ΑΝΩΤΑΤΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΤΜΗΜΑ ΔΙΟΙΚΗΣΗ ΕΠΙΧΕΙΡΗΣΕΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανάλυση, Σχεδιασμός και Υλοποίηση εφαρμογής διαχείρισης φοιτητών εργαστηριακών ομάδων ακαδημαϊκού ιδρύματος e-labs.

Analysis, Design and Implementation of a management application laboratory student groups academic institution e-labs

Σπουδαστές:Μενεξιάδης Γεώργιος

Ξυδιάς Δημήτριος

Φωτόπουλος Παντελής

Εισηγητής: Χαλκιόπουλος Κωνσταντίνος

ΠΑΤΡΑ 2014

Περιεχόμενα

Περίληψη

Με την εξέλιξη της τεχνολογίας και στα πλαίσια μείωσης κόστους και αύξησης της αποτελεσματικότητας, κρίθηκε αναγκαίο να αυτοματοποιηθούν όσο είναι δυνατόν οι διαδικασίες εγγραφής των φοιτητών στα μαθήματα ενός ΤΕΙ.

Η οργάνωση και συμμετοχή των φοιτητών στις εργαστηριακές ομάδες είναι μια διαδικασία με ιδιαίτερες απαιτήσεις σε χρόνο αλλά και συγχρονισμό. Επίσης κατά την διάρκεια του ακαδημαϊκού εξαμήνου θα πρέπει να παρακολουθείται η παρουσία των φοιτητών.

Στόχος της πτυχιακής είναι η ανάλυση, σχεδιασμός και υλοποίηση μιας διαδικτυακής εφαρμογής για την διαχείριση των φοιτητών των εργαστηριακών ομάδων. Στην συγκεκριμένη εφαρμογή θα πρέπει να υλοποιηθούν οι διαδικασίες εγγραφής και κατανομής των φοιτητών στις εργαστηριακές ομάδες καθώς και η παρακολούθηση των παρουσιών των φοιτητών(ηλεκτρονικό παρουσιολόγιο). Στα πλαίσια της πτυχιακής θα πρέπει να πραγματοποιηθεί ο καθορισμός των απαιτήσεων της συγκεκριμένης εφαρμογής. Ο σπουδαστής θα πρέπει στην συνέχεια να πραγματοποιήσει, με την χρήση των σύγχρονων τεχνικών, ανάλυση, σχεδιασμό και υλοποίηση του συστήματος. Η επιλογή της πλατφόρμας υλοποίησης θα γίνει από τον σπουδαστή.

Abstract

With the evolution of technology and the context of reducing costs and increasing efficiency, it was necessary to automate as much as possible the procedures of registration of students in the courses of colleges.

The organization and participation of students in laboratory groups is a process with specific requirements and time synchronization. Also during the semester should be monitored for the presence of students.

The aim of the thesis is the analysis, design and implementation of a web-based application for managing student laboratory groups. In this particular application will be implemented registration procedures and allocation of students to laboratory groups and monitor the attendance of students (electronic attendance). As part of the thesis will be carried out to define the requirements of a particular application. The student should then be made with the use of modern techniques, analysis, design and implementation of the system. The choice of platform implementation will be done by the student.

Εισαγωγή

Σκοπός της πτυχιακής εργασίας είναι η μελέτη και ανάπτυξη ενός συστήματος συναλλαγής των σπουδαστών με τη Γραμματεία μέσω του διαδικτύου. Το σύστημα που δημιουργήθηκε επιτρέπει στο σπουδαστή να εισέρχεται στον ιστοχώρο της Γραμματείας και με χρήση προσωπικού λογαριασμού να δηλώνει τα εργαστηριακά μαθημάτα, απουσίες κ.ά.

Απώτερος σκοπός δημιουργίας της εφαρμογής, είναι να αποτελέσει την απαρχή για μια συντονισμένη βελτίωση των ηλεκτρονικών υπηρεσιών που παρέχονται από τη γραμματεία του Τμήματος Εφαρμογή Πληροφορικής στην Διοίκηση και Οικονομίας με στόχο την έγκυρη και άμεση εξυπηρέτηση των σπουδαστών του. Ωστόσο η εφαρμογή μπορεί να αποδειχτεί χρήσιμη και πολύ αποτελεσματική στην αποσυμφόρηση του όγκου των καθημερινών λειτουργιών της γραμματείας.

Η νέα αυτή εφαρμογή εντάσσεται σε ένα γενικότερο πλαίσιο δημιουργίας διαδικτυακών εφαρμογών με στόχο τη βελτίωση της ποιότητας των ηλεκτρονικών υπηρεσιών που παρέχονται στους σπουδαστές. Η εργασία πραγματοποιήθηκε με χρήση της γλώσσας προγραμματισμού για διαδικτυακές εφαρμογές PHP. Η PHP είναι μια γλώσσα script της οποίας ο κώδικας ενσωματώνεται σε μια HTML σελίδα και εκτελείται κάθε φορά που επισκεπτόμαστε την συγκεκριμένη σελίδα. Η βάση δεδομένων δημιουργήθηκε με την MySQL. Οι δύο αυτές τεχνολογίες συνεργάστηκαν με τον διαδικτυακό διακομιστή (web server) Apache Server.

Στο πρώτο κεφάλαιο αναφέρεται τι είναι διαδικτυακές εφαρμογές, ποιές τεχνολογίες χρησιμοποιούν και πως κατηγοριοποιούνται. Μελετώνται, διεξοδικά, όλα τα μοντέλα ανάπτυξης λογισμικού και όλα τα είδη προγραμματισμού.

Στο δεύτερο κεφάλαιο γίνεται η ανάλυση και ο σχεδιασμός της εφαρμογής. Συγκεκριμένα αναφέρονται οι λειτουργικές και μη λειτουργικές απαιτήσεις. Σχεδιάζονται όλα τα σενάρια χρήσης και με την χρήσης της δηλωτικής γλώσσας UML απεικονίζεται το use case. Στην συνέχει του κεφαλαίου γίνεται μια αναφορά στις βάσεις δεδομένων και στα Συστήματα Διαχείρισης αυτών.

Στο τρίτο κεφάλαιο αναφέρεται το πώς υλοποιήθηκε η εφαρμογή. Αναλυτικά αναφέρονται τα εργαλεία που χρησιμοποιήθηκαν για την κατασκευή της ενώ γίνεται ιδιαίτερη μνεία για την MySQL.

Στο τέταρτο κεφάλαιο παρουσιάζεται η εφαρμογή. Συγκεκριμένα με την χρήση απεικονίσεων από την εφαρμογή δείχνεται το εγχειρίδιο για τον φοιτητή, το εγχειρίδιο για τον καθηγητή και το εγχειρίδιο για την γραμματεία.

Στο πέμπτο και τελευταίο κεφάλαιο παρουσιάζονται τα συμπεράσματα που προκύπτουν. Ειδικά αναφέρονται οι μελλοντικές δυνατότητες που μπορούν να υπάρξουν καθώς και τα προσδοκώμενα οφέλη από την χρήση της εφαρμογής.

Θεωρητικό υπόβαθρο

1 Διαδικτυακές εφαρμογές

1.1 Διαδικτυακές εφαρμογές

Για να γίνει κατανοητό τι είναι μια web εφαρμογή θα πρέπει πρώτα να αναφέρουμε το τι είναι μια εφαρμογή. Εφαρμογή ονομάζεται το λογισμικό το οποίο εγκαθίσταται σε έναν ηλεκτρονικό υπολογιστή και έχει σχεδιαστεί ώστε να πραγματοποιεί συγκεκριμένες διεργασίες, να επιτυγχάνει συγκεκριμένους στόχους και να εξάγει στον χρήστη την επιθυμητή πληροφορία ή αποτέλεσμα. Η Web εφαρμογή είναι το ίδιο με την μόνη διαφορά ότι δεν εγκαθίσταται σε έναν ή περισσότερους ηλεκτρονικούς υπολογιστές αλλά είναι προσβάσιμο με έναν φυλλομετρητή (web browser) μέσω του internet ή κάποιου τοπικού δικτύου.

Αυτό που διαχωρίζει μια web εφαρμογή από μια ιστοσελίδα είναι ότι ο βασικός σκοπός μιας ιστοσελίδας είναι να πληροφορήσει τον χρήστη προβάλλοντας κείμενο, εικόνες ή video, ενώ ο σκοπός μιας web εφαρμογής είναι να προσφέρει στον χρήστη ένα περιβάλλον εργασίας στο οποίο μπορεί να πάρει αλλά και να δώσει πληροφορίες, να εκτελέσει διεργασίες, να επεξεργαστεί δεδομένα και να πετύχει κάποιο στόχο. Ένα πολύ καλό και γνωστό παράδειγμα μιας web εφαρμογής είναι το Hotmail στο οποίο ο χρήστης δεν είναι απλά ένας επισκέπτης αλλά αλληλεπιδρά με το σύστημα.

Η web εφαρμογή δεν έρχεται να υποβαθμίσει την έννοια της ιστοσελίδας η οποία είναι πλέον ανεκτίμητη και αναντικατάστατη, αλλά να προσφέρει ακόμα περισσότερες λύσεις σε εξειδικευμένες ανάγκες. Η web εφαρμογή είναι το απαραίτητο εργαλείο για τις επιχειρήσεις που θέλουν να προσφέρουν ακόμα πιο προηγμένες υπηρεσίες στους πελάτες τους ή στους συνεργάτες τους. Η web εφαρμογή μπορεί να είναι προσβάσιμη στο ευρύτερο κοινό μέσω του internet ή μόνο στο προσωπικό της επιχείρησης μέσω ενός ιδιωτικού τοπικού δικτύου.

1.2 Από το Web 1.0 στο Web 2.0

Το Web 2.0, ως φράση, χρησιμοποιήθηκε από την O' Reilly Media το 2004 με σκοπό να περιγραφεί η δεύτερη γενιά υπηρεσιών που χρησιμοποιούν τον παγκόσμιο ιστό ως πλατφόρμα, και δίνουν έμφαση στην ηλεκτρονική συνεργασία και ανταλλαγή πληροφοριών μεταξύ των χρηστών, με νέους τρόπους, αξιοποιώντας εργαλεία των σύγχρονων τεχνολογιών του διαδικτύου (http://oreilly.com/).

Ένας πρόσφατος ορισμός του Web 2.0, σύμφωνα με τον Tim O' Reilly (ιδρυτή της O' Reilly Media) στην προσπάθεια να περιγραφεί ο όρος, είναι ο ακόλουθος: «Το Web 2.0 είναι η επιχειρηματική επανάσταση στη βιομηχανία των υπολογιστών που προέρχεται από την μετακίνηση στο Διαδίκτυο ως πλατφόρμα, και μια προσπάθεια να κατανοήσουμε τους κανόνες της επιτυχίας αυτής της νέας πλατφόρμας. Ο βασικότερος ανάμεσα στους κανόνες είναι ο εξής: Χτίσιμο εφαρμογών που εκμεταλλεύονται τις επιδράσεις των δικτύων, ώστε να γίνονται καλύτερες, όσο περισσότερο τις χρησιμοποιούν οι άνθρωποι». Αυτό είναι που ο Tim O'Reilly αποκαλεί «εκμετάλλευση της συλλογικής νοημοσύνης». Σύμφωνα Schonfeld E. το κύριο χαρακτηριστικό του Web 2.0 είναι η παγκόσμια διάδοση των νεωτεριστικών ιστοθέσεων. Μόλις αναπτυχθεί μία επιτυχημένη ιδέα σαν ιστόθεση σε μία χώρα, παρόμοιες ιστοθέσεις εμφανίζονται σε όλο τον κόσμο.

Οι εφαρμογές, τα εργαλεία και οι υπηρεσίες που προσφέρονται σήμερα στο διαδίκτυο εντείνουν τη συμμετοχικότητα και την ανοιχτή πρόσβαση των χρηστών σε κοινωνικά μέσα (Social Media), που αποτελούν περιβάλλοντα συνεργασίας και ανταλλαγής και διαμοιρασμού πληροφορίας. Η παρακάτω εικόνα αποτυπώνει ένα οδικό χάρτη του Web 2.0 συμπεριλαμβάνοντας όρους και τεχνολογίες που περιλαμβάνονται σε αυτό (http://oreilly.com/).

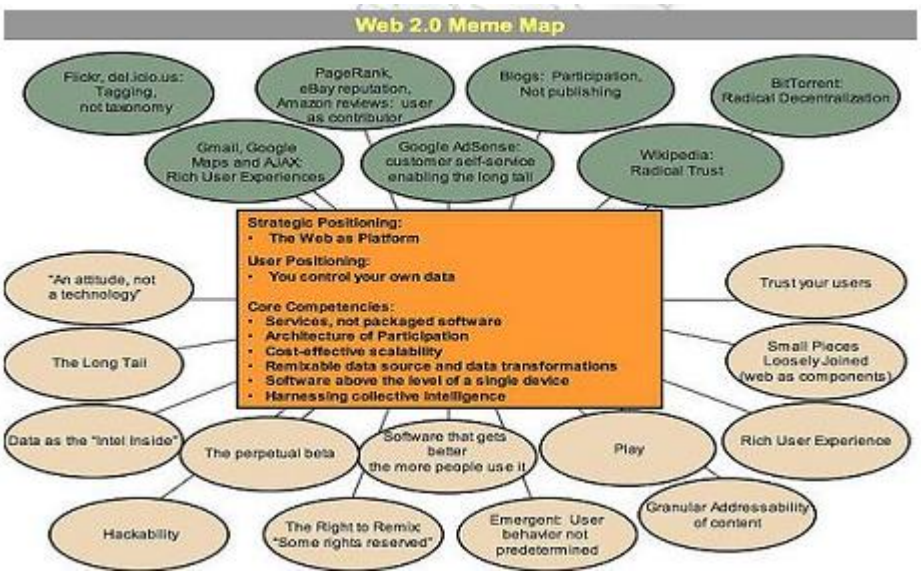
Εικόνα 1 Οδικός Χάρτης του Web 2.0 από τη Wikipedia.com

Στο περιβάλλον του Web 2.0 ο χρήστης αποτελεί στο επίκεντρο, ως κεντρικός παράγοντας που επηρεάζει τη μορφή και χαρακτηριστικά των εργαλείων και υπηρεσιών που προσφέρονται σε αυτόν σήμερα. Τις περισσότερες φορές οι εφαρμογές αξιολογούνται και συνδιαμορφώνονται από τους χρήστες, υιοθετώντας όσο το δυνατόν περισσότερα επιθυμητά, κατά τους χρήστες, χαρακτηριστικά.

Τα κύρια χαρακτηριστικά του Web 2.0, δίνοντας έμφαση σε αυτά που έχουν σχέση με την υγεία και το Web 2.0, μπορούν να συνοψιστούν στα (http://oreilly.com/).:

- Συμμετοχή (Participation)
- Διαμοιρασμός (Sharing)
- Χρηστικότητα (Usability)
- Έλεγχος από τους χρήστες (User control)
- Επικοινωνία και διευκόλυνση – υποστήριξη της δημιουργίας ιδεατών κοινοτήτων (Communication - communities)
- Χαμηλό έως ανύπαρκτο κόστος (Low cost)
- Ανοιχτή πρόσβαση (Free access)
- Αποκέντρωση (Decentralization)

Ένας χάρτης εφαρμογών που παρουσιάζεται στην ιστοσελίδα του O'Reilly με την ονομασία "meme map" του παγκόσμιου ιστού 2.0 και ο οποίος δημιουργήθηκε κατά τη διάρκεια μίας περιόδου brainstorming στα πλαίσια του FOO Camp, ενός συνεδρίου στην εταιρία O'Reilly Media, παρουσιάζεται στην εικόνα. Η εικόνα προσπαθεί να αναπαραστήσει πολλές ιδέες που αναπτύσσονται στα πλαίσια του Web 2.0 και προέρχονται από τον πυρήνα της δεύτερης αυτής γενεάς εφαρμογών. (http://oreilly.com/).



Εικόνα 2 Χάρτης εφαρμογών του Web 2.0 Πηγή: oreilly.com

Ορισμένα από τα στατιστικά στοιχεία που αφορούν χρονικά τον Ιανουάριο του 2007 και απεικονίζουν τη δημοτικότητα εφαρμογών Web 2.0 παρουσιάζονται ακολούθως.

- Υπάρχουν περισσότερα από 53 εκατομμύρια blogs
- Δημιουργούνται 150000 blogs κάθε μέρα ή κατά μέσο όρο δύο blogs ανά δευτερόλεπτο.
- Η σφαίρα των blogs διπλασιάζεται σε μέγεθος κάθε πέντε έως επτά μήνες.
- Το 40% αυτών που αρχίζουν ένα blog συνεχίζουν να δημοσιεύουν σε αυτό τρεις μήνες αργότερα.
- Οι χρήστες αναφορτώνουν 70000 νέα βίντεο στην ιστόθεση Youtube κάθε μέρα.
- Οι χρήστες του YouTube παρακολουθούν 100 εκατομμύρια βίντεο την ημέρα.

Τα κοινωνικά μέσα, αποτελούν μία αναπτυσσόμενη πτυχή του διαδικτύου και η συνεχής εξέλιξη τους, οδηγεί σε νέους τρόπους επικοινωνίας, συνεργασίας και διαμοιρασμού πληροφοριών και γνώσεων. (http://ebennett.org/hsnl/).

1.3 Κατηγοριοποίηση ως προς τις εφαρμογές και παραδείγματα

Στην προσπάθεια να κατηγοριοποιηθούν τα κοινωνικά μέσα, σήμερα στο Web 2.0 υπάρχουν εφαρμογές διαφορετικών χαρακτηριστικών και δυνατοτήτων, πληροφορίες για τα χαρακτηριστικά των οποίων μας παρέχονται και από τη βιβλιογραφία (http://oreilly.com/).

Ακολούθως, παρουσιάζονται παραδείγματα, βάση μίας σύντομης προσέγγισης διαφορετικών κατηγοριών εφαρμογών.

- Ιστολόγια (Blogs)

Αποτελούν, κατά κύριο λόγο, μία μορφή διαδικτυακών ημερολογίων, με θέματα ταξινομημένα ημερολογιακά σε συνάρτηση με την ημερομηνία ανάρτησης των θεμάτων. Η λέξη blog προέρχεται από τις λέξεις web και log. Γνωστές πλατφόρμες αποτελούν οι Blogger.com, Wordpress.com και Blog.com. Σε αυτή την κατηγορία, μπορούν να περιληφθούν νέοι όροι, όπως το microblogging, με παραδείγματα το δημοφιλές Twitter ή το Jaiku.

- Κοινωνικά δίκτυα (Social Networks)

Τα κοινωνικά δίκτυα, αποτελούν εφαρμογές, που επιτρέπουν σε χρήστες να δημιουργούν προσωπικές ιστοσελίδες, και μετά να συνδέονται με άλλους χρήστες, σύμφωνα με τα ενδιαφέροντά τους, διευκολύνοντας την επικοινωνία και το διαμοιρασμό πληροφορίας. Σημαντικά παραδείγματα, αποτελούν το Facebook, MySpace, Hi5 κ.α. Τα forums, τα οποία προϋπήρχαν του Web 2.0 και έχουν υποστηριχτές πολλούς χρήστες, ως δημοφιλείς εφαρμογές, συχνά συναντώνται στο σύγχρονο διαδικτυακό περιβάλλον. Με τις λειτουργίες που μπορούν να προσαρμοστούν σήμερα, σε αυτή τη κατηγορία εφαρμογών, μπορούν, επίσης, να θεωρηθούν ως κοινότητες χρηστών, στα πλαίσια των οποίων αναπτύσσονται χαρακτηριστικά κοινωνικών δικτύων. Μία βασική διαφορά, η οποία μπορεί όμως να αναδειχθεί, συγκρίνοντας τις δύο εφαρμογές, και αποτελεί χαρακτηριστικό τους, είναι ότι σε ένα κοινωνικό δίκτυο, συνήθως, το επίκεντρο αποτελεί ο χρήστης, ενώ σε ένα

forum το επίκεντρο αποτελούν τα θέματα συζητήσεων που ξεκινούν οι χρήστες, που δραστηριοποιούνται με τη χρήση του.

- Κοινότητες περιεχομένου (Content Communities)

Πρόκειται για εφαρμογές, που επιτρέπουν σε χρήστες να οργανώνουν και να διαμοιράζουν υλικό, όπως βίντεο ή φωτογραφίες ή άλλα ηλεκτρονικά αρχεία, φιλοξενώντας τα σε υπηρεσίες. Στην κατηγορία αυτή, περιλαμβάνονται εφαρμογές, όπως τα RSS feeds, tags, bookmarks, podcasts. Οι εφαρμογές αυτές μπορούν να βελτιώσουν τη χρηστικότητα κατά την πλοήγηση ενός χρήστη στο διαδίκτυο. Επίσης, το σημαντικό χαρακτηριστικό τους, είναι ότι προσφέρουν δυνατότητες διαμοιρασμού του περιεχομένου τους, μεταξύ των συμμετεχόντων.

- Ηλεκτρονικές συνεργατικές εγκυκλοπαίδειες (Wikis)

Πρόκειται για διαδικτυακές εφαρμογές, που επιτρέπουν σε πολλούς χρήστες να προσθέτουν, να τροποποιούν και να επεξεργάζονται πληροφορίες συνεργατικά. Συγκεντρώνουν γνώσεις και εμπειρίες σε θέματα κοινού ενδιαφέροντος, με αποτέλεσμα να αποτελούν πηγή πληροφόρησης, για τους χρήστες του διαδικτύου. Η πιο δημοφιλής ηλεκτρονική συνεργατική εγκυκλοπαίδεια είναι η Wikipedia.org.

- Εικονικοί κόσμοι (Virtual worlds)

Αποτελούν τρισδιάστατα εικονικά περιβάλλοντα, στα οποία οι χρήστες δραστηριοποιούνται μέσω τρισδιάστατων ειδώλων (avatars) και μπορούν να πλοηγούνται στον εικονικό κόσμο, αλληλεπιδρώντας με το περιεχόμενό του και άλλους χρήστες. Μπορούν να κάνουν χρήση εφαρμογών που επιτρέπουν το διαμοιρασμό πληροφοριών, το “χτίσιμο” εγκαταστάσεων, δημιουργώντας εικονικά αντικείμενα κ.α. Παραδείγματα από πλατφόρμες εικονικών κόσμων, που επιτρέπουν στους χρήστες τους να αλληλεπιδρούν με άλλους στα πλαίσια χρήσης τους, αποτελούν το Second Life, το Active Worlds και το Open Croquet.

- Διαδικτυακοί χάρτες (Web maps)

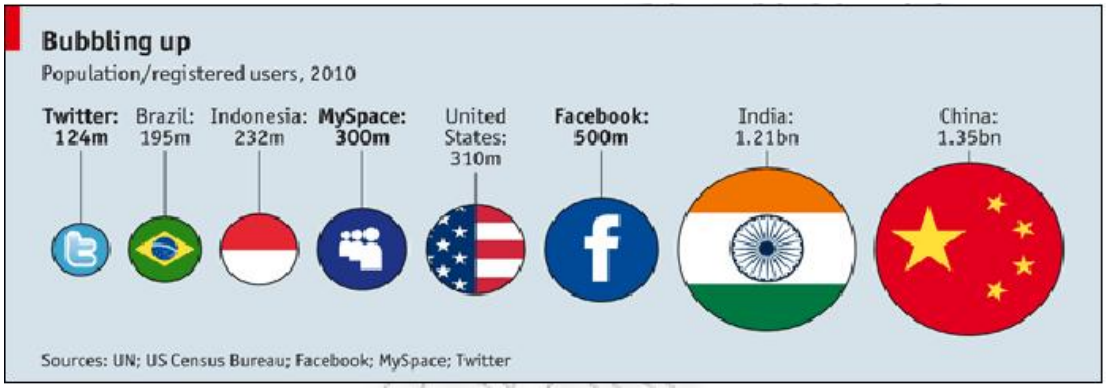
Πρόκειται για χάρτες, που είναι προσπελάσιμοι σε χρήστες μέσω του διαδικτύου, μέσω των οποίων ενημερώνονται σχετικά με γεωγραφικά δεδομένα, ανάλογα με τις δυνατότητες και τις προδιαγραφές απεικόνισης, που παρέχει η εκάστοτε υπηρεσία. Αν η υπηρεσία το επιτρέπει, οι χρήστες μπορούν και δημοσιεύουν περιεχόμενο, με

βάση τις γεωγραφικές πληροφορίες, που παρέχει η υπηρεσία διαδικτυακών χαρτών. Μία πολύ γνωστή εφαρμογή που παρέχει γεωγραφικές πληροφορίες, είναι το Google Earth. Επίσης, δημοφιλείς mashup εφαρμογές χαρτών αποτελούν οι Google maps, Yahoo maps και Bing maps, οι οποίες μπορούν να ενσωματωθούν, σε άλλες εφαρμογές του παγκόσμιου ιστού και να προσφέρουν γεωγραφικές πληροφορίες, σχετικές με τις ανάγκες της υπηρεσίας που παρέχεται, βασισμένη στη διαδικτυακή εφαρμογή στην οποία ενσωματώνονται (http://oreilly.com/)..

