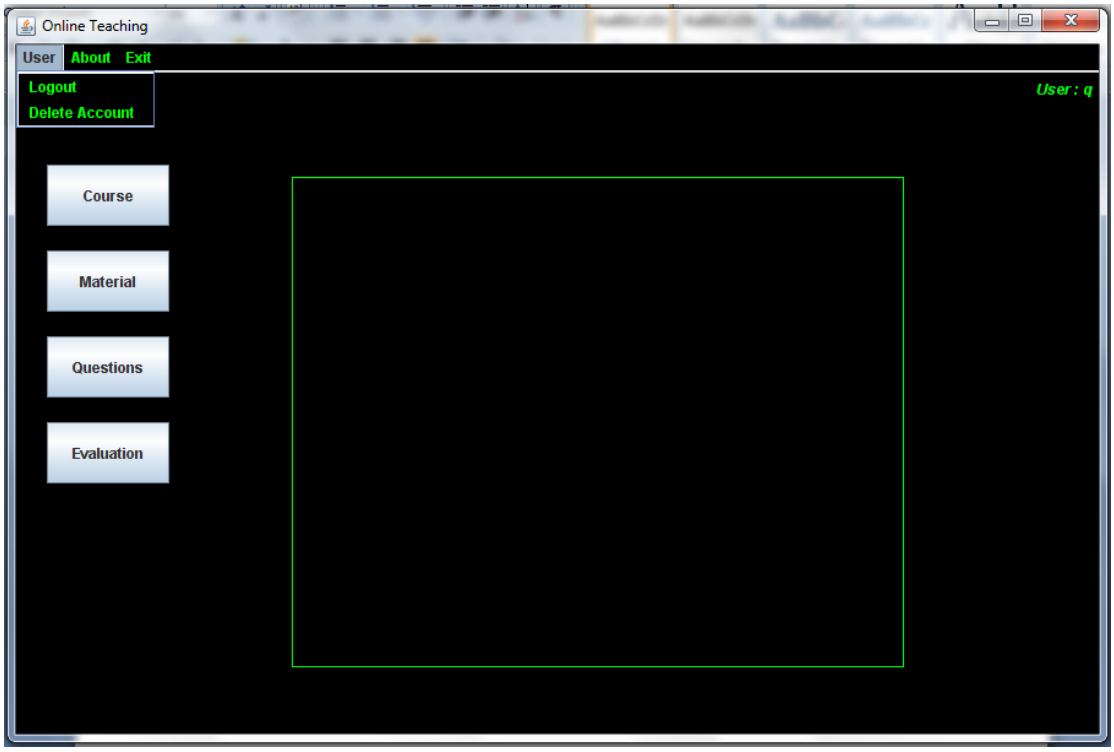
Σε περίπτωση που εισάγει τα στοιχεία του σωστά (αφού ελεγχθούν και επαληθευτούν με τα στοιχεία που είναι καταχωρημένα στη βάση δεδομένων, θα εμφανιστεί μήνυμα καλωσορίσματος στο χρήστη, όπως φαίνεται στην εικόνα 8.5.

Εικόνα 8.5. Παράθυρο μηνύματος καλωσορίσματος χρήστη

Ανάλογα με το αν έκανε είσοδο σα μαθητής ή καθηγητής, θα του εμφανίσει και το παράθυρο με τις αντίστοιχες λειτουργίες. Θα αναλύσουμε αυτές τις περιπτώσεις:

8.2. Είσοδος σαν καθηγητής

Θα του εμφανίσει την ακόλουθη οθόνη(εικόνα 8.6)



Εικόνα 8.6. interface καθηγητή

Όπως φαίνεται και στην εικόνα, πλέον αλλάζει το αρχικό μενού και ο χρήστης μπορεί να κάνει έξοδο από το πρόγραμμα με την επιλογή Logout και διαγραφή του λογαριασμού του με την επιλογή Delete Account. Στο πάνω δεξί μέρος της οθόνης φαίνεται το username του καθηγητή με το οποίο έκανε login και αριστερά υπάρχουν κάποια κουμπιά.

8.2.1. Λειτουργία “Course”

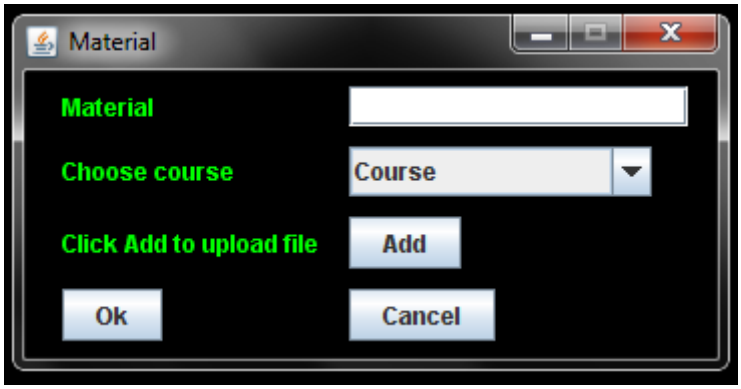
Πατώντας αυτό το κουμπί, ο καθηγητής μπορεί να εισάγει μάθημα το οποίο διδάσκει. Θα του εμφανίσει το ακόλουθο παράθυρο (εικόνα 8.7).

Εικόνα 8.7. Εισαγωγή μαθήματος

Μπορεί να εισάγει το όνομα του μαθήματος και αμέσως θα αποθηκευτεί στη βάση δεδομένων.

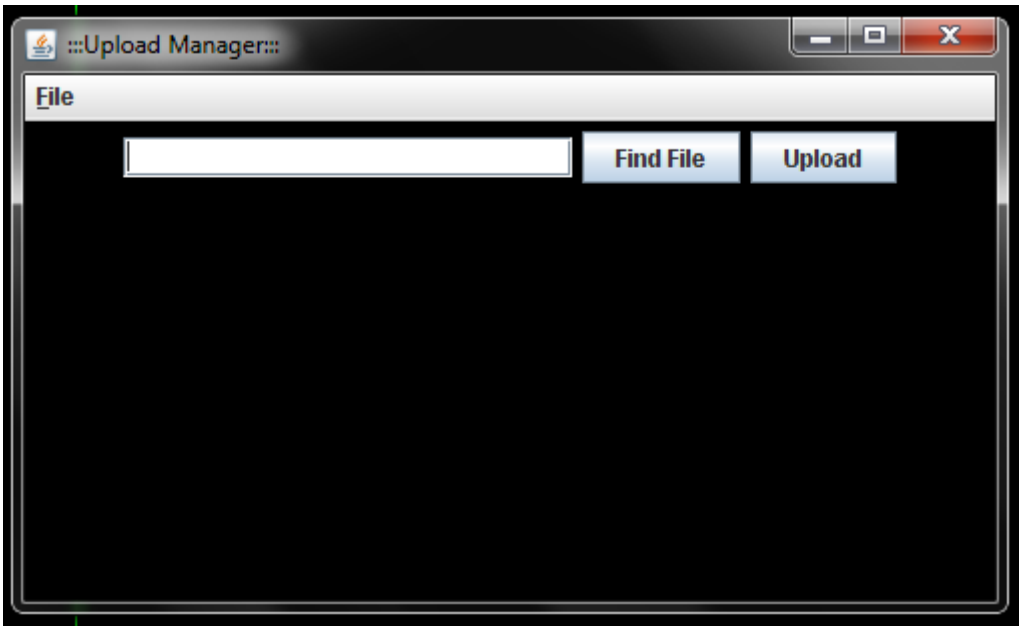
8.2.2. Λειτουργία “Material”

Μόλις επιλέξει να εισάγει ύλη του εμφανίζεται το εξής παράθυρο με τις αντίστοιχες επιλογές (εικόνα 8.8).



