ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΑΝΑΛΥΣΗ & ΣΧΕΔΙΑΣΜΟΣ ΣΥΣΤΗΜΑΤΟΣ ΕΛΕΓΧΟΥ ΔΙΟΙΚΗΣΗΣ (MANAGEMENT CONTROL SYSTEM) ΜΕ ΤΗΝ ΧΡΗΣΗ ΤΗΣ UNIFIED MODELING LANGUAGE (UML)

Η εργασία αυτή υποβάλλεται για την μερική κάλυψη των απαιτήσεων με στόχο την απόκτηση του διπλώματος

ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ «ΕΦΟΔΙΑΣΜΟΣ & ΔΙΑΚΙΝΗΣΗ ΠΡΟΪΟΝΤΩΝ (LOGISTICS)»

από

ΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ ΚΑΙ ΤΟ ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΠΑΠΑΟΙΚΟΝΟΜΟΥ ΕΜΜΑΝΟΥΗΛ

ΤΜΗΜΑ ΒΙΟΜΗΧΑΝΙΚΗΣ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ, ΙΑΝΟΥΑΡΙΟΣ 2006

Ευχαριστίες

Με την ολοκλήρωση της Διπλωματική μου εργασία θα ήθελα να ευχαριστήσω θερμά:

- Τον Αναπληρωτή Καθηγητή του τμήματος Βιομηχανικής Διοίκησης και Τεχνολογίας του Πανεπιστημίου Πειραιώς Κο. Γρηγόρη Χονδροκούκη για την εμπιστοσύνη που έδειξε στο πρόσωπο μου με την ανάθεση της παρούσας εργασίας.
- Τον διευθυντή Πληροφορικής της εταιρείας ΒΙΟΣΕΡ Α.Ε. Κο. Σταμάτη Μπίλα για τις αξιόλογες επιστημονικές του παρατηρήσεις, το ενδιαφέρον του και την πολύτιμη βοήθεια του σε όλα τα στάδια της διπλωματικής μου εργασίας.
- Του συναδέλφους μου στο τμήμα Πληροφορικής της εταιρείας ΒΙΟΣΕΡ Α.Ε. για τις πολύτιμες παρατηρήσεις, διορθώσεις και συμβουλές τους για την συγγραφή του πρακτικού μέρους της επίλυσης του πραγματικού προβλήματος, που αντιμετωπίσαμε στην εταιρεία.

Περιεχόμενα

Λίστα Διαγραμμάτων

Λίστα Πινάκων

Γλωσσάριο Όρων

Activity Διάγραμμα

Το διάγραμμα της UML που δείχνει τον τρόπο με τον οποίο θα επιτευχθεί μια λειτουργία του συστήματος. Με τα Activity διαγράμματα μπορούμε να αναπαραστήσουμε πως υλοποιούνται τα διάφορα workflows στο σύστημα (ροή της εργασίας), ποια μονοπάτια απόφασης (decision paths) πρέπει να ακολουθηθούν, οι διάφορες περιπτώσεις που υπάρχουν για να ολοκληρωθεί η λειτουργία και πως ολοκληρώνεται το workflow. Τα διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν κάποιο σενάριο εκτέλεσης μιας λειτουργίας του συστήματος που έχει περιγραφεί από ένα Use Case διάγραμμα.

Balanced Scorecard (BSC)

Ένα σύστημα μέτρησης της απόδοσης ενός οργανισμού το οποίο μεταφράζει την αποστολή και την στρατηγική του σε ένα σύνολο μονάδων μέτρησης της απόδοσης παρέχοντας ένα πλαίσιο για ένα στρατηγικό σύστημα διοίκησης. Η BSC μετράει την απόδοση του οργανισμού χρησιμοποιώντας 4 διαφορετικές συνιστώσες τις οποίες προσπαθεί να ισορροπήσει: τα *Χρηματοοικονομικά στοιχεία*, τους *Πελάτες*, τις *Εσωτερικές Επιχειρηματικές Λειτουργίες* και την *Μάθηση και Ανάπτυξη* των εργαζόμενων του οργανισμού.