Τα εργαλεία, οι εφαρμογές και οι υπηρεσίες που έχουν αναπτυχθεί και διαδοθεί, μέσα από το μεγάλο αριθμό χρηστών που τα χρησιμοποιεί, συγκέντρωσαν το ενδιαφέρον πολλών επιστημονικών κοινοτήτων και τομέων, εξαιτίας των σημαντικών πλεονεκτημάτων που προσφέρουν. Παραδείγματα που υπάρχουν σε διάφορους τομείς που, σε πολλά σημεία, αλληλοεξαρτώνται και υπάρχει συχνή ανταλλαγή πρακτικών και αποτελεσμάτων από τη χρήση των σύγχρονων εφαρμογών. Τέτοιοι επιστημονικοί τομείς δείχνουν να είναι η υγεία και η εκπαίδευση. Συγκεκριμένα, η επιστημονική κοινότητα της υγείας, δείχνει συνεχώς να εξετάζει νέες εφαρμογές και πρακτικές, που δημοσιεύονται σε σχέση με ευέλικτες πρακτικές εξ' αποστάσεως μάθησης και εκπαίδευσης. Αυτό, διότι η εκπαίδευση μπορεί να βοηθήσει τόσο επαγγελματίες υγείας στην επιμόρφωσή τους, όσο και ασθενείς στην εξοικείωσή τους με Web 2.0 εφαρμογές. Αυτό μπορεί να επιτευχθεί αξιοποιώντας διαδικτυακά βοηθήματα και τεκμηρίωση που παρέχεται στα πλαίσια χρήσης μιας διαδικτυακής υπηρεσίας. Πληθώρα δημοσιεύσεων, προβάλλει συμπεράσματα, πλεονεκτήματα και προβληματισμούς από τη χρήση των Web 2.0 εφαρμογών, ως εργαλείων, στους χώρους της εκπαίδευσης και της μάθησης ακόμα και μέσα από εικονικά περιβάλλοντα που ενσωματώνουν τεχνικές εικονικής πραγματικότητας. Οι Web 2.0 εφαρμογές, χρησιμοποιούνται ως εργαλεία για την προαγωγή δραστηριοτήτων μάθησης και τη διευκόλυνση εκπαιδευτικών δραστηριοτήτων, διαμορφώνοντας ένα χώρο μεταγενέστερο του eLearning, που αντιπροσωπεύεται από όρους όπως Learning 2.0, που επηρεάζουν διαφόρους τομείς (http://oreilly.com/)..

Οι εφαρμογές του Web 2.0, αποτελούν ένα εργαλείο της καθημερινότητας, για την πλειοψηφία των χρηστών του διαδικτύου σήμερα. Αξίζει να αναφερθεί ότι τη στιγμή, που γράφεται η παρούσα διπλωματική, το κοινωνικό δίκτυο Facebook, έχει ήδη ξεπεράσει τους 500 εκατομμύρια χρήστες σύμφωνα το περιοδικό Economist (http:// Economist.com). Στην ακόλουθη εικόνα, μπορεί κανείς να δει πως θα μπορούσε να

απεικονιστεί συγκριτικά το μέγεθος των μεγαλυτέρων παγκοσμίως χωρών, σε σχέση με τον αριθμό των χρηστών, που απαρτίζουν τα κοινωνικά δίκτυα Facebook, MySpace και Twitter.



Εικόνα 3 Πληθυσμιακή αναπαράσταση κοινωνικών μέσων σε σχέση με κράτη από το Economist.com

Η επόμενη γενιά του παγκόσμιου ιστού, σύμφωνα με τους Tim O'Reilly και John Battele, σε άρθρο τους (O'Reilly, 2011) το 2009 και πέντε χρόνια μετά το πρώτο Web 2.0 συνέδριο, είναι ο Web^2 (Web Squared), δηλαδή ο παγκόσμιος ιστός στο τετράγωνο. Σύμφωνα με το άρθρο, από την πρώτη στιγμή που εισήχθη ο όρος παγκόσμιος ιστός 2.0 (Web 2.0), ο κόσμος διερωτήθηκε ποια είναι η επόμενη γενιά του παγκόσμιο ιστού, όντας κατά κύριο λόγο επικεντρωμένοι κυρίως στον όρο Web 3.0, όπως για παράδειγμα συμβαίνει στις εκδόσεις λογισμικού, λαμβάνοντας το Web 2.0 ως έναν αριθμητικό όρο και όχι ως μία δήλωση. Το ίδιο ερώτημα ισχύει ως προς τη διαδοχή του Web 2.0, για παράδειγμα, από τον σημασιολογικό ιστό (semantic web), τον κοινωνικό ιστό (social web), τον κινητό ιστό (mobile web) ή κάποια μορφή εικονικής πραγματικότητας. Όλα τα παραπάνω, σύμφωνα με τους συγγραφείς, μπορούν να αντιπροσωπευτούν από τον όρο του Web Squared. Στον τομέα των ηλεκτρονικών υπηρεσιών υγείας, όμως, όπως θα δούμε στη συνέχεια, το web 2.0 διαμορφώνει, σήμερα, ένα νέα χώρο συνεργατικής δραστηριοποίησης μεταξύ των όλων εμπλεκομένων σε αυτό.

1.4 Τεχνολογίες και Υπηρεσίες στο Web 2.0

Η ανάπτυξη στους τομείς της Πληροφορικής και των Τηλεπικοινωνιών, καθώς και η σημαντικά αυξανόμενη συμμετοχή των χρηστών του διαδικτύου σε νέες μορφές επικοινωνίας, ενημέρωσης και κοινωνικοποίησης, με τη χρήση Web 2.0 εφαρμογών, δημιουργούν ανάγκες εξέλιξης και αναβάθμισης των υπολογιστικών συστημάτων, των τεχνικών, των μοντέλων και των αρχιτεκτονικών που χρησιμοποιούνται για την

επίτευξη των σύγχρονων επικοινωνιών. Σήμερα, στο διαδίκτυο, οι εφαρμογές που αναπτύσσονται, στηρίζονται σε μεγάλο ποσοστό στο συνδυασμό ανοιχτών προτύπων με τη χρήση των τεχνολογιών Linux (ως λειτουργικό σύστημα), Apache HTTP Server, MySQL (ως λογισμικό βάσεων δεδομένων), PHP ή Perl ή Python (γλώσσες προγραμματισμού ανάπτυξης εφαρμογών). Οι τεχνολογίες αυτές συνδυάζονται σε πολλές περιπτώσεις με σκοπό τη δημιουργία Web 2.0 εφαρμογών. Τα αρχικά των λέξεων μπορούν να σχηματίσουν το αρκτικόλεξο LAMP, στο οποίο αναφέρονται πολλοί προγραμματιστές θέλοντας να περιγράψουν το συνδυασμό αυτών των τεχνολογιών. Διαδεδομένη και δημοφιλής είναι, επίσης, η χρήση της τεχνολογίας AJAX (Asynchronous Java Script and XML) για τη δημιουργία διαδραστικών διαδικτυακών εφαρμογών, καθώς ο συνδυασμός αυτών των γλωσσών προγραμματισμού μπορεί να προσφέρει πολλά πλεονεκτήματα που συνδέονται με τα επιθυμητά χαρακτηριστικά του Web 2.0. (http://oreilly.com/)

Οι σύγχρονες τάσεις στο διαδίκτυο, έχουν σε πολλές περιπτώσεις υιοθετήσει υπηρεσίες και λύσεις που φιλοξενούνται στο «νέφος» (cloud), στην κατεύθυνση της παροχής του Λογισμικού ως Υπηρεσία (Software as a Service - SaaS), της Πλατφόρμας ως Υπηρεσία (Platform as a Software) ακόμα και του Υλικού ή Υποδομής ως Υπηρεσία (Hardware as a Service – HaaS ή IaaS – Infrastructure as a Service). Οι υπηρεσίες που περιγράφονται τόσο στο παρόν κεφάλαιο, αλλά και στην υπόλοιπη εργασία, στην πλειοψηφία τους, υιοθετούν λύσεις που βασίζονται στο νέφος, δηλαδή στην τεχνολογία του “Υπολογιστικού Νέφους” (Cloud Computing) με σκοπό την εκμετάλλευση των πλεονεκτημάτων που προσφέρει η χρήση του. Να ασχοληθούμε λίγο, όμως, με κάθε όρο ξεχωριστά με σκοπό να κατανοήσουμε τη διαμόρφωση των σύγχρονων μοντέλων αποθήκευσης και επικοινωνίας στο διαδίκτυο και τις αλληλεπιδράσεις που προκύπτουν ή μπορούν να προκύψουν σε μία εποχή πέρα από το Web 2.0, στα πλαίσια της συνεχούς αναβάθμισης των εφαρμογών, των υπηρεσιών, της διαλειτουργικότητας των διασυνδεδεμένων συσκευών και της εξέλιξης της Human - Computer Interaction. (http://oreilly.com/)

Τα νέα μοντέλα υπηρεσιών που αναπτύσσονται, βάση των όρων που περιγράφηκαν παραπάνω, οδηγούν σε νέες μορφές αρχιτεκτονικών, με διαφοροποίηση των απαιτήσεων από τους παρόχους, τους σχεδιαστές και τους καταναλωτές – χρήστες των υπηρεσιών. Προκύπτουν, όμως, και νέα θέματα ασφάλειας, καθώς οι υπηρεσίες που προσφέρονται βασισμένες στο νέφος, επιφέρουν προβληματισμούς ως προς την

ασφάλεια και τους κινδύνους από την μετακίνηση μία υπηρεσίας στο νέφος. Από την έρευνα (CSA, 2010), προκύπτουν οι ακόλουθες απειλές: α. η κατάχρηση και η φαύλη χρήση του υπολογιστικού νέφους, β. οι μη ασφαλείς διεπαφές και APIs, γ. κακόβουλοι εσωτερικοί καταναλωτές, δ. θέματα διαμοιραζόμενων τεχνολογιών, ε. απώλεια δεδομένων ή διαρροή, στ. πειρατεία λογαριασμού ή υπηρεσίας, ζ. άγνωστο προφίλ επικινδυνότητας.

Τα ζητήματα αυτά, αποτελούν θέματα που απασχολούν τους χρήστες ή τους συνεργάτες μίας υπηρεσίας βασισμένης στο υπολογιστικό νέφος. Στην υγεία μπορεί να προκύψουν προβληματισμοί, ιδίως όταν πρόκειται, για παράδειγμα, για δεδομένα που αποθηκεύονται σε έναν ηλεκτρονικό φάκελο υγείας που προσφέρεται ως υπηρεσία βασισμένη στο νέφος.

Μετά την μελέτη των παραπάνω κρίνετε αναγκαίο η περιγραφή της έννοιας της πολυνηματικότητας.

1.5 Πολυνηματικότητα (Multithreaded)

Πολυνηματικότητα (multithreading) καλείται η τεχνική δημιουργίας εφαρμογών που βασίζονται στα πολλαπλά «νήματα» (threads). Ένα νήμα είναι μία ακολουθία εντολών η οποία μπορεί να εκτελείται παράλληλα με άλλες παρόμοιες. Είναι δηλαδή ένα είδος παραλληλισμού του κώδικα, ο οποίος παρ’ όλα αυτά μπορεί να τρέχει και σε έναν επεξεργαστή. Η δυσκολία του να γραφούν προγράμματα που να αντιμετωπίζουν καταστάσεις όπου πολλά πράγματα ταυτόχρονα πρέπει να συμβαίνουν, ανάγεται στη δυσκολία εξαγωγής ενός παράλληλου προγράμματος από το αντίστοιχο σειριακό, με τα αντίστοιχα προβλήματα συγχρονισμού, αδιεξόδων, αποτυχιών.

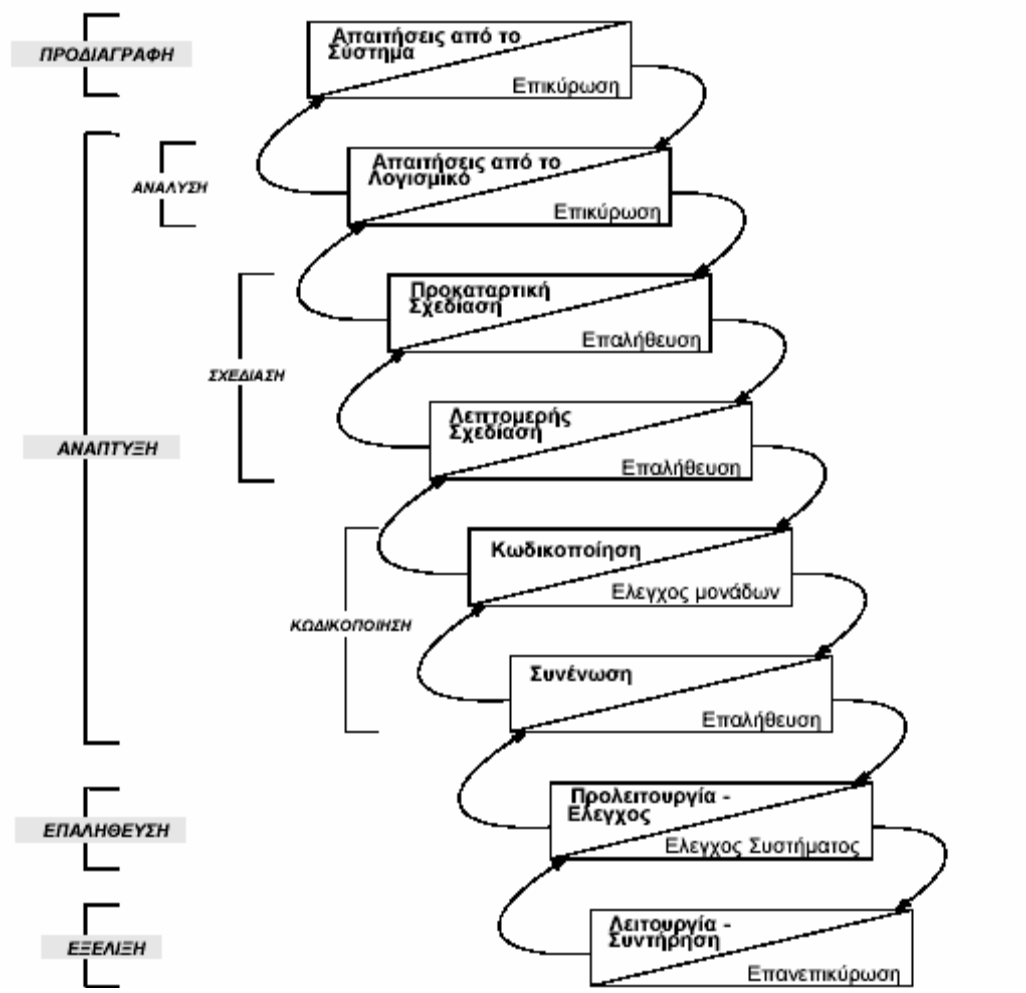
Άλλα χαρακτηριστικά της πολυνηματικότητας είναι η καλύτερη αλληλεπίδραση πραγματικού χρόνου. Σε αυτόνομα περιβάλλοντα Java, η απόκριση προσεγγίζει πολύ τις απαιτήσεις απόκρισης πραγματικού χρόνου. Αν όμως αυτό το περιβάλλον βρίσκεται κάτω από ένα διαφορετικό λειτουργικό σύστημα (π.χ. UNIX, Windows, Macintosh) η απόδοση αυτή φυσικό είναι να παρουσιάζεται μειωμένη.

Πριν προχωρήσουμε στην περιγραφή των εργαλείων που χρησιμοποιήσαμε θα πρέπει να αναφέρουμε τα είδη προγραμματισμού που υπάρχουν.

1.6 Μοντέλα Ανάπτυξης Λογισμικού

1.6.1 Μοντέλο Καταρράκτη και παραλλαγές του

Το μοντέλο του Καταρράκτη (waterfall ή linear sequential model) αναπτύχθηκε από τον Royce, το 1970 [Royce, 1970; Boehm, 1976; Boehm, 1984; Ramamoorthy, 1984] και περιλαμβάνει 8 διακριτές φάσεις.



Εικόνα 4: Φάσεις Μοντέλου Καταρράκτη

Ήταν το πρώτο μοντέλο που δημιουργήθηκε και έγινε ευρέως αποδεκτό, ενώ Ακόμα παραμένει δημοφιλές ιδιαίτερα για μικρά ή μεσαία μεγέθη εφαρμογών αφού συμβάλλει στην επιτυχή κατασκευή αξιόπιστων προϊόντων σε μικρό χρονικό διάστημα.

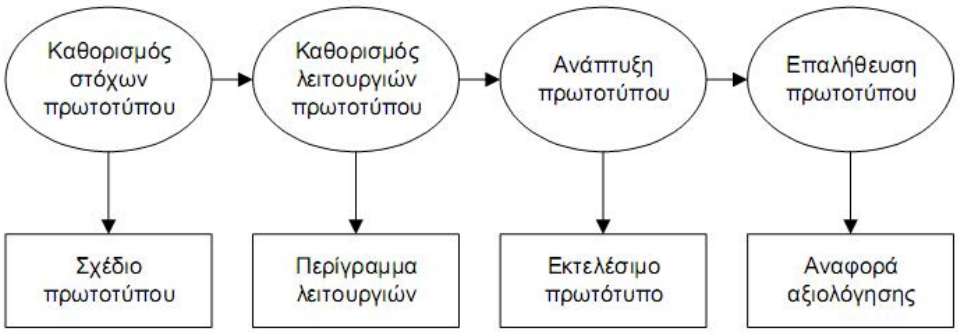
Στο μοντέλο αυτό οι διάφορες φάσεις διαχωρίζονται και ακολουθούνται σειριακά. Η κάθε φάση παράγει ενδιάμεσα προϊόντα τα οποία χρησιμοποιούντα από τις επόμενες φάσεις και κορυφώνεται από μια διαδικασία επικύρωσης ή επαλήθευσης των προϊόντων που παράγονται, με σκοπό να απαλειφθούν τυχόν σφάλματα.

Τα μειονεκτήματα που παρουσιάζει το μοντέλο του καταρράκτη δηλαδή το ότι γνωρίζουμε αν θα είναι ικανοποιημένος ο χρήστης μόνο στο τελικό στάδιο (ουσιαστικά αργά), επίσης το ότι δεν προβλέπει επαναχρησιμοποίηση του λογισμικού που πιθανά υπάρχει και ότι είναι απαιτητικό τόσο σε χρόνο όσο και σε κόστος και αρκετά δύσκαμπτο εργαλείο, προσπαθούν να απαλείψουν τα μοντέλα IEEE (IEEE variant of a life cycle model) και το μοντέλο V (V model) που αποτελούν παραλλαγές του.

Οι παραλλαγές αυτές διαφοροποιούνται περισσότερο στον τρόπο που γίνεται ο έλεγχος, η διόρθωση, η επαλήθευση, η επικύρωση καθώς και ο τρόπος που θα πραγματοποιηθούν αυτές δηλαδή σε ποια φάση θα πρέπει να επιστρέψουμε αν χρειαστεί ενώ παράλληλα οι φάσεις που περιλαμβάνονται χαρακτηρίζονται είτε από την σύμπτυξη είτε από τον επιμέρους διαχωρισμό των αντίστοιχων φάσεων του αντίστοιχου μμοντέλου του καταρράκτη.

1.6.2 Μοντέλο Προτυποποίησης

Ένας τρόπος να δούμε την προτυποποίηση είναι ως μια τεχνική για μείωση του ρίσκου. Το πιο σημαντικό ρίσκο στην ανάπτυξη λογισμικού είναι τα λάθη και περισσότερο οι παραλείψεις που προκύπτουν από μη σαφείς απαιτήσεις των χρηστών για το τελικό σύστημα. Το κόστος της διόρθωσης αυτών των λαθών και παραλείψεων σε επόμενα στάδια μπορεί να είναι πολύ υψηλό. Είναι προφανές ότι η δημιουργία ενός πρωτοτύπου μπορεί να μειώσει τον αριθμό των προβλημάτων των απαιτήσεων και ως εκ τούτου να μειώσει το συνολικό κόστος ανάπτυξης. Μια σχηματική αναπαράσταση της διαδικασίας ανάπτυξης ενός πρωτοτύπου φαίνεται στο παρακάτω σχήμα.



Εικόνα 5: Σχηματική αναπαράσταση της διαδικασίας ανάπτυξης ενός πρωτοτύπου

Αρχικά θα πρέπει να καθοριστούν επακριβώς οι στόχοι του πρωτότυπου. Το πρωτότυπο σύστημα μπορεί να αφορά τη διεπιφάνεια χρήστη ή να περιέχει τις λειτουργίες εκείνες που θεωρούνται περισσότερο κρίσιμες. Είναι προφανές ότι ένα πρωτότυπο δεν μπορεί να καλύπτει όλες τις απαιτήσεις του συστήματος. Για τον λόγο αυτό κάθε φορά θα πρέπει να ορίζονται πλήρως οι απαιτήσεις που αυτό θα καλύπτει, αλλιώς τελικά μπορεί να μην λάβουμε τα πλεονεκτήματα που μας προσφέρει η μέθοδος αυτή. Το επόμενο στάδιο αφορά στο να αποφασιστεί ποιες ενδεχομένως λειτουργίες του τελικού συστήματος δεν θα πρέπει να συμπεριληφθούν γιατί έχει πολύ υψηλό κόστος εάν το πρωτότυπο δημιουργηθεί με όλες τις λειτουργίες του τελικού συστήματος.

Βέβαια θα μμπορούσε να αποφασιστεί να περιλαμβάνονται όλες οι λειτουργίες που έχει αποφασιστεί αλλά σε μειωμένο επίπεδο (πχ. χωρίς διαχείριση λαθών). Η τελευταία φάση, μετά την ανάπτυξη του πρωτοτύπου, είναι η επαλήθευση του πρωτότυπου και είναι ίσως η πιο σημαντική φάση. Θα πρέπει να καταγραφούν συμπεράσματα για το πως νιώθουν οι χρήστες με το σύστημα, αν γίνεται κατανοητό το περιβάλλον και η λειτουργία του και να βρεθούν τυχόν λάθη και προβλήματα.

Τα **πλεονεκτήματα** της χρήσης πρωτότυπου είναι ότι ανακαλύπτονται και διορθώνονται:

• Παρεξηγήσεις μεταξύ των χρηστών και των δημιουργών

• Παραλειπόμενες υπηρεσίες στο σύστημα

• Δυσκολίες στη χρήση

• Ασυνέχειες και κενά στις προδιαγραφές

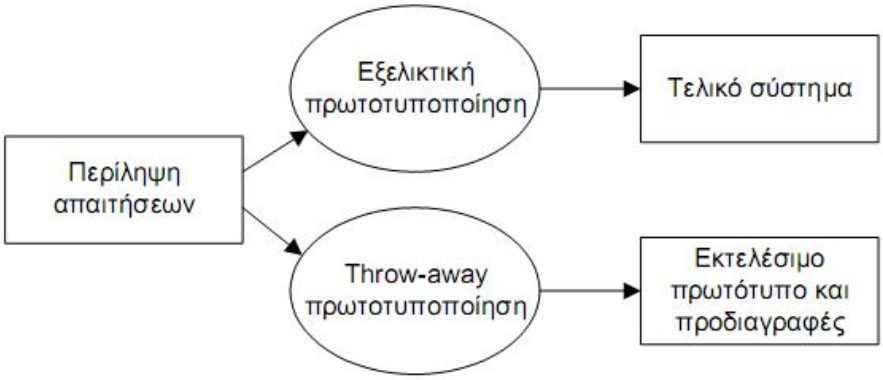
Η προτυποποίηση μπορεί να χρησιμοποιηθεί και για άλλους σκοπούς, όπως στην εκπαίδευση των χρηστών, δηλαδή το πρωτότυπο να χρησιμοποιηθεί ως εκπαιδευτικό

εργαλείο για την εκμάθηση του τελικού συστήματος, αλλά ταυτόχρονα είναι και ένας τρόπος μείωσης του ρίσκου, αφού περιορίζονται τα λάθη και οι παραλείψεις. Αν τα λάθη αφεθούν για διόρθωση στις τελευταίες φάσεις του κύκλου ζωής το κόστος αυξάνεται κατακόρυφα.

Το βασικό **μειονέκτημα** του μοντέλου της προτυποποίησης είναι ότι το κόστος ανάπτυξής του αποτελεί ένα μεγάλο μέρος του συνολικού κόστους του συστήματος που αναπτύσσεται. Πολλές φορές είναι οικονομικά πιο συμφέρον να μεταβληθεί το τελικό προϊόν από το να δημιουργηθεί ένα πρωτότυπο.

Είναι προφανές, ότι είναι πολύ δύσκολο να προβλεφτεί ποιες ακριβώς δυσκολίες θα αντιμετωπίσει ο τελικός χρήστης από την καθημερινή χρήση ενός νέου συστήματος λογισμικού. Ιδιαίτερα εάν αναφερόμαστε σε μεγάλα συστήματα λογισμικού η δυσκολία αυτή μπορεί να καταφανεί μόνο όταν το ολοκληρωμένο σύστημα αναπτυχθεί και τεθεί σε λειτουργία. Για να αντιμετωπιστεί αυτή η δυσκολία μπορεί να χρησιμοποιηθεί η **εξελικτική (evolutionary) προτυποποίηση**. Κατά τη προσέγγιση αυτή δημιουργείται μια περιορισμένη (ατελής) έκδοση του συστήματος πάνω στην οποία γίνονται διορθώσεις και προσθήκες καθώς απαιτήσεις είτε διευκρινίζονται είτε ανακαλύπτονται είτε βελτιώνονται μέχρι να καταλήξουμε σε μια έκδοση που να είναι επαρκής και ικανοποιητική.

Το βασικό **πρόβλημα** του μοντέλου της εξελικτικής προτυποποίησης είναι ότι με τις συνεχείς διορθωτικές και προσθετικές παρεμβολές παράγεται *'μπλεγμένος' κώδικας* (spaghetti code) που είναι πολύ δύσκολα συντηρήσιμος. Εναλλακτικά και προκειμένου να αποφύγουμε το φαινόμενο του 'μπλεγμένου κώδικα', θα μπορούσε να δημιουργηθεί ένα πρωτότυπο το οποίο σκοπό αποκλειστικό θα είχε την αποσαφήνιση των απαιτήσεων και να παράσχει πληροφορίες για την εκτίμηση του ρίσκου του τελικού συστήματος. Μετά την αξιολόγηση το πρωτότυπο σύστημα '**πετιέται**' και δεν χρησιμοποιείται για την ανάπτυξη του συστήματος **(throw-away prototype)**.