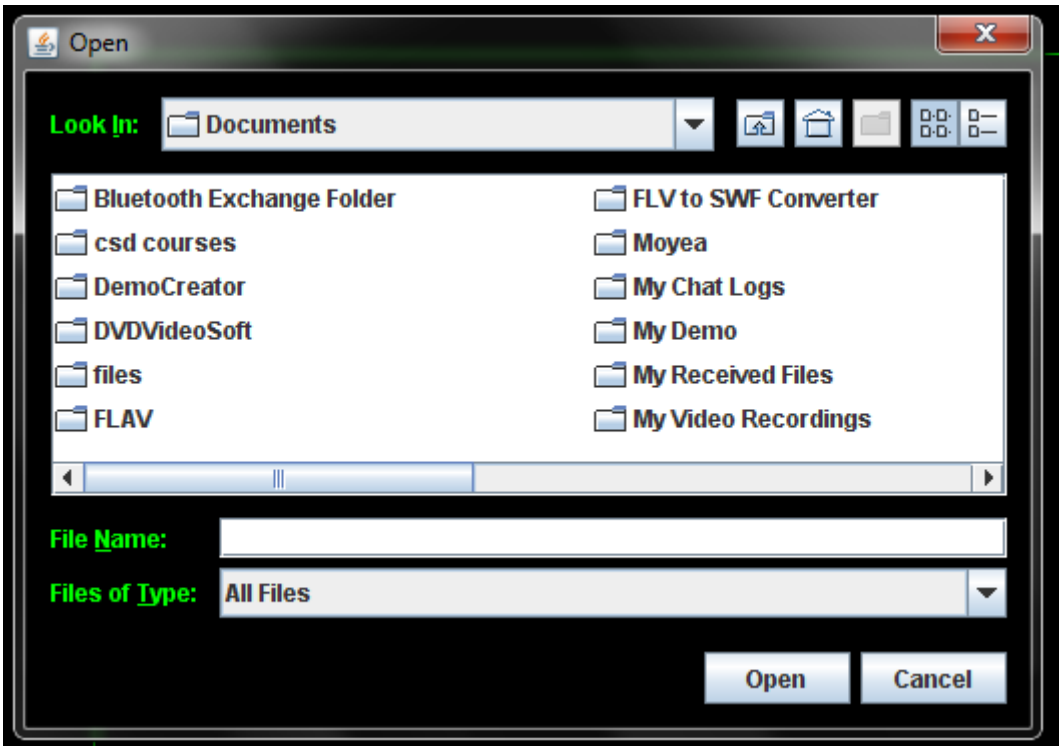
Εικόνα 8.8. Εισαγωγή ύλης

Στο πεδίο Material, εισάγει το όνομα της ύλης που θέλει να προσθέσει. Στην αμέσως επόμενη επιλογή, μπορεί να επιλέξει ένα από τα ήδη υπάρχοντα μαθήματα που έχει εισάγει ο συγκεκριμένος καθηγητής. Αφού πατήσει πάνω στο Course, θα του ανοίξει μια λίστα με όλα τα διαθέσιμα μαθήματα. Αφού επιλέξει και το μάθημα, μπορεί να ανεβάσει αρχείο για τη συγκεκριμένη ύλη, πατώντας το κουμπί Add. Όταν το πατήσει θα του εμφανιστεί ένα παράθυρο που θα τον παραπέμπει να ψάξει αρχείο που είναι αποθηκευμένο στον υπολογιστή του για να το ανεβάσει στη βάση δεδομένων.(εικόνα 8.9).



Εικόνα 8.9. Εύρεση αρχείου

Όπως φαίνεται στην εικόνα 8.9, αφού πατήσει το κουμπί Find File, θα του εμφανιστεί το ακόλουθο παράθυρο (εικόνα 8.10) και αφού επιλέξει το αρχείο και πατήσει το Open, όπου θα επιστρέψει στο προηγούμενο παράθυρο (εικόνα 8.9) για να πατήσει στη συνέχεια Upload.

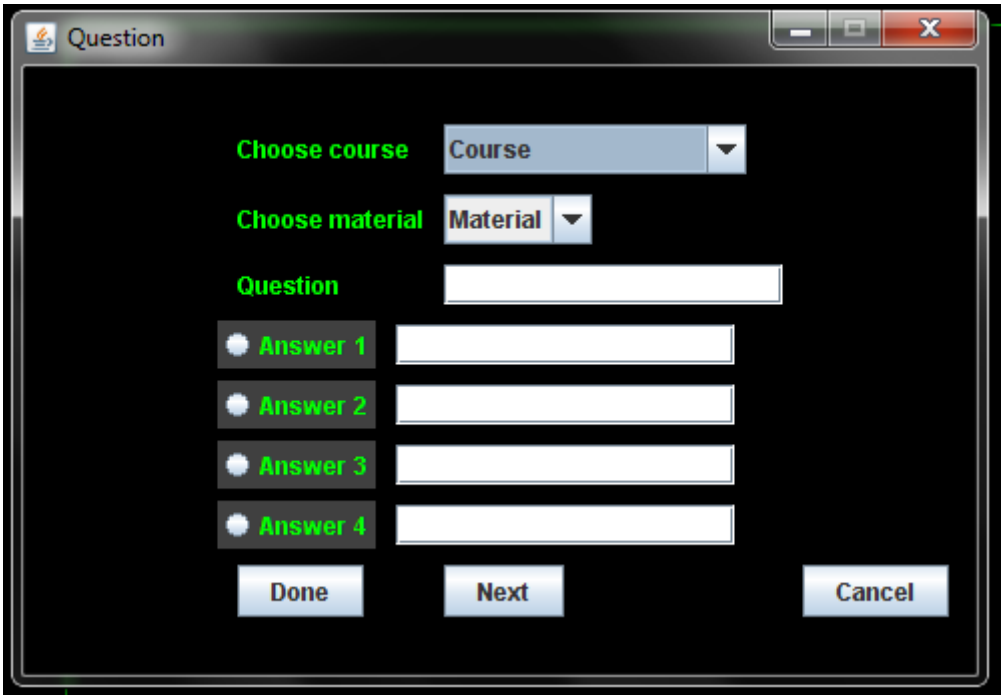


Εικόνα 8.10. Αναζήτηση αρχείου στον υπολογιστή

Αφού, λοιπόν, έχει επιλέξει και το αρχείο που θέλει να ανεβάσει, πατάει Ok στο αρχικό μενού για να εισάγει την ύλη στη βάση δεδομένων ή Cancel για να ακυρώσει την όλη διαδικασία.

8.2.3. Λειτουργία “Questions”

Μόλις επιλέξει τη συγκεκριμένη λειτουργία, θα του εμφανιστεί το ακόλουθο παράθυρο (εικόνα 8.11) για να μπορέσει να εισάγει ερώτηση για την ύλη που έχει ήδη εισάγει. Έτσι, μπορεί να επιλέξει κάποιο από τα μαθήματα που έχει ήδη εισάγει και αυτόματα θα ανανεωθεί και η λίστα με τη διαθέσιμη ύλη που μπορεί να επιλέξει και που αφορά αποκλειστικά στο συγκεκριμένο μάθημα. Στο πεδίο Question, εισάγει το όνομα της ερώτησης και στα πεδία που ακολουθούν εισάγει τις διαθέσιμες απαντήσεις για τη συγκεκριμένη ερώτηση και επιλέγοντας με το ποντίκι κάποιο από Answer 1 έως Answer 4, στην ουσία επιλέγει τη σωστή απάντηση. Μπορεί να επιλέξει το κουμπί Done και να εισάγει την τρέχουσα ερώτηση στη συγκεκριμένη ύλη του μαθήματος που επέλεξε ή να πατήσει το κουμπί Next και να εισάγει και άλλες ερωτήσεις για το συγκεκριμένο μάθημα, αλλά με τη δυνατότητα η ερώτηση που θα εισάγει να αφορά διαφορετική ύλη. Πατώντας το κουμπί Cancel, ακυρώνει την τρέχουσα ερώτηση από την εισαγωγή της στη βάση δεδομένων.