Class Διάγραμμα

Το διάγραμμα της UML που περιγράφει την στατική δομή της εφαρμογής και περιλαμβάνει τις οντότητες (κλάσεις) που θα χρησιμοποιηθούν σε αυτή. Ένα Class διάγραμμα περιγράφει την δομή (structural view) του συστήματος δηλαδή τους τύπους των αντικειμένων που θα χρησιμοποιηθούν, την στατική δομή των αντικειμένων αυτών και το είδος των συσχετίσεων που υπάρχουν ανάμεσα στα αντικείμενα. Μέσα από την ανάλυση του Class διαγράμματος βλέπουμε τις κλάσεις που απαιτούνται για την υλοποίηση της εφαρμογής.

Collaboration Διάγραμμα

Είναι μια παραλλαγή των Sequence διαγραμμάτων όπου παρουσιάζονται τα αντικείμενα και η επικοινωνία μεταξύ τους. Η διαφορά με τα Sequence διαγράμματα είναι ότι δεν εμφανίζονται τα μηνύματα που αποστέλλονται μεταξύ των αντικειμένων στην διάρκεια του χρόνου (σειρά ανάγνωσης από πάνω προς τα κάτω) αλλά εμφανίζονται με αύξουσα αρίθμηση.

Component Διάγραμμα

Το διάγραμμα της UML που χρησιμοποιείται για να περιγράψει την φυσική υλοποίηση τμημάτων λογισμικού του συστήματος. Περιγράφει την φυσική υλοποίηση (physical implementation) της εφαρμογής παρουσιάζοντας τα τμήματα πηγαίου κώδικα (source code) και τα εκτελέσιμα αρχεία (executable files) που χρησιμοποιούνται στην εφαρμογή.

Deployment Διάγραμμα

Το διάγραμμα της UML που χρησιμοποιείται για να αναπαραστήσει την αρχιτεκτονική του hardware του συστήματος και τον προσδιορισμό των τμημάτων του συστήματος στα οποία γίνονται οι διάφορες εργασίες. Σκοπός είναι να παρουσιαστεί η φυσική υλοποίηση του συστήματος (hardware της εφαρμογής) και συνήθως χρησιμοποιείται σε συνεργασία με τα Component διαγράμματα για μια ολοκληρωμένη εικόνα του συστήματος.

Formal Control Process

Μια επίσημη διαδικασία ελέγχου της διοίκησης ενός οργανισμού με την έννοια ότι η διαδικασίας αυτή είναι καταγεγραμμένη και είναι γνωστή σε όλους τους εργαζόμενους του οργανισμού. Η διαδικασία αυτή αναφέρεται και σαν Σύστημα Ελέγχου Διοίκησης (Management Control System).

Mainframe

Ο κεντρικός υπολογιστής, που χρησιμοποιείται στην Αρχιτεκτονική 1-tier, η οποία είναι ένας τύπος αρχιτεκτονικής υπολογιστών, που χρησιμοποιούνταν στις πρώτες δεκαετίες ανάπτυξης εφαρμογών (δεκαετίες ’70 και ’80). Στην αρχιτεκτονική αυτή γινόταν χρήση ενός κεντρικού υπολογιστή (γνωστός και σαν mainframe) Όλη η επεξεργασία των δεδομένων γίνεται στον κεντρικό υπολογιστή, στον οποίο «τρέχουν» και οι εφαρμογές του οργανισμού. Στον κεντρικό υπολογιστή συνδέονται απλά τερματικά (terminals) μόνο για την παρουσίαση των δεδομένων.

Management Control Systems (Συστήματα Ελέγχου Διοίκησης)

Όλες εκείνες οι διαδικασίες και οι λειτουργίες που χρησιμοποιούν οι διευθυντές ενός οργανισμού έτσι ώστε να πετύχουν την υλοποίηση των στόχων τους και ταυτόχρονα τους στόχους του οργανισμού. Σκοπός του Ελέγχου Διοίκησης είναι να λάβουν οι διευθυντές τις κατάλληλες αποφάσεις για τον οργανισμό και να διαχειριστούν τους πόρους αποδοτικά και αποτελεσματικά έτσι ώστε να επιτευχθούν οι στόχοι που έχουν τεθεί.

Object Management Group (OMG)

Ένα ανοιχτό consortium εταιρειών πληροφορικής – μη κερδοσκοπικού χαρακτήρα – το οποίο παράγει και συντηρεί ειδικές προδιαγραφές προϊόντων λογισμικού. Η σημαντικότερη προδιαγραφή του οργανισμού είναι η UML, που καθιερώθηκε σαν πρότυπο τον Νοέμβριο του 1997. Άλλες προδιαγραφές του οργανισμού είναι η Model Driven Architecture (MDA), CORBA κ.α.