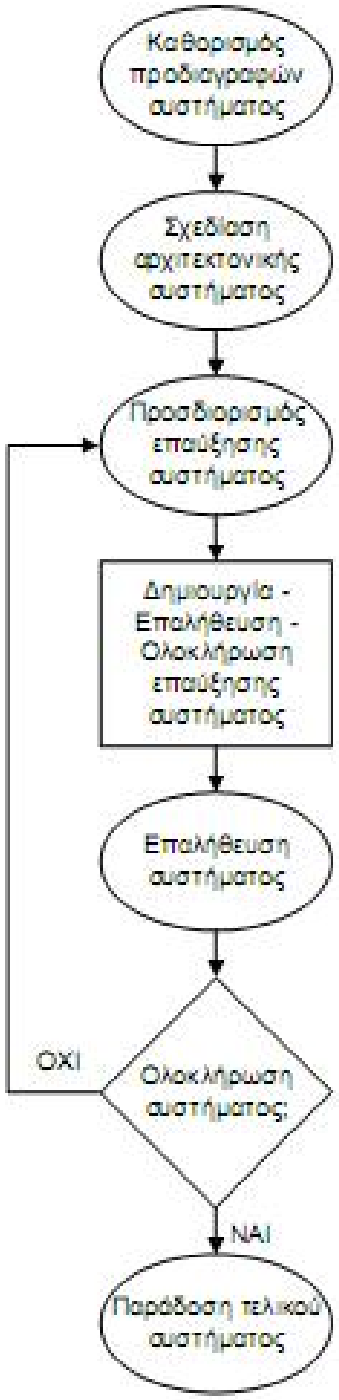


Εικόνα 6: Throw-away prototype

Ο χρόνος που απαιτείται για την ανάπτυξη ενός συστήματος μπορεί να μειωθεί ακόμα περισσότερο εάν κάποια τμήματα του συστήματος μπορούν να επαναχρησιμοποιηθούν. Έτσι τα πρωτότυπα μπορούν να κατασκευαστούν ακόμα πιο γρήγορα εάν υπάρχει μια βιβλιοθήκη με επαναχρησιμοποιήσιμες ψηφίδες (reusable components) και φυσικά κάποιος τρόπος σύνθεσης των ψηφίδων αυτών. Η επαναχρησιμοποίηση ψηφίδων αρμόζει περισσότερο στην throw – away προσέγγιση της πρωτοτυποποίησης.

1.6.3 Μοντέλο Λειτουργικής Επαύξησης

Μια εναλλακτική διαδικασία που συνδυάζει τα πλεονεκτήματα της εξελικτικής προσέγγισης με τον έλεγχο που απαιτείται για μεγάλα συστήματα είναι η **λειτουργική επαύξηση (incremental development)**. Σύμφωνα με το μοντέλο αυτό αρχικά αναπτύσσεται μια έκδοση του συστήματος που περιέχει τις περισσότερο σημαντικές και κρίσιμες λειτουργίες. Από την χρησιμοποίηση αυτής της έκδοσης κερδίζεται εμπειρία η οποία χρησιμοποιείται για την βελτίωσή της. Στην συνέχεια γίνεται μια προσαύξηση η οποία επεκτείνει την προηγούμενη έκδοση η οποία περιέχει και άλλες λειτουργίες. Η νέα έκδοση εκλεπτύνεται και προσαυξάνεται με την σειρά της με τον ίδιο τρόπο έως ότου κατασκευαστεί η τελική έκδοση. Το μοντέλο ενδείκνυται στις περιπτώσεις που υπάρχει σαφής γνώση και πολύ μικρή ή καθόλου μεταβλητότητα των απαιτήσεων του υπό ανάπτυξη λογισμικού. Άρα πρόκειται για μοντέλο που χρησιμοποιείται σε λίγες περιπτώσεις μια και το βασικό πρόβλημα της ανάπτυξης λογισμικού είναι η ασάφεια (σε μικρότερο ή μεγαλύτερο βαθμό) των απαιτήσεων του συστήματος.



Εικόνα 7: Μοντέλο Λειτουργικής Επαύξησης

Με αυτό το μοντέλο αποφεύγονται προβλήματα που προκύπτουν από τις συνεχείς αλλαγές, όπως στην εξελικτική προτυποποίηση. Η αρχιτεκτονική του συστήματος καθορίζεται σχετικά νωρίς, και λειτουργεί σαν πλαίσιο. Τα μέρη που αποτελούν το σύστημα αναπτύσσονται με επαυξήσεις και παραδίδονται με αυτό τον τρόπο (βλέπε και Βεσκούκης, 2000).

1.6.4 Εναλλακτικά Μοντέλα Ανάπτυξης Λογισμικού

Τα εναλλακτικά μοντέλα έχουν προταθεί ως μια εναλλακτική λύση στα συμβατικά με σκοπό να ελαττώσουν τα προβλήματα και τις αδυναμίες που έχουν τα συμβατικά μοντέλα (Σκορδαλάκης, 1991; Sommerville, 2001).

Το Λειτουργικό Μοντέλο (operational model) [Zave, 1984] χρησιμοποιεί τις προδιαγραφές που είναι λειτουργικές και περιγράφουν τι θα κάνει το σύστημα έμμεσα, μέσα από μια περιγραφή η οποία δείχνει πως αυτό θα λειτουργεί. Η περιγραφή αυτή γίνεται σε μια γλώσσα που να μπορεί να εκτελεστεί ώστε να αξιολογηθεί και έτσι να γίνει φανερή η συμπεριφορά του συστήματος. Οι λειτουργικές προδιαγραφές που χρησιμοποιούνται σύμφωνα με το μοντέλο αυτό είναι ένα είδος πρωτότυπου στο οποίο είναι εμφανής όλη η λειτουργική συμπεριφορά του συστήματος, χρησιμοποιώντας διαφορετικά μέσα από αυτά που θα χρησιμοποιούσε το τελικό σύστημα.

Αξιολογώντας τη συμπεριφορά του συστήματος, οι χρήστες μπορούν να κάνουν παρατηρήσεις και αλλαγές στις λειτουργικές προδιαγραφές. Ο κύκλος αυτός αξιολόγησης – αλλαγών επαναλαμβάνεται έως ότου θεωρηθεί ότι το σύστημα έχει την επιθυμητή λειτουργικότητα. Έτσι ολοκληρώνεται η φάση των απαιτήσεων και στην συνέχεια μπορεί να χρησιμοποιηθεί για παράδειγμα το μοντέλο του καταρράκτη από την φάση της σχεδίασης και κάτω.

Βασικό πρόβλημα του μοντέλου αποτελεί ότι χρησιμοποιεί εκτελέσιμες γλώσσες προδιαγραφών (executable specification languages) που είναι αυστηρά τυπικές (formal) και απαιτούν ιδιαίτερες γνώσεις από την ομάδα ανάπτυξης αλλά και εργαλεία υλοποίησης (compilers – interpreters, specification languages). (Madnick, 1991) (Ian Sommerville, 2008)

1.6.5 Μοντέλο Αυτόματου Προγραμματισμού

Το μοντέλο αυτόματου προγραμματισμού (automatic programming model) βασίζεται στην ιδέα της δημιουργίας ενός συστήματος που να μπορεί να δημιουργήσει λογισμικό αυτόματα αφού πρώτα του δοθούν οι προδιαγραφές του προβλήματος (Σχήμα 3.4)

Εικόνα 8: Μοντέλο αυτόματου προγραμματισμού

Η αυτοματοποιημένη δημιουργία λογισμικού είναι μια παλιά ιδέα που χρησιμοποιήθηκε και για την δημιουργία μεταφραστών γλωσσών προγραμματισμού (μετα-φραστές / meta-translators). Το μοντέλο αυτόματου προγραμματισμού ενδείκνυται για τις περιπτώσεις που οι απαιτήσεις του συστήματος είναι σαφώς καθορισμένες ώστε να μπορούν να περιγραφούν με ένα πολύ τυπικό τρόπο όπως είναι οι γραμματικές χωρίς συμφραζόμενα κλπ (Πιντέλας, 2001).

Τα εργαλεία που χρησιμοποιούνται (μετα-μεταφραστέές και οι αντίστοιχες γραμματικές) απαιτούν υψηλές γνώσεις πληροφορικής και επομένως πολύ εξειδικευμένη και έμπειρη ομάδα ανάπτυξης.

1.6.6 Μετασχηματιστικό Μοντέλο

Το μετασχηματιστικό μοντέλο (transformational model) [Partsch, 1983; Agresti, 1986] είναι μια παραλλαγή του μοντέλου αυτόματου προγραμματισμού και υποστηρίζει τον μετασχηματιστικό προγραμματισμό. Αρχικά ορίζονται οι προδιαγραφές του συστήματος με ένα τυπικό τρόπο και στη συνέχεια με μια σειρά από αυτόματους μετασχηματισμούς μετατρέπονται σε κώδικα.

Πλεονέκτημα του μοντέλου είναι ότι δεν χρειάζεται έλεγχος ορθότητας, παρά μόνον για τις τυπικές προδιαγραφές. Η συντήρηση γίνεται με αλλαγές στις τυπικές προδιαγραφές.

Μειονεκτήματα του μοντέλου αυτού είναι τα ίδια με εκείνα του μοντέλου του αυτόματου προγραμματισμού.

1.6.7 Μοντέλο Επαναχρησιμοποίησης Λογισμικού

Με το μοντέλο επαναχρησιμοποίησης λογισμικού (software reusability model) γίνεται χρήση ήδη υπάρχοντος και δοκιμασμένου λογισμικού, σχεδίων και κώδικα. Οι υπάρχουσες ψηφίδες λογισμικού (με ελεγμένη ορθότητα) ενσωματώνονται σε νέα προϊόντα λογισμικού.

Η διαδικασία αυτή δεν είναι εύκολη, αφού παρουσιάζονται δυσκολίες, λόγω της ανυπαρξίας εργαλείων και τεχνικών καταλλήλων για αυτή τη δουλειά, αλλά και της έλλειψης προτύπων κατασκευής ψηφίδων λογισμικού που να μπορούν να επαναχρησιμοποιηθούν.

Τα βασικά πλεονεκτήματα του μοντέλου είναι η συντόμευση του χρονικού διαστήματος κατασκευής λογισμικού αλλά και η βελτίωση της αξιοπιστίας του αφού στηρίζεται σε έτοιμα, δοκιμασμένα και άρα αξιόπιστα τμήματα λογισμικού. Τα συστατικά λογισμικού που θα επαναχρησιμοποιηθούν μπορεί να είναι πολλών και διαφόρων μεγεθών, όπως για παράδειγμα :

- Επαναχρησιμοποίηση ολόκληρων συστημάτων εφαρμογών (application system reuse) τα οποία είτε ενσωματώνονται στο καινούριο σύστημα χωρίς αλλαγή είτε δημιουργούνται ολόκληρες οικογένειες εφαρμογών που μπορούν να τρέξουν σε διαφορετικές πλατφόρμες ώστε να ικανοποιήσουν συγκεκριμένες ανάγκες.

- Επαναχρησιμοποίηση ψηφίδων (component reuse) όπου διάφορα ‘συστατικά’ μιας εφαρμογής από ένα υποσύστημα μέχρι ένα μεμονωμένο αντικείμενο μπορούν να επαναχρησιμοποιηθούν. Για παράδειγμα αντικείμενο τύπου ‘χρονόμετρο’ μπορεί να επαναχρησιμοποιηθεί πολλές φορές στην ανάπτυξη ενός λογισμικού φυσικής, χημείας κλπ.

- Επαναχρησιμοποίηση κάποιας λειτουργίας ή συνάρτησης (function reuse) όπου συστατικά (συναρτήσεις, διαδικασίες) που υλοποιούν μια απλή λειτουργία όπως μια μαθηματική συνάρτηση επαναχρησιμοποιούνται σε άλλες εφαρμογές. Αυτού του είδους η επαναχρησιμοποίηση που βασίζεται στις βιβλιοθήκες είναι και η πιο διαδεδομένη.

Το προφανές κέρδος από την επαναχρησιμοποίηση είναι η μείωση του κόστους ανάπτυξης δεδομένου ότι λιγότερα συστατικά του συστήματος χρειάζεται να προσδιοριστούν, να σχεδιαστούν, να υλοποιηθούν και να αξιολογηθούν.

1.6.8 Αντικειμενοστραφές Μοντέλο

Το αντικειμενοστραφές μοντέλο (object-oriented model) βασίζεται στον αντικειμενοστραφή προγραμματισμό (βλέπε και Βεσκούκης, 2000). Αναπτύσσεται με τρόπο παρόμοιο με το μοντέλο του καταρράκτη, αλλά διαφέρει σε δύο βασικά σημεία:

- Οι διάφορες φάσεις υπερκαλύπτονται μεταξύ τους
- Η ανάπτυξη του, αν χρειαστεί οπισθοδρομεί στην προηγούμενη φάση, εκτός από την τελευταία που οπισθοδρομεί στην αρχή.

Το κύριο **πλεονέκτημα** του μοντέλου είναι ότι κάνει χρήση επαναχρησιμοποιήσιμων μονάδων και με αυτό τον τρόπο συντομεύεται τόσο η φάση της ανάπτυξης όσο και η φάση της συντήρησης.

1.6.9 Σπειροειδές μοντέλο

Το σπειροειδές μοντέλο ή αλλιώς μοντέλο του Boehm (spiral model ή Boehm's spiral model) (Boehm, 1988) είναι ένα δημοφιλές μοντέλο που ουσιαστικά είναι γενίκευση των μοντέλων της λειτουργικής επαύξησης και της πρωτοτυποποίησης και διαφέρει από όλα τα άλλα μοντέλα που βασίζονται στο ακολουθιακό μοντέλο του καταρράκτη. Το μοντέλο του Boehm έχει τη μορφή μιας σπείρας (βλέπε και Βεσκούκης, 2000) όπου κάθε γύρος στην σπείρα αναπαριστά και μια φάση έτσι όπως αυτή καθορίζεται κάθε φορά από την διοίκηση.

Η διαδοχή των φάσεων δεν γίνεται ούτε σταθερά ούτε γραμμικά, ενώ η εκτέλεσή τους μπορεί να γίνει είτε με τη φορά της σπείρας, είτε με την αντίθετη φορά, ανάλογα με το ρίσκο που λαμβάνεται και το οποίο αποτελεί θεμελιώδη έννοια στο σπειροειδές μοντέλο.

Σε κάθε γύρο που διανύεται υπάρχουν οι εξής φάσεις:

• Καθορισμός στόχων, εναλλακτικών λύσεων και υπολογισμός περιορισμών.

• Ανάλυση και υπολογισμός του ρίσκου και προσπάθεια μείωσης του.

• Ανάπτυξη και επαλήθευση ενδιάμεσου προϊόντος – εφόσον η προηγούμενη φάση δεν έδειξε κάποιο σοβαρό ρίσκο – πρόσθεση νέων λειτουργικών προδιαγραφών.

• Σχεδιασμός των επόμενων βημάτων.

Η βασική διαφορά του παραπάνω μοντέλου από τα υπόλοιπα μοντέλα είναι ότι σε αυτό υπολογίζεται πριν την έναρξη κάθε φάσης το ρίσκο, γεγονός που ουσιαστικά αποτελεί και το βασικό του **πλεονέκτημα** αν και ο υπολογισμός και η ανάλυση του ρίσκου δεν είναι εύκολη υπόθεση πρακτικά.

1.6.9.1 Σχόλιο μελέτης:

Η έννοια ρίσκο είναι δύσκολο να καθοριστεί πλήρως. Μια απλή εξήγηση που θα μπορούσαμε να δώσουμε για το ρίσκο είναι ότι είναι οτιδήποτε θεωρείται ότι μπορεί να πάει στραβά. Για παράδειγμα, αν χρησιμοποιηθεί ένα καινούριο πακέτο για τρισδιάστατα γραφικά, ρίσκο θα αποτελέσει κατά πόσο είναι δύσκολο να το μάθουν γρήγορα οι εμπλεκόμενοι στην ομάδα ανάπτυξης. Άλλο σημαντικό ρίσκο στην ανάπτυξη λογισμικού είναι τα λάθη και περισσότερο οι παραλείψεις των απαιτήσεων του συστήματος. Τα ρίσκα είναι συνέπεια ελλιπούς πληροφόρησης. Για την μείωση ή εξάλειψη τους χρειαζόμαστε περισσότερες πληροφορίες. Για το παραπάνω παράδειγμα, το ρίσκο μπορεί να μειωθεί κάνοντας στην ομάδα ανάπτυξης ένα ταχύρυθμο πρόγραμμα εκμάθησης, ή προσλαμβάνοντας νέα ομάδα ανάπτυξης με γνώση του συγκεκριμένου πακέτου. Η μελέτη των παραπάνω μοντέλων ήταν απαραίτητη για την απόκτηση της αναγκαίας γνώσης. Αφού εξετάσαμε τα μοντέλα ανάπτυξης λογισμικού, πρέπει να διαλέξουμε εκείνο που εξυπηρετεί καλύτερα τις ανάγκες μας.

1.7 Ποιο είναι το Κατάλληλο Μοντέλο για Ανάπτυξη Λογισμικού;

Από όσα αναφέρθηκαν στις προηγούμενες ενότητες η επιλογή του κατάλληλου μοντέλου ανάπτυξης λογισμικού είναι λογικό να μην περιλαμβάνει τα μοντέλα αυτόματου προγραμματισμού. Και αυτό γιατί η ανάπτυξη του λογισμικού είναι μια δημιουργική διαδικασία που εμπλέκει πολλών περιοχών ειδικούς και όχι μια αυτόματη διαδικασία. Πολλά, επίσης, μοντέλα όπως το λειτουργικό χρησιμοποιούνται περισσότερο για την ανάπτυξη ειδικών περιπτώσεων λογισμικού.
Το μοντέλο του καταρράκτη, και όλα τα μοντέλα αυτής της μορφής, αν και είναι δημοφιλής επιλογή, είναι επίσης απορριπτέα γιατί όπως έχει αναφερθεί, ξέρουμε εάν έχουμε κατασκευάσει αυτό που θέλαμε μόνο όταν η διαδικασία ανάπτυξης έχει ολοκληρωθεί. Επίσης, δεδομένου ότι το προϊόν κάθε φάσης θεωρείται τελικό είναι σχεδόν αδύνατο να παρέμβουμε σε αυτό όταν βρισκόμαστε σε αρκετά μεταγενέστερο

στάδιο ανάπτυξης στο οποίο θα παρουσιαστεί κάποιο πρόβλημα ή λάθος. Και αυτό γιατί αυξάνεται γεωμετρικά το κόστος της διόρθωσης.

Το μοντέλο επαναχρησιμοποίησης λογισμικού δεν θα μπορούσε να οδηγήσει σε ολοκληρωμένη ανάπτυξη λογισμικού παρόλα αυτά θα μπορούσε να βοηθήσει ιδιαίτερα ως τμήμα ενός άλλου μοντέλου.

Το μοντέλο πρωτοτυποποίησης θα μπορούσε να δώσει έναν τρόπο ανάπτυξης δεδομένου ότι ειδικά στην εξελικτική του μορφή μπορεί να δώσει πληροφορίες που βοηθούν στην ανάπτυξη πολύ πιο γρήγορα από το μοντέλο του καταρράκτη, τόσο ως προς την εφικτότητα κατασκευής του τελικού συστήματος όσο και ως προς την ακρίβεια των προδιαγραφών των απαιτήσεων από το λογισμικό.

Επίσης, το μοντέλο της throw-away πρωτοτυποποίησης είναι μια σοβαρή υποψηφιότητα για την ανάπτυξη λογισμικού διότι αφενός δίνει πληροφορίες για την εφικτότητα του τελικού συστήματος αλλά και γιατί αποσαφηνίζει τις απαιτήσεις του συστήματος. Αυτό συμβαίνει διότι το τελικό σύστημα όπως προκύπτει με την χρήση αυτού του μοντέλου είναι πιο συντηρήσιμο, τελικά για ολόκληρη της διάρκεια ζωής του λογισμικού προκύπτει σημαντική μείωση του συνολικού κόστους αν και όπως έχει ήδη αναφερθεί επειδή το αρχικό πρωτότυπο θα ‘πεταχτεί’, αυξάνεται ως ένα βαθμό το κόστος ανάπτυξης από το γεγονός αυτό.

Το σπειροειδές μοντέλο ανάπτυξης διαφέρει από όλα τα προηγούμενα κυρίως στο ότι σε κάθε φάση ανάπτυξης υπάρχει μια άμεση εκτίμηση του ρίσκου. Καταρχήν σε αυτό καθορίζονται οι στόχοι, οι εναλλακτικές λύσεις και οι περιορισμοί τους και στη συνέχεια αξιολογούνται αυτές οι λύσεις και καθορίζονται στρατηγικές επίλυσης του ρίσκου εάν αυτό είναι μεγάλο. Με αυτό τον τρόπο είναι δυνατό να αναπτυχθεί το λογισμικό χωρίς να υπάρχει ο κίνδυνος της μη ολοκλήρωσης μιας φάσης ανάπτυξής του και συγχρόνως μετά από κάθε φάση έχουμε ένα ενδιάμεσο πρωτότυπο που μπορεί σταδιακά να αξιολογείται.

Λαμβάνοντας υπόψη ότι η σύγχρονη τάση στην ανάπτυξη λογισμικού ουσιαστικά χρησιμοποιεί κάποιες γενικές κατευθύνσεις από τις υπάρχουσες ιδέες (μοντέλα) αλλά αφήνει αρκετούς βαθμούς ελευθερίας στον κατασκευαστή, μπορούμε να συμπεράνουμε ότι το μοντέλο καταρράκτη μπορεί να αποτελέσει βάση για την ανάπτυξη λογισμικού με ορισμένες παραλλαγές οι οποίες εξειδικεύονται κάθε φορά ανάλογα με το περιβάλλον ανάπτυξης, το συγκεκριμένο πρόβλημα κλπ.

1.8 Είδη προγραμματισμού

1.8.1 Δομημένος προγραμματισμός

Στην επιστήμη υπολογιστών δομημένος προγραμματισμός (structured programming) ή διαδικαστικός προγραμματισμός (procedural programming) είναι μία προσέγγιση στον προγραμματισμό[1], η οποία βασίζεται στην έννοια της κλήσης διαδικασίας. Η διαδικασία, γνωστή επίσης και ως ρουτίνα, υπορουτίνα, μέθοδος ή συνάρτηση (δεν σχετίζεται άμεσα με τη μαθηματική έννοια της συνάρτησης), είναι απλά ένα αυτοτελές σύνολο εντολών προς εκτέλεση.

Ο δομημένος προγραμματισμός βασίζεται στην αρχή του διαίρει και βασίλευε, καθώς διασπά το βασικό πρόβλημα σε μικρότερα υποπροβλήματα (γνωστά επίσης και ως εργασίες). Κάθε εργασία με πολύπλοκη περιγραφή διαιρείται σε μικρότερες, έως ότου οι εργασίες να είναι αρκετά μικρές, περιεκτικές και εύκολες προς κατανόηση.

1.8.1.1 Ιστορική διαδρομή

Ιστορικά ο δομημένος προγραμματισμός αναπτύχθηκε ύστερα από έρευνα κατά τη δεκαετία του 1960, ως βελτίωση του ήδη υπάρχοντος διαδικαστικού προγραμματισμού. Ένα από τα πιο σημαντικά αποτελέσματα αυτής της έρευνας ήταν η ανάπτυξη της γλώσσας Pascal (γλώσσα προγραμματισμού), από τον Νίκλαους Βιρτ (Niklaus Wirth) το 1971, η οποία σύντομα έγινε η προτιμώμενη γλώσσα διδασκαλίας σε πολλά πανεπιστήμια[2]. Η έννοια της διαδικασίας επομένως ήταν προϋπάρχουσα αλλά δεν έπαιζε τόσο σημαντικό ρόλο στη δομή των υπό συγγραφή εφαρμογών, καθώς τα δεδομένα ήταν αρκετά διαχωρισμένα από τις διαδικασίες και έπρεπε ο προγραμματιστής να θυμάται για κάθε διαδικασία ποια άλλη καλούσε, αλλά και ποια δεδομένα διαφοροποιούνταν. Καθώς όμως οι περισσότερες διαδικαστικές γλώσσες γρήγορα υιοθέτησαν στοιχεία ώστε να υποστηρίζουν δομημένο προγραμματισμό, οι δύο όροι σήμερα έχουν πρακτικώς ταυτιστεί. Με τον καιρό οι δομημένες γλώσσες έφτασαν να μην επαρκούν για τη συγγραφή προγραμμάτων, επεκτάθηκαν και ως λύση υιοθετήθηκε ο αντικειμενοστραφής προγραμματισμός.

1.8.2 Πλεονεκτήματα

Ο δομημένος προγραμματισμός βοηθάει την ευκολότερη συγγραφή πολύπλοκων προγραμμάτων. Με τον δομημένο προγραμματισμό γίνεται πιο εύκολη και ταχύτερη η διαχείριση, η συντήρηση και η αποσφαλμάτωση, καθώς βασίζεται σε μεγαλύτερες,

πολυπλοκότερες και περιεκτικότερες μονάδες, όπως οι διαδικασίες, αντί για μεμονωμένες εντολές.

1.8.3 Συναρτησιακός προγραμματισμός

Στην επιστήμη υπολογιστών, συναρτησιακός προγραμματισμός είναι ένα προγραμματιστικό παράδειγμα που αντιμετωπίζει τον υπολογισμό ως την αποτίμηση μαθηματικών συναρτήσεων και αποφεύγει την κατάσταση προγράμματος και τα μεταβλητά δεδομένα. Δίνει έμφαση στην εφαρμογή συναρτήσεων, σε αντίθεση με τον προστακτικό προγραμματισμό, ο οποίος δίνει έμφαση στις αλλαγές κατάστασης. Ο συναρτησιακός προγραμματισμός έχει τις ρίζες του στο λογισμό λάμδα, ένα τυπικό σύστημα που αναπτύχθηκε τη δεκαετία 1930 για τη διερεύνηση του ορισμού συναρτήσεων, της εφαρμογής συναρτήσεων και της αναδρομής. Πολλές συναρτησιακές γλώσσες προγραμματισμού μπορούν να θεωρηθούν ως επεκτάσεις του λογισμού λάμδα.