Εικόνα 8.11. Εισαγωγή ερώτησης/ερωτήσεων

8.2.4. Λειτουργία “Evaluation”

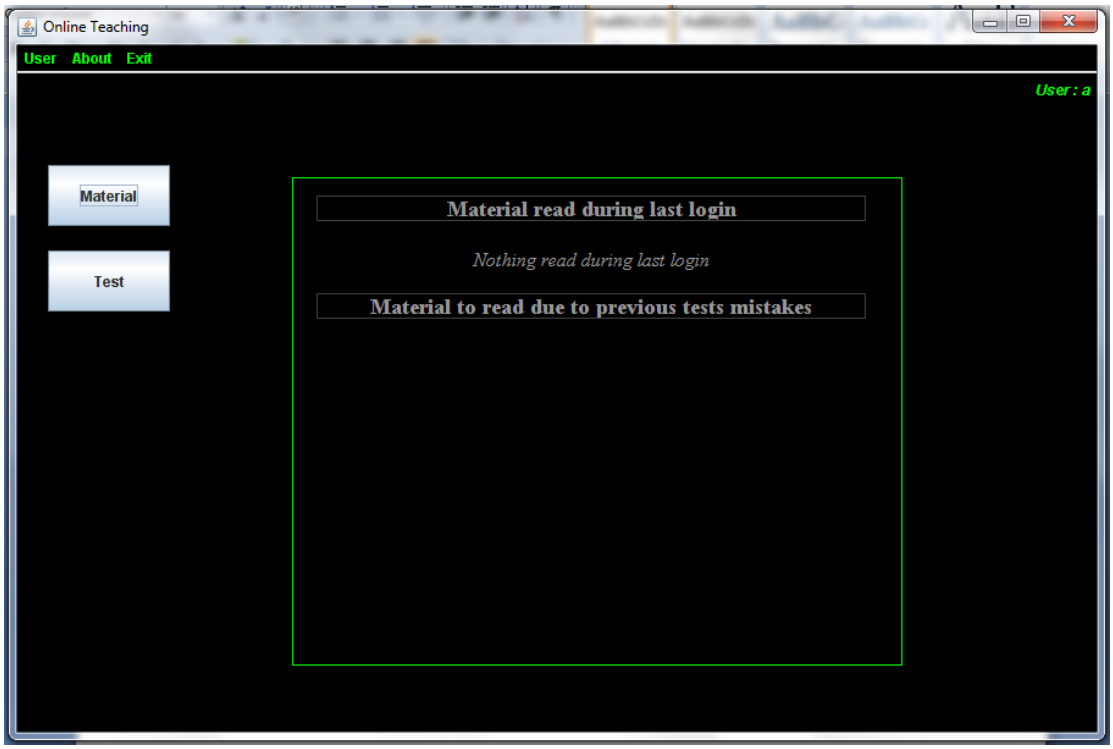
Με αυτή την επιλογή, του εμφανίζεται το ακόλουθο παράθυρο (εικόνα 8.12) και πατώντας πάνω στο Course, μπορεί να επιλέξει ένα από τα μαθήματα που διδάσκει και να του εμφανιστούν όλοι οι μαθητές που είναι εγγεγραμμένοι στο συγκεκριμένο μάθημα με την προϋπόθεση ο κάθε ένας να έχει κάνει τεστ έστω και μια φορά. Μαζί με το ονοματεπώνυμο του μαθητή, μπορεί να δει και τη βαθμολογία του σε απόλυτο και σχετικό βαθμό, δηλαδή από 0 έως 100 αλλά και από Α έως και Ε.

Εικόνα 8.12. Επιλογή μαθήματος για εμφάνιση αξιολόγησης μαθητών

Ο καθηγητής, μπορεί να επιλέξει τις διαθέσιμες αυτές λειτουργίες όσες φορές επιθυμεί και στη συνέχεια να πατήσει Logout από τη μπάρα του User και να αποσυνδεθεί από το πρόγραμμα.

8.3. Είσοδος σα μαθητής

Αφού ελεγχθεί το username και το password του χρήστη και δια πιστωθεί ότι ανήκουν σε μαθητή, θα του εμφανιστεί το παραπάνω παράθυρο (εικόνα 8.13). Στο κέντρο του παραθύρου βλέπουμε ότι έχουν εμφανιστεί δυο κύριοι τίτλοι, “Material read during last login” και “Material to read due to previous test mistakes”. Πιο συγκεκριμένα, εκεί που βλέπουμε να γράφει “Nothing read during last login”, σημαίνει ότι την προηγούμενη φορά που έκανε είσοδο ο συγκεκριμένος χρήστης, δε διάβασε καμία ύλη, σε διαφορετική περίπτωση θα του εμφάνιζε το όνομα της ύλης που είχε διαβάσει. Στη δεύτερη κατηγορία δεν εμφανίζει κάτι, που σημαίνει ότι δεν έχει κάνει ακόμα κάποιο test ή στα tests που έχει κάνει δεν έχει κάνει κανένα λάθος ή ακόμα και αν είχε κάνει λάθος, έχει διαβάσει τη συγκεκριμένη ύλη οπότε και του έχει εξαφανιστεί από την οθόνη. Την τελευταία περίπτωση θα την αναλύσουμε παρακάτω.

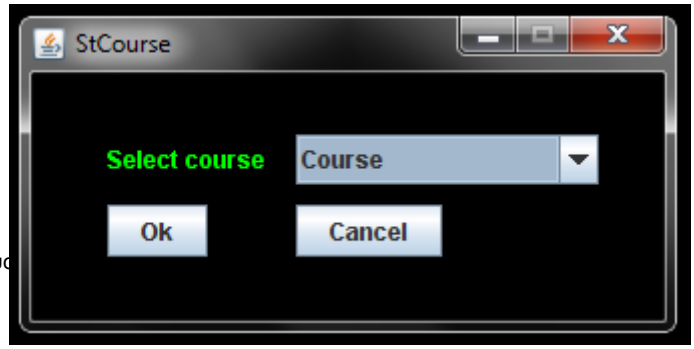


Εικόνα 8.13. interface μαθητή

Στο αριστερό μέρος της οθόνης βλέπουμε πως έχουν εμφανιστεί δυο κουμπιά, τα Material και Test.

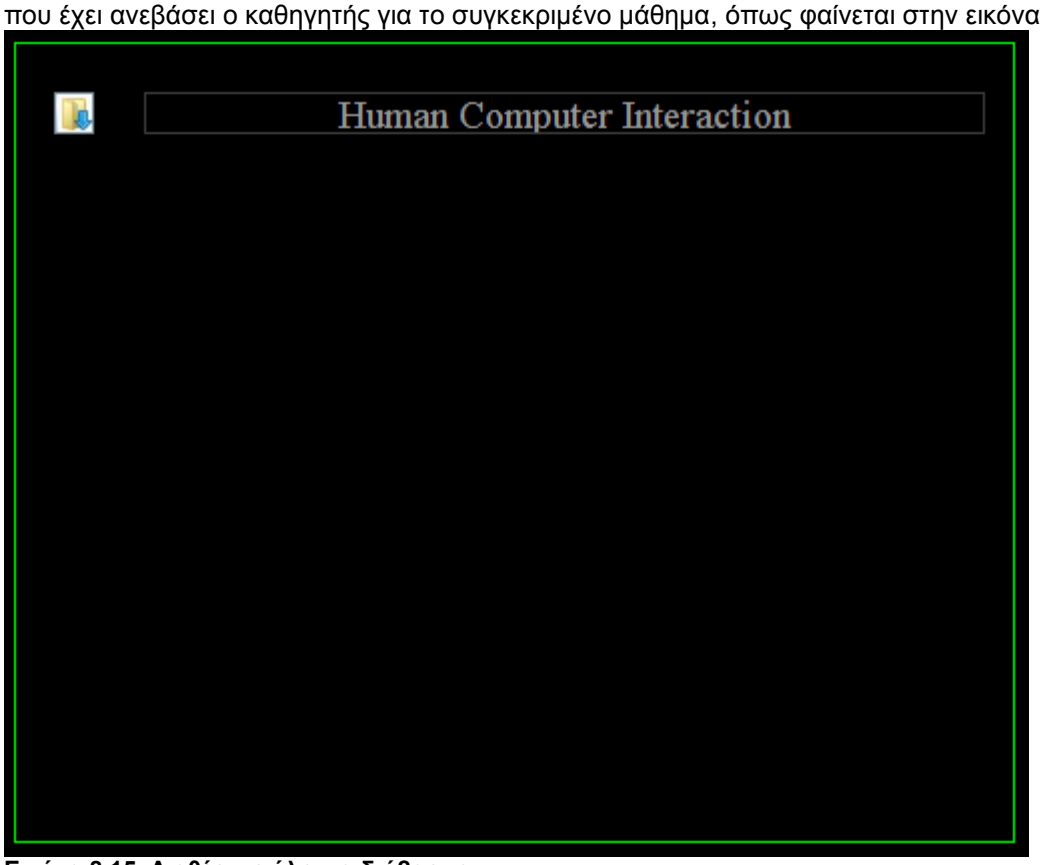
8.3.1. Λειτουργία “Material”