Sequence Διάγραμμα

Το διάγραμμα της UML που χρησιμοποιείται για να αναπαραστήσει την αλληλεπίδραση των αντικειμένων του συστήματος στη διάρκεια του χρόνου (ανταλλαγή μηνυμάτων). Δείχνει την συμμετοχή των αντικειμένων στην λειτουργία που περιγράφεται και την σειρά με την οποία ανταλλάσσονται τα μηνύματα. Η σειρά των μηνυμάτων παρουσιάζεται από πάνω προς τα κάτω).

Statechart Διάγραμμα

Το διάγραμμα της UML που χρησιμοποιείται για να αναπαραστήσει τον κύκλο ζωής των διαφόρων αντικειμένων του συστήματος. Ο κύκλος ζωής των αντικειμένων αναφέρεται στις διάφορες καταστάσεις (states) τις οποίες έχουν αυτά τα αντικείμενα καθώς και στα γεγονότα (events) τα οποία δημιουργούν αυτές τις καταστάσεις. Περιγράφει όλες τις πιθανές καταστάσεις στις οποίες μπορεί να βρεθεί ένα αντικείμενο όταν αντίστοιχα γεγονότα συμβούν.

UML όψεις (Views)

Το σύνολο UML διαγραμμάτων που περιγράφει μια πλευρά του συστήματος. Οι όψεις του συστήματος είναι της λειτουργικότητας (functionality – τι κάνει το σύστημα), της στατικής (static) δομής του και οι δυναμικές (dynamic) αλληλεπιδράσεις των στοιχείων του. Μέσα από τον συνδυασμό των όψεων της UML μπορούμε να έχουμε μια ολοκληρωμένη εικόνα του συστήματος.

Unified Modeling Language (UML)

Μια γραφική γλώσσα η οποία συμβάλλει στην οπτικοποίηση (visualizing), προσδιορισμό (specifying), κατασκευή (constructing) και τεκμηρίωση (documenting) συστημάτων λογισμικού. Καθιερώθηκε σαν πρότυπο τον Νοέμβριο του 1997 και αποτελεί συνένωση των μεθοδολογιών Object Modeling Technique (OMT) του Jim Rumbaugh, Booch του Grady Booch και της μεθοδολογίας Objectory του Ivar Jacobson.

Use Case Διάγραμμα

Το διάγραμμα της UML που δείχνει μια γενική όψη του συστήματος από την πλευρά του χρήστη δείχνει δηλαδή τις λειτουργίες του συστήματος όπως τις αντιλαμβάνεται ένας εξωτερικός παρατηρητής. Παρουσιάζονται οι απαιτήσεις (requirements) των χρηστών από το σύστημα και αναπαριστάται η λειτουργικότητα του συστήματος (functionality).

Use case σενάρια

Τα σενάρια (βήματα) που αναπαριστούν κάποιες περιπτώσεις οι οποίες είναι πολύ πιθανό να συμβούν κατά τη διάρκεια εκτέλεσης του συστήματος. Με τον τρόπο αυτό μπορούμε να καταγράψουμε την συμπεριφορά του συστήματος σε διαφορετικές περιπτώσεις και να λάβουμε πολύτιμα συμπεράσματα σχετικά με την πορεία του σχεδιασμού του συστήματος.

Αντικείμενα (Objects)

Τα αντικείμενα στον αντικειμενοστραφή προγραμματισμό είναι οι βασικές οντότητες της εφαρμογής. Ένα αντικείμενο είναι ένα τμήμα του κώδικα κάτω από μια κοινή ονομασία που περιλαμβάνει τις μεταβλητές και τις συναρτήσεις που σχετίζονται με το αντικείμενο αυτό. Το αντικείμενο αναφέρεται και σαν Κλάση (Class).

Αντικείμενο

Η οντότητα που αποθηκεύει τις επιλογές του χρήστη για μια Παρουσίαση. Ένας χρήστης ανάλογα με τον Ρόλο στον οποίο ανήκει έχει Δικαίωμα να δει ορισμένες Παρουσιάσεις. Στην συνέχεια για κάθε Παρουσίαση μπορεί να επιλέξει ορισμένα στοιχεία των Δομών της κάθε διάστασης της Παρουσίασης και να τα αποθηκεύσει. Όποτε ο χρήστης θέλει να δει δεδομένα για τα στοιχεία που τον ενδιαφέρουν επιλέγει να δει το συγκεκριμένο Αντικείμενο.