Πρακτικά, η διαφορά μεταξύ μιας μαθηματικής συνάρτησης και της έννοιας της "συνάρτησης" που χρησιμοποιείται στον προστακτικό προγραμματισμό είναι ότι οι προστακτικές συναρτήσεις μπορούν να έχουν παρενέργειες, αλλάζοντας την τιμή των ήδη αποτιμημένων υπολογισμών. Λόγω αυτού, δεν έχουν διαφάνεια αναφορικότητας, δηλαδή η ίδια έκφραση της γλώσσας μπορεί να δώσει διαφορετικές τιμές ανάλογα με την κατάσταση του εκτελούμενου προγράμματος. Αντίστροφα, σε συναρτησιακά προγράμματα, η τιμή που επιστρέφει μια συνάρτηση εξαρτάται μόνο από τα ορίσματα που αποτελούν την είσοδο της συνάρτησης. Έτσι, η κλήση μιας συνάρτησης f με ένα όρισμα x θα δώσει το ίδιο αποτέλεσμα f(x) και τις δύο φορές. Η εξάλειψη των παρενεργειών μπορεί να κάνει πολύ ευκολότερο το να κατανοηθεί και να προβλεφθεί η συμπεριφορά ενός προγράμματος. Αυτό είναι ένα από τα κίνητρα για την ανάπτυξη του συναρτησιακού προγραμματισμού.

Οι συναρτησιακές γλώσσες προγραμματισμού, και ειδικότερα οι αμιγώς συναρτησιακές συναντώνται περισσότερο στο ακαδημαϊκό παρά στο εμπορικό ή βιομηχανικό περιβάλλον. Παρ' όλα αυτά, αξιοσημείωτες συναρτησιακές γλώσσες που χρησιμοποιούνται στη βιομηχανία και στην ανάπτυξη εμπορικών εφαρμογών είναι ανάμεσα σε άλλες η Erlang, η OCaml, η Haskell, η Scheme, αλλά και γλώσσες ειδικών πεδίων όπως η R για τη στατιστική, η Mathematica για τα συμβολικά μαθηματικά, ή η K για την χρηματιστηριακή ανάλυση. Διαδεδομένες γλώσσες

ειδικού πεδίου όπως η SQL και τα Lex/Yacc χρησιμοποιούν στοιχεία συναρτησιακού προγραμματισμού, ειδικά για να αποφύγουν μεταβλητές τιμές. Τα λογιστικά φύλλα μπορούν επίσης να θεωρηθούν συναρτησιακές γλώσσες προγραμματισμού.

Προγραμματισμός σε συναρτησιακό στυλ μπορεί να επιτευχθεί και σε μη συναρτησιακές γλώσσες όπως η C, η Java, η Python, που δεν είναι ειδικά σχεδιασμένες για τέτοια χρήση.

1.8.4 Προστακτικός προγραμματισμός

Στην επιστήμη υπολογιστών καλούμε προστακτικό προγραμματισμό, σε αντίθεση με το δηλωτικό προγραμματισμό, ένα προγραμματιστικό υπόδειγμα όπου το ζητούμενο κατασκευάζεται / υπολογίζεται αλλάζοντας την κατάσταση του υπολογιστή μέσω εντολών.

Το υπόδειγμα αυτό ακολουθούν οι διαδικαστικές γλώσσες προγραμματισμού, όπως οι Pascal, C , Fortran κ.α., αλλά και οι αντικειμενοστρεφείς γλώσσες όπως η Java, η C++, η C# κ.α. (wikipedia.org)

1.8.5 Δηλωτικός προγραμματισμός

Στην επιστήμη υπολογιστών δηλωτικός προγραμματισμός (declarative programming) είναι ένα προγραμματιστικό υπόδειγμα όπου, σε αντίθεση με τον προστακτικό προγραμματισμό, το ζητούμενο αποτέλεσμα υπολογίζεται περιγράφοντας απλώς τις επιθυμητές ιδιότητες του. Παραδείγματα γλωσσών δηλωτικού προγραμματισμού είναι η Haskell, η Prolog, η SQL, η HTML και η CSS.

1.8.6 Αντικειμενοστρεφής προγραμματισμός

Στην επιστήμη υπολογιστών αντικειμενοστρεφή προγραμματισμό (object-oriented programming), ή ΑΠ, ονομάζουμε ένα προγραμματιστικό υπόδειγμα το οποίο εμφανίστηκε στα τέλη της δεκαετίας του 1960 και καθιερώθηκε κατά τη δεκαετία του 1990, αντικαθιστώντας σε μεγάλο βαθμό το παραδοσιακό υπόδειγμα του δομημένου προγραμματισμού. Πρόκειται για μία μεθοδολογία ανάπτυξης προγραμμάτων, υποστηριζόμενη από κατάλληλες γλώσσες προγραμματισμού, όπου ο χειρισμός σχετιζόμενων δεδομένων και των διαδικασιών που επενεργούν σε αυτά γίνεται από κοινού, μέσω μίας δομής δεδομένων που τα περιβάλλει ως αυτόνομη οντότητα με ταυτότητα και δικά της χαρακτηριστικά. Αυτή η δομή δεδομένων καλείται αντικείμενο και αποτελεί πραγματικό στιγμιότυπο στη μνήμη ενός σύνθετου, και

πιθανώς οριζόμενου από τον χρήστη, τύπου δεδομένων ονόματι κλάση. Η κλάση προδιαγράφει τόσο δεδομένα όσο και τις διαδικασίες οι οποίες επιδρούν επάνω τους• αυτή υπήρξε η πρωταρχική καινοτομία του ΑΠ.

Στην επιστήμη υπολογιστών αντικειμενοστρεφή προγραμματισμό (object-oriented programming), ή ΑΠ, ονομάζουμε ένα προγραμματιστικό υπόδειγμα το οποίο εμφανίστηκε στα τέλη της δεκαετίας του 1960 και καθιερώθηκε κατά τη δεκαετία του 1990, αντικαθιστώντας σε μεγάλο βαθμό το παραδοσιακό υπόδειγμα του δομημένου προγραμματισμού. Πρόκειται για μία μεθοδολογία ανάπτυξης προγραμμάτων, υποστηριζόμενη από κατάλληλες γλώσσες προγραμματισμού, όπου ο χειρισμός σχετιζόμενων δεδομένων και των διαδικασιών που επενεργούν σε αυτά γίνεται από κοινού, μέσω μίας δομής δεδομένων που τα περιβάλλει ως αυτόνομη οντότητα με ταυτότητα και δικά της χαρακτηριστικά. Αυτή η δομή δεδομένων καλείται αντικείμενο και αποτελεί πραγματικό στιγμιότυπο στη μνήμη ενός σύνθετου, και πιθανώς οριζόμενου από τον χρήστη, τύπου δεδομένων ονόματι κλάση. Η κλάση προδιαγράφει τόσο δεδομένα όσο και τις διαδικασίες οι οποίες επιδρούν επάνω τους• αυτή υπήρξε η πρωταρχική καινοτομία του ΑΠ.

Έτσι μπορεί να οριστεί μία προδιαγραφή δομής αποθήκευσης (π.χ. μία κλάση "τηλεόραση") η οποία να περιέχει τόσο ιδιότητες (π.χ. μία μεταβλητή "τρέχον κανάλι") όσο και πράξεις ή χειρισμούς επί αυτών των ιδιοτήτων (π.χ. μία διαδικασία "άνοιγμα της τηλεόρασης"). Στο εν λόγω παράδειγμα κάθε υλική τηλεόραση (κάθε αντικείμενο αποθηκευμένο πραγματικά στη μνήμη) αναπαρίσταται ως ξεχωριστό, "φυσικό" στιγμιότυπο αυτής της πρότυπης, ιδεατής κλάσης. Επομένως μόνο τα αντικείμενα καταλαμβάνουν χώρο στη μνήμη του υπολογιστή ενώ οι κλάσεις αποτελούν απλώς "καλούπια". Οι αιτίες που ώθησαν στην ανάπτυξη του ΑΠ ήταν οι ίδιες με αυτές που οδήγησαν στην ανάπτυξη του δομημένου προγραμματισμού (ευκολία συντήρησης, οργάνωσης, χειρισμού και επαναχρησιμοποίησης κώδικα μεγάλων και πολύπλοκων εφαρμογών), όμως τελικώς η αντικειμενοστρέφεια επικράτησε καθώς μπορούσε να αντεπεξέλθει σε προγράμματα πολύ μεγαλύτερου όγκου και πολυπλοκότητας.

1.8.7 Αρχές αντικειμενοστρεφούς σχεδίασης

Με το πέρασμα του χρόνου κωδικοποιήθηκαν κάποιες ανεπίσημες αρχές για την ορθή σχεδίαση αντικειμενοστρεφών συστημάτων λογισμικού. Οι αρχές αυτές

παρουσιάστηκαν κατά καιρούς σε βιβλία και άρθρα ακαδημαϊκών και αναγνωρισμένων μηχανικών λογισμικού. Οι σπουδαιότερες αρχές είναι οι παρακάτω:

- Αρχή ανοιχτότητας-κλειστότητας (open-closed principle), του δημιουργού της γλώσσας προγραμματισμού Eiffel Μπέρτραντ Μέιερ. Η αρχή αυτή δηλώνει πως τα συστατικά ενός προγράμματος πρέπει να είναι "ανοιχτά" ως προς την επέκταση των δυνατοτήτων του συστήματος αλλά "κλειστά" ως προς αλλαγές στην υλοποίηση του. Πρακτικώς αυτό σημαίνει οι διάφορες κλάσεις και τα υπόλοιπα τμήματα λογισμικού να μη χρειάζεται να τροποποιηθούν σε περίπτωση που προστεθεί νέα λειτουργικότητα στο σύστημα (π.χ. μία νέα κλάση) προκειμένου να την αξιοποιήσουν. Βεβαίως είναι αδύνατο να μη χρειάζεται να τροποποιηθεί τίποτα, οπότε αυτό που επιτάσσει στην πραγματικότητα η εν λόγω αρχή είναι η ελαχιστοποίηση και η συγκέντρωση, κατά προτίμηση σε ένα μικρό τμήμα του κώδικα, των γραμμών που θα πρέπει να αλλάξουν. Αυτό συνήθως επιτυγχάνεται μέσω αφαίρεσης (με αφηρημένες κλάσεις ή διασυνδέσεις και πραγματικές κλάσεις που κληρονομούν από αυτές) και με χρήση του πολυμορφισμού.

Μία συνηθισμένη τακτική για διασφάλιση της κλειστότητας του ολικού προγράμματος ως προς την υλοποίηση μίας κλάσης, είναι η συνειδητή προσπάθεια για δήλωση όλων των γνωρισμάτων της ως ιδιωτικών. Έτσι η προσπέλαση των πεδίων της κλάσης μπορεί να ελεγχθεί εξ ολοκλήρου μέσω ειδικών δημόσιων μεθόδων της, γεγονός που διευκολύνει κατά πολύ την αποσφαλμάτωση: στις μεθόδους αυτές συγκεντρώνονται οι έλεγχοι επιτρεπτών τιμών για τα πεδία, έλεγχοι κατάλληλων συνθηκών κλπ.

- Αρχή υποκατάστασης Λίσκοφ (Liskov substitution principle), της επιστήμονα υπολογιστών Μπάρμπαρα Λίσκοφ. Η αρχή αυτή συμπυκνώνεται στον παρακάτω κανόνα για σχηματισμό μίας ορθής ιεραρχίας κλάσεων: μία κλάση Κ1 μπορεί να υλοποιηθεί ως υποκλάση μίας κλάσης Κ2 αν κάθε πρόγραμμα Π το οποίο λειτουργεί με αντικείμενα Κ2 συμπεριφέρεται με τον ίδιο τρόπο και με αντίστοιχα αντικείμενα Κ1. Έτσι με την αρχή υποκατάστασης Λίσκοφ φαίνεται πως για να οριστεί μία κλάση ως θυγατρική μίας άλλης δεν αρκεί να έχουν διαισθητικά μία ανάλογη εννοιολογική σχέση (π.χ. μία κλάση που

αναπαριστά όχημα και μία που αναπαριστά αυτοκίνητο) αλλά, στο πλαίσιο του υπό εξέταση προγράμματος, τα αντικείμενα της υποκλάσης να έχουν πάντα την ίδια προγραμματιστική συμπεριφορά με τα αντικείμενα της υπερκλάσης υπό τις ίδιες συνθήκες.

- Αρχή αντιστροφής εξαρτήσεων (dependency inversion principle), του γνωστού μηχανικού λογισμικού Ρόμπερτ Σέσιλ Μάρτιν. Η αρχή αυτή πρακτικά αποτελεί εκλέπτυνση της αρχής ανοιχτότητας-κλειστότητας, προϋποθέτοντας όμως χρήση και της αρχής υποκατάστασης Λίσκοφ. Αφορά ιεραρχίες κληρονομικότητας κλάσεων και τη χρήση αντικειμένων αυτών των ιεραρχιών από εξωτερικά προγράμματα. Στα πλαίσια της αρχής αντιστροφής εξαρτήσεων ένα τμήμα λογισμικού Α (π.χ. μία κλάση) το οποίο χρησιμοποιεί τις υπηρεσίες που παρέχει ένα άλλο τμήμα λογισμικού Β, καλώντας για παράδειγμα μία μέθοδό του, θεωρείται στοιχείο "υψηλότερου επιπέδου" σε σχέση με το Β. Η αρχή λέει πως τα υψηλού επιπέδου στοιχεία δεν πρέπει να εξαρτώνται από την υλοποίηση χαμηλότερου επιπέδου στοιχείων, αλλά πως και τα δύο πρέπει να βασίζονται σε ενδιάμεσα επίπεδα αφαίρεσης. Στην πράξη αυτή η αφαίρεση είναι μία διασύνδεση (ή αφηρημένη κλάση) την οποία γνωρίζει το υψηλού επιπέδου στοιχείο Α και υλοποιεί το χαμηλού επιπέδου στοιχείο Β. Ακόμα και αν το Β αλλαχθεί με μία κλάση Γ η οποία επίσης υλοποιεί την ίδια διασύνδεση, το Α θα πρέπει να συνεχίσει να λειτουργεί χωρίς καμία τροποποίηση. Η αρχή αντιστροφής εξαρτήσεων δεν είναι παρά ένα απτό παράδειγμα χρήσης ιεραρχικών επιπέδων με τη βοήθεια ενδιάμεσων αφαιρέσεων, μίας πρακτικής που εφαρμόζεται κατά κόρον στην επιστήμη υπολογιστών (για ένα άλλο παράδειγμα βλέπε δίκτυα υπολογιστών).
- Αρχή διαχωρισμού διασυνδέσεων (interface segregation principle), του μηχανικού λογισμικού Ρόμπερτ Σέσιλ Μάρτιν. Η εν λόγω αρχή σημαίνει ότι σε περιπτώσεις όπου διαφορετικά υποσύνολα μεθόδων μίας κλάσης αφορούν διαφορετικές περιπτώσεις χρήσης της κλάσης, σκόπιμο είναι να ορίζουμε επιμέρους διασυνδέσεις τις οποίες η κλάση θα υλοποιεί. Κάθε τέτοια διασύνδεση θα ορίζει μόνο το αντίστοιχο υποσύνολο των μεθόδων.
- Αρχή μοναδικής αρμοδιότητας (single responsibility principle), των Τομ Ντε Μάρκο και Μέιρ Πέιτζ Τζόουνς. Σύμφωνα με την αρχή αυτή κάθε κλάση θα πρέπει να έχει μόνο μία, καλά ορισμένη και διαχωρισμένη από το υπόλοιπο

πρόγραμμα αρμοδιότητα, η ύπαρξη της οποίας να εξυπηρετεί έναν συγκεκριμένο σκοπό. Αν μπορούμε να εντοπίσουμε σε μία κλάση Α δύο διαφορετικές αρμοδιότητες, τότε η καλύτερη λύση είναι η διάσπαση της σε δύο κλάσεις Β' και Γ', καθεμία από τις οποίες θα λάβει ένα υποσύνολο των πεδίων και των μεθόδων της Α. Τα υποσύνολα αυτά θα είναι ξένα μεταξύ τους, οπότε με το ανάποδο σκεπτικό αν μπορούμε να διασπάσουμε μία κλάση Α σε δύο άλλες κλάσεις (π.χ. σε περίπτωση που κάποιες μέθοδοι δε χρησιμοποιούν κάποια γνωρίσματα, οπότε οι μεν μπορούν να καταλήξουν στη μία κλάση Β' και τα πεδία στην άλλη κλάση Γ') τότε πιθανώς η κλάση να παραβιάζει την αρχή μοναδικής αρμοδιότητας. Έτσι έχουν προταθεί κάποιες μετρικές οι οποίες επιχειρούν να προσδιορίσουν την έλλειψη συνοχής (cohesion) σε μία κλάση, δηλαδή το κατά πόσον οι μέθοδοι της δε σχετίζονται με τα γνωρίσματα της. Συνήθως η συνοχή αντιπαραβάλλεται με τη σύζευξη (coupling), δηλαδή το βαθμό στον οποίον μία κλάση εξαρτάται από κάποια/ες άλλη/ες, και τα δύο αυτά μεγέθη είναι αντιστρόφως ανάλογα.

1.8.8 Λογικός Προγραμματισμός

Ο Λογικός Προγραμματισμός (logic programming) αποτελεί ένα σημαντικό κλάδο της Τεχνικής Νοημοσυνής που βασίζεται σε μεγάλο βαθμό στη Μαθηματική Λογική, αλλά την προεκτείνει εισάγοντας νέες μεθόδους αυτοματοποιημένης συμπερασματολογίας.

Η μελέτη του Λογικού Προγραμματισμού είναι να εξηγηθεί η λογική βάση του. Χωρίς αυτήν την μαθηματική εξήγηση δεν θα γίνει αντιληπτή η σπουδαιότητα του λογικού προγραμματισμού.

Η βάση αυτή θα φανεί χρήσιμη σε διάφορα θέματα όπως :

στη περιγραφή ενός συστήματος (δηλαδή στην κωδικοποίηση της γνώσης) με τη μορφή ενός συνόλου λογικών προτάσεων (δηλαδή με τη μορφή ενός λογικού προγράμματος) με ποιό τρόπο κινούμαστε από την Μαθηματική Λογική στο λογικό προγραμματισμό και στη συνέχεια στην γλώσσα προγραμματισμού Prolog.

Θα μας εφοδιάσει λ.χ. με το τι μπορούν ή δεν μπορούν να εκφράσουν ή υπολογίσουν τα λογικά προγράμματα, εν συντομία, τι σημαίνουν. Επιπλέον, μελετώντας

σημασιολογικά και θεωρητικά ζητήματα θα σχολιασθεί, όπου κρίνεται απαραίτητο, και η πρακτική σπουδαιότητα των υπό μελέτη θεμάτων.

Θα γνωρίσουμε τη γλώσσα της Μαθηματικής Λογικής - Προτασιακής και Κατηγορηματικής Λογικής α' τάξης - ως μιας γλώσσας που μπορεί να χρησιμοποιηθεί για την επίλυση προβλημάτων.

Στη συνέχεια θα δούμε πως η λογική και κατ' επέκταση ο λογικός προγραμματισμός δεν είναι απλά μια ακόμη γλώσσα για την επίλυση συνηθισμένων υπολογιστικών προβλημάτων.

Το κύριο σημείο εδώ είναι ότι, η λογική μπορεί να χρησιμοποιηθεί για την αναπαράσταση γνώσης που είναι συναφής με διάφορους τομείς, και μπορεί να διακινηθεί ή να αυτοματοποιηθεί με πολλούς τρόπους.

Για το σκοπό αυτό θα δούμε διάφορες πρακτικές εφαρμογές που θα βασίζονται στην χρήση του λογικού προγραμματισμού και ιδιαίτερα στη χρήση της γλώσσας Prolog.

Ο λογικός προγραμματισμός έχει τη δυνατότητα να εξυπηρετήσει κάθε υπολογιστικό πλαίσιο που δίνει περιθώρια για αυτοματοποιημένη συμπερασματολογία (mechanized reasoning).

Οι λόγοι για τους οποίους θα ασχοληθούμε με την κλασσική λογική είναι :

- η κλασσική λογική είναι πιο απλή και κατανοητή από τα άλλα είδη λογικής.
- μόνο αυτή έχει διερευνηθεί επαρκώς και με πληρότητα
- ως τώρα, μόνο η κλασσική λογική έχει ανταποκριθεί σε υπολογιστικούς σκοπούς με μεγάλο βαθμό επιτυχίας, και αυτό βέβαια λόγω της γλώσσας Prolog.

Βέβαια, η Κλασσική Λογική, και κατ'επέκταση ο λογικός προγραμματισμός, δεν καλύπτει τις ανάγκες αναπαράστασης της γνώσης.

Όμως αποτελεί ίσως την πιο κατάλληλη 'θύρα' για την ξενάγηση του αναγνώστη στον τεράστιο από πλευράς θεωριών και εφαρμογών κόσμο της Τεχνητής Νοημοσύνης.

Υπάρχουν δύο κύριοι τρόποι προσέγγισης του μαθηματικού περιεχομένου της λογικής :

- η θεωρία μοντέλου (model theory) και
- η θεωρία απόδειξης (proof theory).

Η θεωρία μοντέλου εξετάζει την σχέση μεταξύ λογικών προτάσεων αφού έχουν ερμηνευθεί, με την βοήθεια εξωτερικών στοιχείων, με την βοήθεια δηλαδή της απόδοσης τιμών αλήθειας.

Το λεξιλόγιο της βασικής θεωρίας μοντέλου περιλαμβάνει όρους όπως:

- αλήθεια (true),
- ψεύδος (false),
- ερμηνεία (interpretation),
- ικανοποίηση (satisfaction),
- μοντέλο (model),
- λογικό συμπέρασμα (logical consequence) ή σημασιολογική συνέπεια (semantic consequence).

Η θεωρία απόδειξης μελετά τις σχέσεις μεταξύ προτάσεων, όσον αφορά την δυνατότητα παραγωγής τους από άλλες προτάσεις, μέσω κανόνων που ενεργούν μόνο πάνω στη σύνταξη των προτάσεων αυτών.

Το λεξιλόγιο της θεωρίας απόδειξης χρησιμοποιεί όρους όπως:

- αξίωμα (axiom) ,
- κανόνας συμπερασμού (inference rule),
- θεώρημα (theorem) ,
- απόδειξη (proof),
- συνέπεια (consistency) και
- λογικό συμπέρασμα ή
- συντακτική συνέπεια (syntactic consequence).

Και οι δύο προσεγγίσεις είναι σημαντικές για την κατανόηση του λογικού προγραμματισμού, και θα δούμε ότι στην κλασσική λογική υπάρχουν απλές αλλά ουσιώδεις σχέσεις μεταξύ τους.

Η πιο ισχυρή σχέση είναι ότι τα στοιχεία που θέλουμε να αληθεύουν θα πρέπει να συμπίπτουν με εκείνα που μπορούμε να αποδείξουμε, δηλαδή, οι απαντήσεις που υπονοούνται από ένα πρόγραμμα να συμπίπτουν με τις απαντήσεις που υπολογίζονται απ' αυτό.

Η απαίτηση της εφαρμογής επιτάσσει σαν είδος προγραμματισμού τον διαδικτυακό αντικειμενοστραφή προγραμματισμό. Για αυτό χρειάζονται τα παρακάτω εργαλεία

2 Ανάλυση -Σχεδιαμός Εφαρμογής

2.1 Απαιτήσεις από το Λογισμικό

Μια λειτουργία που θα πρέπει το λογισμικό να επιτελεί ή μια συνθήκη που θα πρέπει να ικανοποιεί όταν θα έχει ολοκληρωθεί η κατασκευή του

- αφορούν τη συμπεριφορά του λογισμικού προς το εξωτερικό
- του περιβάλλον (χρήστης, άλλες εφαρμογές, λογισμικού) και
- όχι εσωτερικά του στοιχεία

Λειτουργικές απαιτήσεις: περιγράφουν τις εργασίες

- (λειτουργίες) που θα πρέπει να εκτελεί το λογισμικό
- καθορίζουν τη συμπεριφορά του συστήματος, δηλ. την
- απόκριση που πρέπει να εμφανίζει στο περιβάλλον του όταν
- ισχύουν συγκεκριμένες συνθήκες

Μη λειτουργικές απαιτήσεις: περιγράφουν χαρακτηριστικά

- που πρέπει να έχει το λογισμικό τα οποία δεν αφορούν την
- εκτέλεση κάποιας λειτουργίας από αυτό
- καθορίζουν ιδιώματα εμφάνισης (αισθητική, επικοινωνία με
- το χρήστη), επιδόσεων (αξιοπιστία, χρόνος εκτέλεσης, χρήση πόρων), υλοποίησης, κ.τ.λ.

2.2 Λειτουργικές Απαιτήσεις

Οι λειτουργικές απαιτήσεις αφορούν στην ανάπτυξη μίας εφαρμογής η οποία θα ικανοποιεί τις ακόλουθες απαιτήσεις :

Τα προγράμματα καταχώρησης που θα αφορούν στα ακόλουθα αντικείμενα :

Σύστημα εγγραφής και πιστοποίησης χρηστή. Τα σύστημα θα μπορεί να εγγράφεται ο εκαστοτέ φοιτητής και το σύστημα να τον αναγνωρίζει.

Σύστημα επιλογής εργαστηρίων και ομάδων. Ο φοιτητής θα είναι σε θέση να δηλώνει την ομάδα και το εργαστήριο που επιθυμεί.

Σύστημα παρουσιολογίου. Ο φοιτητής θα μπορεί να δηλώνει παρόν στην ομάδα που δήλωσε.