Μόλις πατήσει το συγκεκριμένο κουμπί, θα του εμφανιστεί το ακόλουθο παράθυρο (εικόνα 8.14).



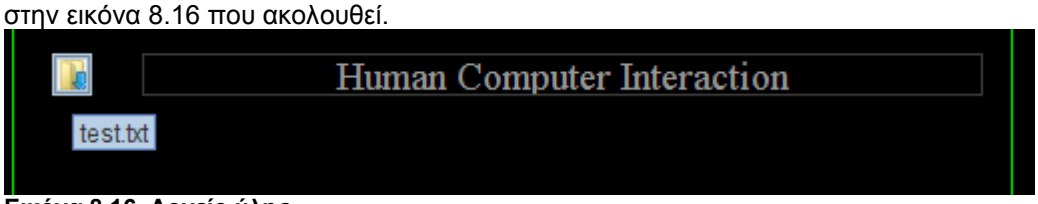
Εικόνα 8.14. Επιλογή ύλης για διάβασμα

Μόλις επιλέξει το μάθημα που επιθυμεί, θα του εμφανιστεί στην κεντρική οθόνη όλη η ύλη που έχει ανεβάσει ο καθηγητής για το συγκεκριμένο μάθημα, όπως φαίνεται στην εικόνα 8.15.



Εικόνα 8.15. Διαθέσιμη ύλη για διάβασμα

Αριστερά του παραθύρου εμφανίζεται μια εικόνα, που μόλις την πατήσει, θα του εμφανιστεί η ύλη που έχει ανεβάσει ο καθηγητής για το συγκεκριμένο μάθημα. Δεξιά της φωτογραφίας εμφανίζεται το όνομα της ύλης. Χαρακτηριστικά, αν ακουμπήσει το ποντίκι πάνω από την εικόνα, θα του εμφανιστεί το όνομα του αρχείου που έχει ανεβάσει ο καθηγητής, όπως φαίνεται στην εικόνα 8.16 που ακολουθεί.

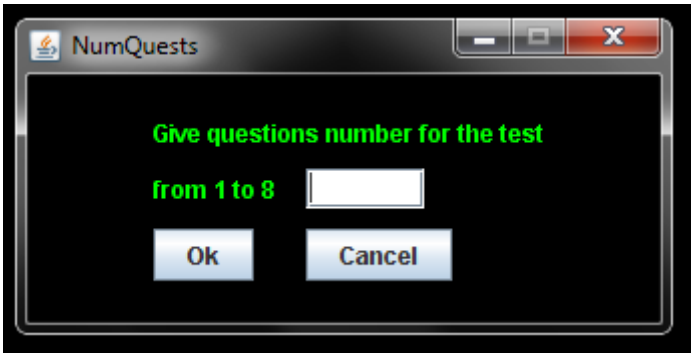


Εικόνα 8.16. Αρχείο ύλης

Μόλις επιλέξει να το διαβάσει, θα σβήσει το όνομα της ύλης από την ύλη που πρέπει να διαβάσει για κάθε λανθασμένη απάντηση στο τεστ που την αφορά, αλλά και θα του εμφανιστεί σαν ύλη που διάβασε την τελευταία φορά που έκανε login στην επόμενη είσοδό του στο πρόγραμμα. Αναφορικά με το τεστ, ακολουθεί περιγραφή.

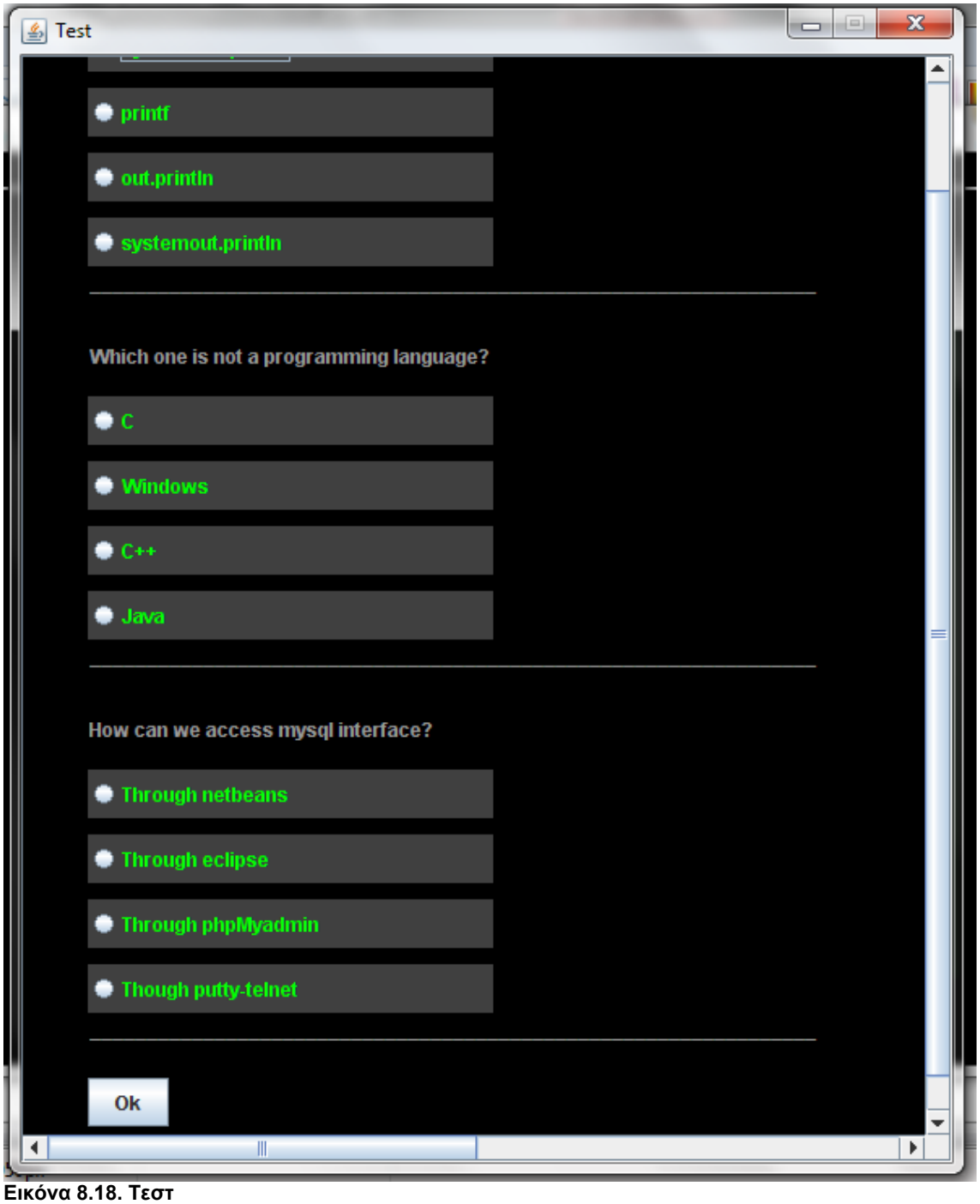
8.3.2. Λειτουργία “Test”