Αντικειμενοστραφής Προγραμματισμός

Ο τρόπος προγραμματισμού με την χρήση αντικειμένων. Η εφαρμογή περιγράφεται σαν ένα σύνολο από ξεχωριστά και ανεξάρτητα αντικείμενα τα οποία επικοινωνούν μεταξύ τους μέσω της ανταλλαγής μηνυμάτων.

Αποτέλεσμα

Η οντότητα που διαχειρίζεται την ομαδοποίηση διαφόρων Παρουσιάσεων και την παρουσίαση του αποτελέσματος. Για παράδειγμα, εάν έχουμε πολλές Παρουσιάσεις που σχετίζονται με έξοδα και έσοδα το Αποτέλεσμα ομαδοποιεί τις Παρουσιάσεις αυτές, κάνει προσθέσεις και αφαιρέσεις (ανάλογα με τον Τύπο τους) και εμφανίζει το τελικό Αποτέλεσμα (άθροισμα) στον χρήστη.

Αρχιτεκτονική N-tier

Η αρχιτεκτονική που σχετίζεται με τον διαχωρισμό των εφαρμογών ενός οργανισμού σε ξεχωριστά επίπεδα (3 ή περισσότερα επίπεδα). Κάθε επίπεδο είναι σαφώς ορισμένο και απομονωμένο από τα υπόλοιπα διαγράμματα και εκφράζει μια γενική λειτουργία της εφαρμογής. Ο διαχωρισμός των επιπέδων είναι φυσικός είτε λογικός.

Αρχιτεκτονική Εφαρμογών

Αναφέρεται στον τρόπο με τον οποίο είναι δομημένες οι εφαρμογές που αναπτύσσονται σε ένα οργανισμό.

Αφαίρεση (Abstraction)

Η αρχή του αντικειμενοστραφούς προγραμματισμού που σχετίζεται με την αφαιρετική παρουσίαση μιας έννοιας του πραγματικού κόσμου παρουσιάζοντας μόνο τις λεπτομέρειες που δείχνουν πως θα χρησιμοποιήσουμε το αντικείμενο.

Βιβλιοθήκες

Η ομαδοποίηση κλάσεων και οθονών για να μπορούμε εύκολα να τις διαχειριστούμε και να έχουμε την δυνατότητα επαναχρησιμοποίησης τους και σε άλλες εφαρμογές. Οι κλάσεις και οι οθόνες μπορούν να χρησιμοποιούνται από διάφορες εφαρμογές και να μπορούν εύκολα να αλλάξουν χωρίς να επηρεάζουν τα υπόλοιπα τμήματα της εφαρμογής.

Δικαιώματα

Το αντικείμενο που διαχειρίζεται τα στοιχεία που έχει δικαίωμα να βλέπει ο χρήστης βάσει του Ρόλου στον οποίο ανήκει. Τα στοιχεία που έχει δικαίωμα να βλέπει ο χρήστης είναι οι Παρουσιάσεις και τα Αντικείμενα.

Δομή

Το αντικείμενο που παρουσιάζει την ιεραρχική δομή διάφορων Τύπων (Δομές Πελατών, Ειδών, Εξόδων κλπ.) Για κάθε Τύπο μπορούμε να έχουμε περισσότερες από μία Δομές προκειμένου να παραστήσουμε την ίδια πληροφορία με διαφορετικό τρόπο. Με μία Δομή μπορούμε να αναπαραστήσουμε και να διαχειριστούμε οποιαδήποτε οντότητα μέσα στην εφαρμογή. Επίσης μπορούμε να συνδυάσουμε δυο ή περισσότερες Δομές για να δείξουμε διάφορες πληροφορίες (π.χ. Έσοδα ανά Πελάτη, Έσοδα ανά Είδος, Πωλήσεις ανά Πελάτη και ανά Είδος κλπ.)

Δομημένος Προγραμματισμός

Ο τρόπος προγραμματισμού με την χρήση διαδικασιών και συναρτήσεων. Χαρακτηριστικό του τρόπου προγραμματισμού αυτού είναι η δόμηση του προγράμματος σε μικρά τμήματα, η επίλυση και η συνένωση τους για την επίλυση του συνολικού προβλήματος.