Σύστημα διαχείρισης. Ο καθηγητής θα μπορεί να δημιουργεί, τροποποιεί και διαγραφεί κατά βούληση, ομάδες, εργαστήρια και φοιτητές.

Σύστημα επικύρωσης. Ο καθηγητής θα επικυρώνει την παρουσία του φοιτητή.

2.3 Μη Λειτουργικές Απαιτήσεις

2.3.1 Απαιτήσεις χρήσης

Το λογισμικό θα πρέπει να περιέχει φιλικό περιβάλλον προς τον χρήστη με υποστήριξη γραφικών και συνδυασμό χρήσης ποντικιού και πληκτρολογίου. Θα υπάρχουν φόρμες καταχώρησης στοιχείων με αυτοματοποίηση πεδίων όπου αυτό επιτρέπεται.

2.3.2 Απαιτήσεις αξιοπιστίας

Η πρόσβαση από πολλούς χρήστες ταυτόχρονα προστατεύεται με δυνατότητα να χρησιμοποιούν την εφαρμογή με δικό τους κωδικό χρήστη και password. Το λογισμικό θα υποστηρίζει «κλείδωμα» των εγγραφών κάθε χρήστη ώστε να μην υπάρχουν προβλήματα λάθους κατά τη διάρκεια ενημέρωσης από άλλο χρήστη.

2.3.3 Απαιτήσεις επιδόσεων

Θα υπάρχει αξιοποίηση των πόρων του συστήματος, όποιων κι αν είναι αυτοί.. Επιπλέον, η απόκριση από το σύστημα βάσεων δεδομένων δεν θα ξεπερνά τα 2 δευτερόλεπτα.

2.3.4 Απαιτήσεις υποστήριξης

Το λογισμικό θα διατίθεται σε περιβάλλον Windows και θα εγκαθίσταται ιδιαίτερα εύκολα σε οποιοδήποτε μηχάνημα με υποστήριξη και αρχείου README το οποίο θα επεξηγεί και θα διευκρινίζει τα θέματα εγκατάστασης και λειτουργίας της εφαρμογής.

2.3.5 Απαιτήσεις υλοποίησης

Θα χρησιμοποιηθεί η γλώσσα PHP λόγω της δυνατότητας ανάπτυξης διαπλατφορμικών εφαρμογών με αντικειμενοστραφή προγραμματισμό σε συνδυασμό με χρήση SQL για υποβολή ερωτημάτων προς τη Βάση Δεδομένων.

2.3.6 Απαιτήσεις Βάσεων Δεδομένων

Θα χρησιμοποιηθεί σύστημα Βάσεων Δεδομένων με τα ακόλουθα βασικά αρχεία/πίνακες : Αρχείο φοιτητών (με τα στοιχεία του κάθε φοιτητή), Αρχείο καθηγητών (με τα στοιχεία του κάθε καθηγητή), Αρχείο μαθημάτων (με τα στοιχεία των υπαρχόντων μαθημάτων)

2.3.7 Φυσικές απαιτήσεις

Το λογισμικό αρχικά θα εγκατασταθεί σε έναν server με ελάχιστη απαίτηση Pentium IV, 128 MB RAM και θα έχει την δυνατότητα εγκατάστασης σε πολλά τερματικά PCs (Pentium II) με λειτουργικό Windows NT στα οποία θα έχει εγκατασταθεί MySQL και Php.

2.4 Εισαγωγικά

Η ανάπτυξη ενός μοντέλου για ένα σύστημα λογισμικού που θα μπορέσει να χρησιμοποιηθεί στη βιομηχανία, προτού δομηθεί ή επανασχεδιαστεί το σύστημα, είναι τόσο απαραίτητο όσο το να υπάρχουν τα αρχιτεκτονικά σχέδια για ένα μεγάλο κτίσμα. Τα καλά μοντέλα είναι απαραίτητα για την επικοινωνία ανάμεσα στις ομάδες ενός έργου και επίσης για να υπάρχει μία ορθή αρχιτεκτονική του συστήματος. Δημιουργούμε μοντέλα των σύνθετων συστημάτων γιατί δεν μπορούμε να τα κατανοήσουμε στην ολότητά τους. Όσο αυξάνει η πολυπλοκότητα των συστημάτων, αυξάνει και η σημασία της ύπαρξης καλών τεχνικών μοντελοποίησης. Υπάρχουν και άλλοι παράγοντες για την επιτυχία ενός έργου, αλλά η ύπαρξη μίας πρότυπης, πλούσιας γλώσσας μοντελοποίησης είναι απαραίτητος παράγοντας. Μία γλώσσα μοντελοποίησης πρέπει να περιλαμβάνει:

- Στοιχεία μοντέλων – στοιχειώδεις έννοιες μοντελοποίησης και τη σημασιολογία τους.
- Συμβολισμό – οπτικές αναπαραστάσεις των στοιχείων μοντελοποίησης.
- Οδηγίες – τρόπους χρήσης στην πράξη.

Στην περίπτωση των όλο και πιο πολύπλοκων συστημάτων η οπτική αναπαράσταση και η μοντελοποίηση γίνονται απαραίτητες.

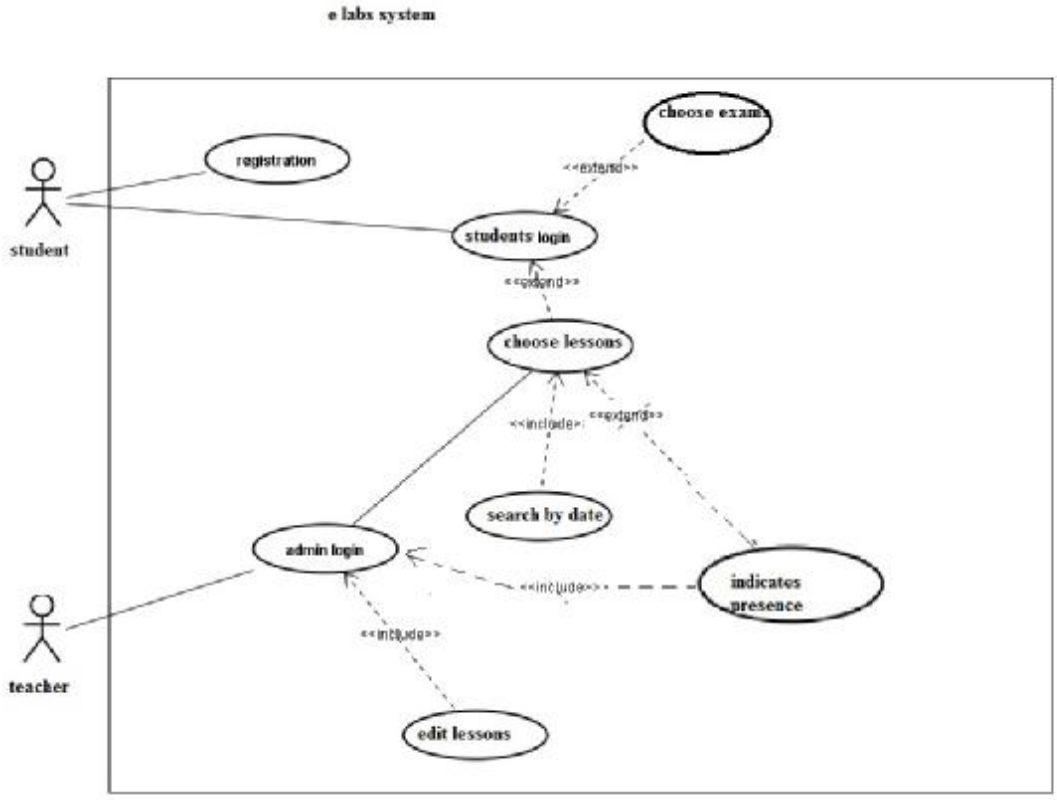
2.5 Use Case Diagram

Τα διαγράμματα περιπτώσεων χρήσης παρουσιάζουν τους actors και τις περιπτώσεις χρήσης ενός συστήματος μαζί με τις σχέσεις μεταξύ τους. Οι περιπτώσεις χρήσης αναπαριστούν λειτουργικότητα ενός συστήματος ή ενός classifier, όπως ένα υποσύστημα ή μία κλάση, όπως παρουσιάζεται σε όσους αλληλεπιδρούν με το σύστημα ή τον classifier, ενώ βρίσκονται εκτός του.

Μία *περίπτωση χρήσης* (use case) είναι ένα είδος classifier που αναπαριστά μία συνεπή μονάδα λειτουργικότητας που παρέχεται από το σύστημα, ένα υποσύστημα, ή μία κλάση, όπως παρουσιάζεται από ακολουθίες μηνυμάτων που ανταλλάσσονται ανάμεσα στο σύστημα και έναν ή περισσότερους εξωτερικούς χρήστες (οι οποίοι ονομάζονται *actors*) μαζί με τις πράξεις που θα πραγματοποιηθούν από το σύστημα.

Ένα *σημείο επέκτασης* (extension point) είναι μία αναφορά σε μία περιοχή μέσα στην περίπτωση χρήσης στην οποία μπορούν να εισαχθούν ακολουθίες πράξεων από άλλες περιπτώσεις χρήσης. Κάθε σημείο επέκτασης έχει ένα μοναδικό όνομα μέσα στην περίπτωση χρήσης και μία περιγραφή της περιοχής μέσα στη συμπεριφορά της περίπτωσης χρήσης.

2.5.1 Διάγραμμα



Το σύστημα ακολουθεί 2 σενάρια που εξελίσονται σειριακά (δηλαδή αφού ολοκληρωθεί το ένα ξεκινάει το άλλο) τα οποία είναι τα εξής:

Σενάριο 1: ΔΗΛΩΣΗ ΜΑΘΗΜΑΤΟΣ - ΕΡΓΑΣΤΗΡΙΟ

- Ο καθηγητής - διαχειριστής εισάγεται στο σύστημα και το σύστημα το αναγνωρίζει
- Ο καθηγητής - διαχειριστής καταχωρεί στο σύστημα τα εργαστήρια (ημερολογιακά και ώρες)
- Ο χρήστης εισάγεται στο σύστημα και το σύστημα το αναγνωρίζει
- Ο χρήστης θα διαλέγει σε ποιο εργαστήριο επιθυμεί να παρακολουθήσει
- Μόλις συμπληρώσει το μάθημα-τμήμα τον απαραίτητο αριθμό φοιτητών (έστω 20) θα κλείνει και θα γίνεται μη διαθέσιμό
- Ο φοιτητής είναι υποχρεωμένος να διαλέξει μια ομάδα βάσει το εξάμηνο του (ή την δήλωση)
- Σε περίπτωση που επιθυμεί άλλο τμήμα στο εργαστήριο του θα πρέπει να συνεννοηθεί με τον διδάσκοντα
- Ο καθηγητής έχει δικαίωμα να αλλάξει τους φοιτητές στα τμήματα

Σενάριο 2: ΠΑΡΟΥΣΙΟΛΟΓΙΟ

- ο φοιτητής θα εισάγεται στο μάθημα της ημέρας και θα δηλώνει την παρουσία του
- ο καθηγητής θα εισάγεται στο μάθημα της ημέρας και θα επικυρώνει την παρουσία του φοιτητή

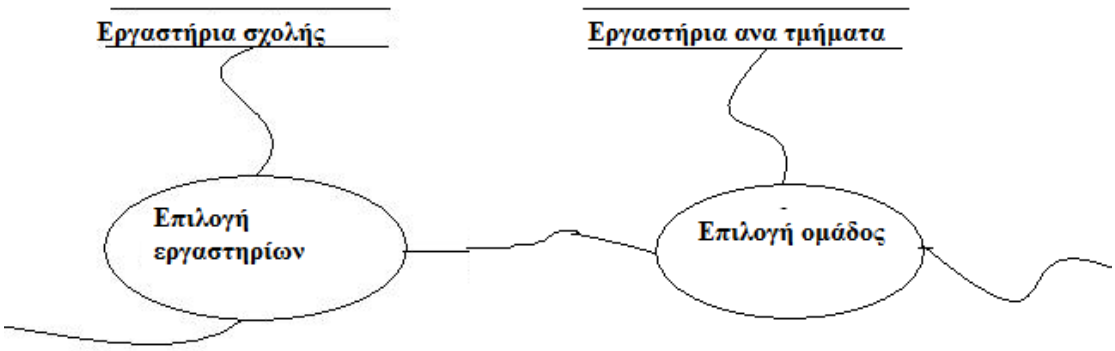
2.6 Διαγράμματα ροής δεδομένων

Τα Διαγράμματα Ροής Δεδομένων – ΔΡΔ (Data Flow Diagrams - DFDs) αναπαριστούν ένα σύστημα λογισμικού με βάση τα δεδομένα που παράγονται ή διακινούνται σ' αυτό. Αποτελούν μια λογική αναπαράσταση του συστήματος, χωρίς να περιέχουν πληροφορίες για το υλικό, το λογισμικό ή τα αρχεία που το αποτελούν. Για τους λόγους αυτούς, τα ΔΡΔ είναι κατάλληλα για την κατανόηση της λειτουργίας του συστήματος και από μη ειδικούς. Σε κάθε ΔΡΔ αναπαρίσταται η πορεία (ροή) που ακολουθεί κάθε τμήμα δεδομένων από το σημείο δημιουργίας του, έως το σημείο εξόδου από το σύστημα.

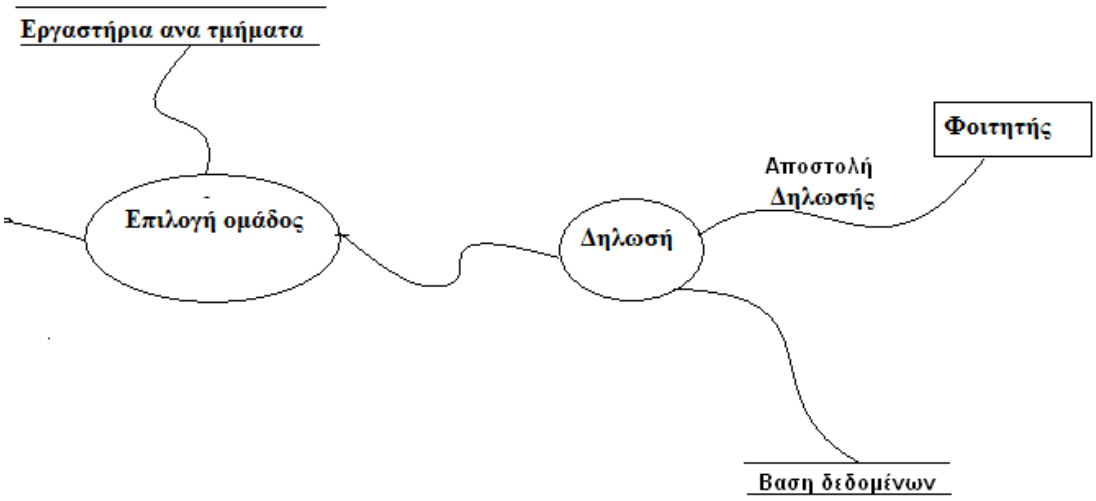
2.6.1 Διάγραμμα ροής δεδομένων σε επίπεδο χρήστη

2.6.1.1 Διάγραμμα ροής δεδομένων σε επίπεδο χρήστη – 1ο στάδιο

2.6.1.2Διάγραμμα ροής δεδομένων σε επίπεδο χρήστη – 2ο στάδιο



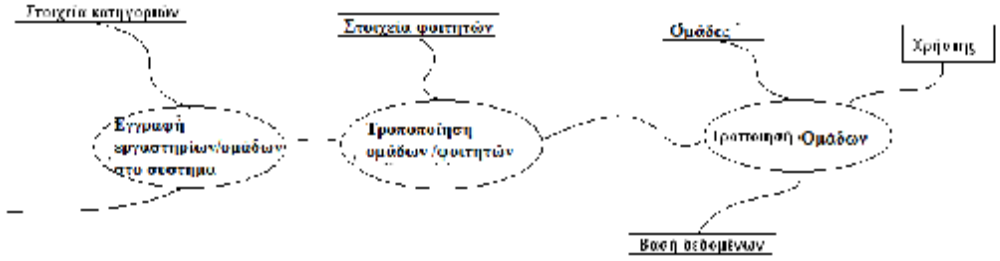
2.6.1.3Διάγραμμα ροής δεδομένων σε επίπεδο χρήστη – 3ο στάδιο



2.6.2 Διάγραμμα ροής δεδομένων σε επίπεδο διαχειριστή

2.6.2.1Διάγραμμα ροής δεδομένων σε επίπεδο διαχειριστή – 1ο στάδιο

2.6.2.2Διάγραμμα ροής δεδομένων σε επίπεδο διαχειριστή – 2ο στάδιο



2.7 Δεδομένα και Πληροφορία

Πριν προχωρήσουμε στην έννοια της Βάσης Δεδομένων πρέπει να διακρίνουμε μεταξύ των εννοιών δεδομένα (data) και πληροφορία (information). Από τα πρώτα χρόνια της χρήσης των Ηλεκτρονικών Υπολογιστών οι έννοιες δεδομένα και πληροφορία συγχέονταν μεταξύ τους. Έτσι τα δεδομένα μπορούν να είναι λέξεις π.χ. {“Νίκος”, “Μιχάλης”, “Μαρία”, “Θάλασσα”, “Αυτοκίνητο”}, αριθμοί, π.χ. {1, 2, 3, 5, 78}, συμβολοσειρές (strings) π.χ. {“Κώστας”, “5621”, “TP–882”, “6&5 #1”, “<>+–/*”}, ή σύμβολα που έχουν νόημα σε συγκεκριμένο πλαίσιο αναφοράς. Ένα τέτοιο παράδειγμα, δεδομένων είναι ψηφιοποιημένες εικόνες με κάποια καθορισμένη τυποποίηση. Αυτές οι εικόνες θα έχουν νόημα μόνο με τη χρήση ενός συγκεκριμένου προγράμματος απεικόνισης, δηλαδή μέσα στο πλαίσιο το προγράμματος απεικόνισης, ενώ έξω από αυτό είναι ακατανόητες συλλογές συμβόλων.

Έτσι όπως ορίσαμε τα δεδομένα είναι φανερό ότι είναι κατάλληλα για αποθήκευση σε ηλεκτρονικό υπολογιστή. Οι άνθρωποι, όμως, χρησιμοποιούσαν δεδομένα, πολύ πριν από την ανακάλυψη των ηλεκτρονικών υπολογιστών, στα πλαίσια ενός πεδίου

αναφοράς για την εξαγωγή συμπερασμάτων ή την λήψη αποφάσεων. Για παράδειγμα, από τα δεδομένα {85, 210, 515} και τα δεδομένα {“Κόρινθος”, “Πάτρα”, “Θεσσαλονίκη”}, αλλά και από τη γνώση (πεδίο αναφοράς) ότι τα πρώτα δεδομένα περιγράφουν αποστάσεις από την Αθήνα για τις πόλεις που ορίζονται στα δεύτερα δεδομένα, κάποιος μπορεί να καταλάβει ότι «η Πάτρα είναι πιο μακριά από την Αθήνα από ότι η Κόρινθος, αλλά πολύ κοντύτερα στην Αθήνα από ότι η Θεσσαλονίκη». Αυτό το συμπέρασμα είναι πληροφορία.

2.8 Βάση δεδομένων

Ένα βασικό χαρακτηριστικό των δεδομένων, έτσι όπως τα παρουσιάσαμε είναι ότι είναι κατάλληλα για να αποθηκευτούν σε ηλεκτρονικό υπολογιστή. Αυτό οδηγεί στο επόμενο βήμα που είναι η δημιουργία και χρήση Βάσεων Δεδομένων (databases).
Η σχετικότητα των δεδομένων είναι σημαντική για τον ορισμό μιας Βάσης Δεδομένων. Δεδομένα που δε σχετίζονται μεταξύ τους και απλά έχουν αποθηκευτεί σε έναν ηλεκτρονικό υπολογιστή δεν αποτελούν μια Βάση Δεδομένων. Μια Βάση Δεδομένων πρέπει να αντικατοπτρίζει ένα περιβάλλον του πραγματικού κόσμου. Τα δεδομένα που αποθηκεύονται στη Βάση Δεδομένων πρέπει να έχουν λογική συνέχεια και νόημα. Επίσης οι λειτουργίες που παρέχει η Βάση Δεδομένων είναι σημαντικές για τον ορισμό μιας Βάσης Δεδομένων. Συλλογές δεδομένων χωρίς τη δυνατότητα ενός αυτόματου, κοινού και κεντρικού τρόπου χειρισμού των δεδομένων αυτών, δεν αποτελούν Βάση Δεδομένων. Απλά είναι δεδομένα αποθηκευμένα στον ηλεκτρονικό υπολογιστή.

2.9 Σύστημα Διαχείρισης Βάσεις Δεδομένων.

Η εξέλιξη των Βάσεων Δεδομένων και οι ανάγκες για δημιουργία όλο και περισσότερων Βάσεων Δεδομένων, οδήγησαν στη δημιουργία των Συστημάτων Διαχείρισης Βάσεων Δεδομένων (Database Management Systems ή DBMS). Το Σύστημα Διαχείρισης Βάσεων Δεδομένων είναι ένα εργαλείο το οποίο διευκολύνει τους χρήστες να εργάζονται με Βάσεις Δεδομένων. Με τη χρήση του Συστήματος Διαχείρισης Βάσεων Δεδομένων οι χρήστες μπορούν να κατασκευάσουν και να χρησιμοποιήσουν Βάσεις Δεδομένων. Ακολουθεί ο ορισμός του Συστήματος Διαχείρισης Βάσεων Δεδομένων.

Ένα Σύστημα Διαχείρισης Βάσεων Δεδομένων κατά κανόνα «φιλοξενεί» πολλές Βάσεις Δεδομένων που έχουν κατασκευαστεί από διαφορετικούς χρήστες. Οι δυνατότητες που παρέχει ένα Σύστημα Διαχείρισης Βάσεων Δεδομένων στους χρήστες συνοψίζονται στις παρακάτω:

- Ορισμός της Βάσης Δεδομένων
- Κατασκευή της Βάσης Δεδομένων
- Διαγραφή της Βάσης Δεδομένων
- Χρήση της Βάσης Δεδομένων

Στον **ορισμό** της Βάσης Δεδομένων ο χρήστης μπορεί να καθορίσει το μοντέλο της Βάσης Δεδομένων, να ορίσει τους τύπους δεδομένων που θα χρησιμοποιήσει και να ελέγξει τη Βάση Δεδομένων χωρίς να προχωρήσει σε κατασκευή της. Στην **κατασκευή** της Βάσης Δεδομένων οι τύποι των δεδομένων και τα δεδομένα αποθηκεύονται στο υλικό (hardware) του ηλεκτρονικού υπολογιστή με διαδικασίες που ελέγχονται από το Σύστημα Διαχείρισης Βάσεων Δεδομένων και δεν απασχολούν τον χρήστη. Στη **διαγραφή** της Βάσης Δεδομένων ο χρήστης αποφασίζει τον τερματισμό μιας Βάσης Δεδομένων και την απομάκρυνση των δεδομένων από το υλικό. Τέλος, κατά τη **χρήση** της Βάσης Δεδομένων ο χρήστης, είτε χειρίζεται τα δεδομένα (εισάγει νέα δεδομένα, τροποποιεί δεδομένα, ή διαγράφει δεδομένα), είτε υποβάλει ερωτήσεις στη Βάση Δεδομένων με στόχο την εξαγωγή πληροφοριών.

2.10 Διαφορά βάσης δεδομένων με σύστημα διαχείρισης βάσεων δεδομένων

Σε αυτό το σημείο πρέπει να είναι ξεκάθαρη σε εσάς η διαφορά μεταξύ ενός Συστήματος Διαχείρισης Βάσεων Δεδομένων και μίας Βάσης Δεδομένων. Το Σύστημα Διαχείρισης Βάσεων Δεδομένων είναι λογισμικό το οποίο διευκολύνει τους χρήστες να υλοποιήσουν Βάσεις Δεδομένων. Αντίθετα η Βάση Δεδομένων υλοποιείται με τη βοήθεια ενός Συστήματος Διαχείρισης Βάσεων Δεδομένων που χειρίζεται δεδομένα τα οποία αποθηκεύονται στο υλικό του υπολογιστή. Η όλη διαδικασία χειρισμού και αποθήκευσης των δεδομένων διευκολύνεται από το Σύστημα Διαχείρισης Βάσεων Δεδομένων, το οποίο αναλαμβάνει τη μετατροπή των εντολών του χρήστη σε εντολές προς τον υπολογιστή και λειτουργίες που σχετίζονται με το χειρισμό των δεδομένων στο υλικό του υπολογιστή.

Ο χρήστης μιας Βάσης Δεδομένων επιτελεί λειτουργίες που μπορούν να ομαδοποιηθούν στις παρακάτω: εισαγωγή δεδομένων, διαγραφή δεδομένων, αλλαγή δεδομένων και ανάκληση δεδομένων. Όλες οι παραπάνω λειτουργίες προϋποθέτουν επικοινωνία με το υλικό του υπολογιστή (π.χ. το μαγνητικό δίσκο του υπολογιστή). Αυτή η επικοινωνία γίνεται μέσω του λογισμικού του Συστήματος Διαχείρισης Βάσεων Δεδομένων. Η επικοινωνία αυτή συνήθως δεν είναι ορατή από τον χρήστη της Βάσης Δεδομένων, ο οποίος απλά βλέπει τα αποτελέσματα από τις λειτουργίες που ζήτησε να επιτελούνται. Αυτό είναι και το σημείο που κάνει αρκετούς χρήστες να συγχέουν τις έννοιες Βάση Δεδομένων και Σύστημα Διαχείρισης Βάσεων Δεδομένων.

2.11 Γνωστά Συστήματα βάσεων Δεδομένων

Εμπορικά

- Oracle
- IBM/DB2
- MS SQL-server
- Sybase
- Informix
- (MS Access, ...)