Αν επιλέξει το κουμπί Test από το παράθυρο, τότε θα του εμφανιστεί η ακόλουθη εικόνα 8.17, όπου μπορεί να επιλέξει τον αριθμό των ερωτήσεων που θα περιλαμβάνει το τεστ που ια ζητηθεί να κάνει. Το εύρος των ερωτήσεων εμφανίζεται ανάλογα με το πόσες ερωτήσεις έχουν εισάγει οι καθηγητές για κάθε μια από τις ύλες που έχει διαβάσει ο μαθητής.



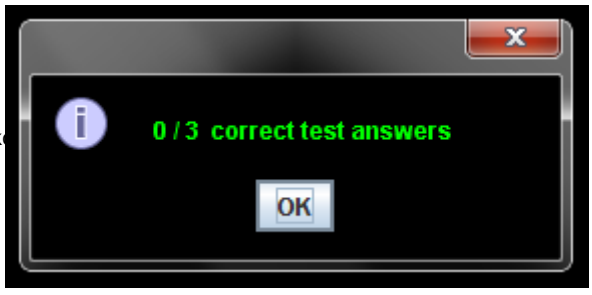
Εικόνα 8.17. Διαθέσιμος αριθμός ερωτήσεων

Αφού επιλέξει τον αριθμό των ερωτήσεων, θα πατήσει το Ok για να προχωρήσει στο τεστ. Το τεστ που θα του εμφανιστεί θα έχει την ακόλουθη μορφή (εικόνα 8.18).



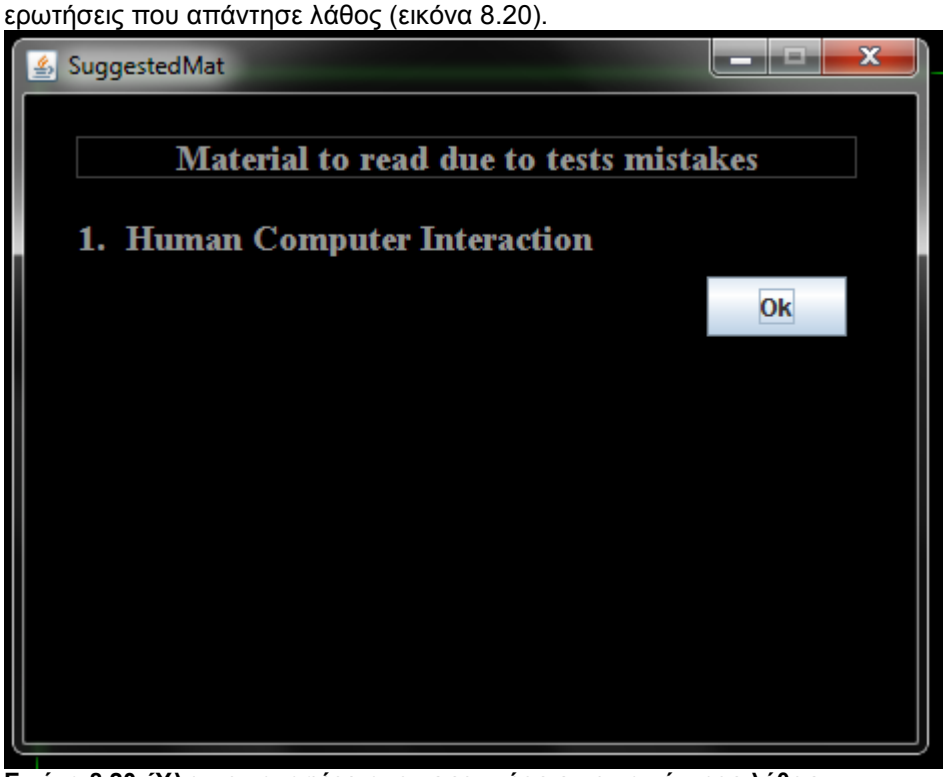
Εικόνα 8.18. Τεστ

Στη συγκεκριμένη περίπτωση ο μαθητής επέλεξε 3 ερωτήσεις για το τεστ του. Αφού διαβάσει τις ερωτήσεις και απαντήσει ανάλογα, θα πατήσει το Ok για να προβεί σε υποβολή του τεστ. Τότε, θα του εμφανιστεί στην οθόνη μήνυμα με το ποσοστό των σωστών ερωτήσεων που απάντησε όπως φαίνεται ακολούθως (εικόνα 8.19).



Εικόνα 8.19. Ποσοστό σωστά απαντηθέντων ερωτήσεων

Στο συγκεκριμένο τεστ, ο μαθητής δεν απάντησε σωστά καμία ερώτηση. Μόλις πατήσει το Ok, θα του εμφανιστεί ένα παράθυρο με την ύλη που πρέπει να διαβάσει, σύμφωνα με τις ερωτήσεις που απάντησε λάθος (εικόνα 8.20).



Εικόνα 8.20. ΄Υλη που αναφέρεται στις ερωτήσεις που απάντησε λάθος

Άρα πρέπει να διαβάσει την αντίστοιχη ύλη. Πλέον, κάθε φορά που θα κάνει είσοδο στο πρόγραμμα, στην αρχική οθόνη θα του εμφανίζεται η συγκεκριμένη ύλη που πρέπει να διαβάσει και μόλις τη διαβάσει, τότε θα αφαιρεθεί η συγκεκριμένη από τη λίστα.

Κεφάλαιο 9
Επεξήγηση κώδικα

9.1. Κατασκευή βάσης δεδομένων

Ένα από τα σημαντικότερα κομμάτια, αν όχι το σημαντικότερο, κάθε συστήματος, είναι αναμφίβολα η βάση δεδομένων στην οποία καταχωρούνται όλα όσα χρειάζονται για την εύρυθμη λειτουργία του συστήματος. Όλα τα δεδομένα που καταχωρούνται στην βάση δεδομένων. Είναι λοιπόν απαραίτητο, αυτά να οργανωθούν με το πιο σωστό τρόπο ώστε να πληρούνται οι εξής προϋποθέσεις:

- Να μην υπάρχει απώλεια δεδομένων
- Να αποφεύγεται η ύπαρξη περιττών δεδομένων (π. χ. διπλοεγγραφές)
- Η αναζήτησή τους μέσα στη βάση να γίνεται με τον ταχύτερο τρόπο
- Να εμφανίζονται πάντα τα ζητούμενα από το χρήστη

9.1.1. Πίνακες της βάσης δεδομένων με τα χαρακτηριστικά τους

- **course** (course, tusername)
- **evaluation** (profname, stname, coursename, grade, relativegrade)
- **material** (matname, profname, coursename, filenm, filename)
- **material_read** (stusername, matname, timesread, lastread)
- **questions** (tusernm, qname, matname, coursename, a1, a2, a3, a4, correctanswer)
- **student** (id, fname, lname, age, username, password)
- **suggestedmat** (matname, coursename, stname, profname, sugmatread)
- **teacher** (tid, tfname, tlname, course, tusername, tpassword)
- **user** (uid, usernm, upass)