Ενθυλάκωση (Encapsulation)

Η αρχή του αντικειμενοστραφούς προγραμματισμού που σχετίζεται με την ενσωμάτωση των χαρακτηριστικών (μεταβλητών) και τον μεθόδων (συναρτήσεων) σε κάθε αντικείμενο έτσι ώστε να εμφανίζονται μόνο οι πληροφορίες χρήσης του αντικειμένου και να κρύβονται οι πληροφορίες λειτουργίας του αντικειμένου.

Επίπεδα (Layers)

Τα λογικά ή/και φυσικά χωρισμένα τμήματα στην Αρχιτεκτονική N-tier των εφαρμογών. Στην αρχιτεκτονική αυτή τα επίπεδα που συνήθως ορίζονται είναι της Παρουσίασης (Presentation), της Επιχειρησιακής Λογικής-Κανόνες (Business Logic), της Πρόσβασης Δεδομένων (Data Access) και της Αποθήκευσης Δεδομένων (Data).

Κέντρα Ευθύνης (Responsibility Centers)

Ένα κέντρο ευθύνης είναι μια επιχειρησιακή μονάδα (organizational unit) η οποία διοικείται από ένα διευθυντή, ο οποίος είναι υπεύθυνος για τις λειτουργίες του κέντρου αυτού. Ο οργανισμός είναι ένα σύνολο από κέντρα ευθύνης κάθε ένα από τα οποία μπορεί να αναπαρασταθεί σαν ένα κουτί στο οργανόγραμμα του οργανισμού. Ένα κέντρο ευθύνης έχει ένα ή περισσότερους σκοπούς (στόχοι-objectives) τους οποίους πρέπει να επιτύχει.

Κλάση

Αντιπροσωπεύει ένα σύνολο αντικειμένων με παρόμοια δομή, συμπεριφορά και συσχετίσεις. Είναι το πρότυπο (pattern) από το οποίο παράγονται τα αντικείμενα της εφαρμογής (στιγμιότυπα της κλάσης). Η κλάση είναι το βασικό δομικό στοιχείο του αντικειμενοστραφούς σχεδιασμού και ανάλυσης όπου δημιουργείται το μοντέλο του συστήματος αποτελούμενο από κλάσεις που περιέχουν χαρακτηριστικά (attributes) και μεθόδους (operations).

Κληρονομικότητα (Inheritance)

Η αρχή του αντικειμενοστραφούς προγραμματισμού που σχετίζεται με την ιδιότητα των αντικειμένων να κληρονομούν χαρακτηριστικά και μεθόδους από άλλα αντικείμενα.

Μοντέλο Πελάτη/Εξυπηρετητή (Client/Server Model

Η αρχιτεκτονική εφαρμογών σύμφωνα με την οποία υπάρχει ένας κεντρικός υπολογιστής (Εξυπηρετητής-Server) από τον οποίο ζητάνε να τους εξυπηρετήσει διάφοροι υπολογιστές-πελάτες (Clients). Είναι γνωστή και σαν Αρχιτεκτονική 2-tier.

Μοντελοποίηση (Modeling)

Η διαδικασία η οποία σχετίζεται με την κατασκευή ενός μοντέλου της εφαρμογής. Στο μοντέλο αυτό συγκεντρώνεται η εργασία των αναλυτών και των σχεδιαστών της εφαρμογής οι οποίοι καταγράφουν τις απαιτήσεις του οργανισμού και αναπαρίσταται οι απαιτήσεις αυτές με ένα σχέδιο. Το σχέδιο αυτό στη συνέχεια κυκλοφορεί και παρουσιάζεται σε όλους τους ενδιαφερόμενους, που σχετίζονται με την εφαρμογή.

Παρουσίαση

Η οντότητα που καθορίζει τι στοιχεία θα περιέχει η κάθε διάσταση. Κατασκευάζεται πάνω σε ένα Προϋπολογισμό ο οποίος καθορίζει τον αριθμό των διαστάσεων που θα περιέχει (από 1 έως 7) καθώς και τον Τύπο που θα έχει η κάθε διάσταση. Για κάθε διάσταση της Παρουσίασης επιλέγουμε μια συγκεκριμένη Δομή, που να ανήκει στον Τύπο της κάθε διάστασης. Με την Παρουσίαση έχουμε την δυνατότητα να ομαδοποιήσουμε διαφορετικά στοιχεία (Πελάτες, Έξοδα, Είδη κλπ.) κάτω από ένα κοινό όνομα.