Ελεύθερο Λογισμικό- Open Source

- Postgres (UCB)
- mySQL, mSQL
- miniBase (Wisc)
- Predator (Cornell)

Εμείς θα χρησιμοποιήσουμε την mysql

Μελέτη περίπτωσης

3 Υλοποίηση εφαρμογής

3.1 Περιγραφή των εργαλείων που χρησιμοποιήθηκαν.

3.1.1 WAMP 2.0

Το wamp είναι ένα ολοκληρωμένο πακέτο Server που περιλαμβάνει apache, php, mysql, ftp server, υποστήριξη SSL και φυσικά όλα αυτά με αυτοματοποιημένη εγκατάσταση και ρύθμιση.
Στην εφαρμογή μας χρησιμοποιήσαμε την έκδοση Wamp 2.0 (Apache 2.2.11-MySQL 5.1.36 - PHP 5.3.0)

3.1.2 PhpMyAdmin

Το PhpMyAdmin είναι ένα εργαλείο γραμμένο σε Php το οποίο διαχειρίζεται την MySQL στο δίκτυο. Μπορεί να χειρίζεται πλήρως βάσεις δεδομένων, πίνακες, πεδία πινάκων αλλά και ολόκληρο τον MySQL Server. Υποστηρίζει 47 γλώσσες μεταξύ των οποίων και τα Ελληνικά και είναι λογισμικό ανοιχτού κώδικα.

3.1.2.1 Δυνατότητες του Php MyAdmin

Το PhpMyAdmin μπορεί να:

1. Δημιουργεί και να διαγράφει βάσεις δεδομένων
2. Δημιουργεί, τροποποιεί, διαγράφει, αντιγράφει και μετονομάζει πίνακες
3. Κάνει συντήρηση της βάσης
4. Προσθέτει, διαγράφει και τροποποιεί πεδία πινάκων
5. Εκτελεί Sql ερωτήματα, ακόμα και ομαδικά (batch)
6. Διαχειρίζεται κλειδιά σε πεδία
7. “Φορτώνει” αρχεία κειμένου σε πίνακες
8. Δημιουργεί και διαβάζει πίνακες (που προέρχονται από dump βάσης)
9. Εξάγει δεδομένα σε μορφή CVS, Latex, XML
10. Διαχειρίζεται πολλούς διακομιστές
11. Διαχειρίζεται τους χρήστες MySQL και τα δικαιώματά τους
12. Ελέγχει την αναφορική ακεραιότητα των δεδομένων των MyISAM πινάκων
13. Δημιουργεί PDF γραφικών του layout της βάσης δεδομένων
14. Εκτελεί αναζητήσεις σε όλη τη βάση ή μέρος αυτής
15. Υποστηρίζει πίνακες InnoDB και ξένα κλειδιά
16. Υποστηρίζει MySQLi, μια βελτιωμένη επέκταση της MySQL

3.1.3 JavaScript

JavaScript είναι μια νέα scripting γλώσσα, η οποία αναπτύσσεται από την Netscape. Με το JavaScript μπορείς εύκολα να αναπτύξεις μια interactive σελίδα (μια σελίδα

δηλαδή που αλληλεπιδρά στον εαυτό της). Αυτό το tutorial δείχνει τι μπορεί να γίνει μέσω της JavaScript - και πιο σημαντικό *πώς* μπορεί να γίνει.

3.2 Κύρια χαρακτηριστικά του MySQL

Η παρακάτω λίστα περιγράφει μερικά από τα πιο σημαντικά χαρακτηριστικά του MySQL :

- Φορητότητα.
- Καμία διαρροή μνήμης.
- Δουλεύει σε πολλές διαφορετικές πλατφόρμες.
- API για C, C++, Java, Perl, PHP, Python και TCL.
- Μπορεί εύκολα να χρησιμοποιήσει πολλαπλές CPU αν είναι διαθέσιμες.
- Πολύ γρήγορα B-tree πίνακες με συμπίεση δεικτών.
- Πολύ γρήγορο σύστημα κατανομής μνήμης.
- Οι SQL λειτουργίες εφαρμόζονται μέσω μιας ιδιαίτερα βελτιστοποιημένης βιβλιοθήκης και πρέπει να είναι όσο το δυνατό γρήγορη.

Τύποι στηλών :

1. Πολλοί τύποι στηλών : υπογεγραμμένοι / ανυπόγραφοι ακέραιοι αριθμοί FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET και ENUM τύποι.

2. Καθορισμένου και μεταβλητού μήκους εγγραφές. 3. Όλες οι στήλες έχουν προκαθορισμένες τιμές. Μπορείτε να χρησιμοποιήσετε το INSERT για να εισάγετε ένα υποσύνολο στηλών ενός πίνακα, αυτές οι στήλες που δεν δίνονται ρητά τιμές, τίθενται στις προκαθορισμένες τιμές.

Εντολές και λειτουργίες :

1. Πλήρης υποστήριξη στα τμήματα SELECT και WHERE των ερωτήσεων.

2. Πλήρης υποστήριξη στις SQL προτάσεις GROUP BY και ORDER BY.

Υποστήριξη λειτουργιών ομαδοποίησης COUNT(), COUNT(DISTINCT ...), AVG(), STD(), SUM(), MAX() και MIN().

3. Υποστήριξη για LEFT OUTER JOIN και RIGHT OUTER JOIN με ANSI SQL και ODBC σύνταξη.

4. Τα ψευδώνυμα στους πίνακες και στις στήλες επιτρέπονται στο πρότυπο SQL92.

5. Τα DELETE, INSERT, REPLACE και UPDATE επιστρέφουν τον αριθμό των γραμμών που άλλαξαν.

6. Η εντολή SHOW μπορεί να χρησιμοποιηθεί για να ανακτηθούν πληροφορίες σχετικές με τη βάση δεδομένων, τους πίνακες και τους δείκτες.
7. Η εντολή EXPLAIN μπορεί αν χρησιμοποιηθεί για να καθορίσει πως επιλύεται μια ερώτηση.
8. Τα ονόματα των λειτουργιών δεν έρχονται σε σύγκρουση με τα ονόματα των πινάκων ή των στηλών. Για παράδειγμα το ABS είναι ένα έγκυρο όνομα στήλης. Ο μόνος περιορισμός είναι το ότι σε μια κλήση λειτουργίας δεν πρέπει να υπάρχει κενό μεταξύ του ονόματος της λειτουργίας και του ΄(΄ που ακολουθεί.
9. Μπορείτε να αναμίξετε πίνακες από διαφορετικές βάσεις δεδομένων στο ίδιο ερώτημα.

- Εξελιξιμότητα και όρια : 1. Διαχειρίζεται μεγάλες βάσεις δεδομένων. Το MySQL χρησιμοποιείτε σε βάσεις που έχουν 50.000.000 εγγραφές και υπάρχουν χρήστες με βάσεις που έχουν 60.000 πίνακες και 5.000.000.000 γραμμές.

2. Μέχρι 32 δείκτες ανά πίνακα επιτρέπονται. Κάθε δείκτες μπορεί να αποτελείτε από 1 ως 6 στήλες ή μέρος των στηλών. Το μέγιστο μέγεθος των δεικτών είναι 500 bytes.

- Συνδετικότητα :

1. Οι πελάτες μπορούν συνδεθούν στο MySQL server χρησιμοποιώντας το TCP/IP.
2. Υποστήριξη ODBC για Win32.

Εντοπισμός :

1. Ο server μπορεί να παρέχει μηνύματα λάθους στους πελάτες σε πολλές γλώσσες.
2. Πλήρης υποστήριξη για πολλά διαφορετικά σύνολα χαρακτήρων, συμπεριλαμβανομένου το ISO-8859-1.
3. Όλα τα δεδομένα σώζονται στο επιλεγμένο σύνολο χαρακτήρων.
4. Η ταξινόμηση γίνεται σύμφωνα με το επιλεγμένο σύνολο χαρακτήρων (εξ ορισμού είναι ο Σουηδικός).

Είναι δυνατό να το αλλάξετε αυτό όταν τίθεται σε λειτουργία ο MySQL server.

Πελάτες και εργαλεία :

1. Περιλαμβάνει την myisamchk, μια πολύ γρήγορη λειτουργία για τον έλεγχο, βελτιστοποίηση και επισκευή πινάκων.

3.3 Πλεονεκτήματα και μειονεκτήματα του MySQL

3.3.1 Πλεονεκτήματα :

- Ταχύτητα : Ο κώδικας του πυρήνα της MySQL γράφτηκε από την αρχή μέχρι το τέλος με άριστη απόδοση ως αρχικός στόχος.
- Αξιοπιστία : Το MySQL έχει κερδίσει τη φήμη του ότι είναι σε θέση να τρέχει αφύλακτο για μέρες ακόμη και μήνες μετά από την αρχική οργάνωση.
- Χαμηλές απαιτήσεις σε πόρους συστήματος : Το MySQL είναι ικανό να κάνει το καλύτερο με τους πόρους που του δίνεται. Φυσικά, όσο περισσότεροι πόροι, τόσο καλύτερη απόδοση να περιμένετε, αλλά ελάχιστοι πόροι δεν θα βγάλουν το MySQL εκτός λειτουργίας.
- Εξελιξιμότητα : Η εμπειρία έδειξε ότι το MySQL αποδίδει καλά σε συστήματα με μέχρι 4 επεξεργαστές και μνήμη μέχρι 4GB εκμεταλλευόμενο πλήρως τους πόρους του συστήματος.
- Ποικιλομορφία πλατφόρμων : Τρέχει σε μια ευρεία ποικιλία υπολογιστικών συστημάτων. Ανάμεσα στα οποία τα πιο δημοφιλή είναι τα Linux, Windows, Solaris και FreeBSD.
- Υποστήριξη για έναν μεγάλο αριθμό γλωσσών : Όταν αναπτύσσετε μια εφαρμογή για τη βάση δεδομένων, μια από τις αρχικές ανησυχίες είναι η διασύνδεση με τον server βάσεων δεδομένων χρησιμοποιώντας κάποια γλώσσα προγραμματισμού.

Αυτός είναι ακόμη ένας τομέας της δύναμης του MySQL. Οι προγραμματιστές μπορούν να επικοινωνήσουν με το MySQL χρησιμοποιώντας C/C++, PHP, Perl, Java, Python, TCL, Ruby και Eiffel.

- Υποστήριξη ODBC : Περιλαμβάνει οδηγό ODBC. Αυτό δίνει τη δυνατότητα στους προγραμματιστές να γράψουν εφαρμογές χρησιμοποιώντας Open Database Connectivity. Το ODBC επιτρέπει στο MySQL να χρησιμοποιηθεί στο Microsoft Access, Excel και άλλα. Επίσης επιτρέπει στο MySQL να χρησιμοποιηθεί σε εφαρμογές σε Visual Basic και Delphi, στην ASP και σε άλλα εργαλεία και περιβάλλοντα ανάπτυξης εφαρμογών.
- Δωρεάν ή χαμηλού κόστους χορήγηση αδειών : Το MySQL διανέμεται κάτω από την άδεια GPL (General Public License). Αυτή η άδεια σας επιτρέπει να το χρησιμοποιείτε και για εμπορικό και για μη εμπορικό σκοπό υπό την

προϋπόθεση ότι οποιοδήποτε παραγόμενο προϊόν πρέπει να διανεμηθεί με ολόκληρο τον πηγαίο κώδικα κάτω από τους όρους της ίδιας άδειας.

- Ανέξοδη εμπορική υποστήριξη : Για όσους σκέφτονται να τρέξουν το MySQL σε ένα περιβάλλον, το ζήτημα υψηλής ποιότητας εμπορικής υποστήριξης είναι πολύ σημαντικό. Το MySQL AB παρέχει ένα ευρύ φάσμα εμπορικής υποστήριξης σε λογική τιμή, συμπεριλαμβανομένου 24x7 τηλεφωνική υποστήριξη.
- Ισχυρή υποστήριξη κοινοτήτων χρηστών : Όπως αναφέρθηκε πιο πάνω το MySQL δεν είναι απλά μια βάση δεδομένων. Οι ιδρυτές του MySQL έχουν εστιάσει τη προσοχή τους στο να δίνουν στη κοινότητα περισσότερα από αυτά που παίρνουν από αυτήν. Η κοινότητα ανταποκρίθηκε με αφοσίωση, σκληρή δουλειά και συναδελφοσύνη. Έτσι μπορεί κάποιος χρήστης να βρει τις απαντήσεις στα προβλήματα που δεν μπορεί να επιλύσει λαμβάνοντας βοήθεια από τη κοινότητα, πέρα από την βοήθεια που υποστηρίζει το MySQL. Μια άλλη πτυχή της ισχυρής κοινότητας είναι το ότι μπορεί να βρει κανείς ένα ειδικό στο MySQL για να εργαστεί για εσάς.
- Διαθεσιμότητα του πηγαίου κώδικα : Η πρόσβαση στον πηγαίο κώδικα είναι κάτι πολύ σημαντικό για τις επιχειρήσεις που απασχολούν πεπειραμένους C/C++ προγραμματιστές. Παρέχει την ευκαιρία να γίνουν διάφορες προσαρμογές, βελτιώσεις, επεκτάσεις και διόρθωση λαθών, χωρίς να περιμένεις των προμηθευτή να το κάνει.

3.3.2 **Αδυναμίες :**

- Έλλειψη ορισμένων χαρακτηριστικών γνωρισμάτων της SQL : Η πιο σοβαρή αδυναμία του MySQL είναι ότι αυτή τη περίοδο δεν υποστηρίζει εμφωλευμένα ερωτήματα, όψεις, αποθηκευμένες διαδικασίες, εναύσματα και επιβολή ξένου κλειδιού.
- Έλλειψη λεπτομερούς δοκιμής σε συγκεκριμένες πλατφόρμες : Το MySQL AB μπορεί να εξασφαλίσει ως ένα επίπεδο την ποιότητα του MySQL σε μια πλατφόρμα από εκεί και μετά, η ποιότητα αυτή θα αυξάνεται όσοι περισσότεροι χρησιμοποιούν το MySQL στη πλατφόρμα αυτή και αποστέλλουν τα σφάλματα που βρίσκουν. Επομένως η εξασφάλιση της ποιότητας εξαρτάτε κατά ένα μεγάλο μέρος από τις δοκιμές που κάνουν οι χρήστες, οπότε όσοι περισσότεροι χρήστες τόσο λιγότερα σφάλματα θα

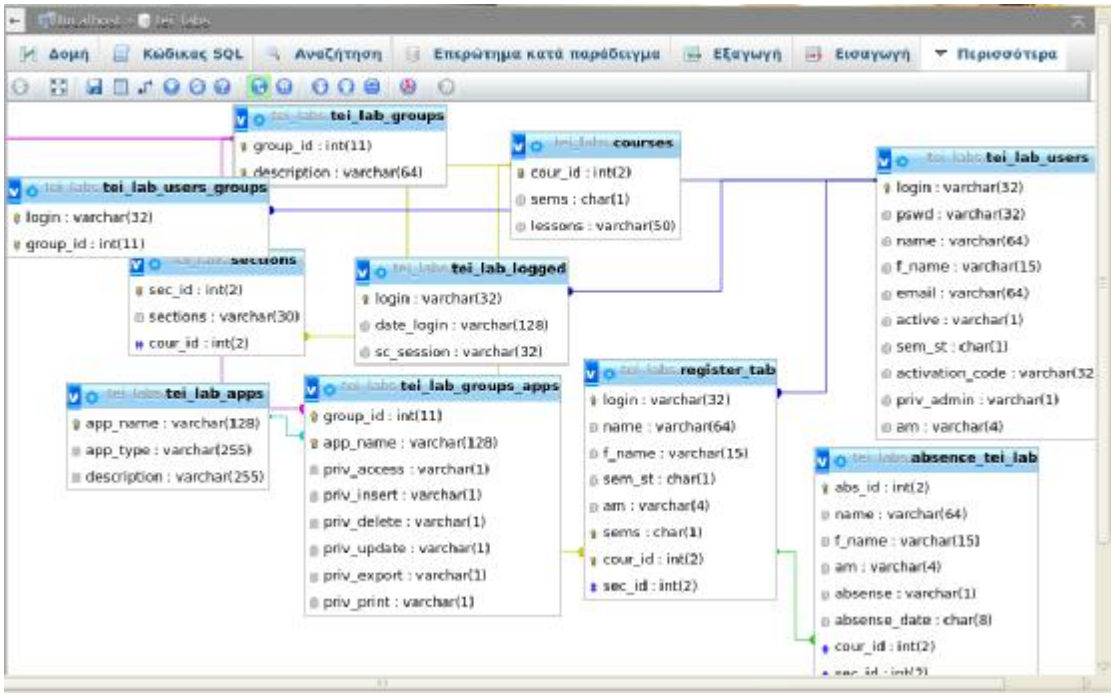
περάσουν απαρατήρητα, για μεγάλο χρονικό διάστημα. Βέβαια αυτό δεν σημαίνει ότι σε πλατφόρμες που δεν χρησιμοποιούν πολλοί χρήστες το MySQL δεν θα τρέχει και μάλιστα καλά, αλλά ίσως να υπάρξουν κάποια προβλήματα.

- Δυσκολία εργασίας με τον πηγαίο κώδικα του server : Το δύσκολο του να δουλεύει κανείς με τον πηγαίο κώδικα που έγραψε κάποιος άλλος είναι ότι πρέπει να τον καταλάβει για να υπάρχει κάποια πρακτική αξία. Ο κώδικας του MySQL server είναι πολύ δύσκολο να το καταλάβει ακόμη και ένας ειδικευμένος πεπειραμένος προγραμματιστής της C/C++. Υπάρχουν δυο κύριοι λόγοι της δυσκολίας αυτής. Πρώτα, είναι ένας server βάσεων δεδομένων, το οποίο σημαίνει ότι πρέπει αν έχει το κώδικα για να οργανώσει τα δεδομένα στο σκληρό δίσκο και να επιλέξει μια στρατηγική για την επίλυση ενός συγκεκριμένου ερωτήματος.

3.4 Περιγραφή της βάσης δεδομένων του συστήματος

Η βάση δεδομένων του συστήματος αποτελείτε από τους εξής πίνακες:

- Τον πίνακα courses που περιλαμβάνει όλα τα στοιχεία των μαθημάτων.
- Τον πίνακα register_tab που περιλαμβάνει τον τρόπο εγγραφής και όλα τα στοιχεία όλων των χρηστών (καθηγητών και φοιτητών).
- Τον πίνακα absense_date για το απουσιολόγιο των φοιτητών.
- Τον πίνακα tei_lab_apps για της εφαρμογές και λειτουργίες του συστήματος.
- Τον πίνακα tei_lab_groups για την ομαδοποίηση των χρηστών και την κατηγοριοποίηση τους.
- Τον πίνακα tei_lab_groups_apps για την αντιστοίχηση των χρηστών σε εφαρμογές του συστήματος.
- Καθώς και αρκετούς βοηθητικούς πίνακες.



Εικόνα 9 Βάση Δεδομένων

3.4.1 Scriptcase 4

Το ScriptCase είναι μια ολοκληρωμένη γεννήτρια κώδικα. Με φιλικό περιβάλλον, το ScriptCase δημιουργεί τις εφαρμογές Ιστού εξαιρετικά γρήγορα με ποιότητα και ολοκληρωμένες, κερδίζοντας χρόνο, χαμηλώνοντας τις δαπάνες και αυξάνοντας την παραγωγικότητα. Το Scriptcase υποστηρίζει τις περισσότερες χρησιμοποιημένες βάσεις δεδομένων, όπως τη Oracle, DB2, MS SQLServer, MySQL, PostgreSQL, Sybase, MS Access καθώς και άλλες.

Ο πηγαίος κώδικας της εφαρμογής είναι PHP και JavaScript και χρησιμοποιεί την τεχνολογία AJAX. Οι εφαρμογές τρέχουν απολύτως ανεξάρτητα από το εργαλείο και είναι συμβατές με τα Windows, τη Unix, AS/400 και άλλα συστήματα. (Scriptcase)

3.4.2 AJAX

Η AJAX δεν είναι μια καινούρια γλώσσα, είναι μια διαφορετική χρησιμοποίηση τεχνολογιών που υπάρχουν εδώ και καιρό. Με τη τεχνολογία AJAX μπορείτε να δημιουργήσετε καλύτερο, γρηγορότερο και πιο φιλικές διαδικτυακές εφαρμογές για το τελικό χρήστη. Η τεχνολογία AJAX βασίζεται στη JavaScript και σε κλήσεις HTTP. Το πρώτο λοιπόν πράμα που θα μάθουμε είναι από πού έχει προκύψει ο όρος AJAX (προφέρεται ΑΓΙΑΞ), έχουμε και λέμε λοιπόν στα αγγλικά τα αρχικά AJAX = Asynchronous JavaScript And XML η στα ελληνικά *AJAX = Ασύγχρονη JavaScript και XML* Τι είναι το Ajax;

Η τεχνολογία AJAX δεν είναι μια νέα γλώσσα προγραμματισμού, αλλά απλά μία νέα τεχνική για τη δημιουργία πιο γρήγορων και πιο φιλικών για το χρήστη διαδικτυακών εφαρμογών. Η τεχνολογία AJAX χρησιμοποιεί JavaScript για την αποστολή και λήψη δεδομένων μεταξύ ενός web browser (φυλλομετρητή, πλοηγό διαδικτύου) και τον εξυπηρετητή (web server). Η τεχνική AJAX κάνει της ιστοσελίδες πιο διαδραστικές επιτρέποντας την αποστολή δεδομένων πίσω από τις σκηνές χωρίς να χρειαστεί ο

χρήστης να μεταφορτώνει μία ιστοσελίδα κάθε φορά που ο χρήστης έκανε κάτι στην σελίδα.

3.4.2.1Η τεχνολογία AJAX είναι μια τεχνολογία που τρέχει στους υπολογιστές μας

Η τεχνολογία AJAX τρέχει στους υπολογιστές που υπάρχει ο φυλλομετρητής. Χρησιμοποιεί ασύγχρονη μεταφορά δεδομένων (HTTP κλήσεις) μεταξύ του φυλλομετρητή και του κεντρικού εξυπηρετητή, επιτρέποντας της σελίδες web να ζητούν μικρές πληροφορίες από τον εξυπηρετητή αντί για πλήρες σελίδες. Η τεχνολογία AJAX είναι πολύ χρήσιμη μιας και επιτρέπει της διαδικτυακές εφαρμογές να είναι μικρές, να φορτώνονται γρήγορα και να είναι πολύ φιλικές για το τελικό χρήστη. Θα πρέπει να σημειωθεί πως η τεχνολογία AJAX είναι κυρίως τεχνολογία που υποστηρίζεται στο φυλλομετρητή (web browser) και όχι τεχνολογία του εξυπηρετητή web server.

3.4.2.2Η τεχνολογία AJAX βασίζεται σε ανοικτά πρότυπα

Η τεχνολογία AJAX βασίζεται σε ανοικτά πρότυπα και αυτό το κάνει πολύ εύχρηστο και ελκυστικό για εταιρείες ανάπτυξης λογισμικού. Μερικές από τις τεχνολογίες του AJAX είναι:

- JavaScript
- XML
- HTML
- CSS

Τα πρότυπα που προαναφέρουμε πως χρησιμοποιούνται από τη τεχνολογία AJAX είναι πλήρως καθορισμένα, και υποστηρίζονται πλήρως από τους πιο γνωστούς φυλλομετρητές. Η εφαρμογές AJAX είναι επίσης γνωστές ως Cross-Platform και Cross-Browser δηλαδή τεχνολογία που τρέχει σε όλες τις πλατφόρμες (λειτουργικά συστήματα) και σε όλους τους φυλλομετρητές. (www.coder.gr)

3.4.3 Macromedia Dreamweaver 8

Ο Dreamweaver MX 2004 αποτελεί το κορυφαίο πρόγραμμα δημιουργίας website και διαδικτυακών εφαρμογών, προσφέρει δυνατότητες δημιουργίας προηγμένων γραφικών, οπτικά εργαλεία περιγράμματος, χαρακτηριστικά ανάπτυξης εφαρμογών και υποστήριξη επεξεργασίας κώδικα μέσω ενός περιβάλλοντος τεχνολογίας ανεξαρτήτου πλατφόρμας.

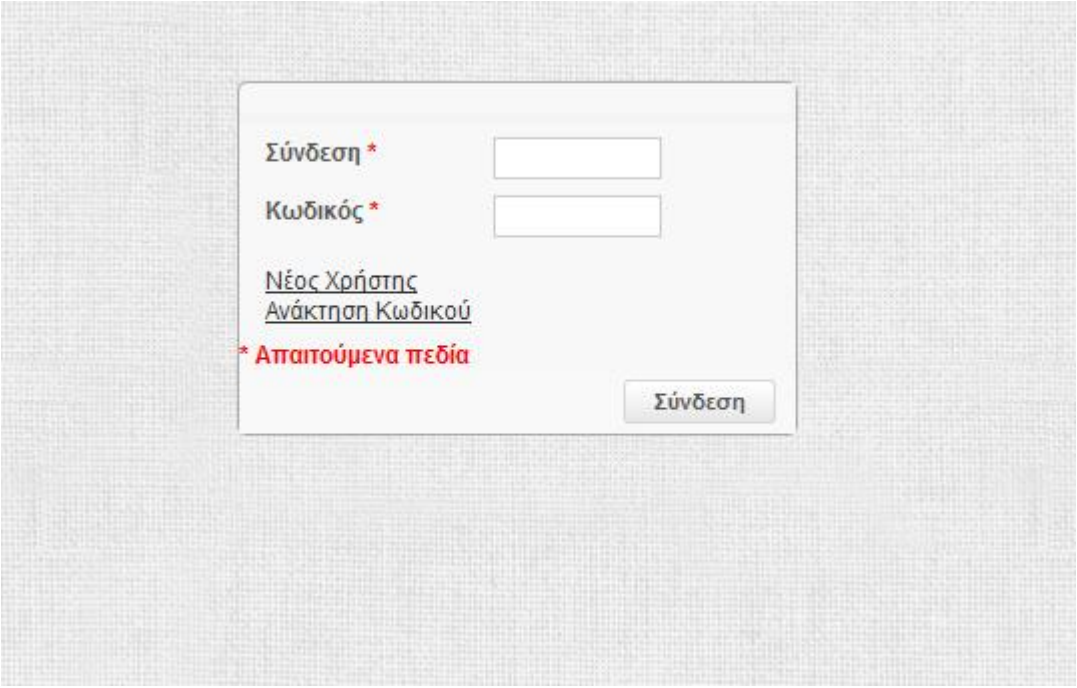
Με το εργαλείο αυτό δημιουργήσαμε τη homepage της σελίδας μας καθώς και ρυθμίσαμε τα μεγέθη των διαφόρων πεδίων ώστε να πετύχουμε ένα πιο ομοιόμορφο οπτικό αποτέλεσμα. (Dreamweaver)

4 Παρουσίαση εφαρμογής

Σε αυτήν την ενότητα ακολουθεί η περιγραφή του εγχειριδίου χρήσης της εφαρμογής, το οποίο χωρίζεται σε δύο μέρη όσα και τα επίπεδα των χρηστών, πελάτης, και διαχειριστής.