9.1.2. Επεξήγηση πινάκων

- Ο πίνακας course αφορά στα μαθήματα που διδάσκει ο κάθε καθηγητής και κάθε φορά που εισάγει ένα μάθημα μέσω του συστήματος, αποθηκεύεται στον πίνακα το όνομα του μαθήματος (course) και το όνομα χρήστη του καθηγητή (tusername).
- Ο πίνακας evaluation αφορά στην αξιολόγηση του κάθε μαθητή και τα δεδομένα του αλλάζουν κάθε φορά που κάποιος μαθητής συμπληρώνει κάποιο ερωτηματολόγιο-test. Στο πίνακα αυτό αποθηκεύεται κάθε φορά το όνομα χρήστη του καθηγητή (profname), το όνομα χρήστη του μαθητή (stname), το όνομα του μαθήματος που αφορούσαν οι ερωτήσεις που υπήρχαν στο τεστ (coursename), ο βαθμός του μαθητή, από 0 μέχρι 100, ο οποίος κάθε φορά που συμπληρώνει άλλο τεστ αλλάζει (grade) και ο σχετικός βαθμός που κυμαίνεται από Α έως Ε (relativegrade).
- Ο πίνακας material αφορά στην ύλη που εισάγει ο κάθε καθηγητής για το μάθημα που διδάσκει. Κάθε φορά που επιλέγει αυτή τη λειτουργία, αποθηκεύεται το όνομα της ύλης που εισάγει (matname), το όνομα χρήστη του καθηγητή (profname), το όνομα του μαθήματος που αφορά η συγκεκριμένη ύλη (coursename), το όνομα του αρχείο που θα αποθηκευτεί στη βάση δεδομένων (filenm), αλλά και το ίδιο το αρχείο (filename).
- Ο πίνακας material_read περιέχει την ύλη που έχει διαβάσει ο κάθε μαθητής που χρησιμοποιεί το σύστημα. Έτσι, κάθε φορά που διαβάσει κάτι, αποθηκεύεται το όνομα χρήστη του μαθητή (stusername), το όνομα της ύλης (matname), πόσες φορές έχει διαβάσει τη συγκεκριμένη ύλη (timesread) και αν τη συγκεκριμένη ύλη τη διάβασε την τελευταία φορά που έκανε είσοδο στο σύστημα (lastread), όπου αποθηκεύεται η τιμή 1 αν το διάβασε, διαφορετικά η τιμή αυτή γίνεται 0.
- Ο πίνακας questions περιέχει τις ερωτήσεις που εισάγει ο κάθε καθηγητής για την ύλη που εισάγει. Κάθε φορά που επιλέγει τη συγκεκριμένη λειτουργία, αποθηκεύεται το όνομα χρήστη του καθηγητή (tusernm), το όνομα της ερώτησης (qname), το όνομα της ύλης (matname) όπου αναφέρεται η συγκεκριμένη ερώτηση, το όνομα του μαθήματος που αναφέρεται η ύλη και κατ' επέκταση η ερώτηση (coursename), οι εναλλακτικές απαντήσεις a1, a2, a3, a4 και η σωστή απάντηση για τη συγκεκριμένη ερώτηση (correctanswer).

- Ο πίνακας student περιλαμβάνει όλα τα στοιχεία του μαθητή που κάνει εγγραφή στο σύστημα και όταν πραγματοποιηθεί αυτό, αποθηκεύεται στον πίνακα ένας χαρακτηριστικός μοναδικός κωδικός για κάθε μαθητή (id), το όνομα του μαθητή (fname), το επίθετό του (lname), η ηλικία του (age), το όνομα χρήστη που θα επιλέξει (username), αλλά και ο κωδικός που θα έχει για να έχει πρόσβαση στο σύστημα (password).
- Ο πίνακας suggestedmat περιέχει την ύλη που πρέπει να διαβάσει ο μαθητής ανάλογα με τα λάθη που έχει κάνει σε προηγούμενα τεστ και περιέχει το όνομα της ύλης όπου αναφερόταν η συγκεκριμένη ερώτηση (matname), το όνομα του μαθήματος (coursename), το όνομα χρήστη του μαθητή (stname), το όνομα χρήστη του καθηγητή (profname) και ένα πεδίο που η τιμή του γίνεται 1 αν πρέπει να διαβάσει τη συγκεκριμένη ύλη (sugmatread).
- Ο πίνακας teacher περιλαμβάνει όλα τα στοιχεία του καθηγητή που κάνει εγγραφή στο σύστημα και περιέχει ένα μοναδικό χαρακτηριστικό αριθμό (tid), το όνομά του (tfname), το επίθετό του (tlname), το μάθημα το οποίο διδάσκει (course), το όνομα χρήστη που επιλέγει (tusername), αλλά και τον κωδικό που πρέπει να διαθέτει για να έχει πρόσβαση στο σύστημα (tpassword).
- Ο πίνακας user περιέχει όλους τους μαθητές και καθηγητές που είναι εγγεγραμμένοι στο σύστημα και το χρησιμοποιούν. Πιο συγκεκριμενα, για τον κάθε ένα περιέχει ένα πεδίο το οποίο έχει την τιμή 1 για καθηγητές και την τιμή 0 για μαθητές (uid), το όνομα χρήστη του καθενός (usernm), αλλά και τον κωδικό που χρησιμοποιεί (upass).

9.2. Κώδικας σε java για την κατασκευή του συστήματος

Στο πρόγραμμα NetBeans έχει υλοποιηθεί το πρόγραμμα, όπως αναφέραμε και παραπάνω. Έχω δημιουργήσει τρεις φακέλους, τον GUI, τον MAIN και τον SQL. Ακολουθεί περιγραφή των φακέλων αυτών.

9.2.1. MAIN

Περιλαμβάνει την κλάση Main.java, δηλαδή αρχικοποιείται το πρόγραμμα και καλείται η συνάρτηση που εμφανίζει το κύριο παράθυρο του προγράμματος. Ο κώδικας ακολουθεί:

```
    public class Main {
        public static void main(String[] args ) throws ClassNotFoundException, SQLException
        {
            Sql_Window window = new Sql_Window();
            window.initialize();
        }

}
```

9.2.2. SQL

Περιλαμβάνει τις κλάσεις Sql_Net.java και User.java.

9.2.2.1. User.java

Περιλαμβάνει όλες τις συναρτήσεις που αφορούν στους χρήστες. Ακολουθεί ο κώδικάς τους.

```
public static String getIn() throws IOException {…}
public static void registerS(String fname, String lname, String age, String usernm, String pswd)
throws IOException, ClassNotFoundException, SQLException {…}
```

public static void registerT(String fname, String lname, String course, String usernm, String pswd) throws IOException, ClassNotFoundException, SQLException {…}
public static int userexistslogin(String usernm, String pswd) throws IOException, ClassNotFoundException, SQLException {…}
public static int userexistsregister(String usernm) throws IOException, ClassNotFoundException, SQLException {…}
public static String getUsername() {…}
public static void setcurrusername(String str) {…}
public static void setcurrusername(String str) {…}

9.2.2.2. Sql_Net.java

Περιλαμβάνει όλες τις συναρτήσεις που επιλέγουν δεδομένα από τη βάση δεδομένων αλλά και καταχωρούν σε αυτή νέα δεδομένα.