4.1 Εγχειρίδιο για τον φοιτητή

Ο φοιτητής εισάγεται με τα στοιχεία πρόσβασης που του έχει δώσει η γραμματεία. Σε περίπτωση που χάσει κάποιος τα στοιχεία πρόσβασης έχει δυνατότητα ανάκτησης του κωδικού βάσει του email του. Επίσης στην εφαρμογή υπάρχει δυνατότητα για δημιουργία νέων φοιτητών (δικαίωμα που το έχει μόνο η γραμματεία).

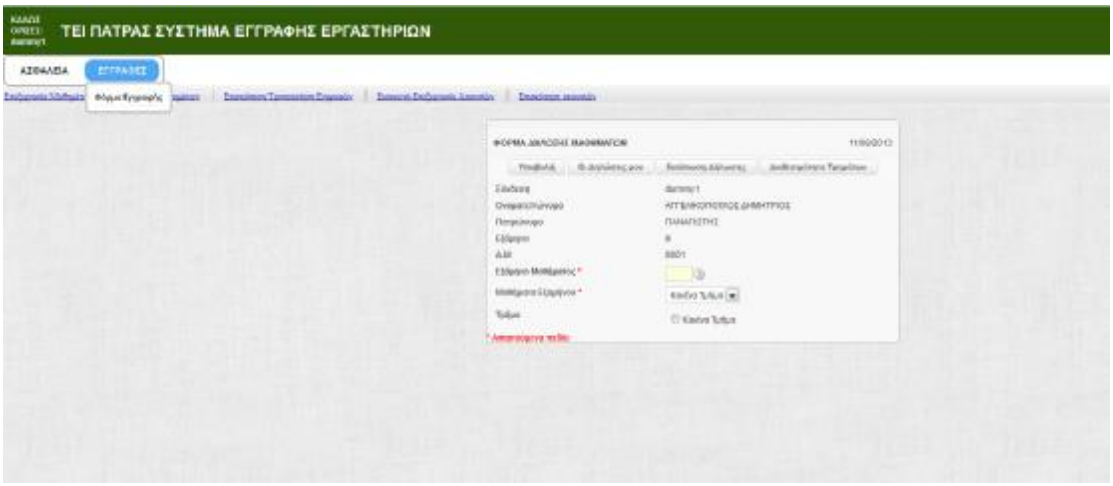


Εικόνα 10 Είσοδός στην εφαρμογή

Στην συνέχεια ο φοιτητής εισάγεται στην πλατφόρμα εγγραφών. Η πλατφόρμα εγγραφών είναι προσωποποιημένη με βάση τον χρήστη.

Εικόνα 11 Πλατφόρμα φοιτητή

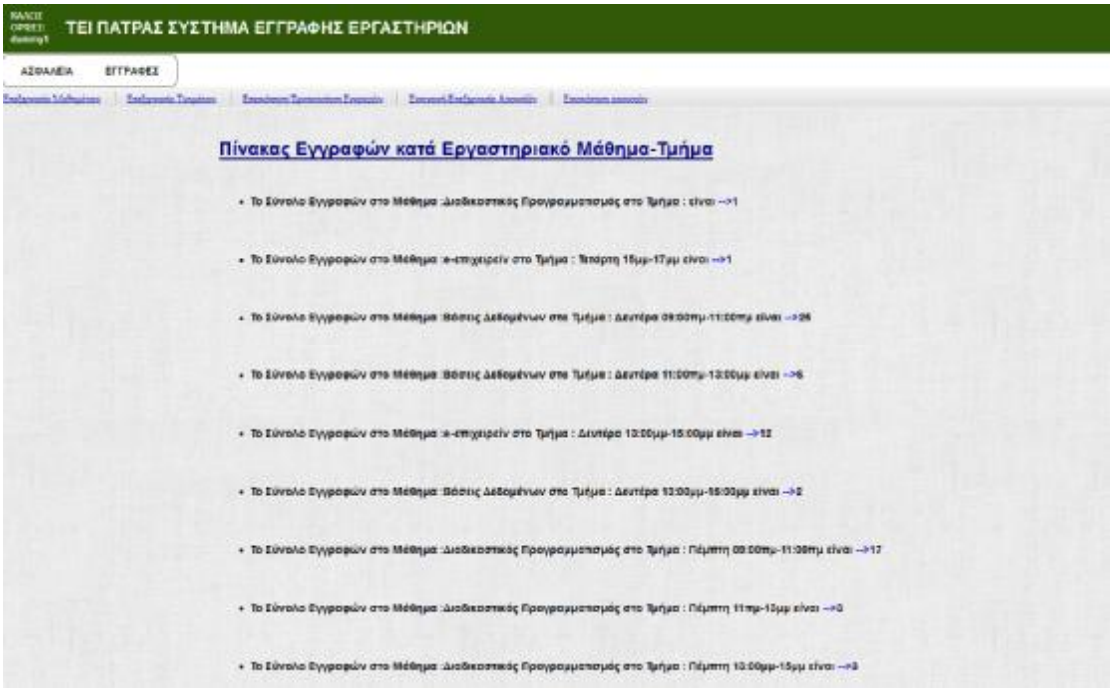
Εδώ ο φοιτητής έχει την εξουσιοδότηση να δει τις απουσίες του, να αλλάξει τον κωδικό του (στο μενού ασφάλεια) και να γραφτεί στα μαθήματα του εξαμήνου.



Εικόνα 12 Εγγραφή σε μαθήματα

Ο φοιτητής δηλώνει το εξάμηνο που θέλει και διαλέγει τα μαθήματα που επιθυμεί όπως και το τμήμα που επιθυμεί. Στην συνέχεια πατάει υποβολή να επιβεβαιώσει την επιλογή. Το σύστημα δεν αφήνει, τον φοιτητή, να δηλώσει μαθήματα μεγαλύτερου εξαμήνου από αυτό που είναι δηλωμένο στην γραμματεία.

Επίσης ο φοιτητής έχει δικαίωμα να εκτυπώσει την λίστα με τα μαθήματα καθώς και να ελέγξει την διαθεσιμότητα.



Εικόνα 13 Διαθεσιμότητα

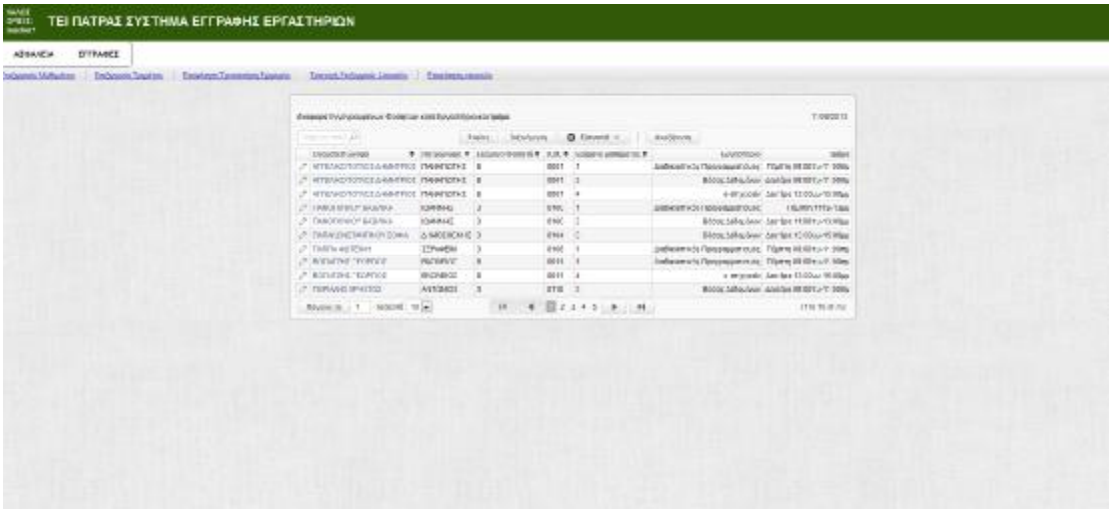
Στο μενού Ασφάλεια -> έξοδος αποσυνδέεται από το τμήμα.

4.2 Εγχειρίδιο για τον καθηγητή

Ο καθηγητής εισάγεται με τα στοιχεία πρόσβασης που του έχει δώσει η γραμματεία. Σε περίπτωση που χάσει κάποιος τα στοιχεία πρόσβασης έχει δυνατότητα ανάκτησης του κωδικού βάσει του email του. Επίσης στην εφαρμογή υπάρχει δυνατότητα για δημιουργία νέων καθηγητών (δικαίωμα που το έχει μόνο η γραμματεία).

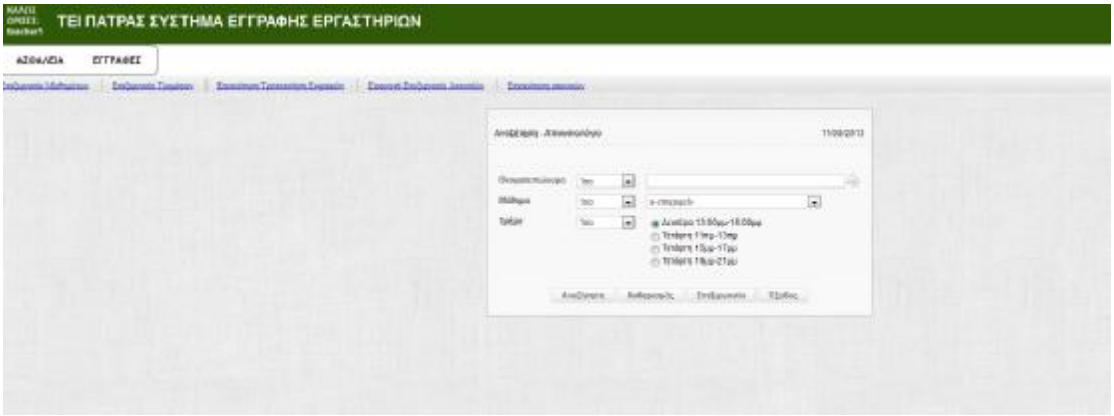
Εικόνα 14 Είσοδός στην εφαρμογή

Στην συνέχεια ο καθηγητής εισάγεται στην πλατφόρμα. Η πλατφόρμα εγγραφών είναι προσωποποιημένη με βάση τον χρήστη.



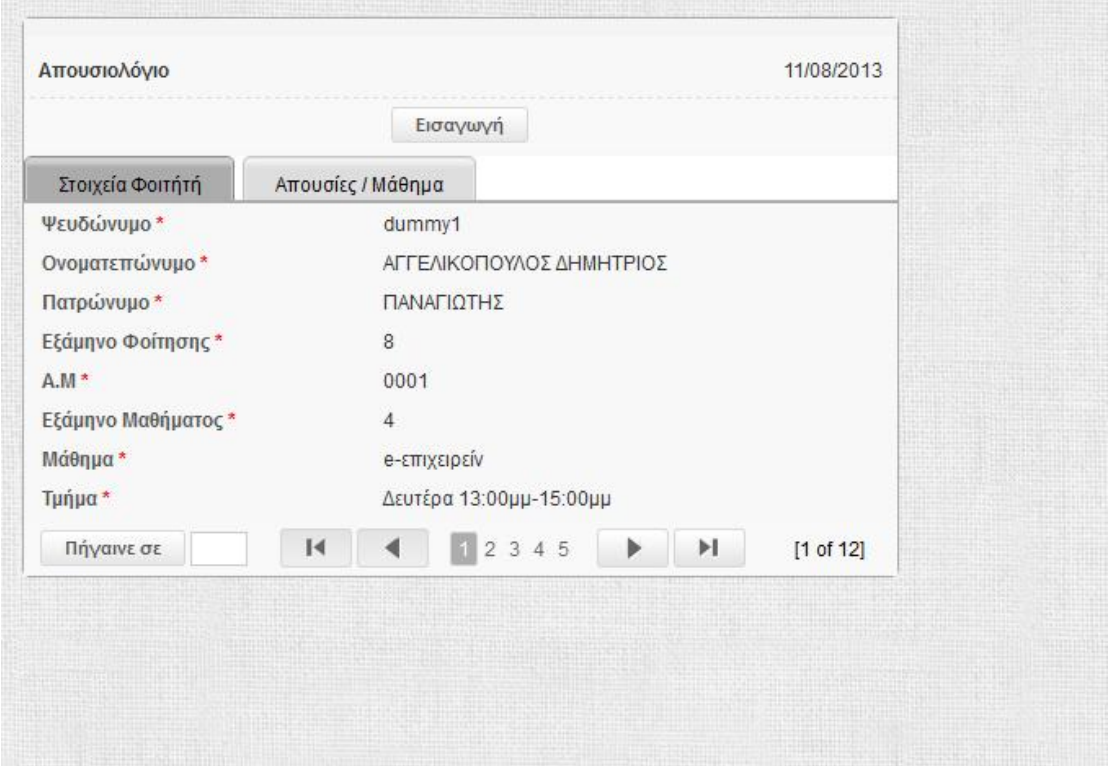
Εικόνα 15 Πλατφόρμα

Ο καθηγητής μπορεί να κάνει οτιδήποτε μπορεί και ο φοιτητής (έγγραφες φοιτητών σε μαθήματα, τροποποιήσεις κλπ) και επιπρόσθετα να τροποποιεί τις εγγραφές των φοιτητών ανά τμήματα (πχ αλλαγή φοιτητή από τμήμα) ή και την διαγραφή ακόμα.

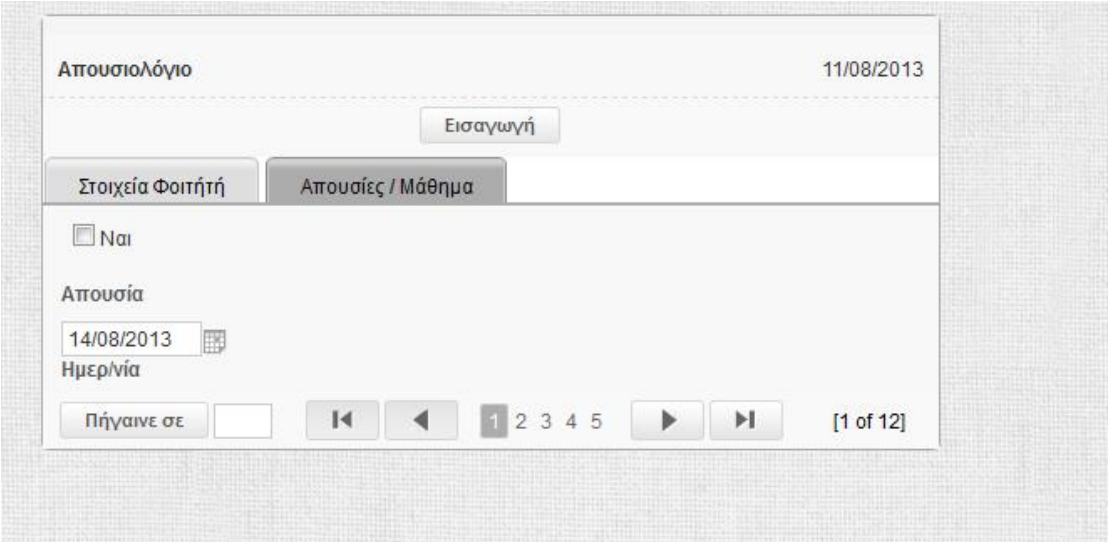


Εικόνα 16 Αναζήτηση απουσιολόγιου

Επίσης μπορεί να μπει στο μάθημα που είναι εξουσιοδοτημένος (από την γραμματεία) και να διαλέξει το τμήμα που επιθυμεί και να βάλει απουσία στον φοιτητή που θέλει (αφού ελέγξει τα στοιχεία του)

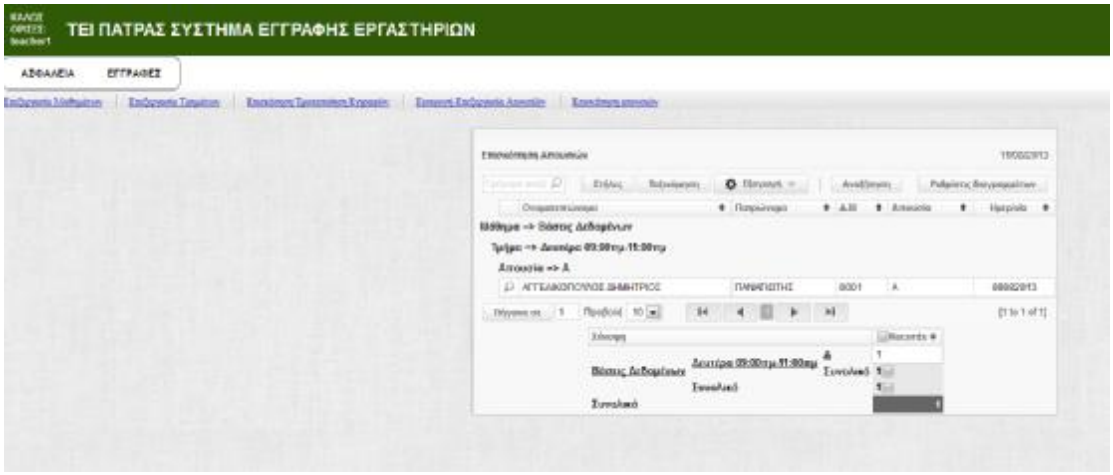


Εικόνα 17 Έλεγχος στοιχείων φοιτητή

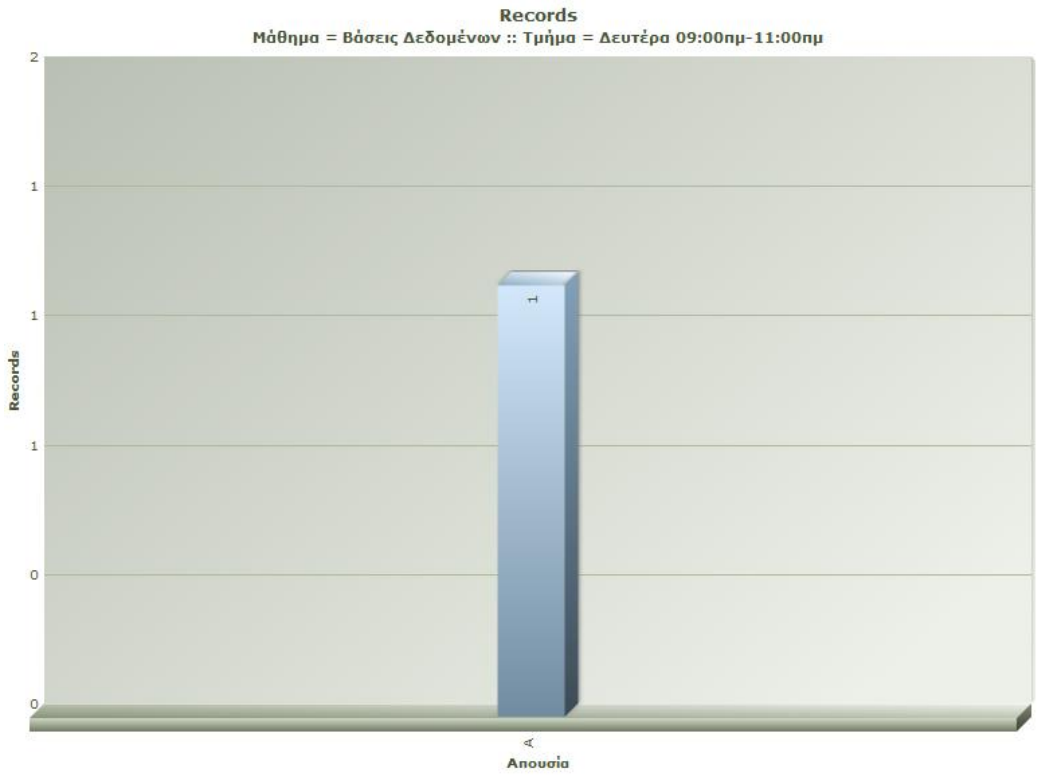


Εικόνα 18 Απουσία σε φοιτητή

Τέλος έχει δικαίωμα να σβήσει απουσία (σε συνεννόηση με την γραμματεία), και να δει το σύνολο των απουσιών των φοιτητών και να βγάλει στατιστικά διαγράμματα ή να τα εξάγει.



Εικόνα 19 Σύνολο απουσιών

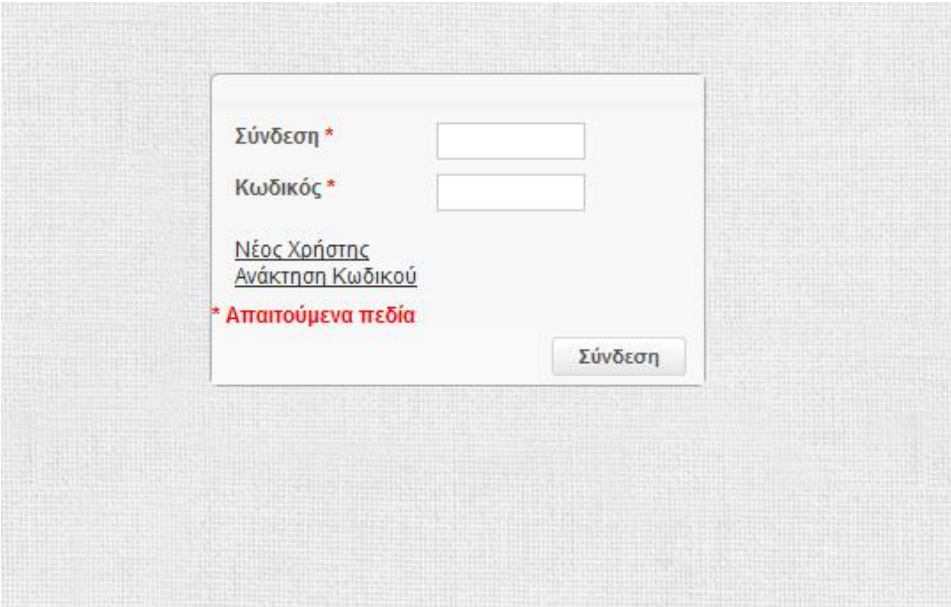


Εικόνα 20 Σύνολό απόντων ανά μάθημα

4.3 Εγχειρίδιο για την γραμματεία

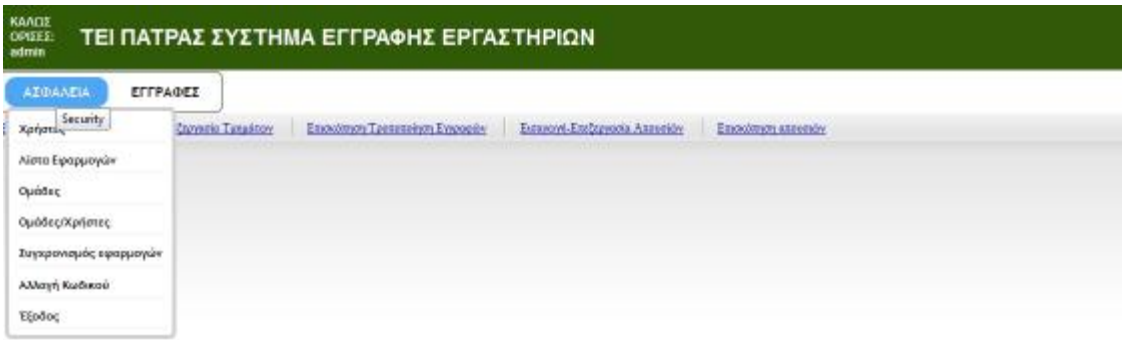
Η γραμματεία εισάγεται με τα στοιχεία πρόσβασης που του έχει δώσει ο κατασκευαστής της εφαρμογής. Σε περίπτωση που χάσει τα στοιχεία πρόσβασης έχει δυνατότητα ανάκτησης του κωδικού βάσει του email της. Επίσης στην εφαρμογή

υπάρχει δυνατότητα για δημιουργία νέων καθηγητών (δικαίωμα που το έχει μόνο η γραμματεία).