public static Connection connectSql() throws ClassNotFoundException, SQLException {…}
public static String checkStudTeach(String usernm) throws ClassNotFoundException, SQLException, IOException {…}
public static int checkUsernamePasswordIfExists(String username, String password) throws ClassNotFoundException, SQLException
public static int userExistsR(String usernm) throws ClassNotFoundException, SQLException {…}
public static int userExistsL(String usernm, String pswd) throws ClassNotFoundException, SQLException {…}
public static int findmax(int x, int y) {…}
public static void insertusertab(int idst, String usern, String psw) throws ClassNotFoundException, SQLException {…}
public static void addcourse(String coursename, String teachname) throws ClassNotFoundException, SQLException {…}
public static void insertTeacher(String fname, String lname, String course, String usernm, String pswd) throws ClassNotFoundException, SQLException, IOException {…}
public static void insertUser(String fname, String lname, String age, String usernm, String pswd) throws ClassNotFoundException, SQLException, IOException {…}
public static void addquest(String prof, String quest, String mat, String cour, String answ1, String answ2, String answ3, String answ4, String corrans) throws ClassNotFoundException, SQLException {…}
public static void addmat(String materialname, String prof, String crs, File fileName) throws ClassNotFoundException, ClassNotFoundException, ClassNotFoundException, ClassNotFoundException, SQLException, FileNotFoundException, IOException {…}
public static String[] getsuggestedmat(String curusername) throws ClassNotFoundException, SQLException {…}
public static String[] getstlastreadmat(String curusername) throws ClassNotFoundException, SQLException {…}
public static boolean isInteger(String s) {…}
public static int getqmcnt(String curuser) throws ClassNotFoundException, SQLException {…}
public static int getmatcnt(String curuser) throws ClassNotFoundException, SQLException {…}
public static int getmaterialquestcnt(String materialnm) throws ClassNotFoundException, SQLException {…}
public static int[] getrandomnums(int num, int numofquests) {…}
public static ResultSet getquestfields(String quest) throws ClassNotFoundException, SQLException {…}
public static boolean getcorrectanswer(String questname, String answer) throws ClassNotFoundException, SQLException {…}
public static String getrelativegrade(int grade) {…}

public static int getEvaluationforcourse(String tname, String course) throws ClassNotFoundException, SQLException {…}
public static String getstudentname(String stusername) throws ClassNotFoundException, SQLException {…}
public static int checkifcoursehastestanswered(String course) throws ClassNotFoundException, SQLException {…}
public static String[] getevalforstudent(String tname, String course) throws ClassNotFoundException, SQLException {…}
public static void updateEvaluation(String stname, String course, int grade) throws ClassNotFoundException, SQLException {…}
public static void countcourseinquest(String curusename, String[] pin) throws ClassNotFoundException, SQLException {…}
public static int findQinstr(String questionsStr, String question) {…}
public static String getcoursefromquest(String quest) throws ClassNotFoundException, SQLException {…}
public static String getquestfromanswer(String answer) throws ClassNotFoundException, {…}
public static int getallquestionsnumber(String curusername, int numgiven) throws ClassNotFoundException, SQLException {…}
public static String[] getallmaterialquestions(String curusername, int numgiven) throws ClassNotFoundException, SQLException {…}
public static String[] gettestquestions(String[] quests, int[] rands) {…}
public static void getsuggestedmaterial(String curusername, String[] queststr) throws ClassNotFoundException, SQLException {…}
public static void getstr(String str[][]) throws ClassNotFoundException, SQLException {…}
public static String[] getsugtemp() {…}
public static void updatesuggestedmat(String curusername, String[][] str) throws ClassNotFoundException, SQLException {…}
public static String getmatfromquest(String quest) throws ClassNotFoundException, SQLException {…}
public static String getteachfromcourse(String crs) throws SQLException, ClassNotFoundException {…}
public static void deletesuggestedmatwhenread(String curusername, String curmat) throws ClassNotFoundException, SQLException {…}
public static void setmatlastreadzero(String curusername) throws ClassNotFoundException, SQLException {…}
public static void setmaterialread(String curusername, String matnm) throws ClassNotFoundException, SQLException {…}
public static String getfilename(String materialnm) throws ClassNotFoundException, SQLException {…}
public static String StGetFile(String materialnm) throws ClassNotFoundException, SQLException, IOException {…}
public static String checkwhologgedin(String usernm) throws ClassNotFoundException, SQLException {…}
public static int same_material_toadd(String profusername, String courseselected, String materialtoadd) throws ClassNotFoundException, SQLException {…}
public static int getteachcoursecnt() {…}
public static int getteachquestcnt() {…}
public static String[] getallcourses() throws SQLException, ClassNotFoundExceptionμ{…}
public static String[] get_all_material(String courseselected) throws ClassNotFoundException, SQLException {…}
public static String[] get_material(String profusername, String courseselected) throws ClassNotFoundException, SQLException {…}
public static String[] getteachcourse(String profusername) throws ClassNotFoundException, SQLException {…}

public static boolean alreadyRead(String usernm, String exhibitnm) throws ClassNotFoundException, SQLException {…}
public static void deleteUser(String value) throws ClassNotFoundException, SQLException {…}
public static String[] chooseQuestions(String[] qnames, int qcnt) throws ClassNotFoundException, SQLException, IOException {…}

9.2.3. GUI

Περιλαμβάνει τις κλάσεις Files.java και Sql_Window.java.

9.2.3.1. Files.java

Περιλαμβάνει όλες τις συναρτήσεις που αφορούν σε αναζήτηση αρχείου από τον υπολογιστή, αποθήκευση αρχείου στον υπολογιστή και ανάκτηση δεδομένων από αρχείο.

public Files() throws ClassNotFoundException, ClassNotFoundException, ClassNotFoundException, SQLException {…}
public static void repaintmatframe() {…}
public static void deleteFile(String filname) {…}
public static File actionAdd() {…}
public static void saveOutputStream(String name, InputStream body) {…}
public File actionFindfile() throws FileNotFoundException, IOException {…}

9.2.3.2. Sql_Window.java

Περιλαμβάνει όλες τις συναρτήσεις που αφορούν στο user interface, δηλαδή το γραφικό περιβάλλον που βλέπει ο κάθε χρήστης όταν χρησιμοποιεί το πρόγραμμα.

public void initialize() throws ClassNotFoundException, SQLException {…}
public void repaintframe() throws SQLException, ClassNotFoundException {…}
public void registeruserframe() {…}
public void registerteachuserframe() {…}
public void addtestframe() throws ClassNotFoundException, SQLException {…}
public void addquestframe() throws ClassNotFoundException, SQLException {…}
public void addmatframe() throws ClassNotFoundException, SQLException{…}
public void addGivequestsnumFrame() throws ClassNotFoundException, SQLException {…}
public void addevalframe() throws ClassNotFoundException, SQLException {…}
public void addsugmatframe() {…}
public void addcourseframe() {…}
public static void CheckQuestnumgiven() throws ClassNotFoundException, SQLException {…}
public void stgetcourseframe() throws SQLException, ClassNotFoundException {…}
public void StaddMaterial() throws ClassNotFoundException, SQLException {…}
public void loginuserframe() {…}
public void addlogRegmenutoframe() {…}
public void addLogoutDelmenutoframe() {…}
public void addmenutoframe() {…}
public void clearregisterfields() {…}
public void clearregisterteachfields() {…}
public void clearloginfields() {…}
public void clearcoursefield() {…}
public static void clearquestnumfield() {…}
public static void clearmatfield() {…}

public void RegisterOkTeachButton() throws IOException, ClassNotFoundException, SQLException {…}
public void RegisterOkButton() throws ClassNotFoundException, SQLException, IOException {…}
public void addbuttonsteach() {…}
public void removecontentstudent() {…}
public void addbuttonsstud() {…}
public void CourseAddButton() throws ClassNotFoundException, SQLException {…}
public static void QuestAddButton() throws ClassNotFoundException, SQLException {…}
public static void MatAddButton() throws ClassNotFoundException {…}
public void LoginOkButton() throws IOException, ClassNotFoundException, SQLException {…}
public static void closeContainerWindow(JFrame frame) {…}
public void actionPerformed(ActionEvent e) {…}
public static void showMessage(String message) {…}

Κεφάλαιο 10
Συμπεράσματα

Το προσαρμοστικό σύστημα διδασκαλίας που υλοποιήθηκε σε πλατφόρμα java, είναι ένα εκπαιδευτικό σύστημα το οποίο χρησιμοποιεί βασικές αρχές σχεδίασης και μπορεί να αξιοποιηθεί από οποιονδήποτε καθηγητή που διδάσκει οποιοδήποτε μάθημα με εντυπωσιακά αποτελέσματα. Το γεγονός που το κάνει τόσο ιδιαίτερο είναι η δυναμικότητά του αναφορικά με τη λειτουργία του, διότι δεν αναφέρεται σε ένα μόνο συγκεκριμένο μάθημα, αλλά δίνει τη δυνατότητα στους καθηγητές να προσθέσουν όση ύλη επιθυμούν και να “ανεβάσουν” στο σύστημα οποιονδήποτε τύπο αρχείου για να μπορέσουν στη συνέχεια να διαβαστούν από τους μαθητές. Παρατηρούμε ότι σχεδιάστηκε για να επιτυγχάνει βασικούς εκπαιδευτικούς σκοπούς.

Η δομή του και ο τρόπος που έχει δημιουργηθεί, του επιτρέπει να έχει τη δυνατότητα να αποτελέσει ένα αναπόσπαστο λογισμικό διδασκαλίας στη δευτεροβάθμια, τριτοβάθμια, αλλά και πανεπιστημιακή εκπαίδευση.

Έχει ως σκοπό όχι την αντικατάσταση του καθηγητή, αλλά τη συμπλήρωση της μαθησιακής και διδακτικής διαδικασίας, χωρίς, όμως, να αποκλείεται η αυτόνομη λειτουργία της.

Από την πλευρά των καθηγητών, έχουν τη δυνατότητα προσθήκης οσωνδήποτε μαθημάτων διδάσκουν, προσθήκη συγκεκριμένης ύλης συνοδευόμενης από αρχεία, αλλά κα-

την καταχώρηση στο σύστημα όσων ερωτήσεων επιθυμούν για την κάθε ύλη που έχουν ήδη εισάγει και όλες αυτές οι λειτουργίες με απίστευτη δυναμικότητα, καθότι δεν υπάρχει κανένας περιορισμός στα μαθήματα, στην ύλη ή τις ερωτήσεις που θα προσθέσουν. Επιπρόσθετα, υπάρχει άμεση ενημέρωσή τους όταν κάποιος από τους εγγεγραμμένους μαθητές πραγματοποιήσουν κάποιο τεστ που περιλαμβάνει ερωτήσεις μαθημάτων τους και είναι σε θέση να παρακολουθούν την εξέλιξη του κάθε μαθητή μέσω της διαδικασίας της αξιολόγησής τους, η οποία γίνεται αυτόματα αμέσως μόλις τελειώσουν και υποβάλλουν κάποιο τεστ.

Από την πλευρά των μαθητών, δεν υπάρχει κανένας περιορισμός στην ύλη που θα διαβάσουν, αρκεί να είναι εγγεγραμμένοι στο συγκεκριμένο μάθημα όπου αυτή αναφέρεται. Έχουν τη δυνατότητα να διαβάζουν ύλη, να ανεβάζουν το γνωστικό τους επίπεδο μέσω της διαδικασίας των ερωτηματολογίων-τεστ και να παρακολουθούν τα λάθη που έχουν κάνει, αλλά και να τους προτείνεται από το σύστημα ύλη για διάβασμα.

Δεν απαιτείται κάποιο εξειδικευμένο υπολογιστικό σύστημα για να λειτουργήσει το προσαρμοστικό σύστημα διδασκαλίας που δημιουργήθηκε, το μόνο που χρειάζεται είναι να εγκατασταθούν τα προγράμματα την πρώτη φορά που θα χρησιμοποιηθεί.

Το σύστημα αυτό είναι user-friendly (φιλικό προς το χρήστη) και, όπως αναφέραμε σε προηγούμενο κεφάλαιο, μπορεί να χρησιμοποιηθεί άνετα και ευχάριστα τόσο από αρχάριους, όσο και από έμπειρους σε σχέση με τους υπολογιστές χρήστες και από την πλευρά των μαθητών, αλλά και των καθηγητών, καθώς όλες οι λειτουργίες πραγματοποιούνται μέσα από παράθυρα, διαλόγους, φόρμες και κουμπιά.

Βιβλιογραφία

[1] Παναγιωτακόπουλος, Χ., Πιερρακέας, Χ., Πιντέλας, Π., Το εκπαιδευτικό λογισμικό και η αξιολόγησή του, Εκδόσεις Μεταίχμιο, 2003.

[2] Ράπτης, Α. & Ράπτη, Α., Μάθηση και Διδασκαλία στην Εποχή της Πληροφορίας, Αθήνα, 2001

[3] Shneiderman, B. (ed), Designing the User Interface. Addison Wesley, NY, 1993

[4] Waterworth J.A., Multimedia Interaction with Computers, ELLIS HORWOOD, 1992.

[5] J.M. Carroll, Minimalist design for active users, Proceedings of IFIP Conference, Interact '84, North-Holland, 1984.

[6] J. Lee & K.-Y. Lai, What's in a design rationale, Human-Computer Interaction, 1991.

[7] J.M.Carroll & M.B.Rosson, Deleberated evolution:stalking the view matcher in design space, Human-computer Interaction, 1991.

[8] A. MacLean, R.M.Young, V.M.E. Belloti & T.P.Moran, Questions, options and criteria: elements of design space analysis, Human-Computer Interaction, 1991.

[9] H.Rittel & W. Kunz, Issues as elements of information systems, Institut fur Grundlagen der Planung I.A., University of Stuttgart, 1970.

[10] D.A.Norman, The psychology of everyday things, Basic Boks, New York, 1988.

[11] R.C. Houghton, Online help systems : a conspectus, Communications of the ACM, 1984.

[12] A.D. Baddeley, Human memory : Theory and Practice, Lawrence Erlbaum Associates, Hove, 1990.
[13] C. Bram & G. Cockton, Design Pronciples for Interactive Software, Chapman and Hall, London, 1996.
[14] D. Hobbs & D. Moore, Human – Computer Interaction, Pitman, 1998.
[15] Csikszentimihalyi, Flow Q The psychology of optimal experience, Harper and Row, N.Y., 1990.
[16] A. Dix, Accelerators and toolbars : learning from the menu, Adjustment Proceedings of HCI, 1995.
[17] A. Fox, B. Johanson, P. Hanrahan & T. Winograd, Integrating information appliances into an interactive workspace, IEEE Computer Graphics and applications, 2000.
[18] L. MacCaulay, C. Fowler, M. Kirby & A. Hutt, USTM : a new approach to requirements specification, Interacting with computers, 1990.
[19] A. Marcus, Graphic design for electronic documents and user interfaces, ACM Press Tutorial Series, New York, 1992.
[20] D. J. Mayhew, Principles and guidelines in software and user interface design, Prentice Hall, Englewood Cliffs, NJ, 1992.
[21] Alan Dix, Janet Finlay, Gregory D. Abowd, Russell Beale, Human-Computer communication, Prentice Hall, 1993.
[22] Dalibor D. Dvorski, INSTALLING, CONFIGURING, AND DEVELOPING WITH XAMPP, March 2007.
[23] Tim Boudreau, Jaroslav Tulach, and Geertjan Wielenga, Rich Client Programming: Plugging into the NetBeans Platform, Prentice Hall, May 2007.
[24] Heiko Boeck, The Definitive Guide to NetBeans Platform 6.5, Apress, June 2009.
[25] Δημητρακοπούλου, Α., Σχεδιάζοντας Εκπαιδευτικά Λογισμικά, 1ο μέρος. Σύγχρονη Εκπαίδευση, 1998.
[26] Cairns, J., Lawton, D. & Gardner, R., Values, Culture and Education – World Yearbook of Education 2001. Kogan page, London, 2001.
[27] Πιντέλας, Π., Σχεδίαση εκπαιδευτικού λογισμικού τεχνικά- τεχνολογικά θέματα, Εκπαιδευτικό Υλικό, 2003.