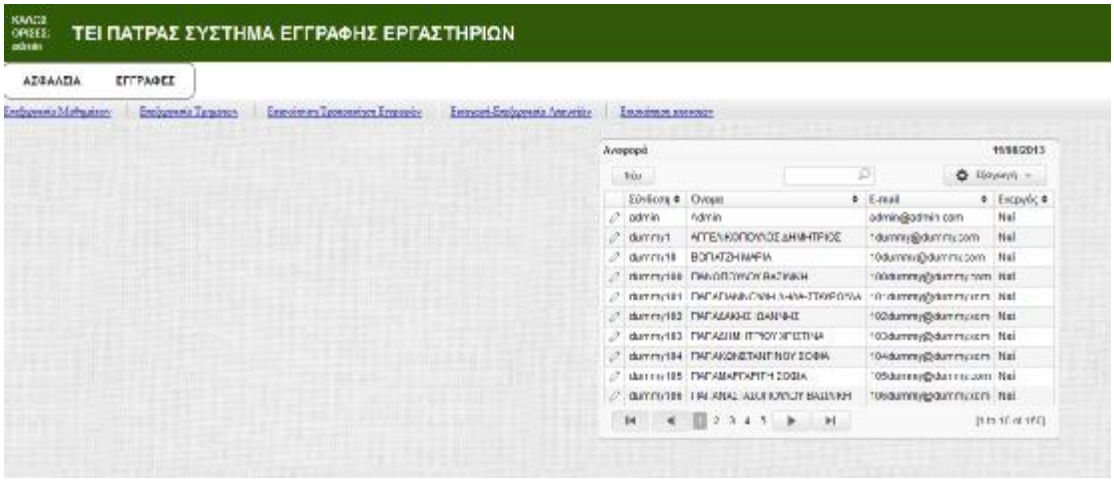
Εικόνα 21 Είσοδος στην εφαρμογή

Στην πλατφόρμα η γραμματεία έχει όλα τα δικαιώματα των καθηγητών και φοιτητών όπως και πολλά επιπρόσθετα.

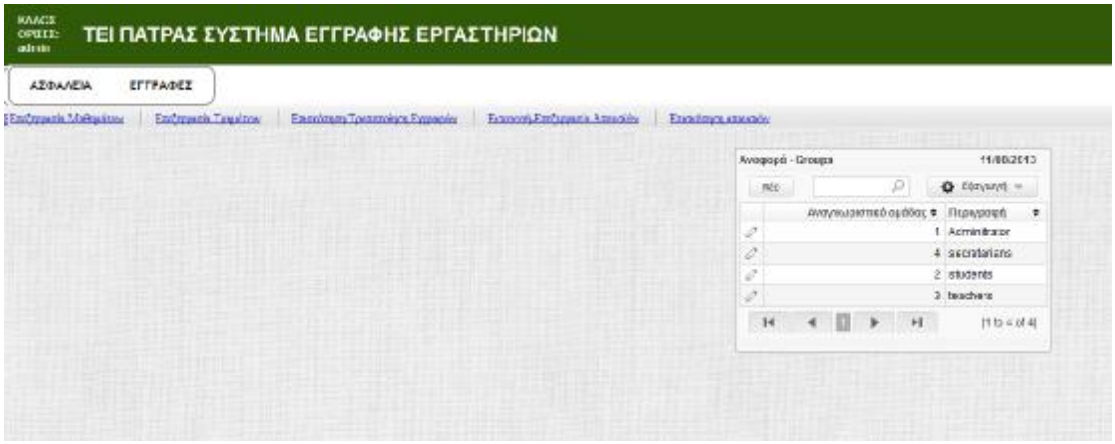


Εικόνα 22 Πλατφόρμα γραμματείας

Συγκεκριμένα στο μενού ασφάλεια η γραμματεία μπορεί να δει και να ορίσει χρήστες (ή και να δημιουργήσει, να δώσει δικαιώματα σε κατηγορίες χρηστών.

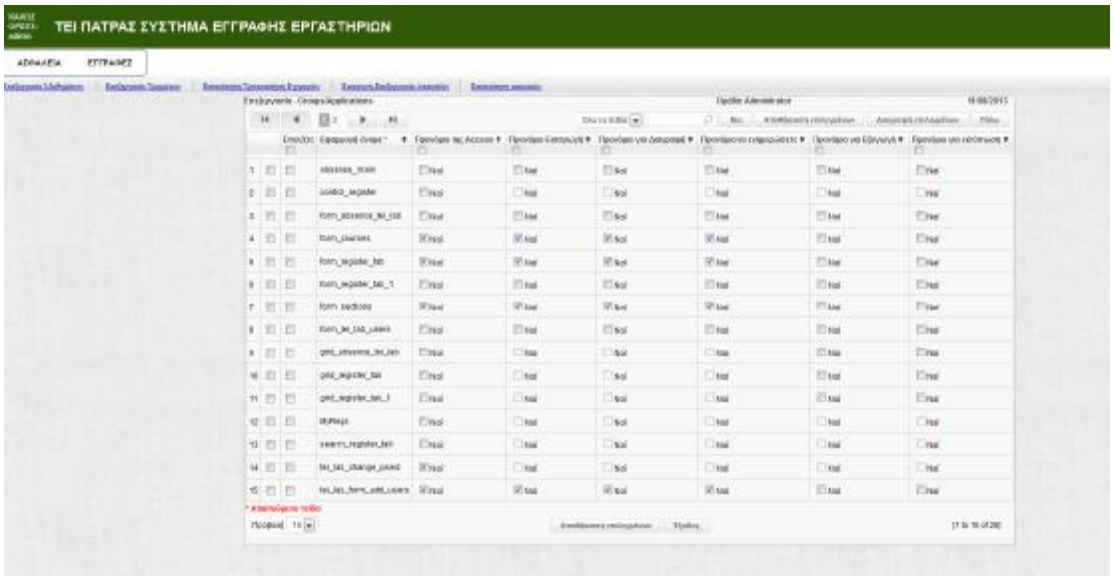


Εικόνα 23 Χρήστες του συστήματος



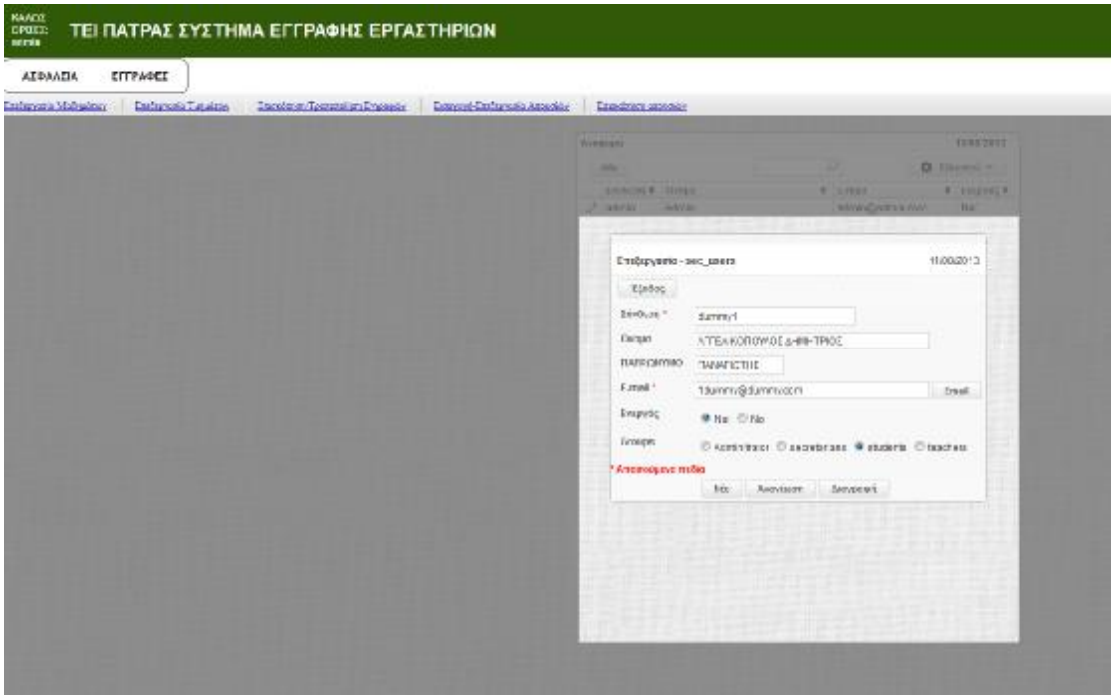
Εικόνα 24 Κατηγορίες χρηστών.

Επίσης έχει δικαίωμα να ορίσει ποια κατηγορία έχει δικαίωμα πρόσβασης σε ποια φόρμα.



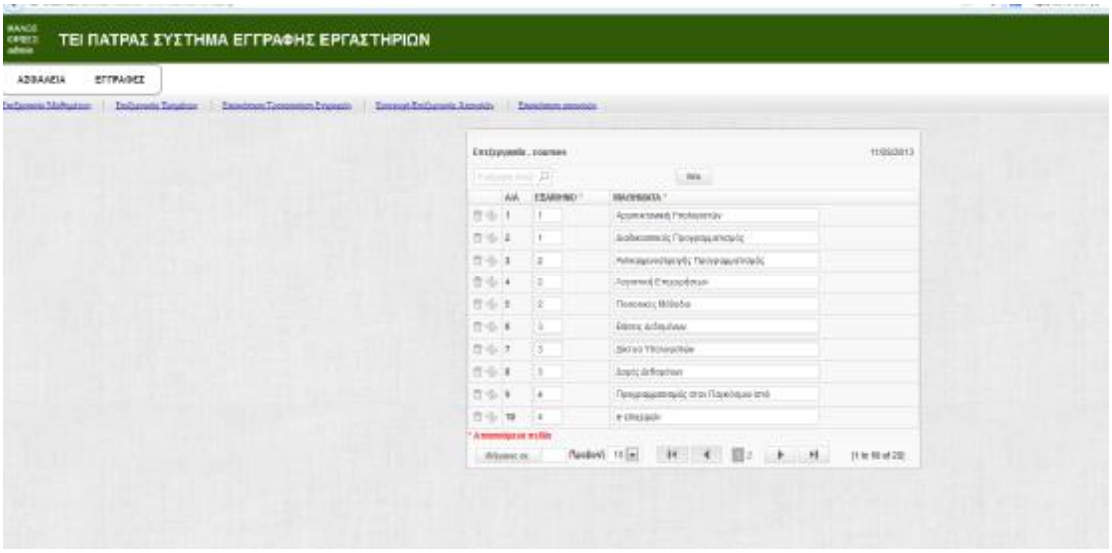
Εικόνα 25 Ορισμός δικαιωμάτων

Επίσης μπορεί να συγχρονίσει (update) τις αλλαγές στο σύστημα. Επίσης μπορεί να δημιουργεί χρήστες και να ορίζει κωδικούς και κατηγορία ανά χρήστη.



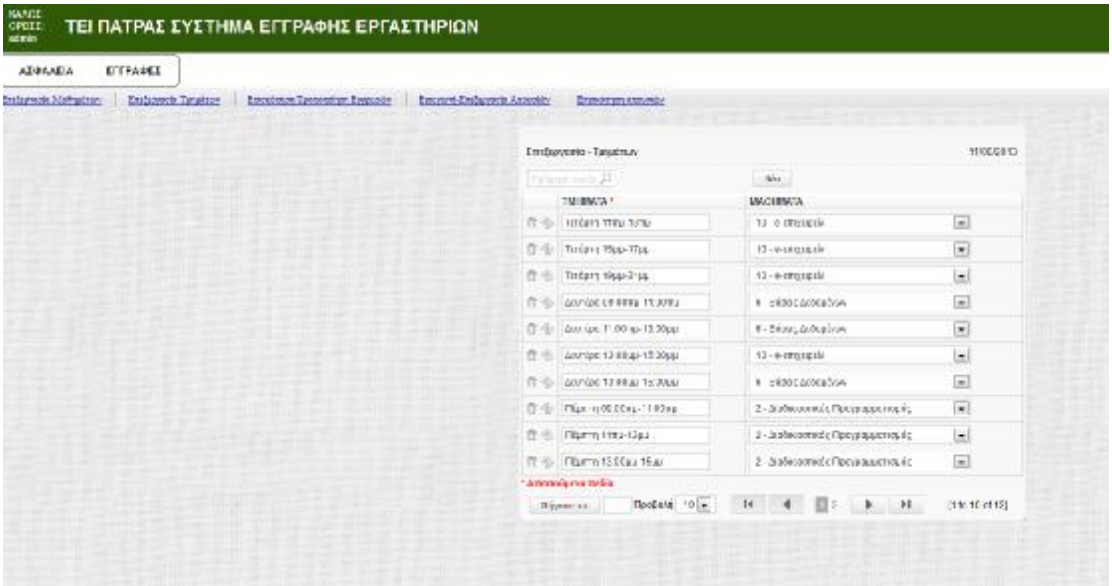
Εικόνα 26 Τροποποίηση χρήστη

Άλλο δικαίωμα της γραμματείας είναι η δημιουργία μαθημάτων και η αντιστοίχηση σε καθηγητή.

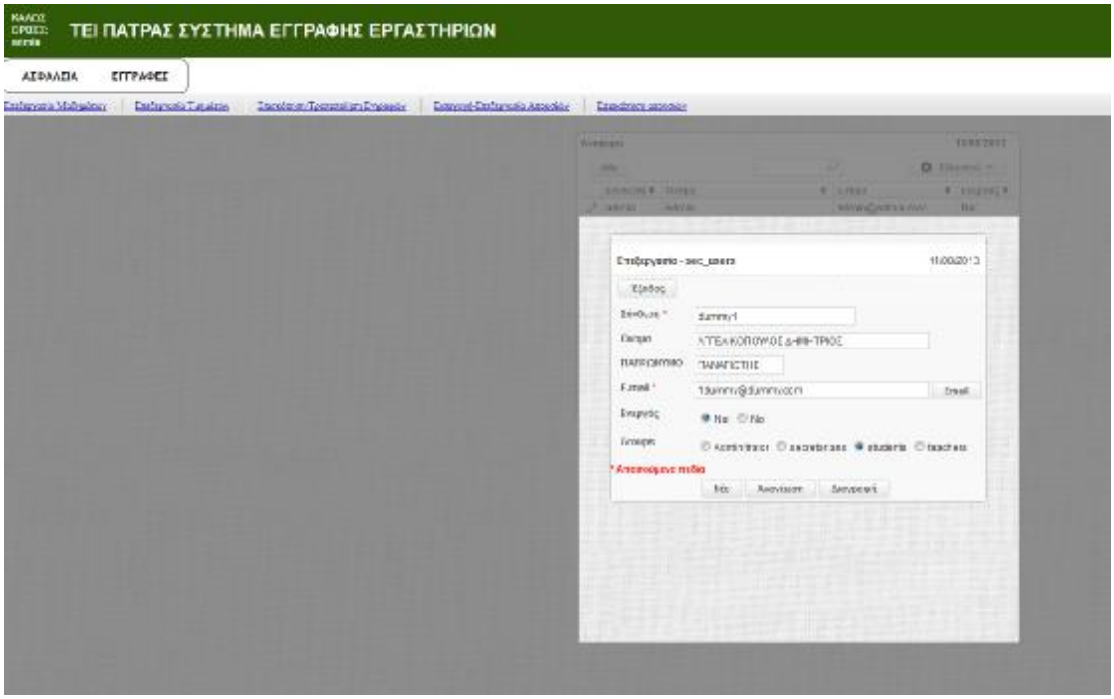


Εικόνα 27 Δημιουργία μαθημάτων

Υπάρχει η δυνατότητα να δημιουργηθούν ή να καταργηθούν τμήματα ανά μάθημα και ποιες ώρες θα γίνονται. Για την καλύτερη λειτουργία του συστήματος, ο μέγιστος αριθμός ατόμων ανά ομάδα είναι τα 25 άτομα.



Εικόνα 28 Δημιουργία τμημάτων



Εικόνα 29 Τροποποίηση χρήστη

5 Συμπεράσματα

Με την παρούσα πτυχιακή πιστεύουμε ότι πετύχαμε τον αρχικό στόχο, δηλαδή την μελέτη όλων των μοντέλων ανάπτυξης λογισμικού με σκοπό την ανάλυση και υλοποίηση διαδικτυακής εφαρμογής e lab για την δήλωση εργαστηριακών μαθημάτων στο ΤΕΙ Πάτρας.

Η εφαρμογή αυτή θα καλύπτει τις βασικές λειτουργίες που γίνονται για την δήλωση ενός μαθήματος. Βέβαια διευκολύνθηκε η διαδικασία δήλωσης ενός μαθήματος που μέχρι στιγμής γινόταν με συμβατά μέσα (κατ' ιδίαν με αίτησή στο ΤΕΙ). Αυτό έχει ως αποτέλεσμά την αύξηση της παραγωγικότητας, της επιχειρηματικότητας καθώς και την διευκόλυνση της γραμματείας και των καθηγητών και επιτάχυνσή των διαδικασιών στο ΤΕΙ Πατρών.

5.1 Μελλοντικές δυνατότητες

Η εφαρμογή προσφέρει (σε μελλοντικό επίπεδο) την δυνατότητα για:

- Συλλογή πληροφοριών όπως για παράδειγμα ιστορικό δηλώσεων και προτιμήσεων, δημογραφικά στοιχεία κ.λ.π. και αξιοποίησή τους για την παροχή όσο το δυνατόν καλύτερων υπηρεσιών προς τους φοιτητές.
- Καλύτερης εξυπηρέτηση φοιτητών
- Αξιολόγηση της μακροπρόθεσμης αξίας του φοιτητή (τακτικός, αιώνιος) (customer analysis).
- Συνεργασία με άλλα πανεπιστημιακά ιδρύματα για προώθηση και ανταλλαγή δεδομένων και πληροφοριών.

Επίσης θα μπορούσε να προστεθεί μελλοντικά δυνατότητές όπως:

- Newsletters με νέα από το ΤΕΙ και την γραμματεία.
- Διαφημιστικές εκστρατείες βασισμένες σε συγκεκριμένες εκδηλώσεις ή ημερίδες, συνέδριά ή και προτάσεις για μεταπτυχιακά τμήματα.
- Response Management ήτοι αυτόματες εφαρμογές αποστολής απαντητικών e-mail για συγκεκριμένες ερωτήσεις ή περιστάσεις

5.2 Οφέλη

Όλα τα παραπάνω με σωστό συντονισμό και οργάνωση θα μπορούσαν να αποφέρουν 3 κύρια οφέλη:

1. Καλύτερη εξυπηρέτηση
2. Μείωση κόστους
3. Αύξηση ευελιξίας στις αλλαγές της αγοράς.

5.2.1 Καλύτερη εξυπηρέτηση

Η Καλύτερη εξυπηρέτηση επιτυγχάνεται ως εξής:

- Βελτιωμένη ανταπόκριση σε απαιτήσεις για πληροφόρηση
- Καλύτερη ικανοποίηση των αναγκών του φοιτητή
- Άμεση παρακολούθηση της προόδου του φοιτητή
- Μεγαλύτερη ποικιλία προσφερόμενων υπηρεσιών στον φοιτητή
- Καλύτερη υποστήριξη

5.2.2 Μείωση κόστους

Η μείωση κόστους προκύπτει από:

- Στη λειτουργία ενός call center – κέντρου εξυπηρέτησης φοιτητών, μέσω της αυτοματοποίησης των διαδικασιών. Το μεγαλύτερο κόστος της λειτουργίας ενός call center είναι το στελεχιακό δυναμικό. Το κόστος λειτουργίας ενός τέτοιου κέντρου μπορεί να μειωθεί μέχρι και 70%.
- Στη βελτίωση της αποτελεσματικότητας του direct mail.
- Στη βελτίωση της αποτελεσματικότητας της διαφήμισης Πανεπιστημιακών ιδρυμάτων.
- Στην αποτελεσματικότερη υποστήριξη των φοιτητών μέσω της καλύτερης πληροφόρησης κλπ.

5.2.3 Άμεση ανταπόκριση στις αλλαγές της αγοράς

Η άμεση ανταπόκριση στις αλλαγές της αγοράς προκύπτει:

Υπάρχουν συστήματα CRM, τα οποία μπορούν να ενσωματωθούν στην εφαρμογή και τα οποία βοήθησαν σημαντικά τις επιχειρήσεις να προσαρμοστούν σε σημαντικές επιχειρησιακές αλλαγές και ένια εφαρμοστούν επιτυχώς σε πανεπιστημιακά ιδρύματα επιπέδου ΤΕΙ και ΑΕΙ. Με την εγκατάσταση των συστημάτων CRM το ίδρυμα έχει τη δυνατότητα να γνωρίζει τους φοιτητές και τις ανάγκες τους ανά πάσα στιγμή, να στοχεύει καλύτερα στην επικοινωνία, στην εξυπηρέτηση και στην μείωση κόστους.

5.2.4 Άλλα οφέλη

Άλλα οφέλη που προκύπτουν είναι:

- Παροχή καλύτερης εξυπηρέτησης, σύμφωνα με τις ανάγκες των φοιτητών.
- Αύξηση της συνολικής αποδοτικότητας μέσω της υλοποίησης διαδικασιών αυτοματοποίησης.
- Αποτελεσματική λειτουργία Κέντρων Παροχής Βοήθειας.
- Διασταυρούμενες κινήσεις και δυνατότητα υλοποίησης ενεργειών προσωποποιημένου marketing («1 προς 1» marketing).
- Απλοποίηση διαδικασιών.
- Αύξηση εσόδων ανά φοιτητή.
- Να μειώσει τα έξοδα
- Να αυξήσει τις συναλλαγές ανά φοιτητή
- Να βελτιώσει τη φήμη του ΤΕΙ (word-of-mouth).

5.3 Σύνοψη

Μελλοντικά πιστεύουμε ότι υπάρχει η δυνατότητα ανάπτυξης περαιτέρω της εφαρμογής καθώς η εμπορευματοποίηση της με ανάλογα οφέλη τόσο για το πανεπιστημιακό ίδρυμα που θα το αναλάβει όσο και για το καταναλωτικό κοινό και στην συγκεκριμένη περίπτωση τους φοιτητές. Με την κατάλληλη οργάνωση, συντονισμό και στήσιμο θα μπορούσε να φέρει σημαντικές αλλαγές στο χώρο των πανεπιστημιακών ιδρυμάτων και να δώσει μία άλλη χροιά στο κλάδο αυτό.

Βιβλιογραφία

- Beck, K. (1999). *Extreme Programming Explained, Embracing Change.* Addison-Wesley Professional.
- Ian Sommerville, M. K. (2008). *Software engineering.* Pearson Education Limited.
- Madachy, R. J. (2008). *Software Process Dynamics.* Wiley-IEEE Press.
- Madnick, T. K.-H. (1991). *Software Project Dynamics: An Integrated Approach.* Prentice-Hall.
- Multithreading. (2007). *Schildt, Herbert.* ΓΚΙΟΥΡΔΑΣ Μ.
- Per Kroll, P. K. (2003). *The rational unified process made easy: a practitioner's guide to the RUP.* Addison-Wesley.
- Βεσκούκης, Β. (2000). *Τεχνολογία Λογισμικού Ι.* ΠΑΤΡΑ: ΕΑΠ.
- Χρυσογιός. (1996). *Απο την αλάνα στο σύγχρονο γήπεδο.* Αθήνα: Ελληνικά Γράμματα
- Melony, J. C.(2000) *PHP, MySql και Apache.* Μ.Γκιούρδας.

Πηγές από διαδίκτυο

- *1keydata.com.*
- http://www.adobe.com/products/dreamweaver/:
- *en.wikipedia.org*
- http://en.wikipedia.org/wiki/Web_service:

- *http://www.sql.org* .
- www.scriptcase.net:
- http://www.en.wikipedia.org/wiki/
- *www.coder.gr*.
- *www.it.uom.grwww.users.hol.gr*.
- *www.w3schools.com*.
- http://oreilly.com/
- http://ebennett.org/hsnl/

Παράρτημα

Script της βάσης δεδομένων

absence_tei_lab

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
abs_id	int(2)	Όχι		
name	varchar(64)	Όχι		
f_name	varchar(15)	Όχι		
am	varchar(4)	Όχι		
absense	varchar(1)	Όχι		
absense_date	char(8)	Όχι		
cour_id	int(2)	Όχι		
sec_id	int(2)	Όχι		

courses

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
cour_id	int(2)	Όχι		
sems	char(1)	Όχι		
lessons	varchar(50)	Όχι		

register_tab

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
login	varchar(32)	Όχι		
name	varchar(64)	Όχι		

f_name	varchar(15)	Όχι	
sem_st	char(1)	Όχι	
am	varchar(4)	Όχι	
sems	char(1)	Όχι	
cour_id	int(2)	Όχι	
sec_id	int(2)	Όχι	

sections

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
sec_id	int(2)	Όχι		
sections	varchar(30)	Όχι		
cour_id	int(2)	Όχι		

tei_lab_apps

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
app_name	varchar(128)	Όχι		
app_type	varchar(255)	Ναι	*NULL*	
description	varchar(255)	Ναι	*NULL*	

tei_lab_groups

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
group_id	int(11)	Όχι		
description	varchar(64)	Ναι	*NULL*	

tei_lab_groups_apps

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
group_id	int(11)	Όχι		
app_name	varchar(128)	Όχι		
priv_access	varchar(1)	Ναι	*NULL*	
priv_insert	varchar(1)	Ναι	*NULL*	
priv_delete	varchar(1)	Ναι	*NULL*	
priv_update	varchar(1)	Ναι	*NULL*	
priv_export	varchar(1)	Ναι	*NULL*	
priv_print	varchar(1)	Ναι	*NULL*	

tei_lab_logged

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
login	varchar(32)	Όχι		
date_login	varchar(128)	Ναι	*NULL*	
sc_session	varchar(32)	Ναι	*NULL*	

tei_lab_users

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
login	varchar(32)	Όχι		
pswd	varchar(32)	Όχι		
name	varchar(64)	Ναι	*NULL*	
f_name	varchar(15)	Όχι		
email	varchar(64)	Ναι	*NULL*	

active	varchar(1)	Ναι	*NULL*
sem_st	char(1)	Όχι	
activation_code	varchar(32)	Ναι	*NULL*
priv_admin	varchar(1)	Ναι	*NULL*
am	varchar(4)	Όχι	

tei_lab_users_groups

Στήλη	Τύπος	Κενό	Προεπιλογή	Σχόλια
login	varchar(32)	Όχι		
group_id	int(11)	Όχι		