Περίοδος

Η διάσταση του χρόνου βάσει του οποίου θέλουμε να γίνει η παρουσίαση των δεδομένων. Για παράδειγμα έχουμε μηνιαία παρουσίαση (12 μήνες του έτους), κυλιόμενες περιόδους (από κάποιο Μήνα/Ετος έως κάποιο Μήνα/Ετος) ή μια Δομή Περιόδου (ιεραρχική παρουσίαση του χρόνου, που έχει δημιουργηθεί από τον χρήστη, π.χ. προβολή ανά ΕΤΟΣ-ΤΕΤΡΑΜΗΝΑ-ΜΗΝΑΣ).

Πολυμορφισμός (Polymorphism)

Η αρχή του αντικειμενοστραφούς προγραμματισμού που σχετίζεται με την ιδιότητα των αντικειμένων να υλοποιούν με διαφορετικό τρόπο τις μεθόδους που περιέχουν, που έχουν ίδιο όνομα με μεθόδους άλλων αντικειμένων.

Προϋπολογισμοί

Ένας «καμβάς» πάνω στον οποίο δημιουργούμε από 1 έως 7 διαστάσεις του προϋπολογισμού που περιλαμβάνουν συγκεκριμένους Τύπους. Ένας Προϋπολογισμός δεν εκτελεί κάποια συγκεκριμένη λειτουργία αλλά χρησιμοποιείται για να κατασκευαστούν πάνω σε αυτόν οι Παρουσιάσεις που χρησιμοποιούνται στις εφαρμογές.

Προϋπολογισμός (Budgeting)

Ένα χρήσιμο εργαλείο για βραχυπρόθεσμο προγραμματισμό και έλεγχο σε ένα οργανισμό. Ένας λειτουργικός προϋπολογισμός (operational budget) συνήθως αφορά στο τρέχον ημερολογιακό έτος και παρουσιάζει των προγραμματισμό των εσόδων και των εξόδων για τον οργανισμό για το έτος αυτό.

Ρόλος

Ο ρόλος που έχει κάθε χρήστης που συνδέεται στην εφαρμογή. Βάσει του ρόλου ορίζεται τι έχει δικαίωμα να βλέπει ο χρήστης. Ένας ρόλος μπορεί να είναι μια συγκεκριμένη οντότητα (ΔΙΕΥΘΥΝΤΗΣ, ΥΠΕΥΘΥΝΟΣ, ΧΡΗΣΤΗΣ) ή κάποιο τμήμα του οργανογράμματος του οργανισμού (ΤΜΗΜΑ ΠΩΛΗΣΕΩΝ, ΛΟΓΙΣΤΗΡΙΟ που περιέχει μέσα του διάφορους ρόλους).

Στρατηγικός Προγραμματισμός

Η δραστηριότητα στην οποία αποφασίζονται τα κύρια βήματα που θα ακολουθηθούν καθώς και το προβλεπόμενο ύψος των πόρων που θα χρησιμοποιηθούν για να υλοποιηθεί η στρατηγική που έχει καθοριστεί για τον οργανισμό.

Στρατηγικός Χάρτης

Μια γενική αναπαράσταση του τρόπου με τον οποίο συνδέονται οι συνιστώσες της BSC και οι στόχοι του οργανισμού. Είναι μια οπτική αναπαράσταση των σχέσεων αιτίας-αποτελέσματος μεταξύ των τμημάτων της στρατηγικής του οργανισμού. Ένας Στρατηγικός Χάρτης είναι ένας ενιαίος και συνεπής τρόπος περιγραφής της στρατηγικής ενός οργανισμού έτσι ώστε οι στρατηγικοί στόχοι και τα στρατηγικά έργα του οργανισμού να καθορίζονται και να διαχειρίζονται.

Συνταύτιση στόχων (Goal Congruence)

Το κεντρικό σημείου ενός Συστήματος Ελέγχου Διοίκησης είναι η εξασφάλιση της συνταύτισης στόχων μεταξύ του οργανισμού και των εμπλεκομένων σε αυτόν (διεύθυνση, εργαζόμενοι κλπ.). Αυτό σημαίνει ότι οι ενέργειες που κάνουν τα άτομα για να επιτύχουν τους προσωπικούς τους στόχους συμφωνούν με τις ενέργειες που γίνονται από πλευράς του οργανισμού για την επίτευξη των στόχων του.

Τύποι

Μια γενική οντότητα του συστήματος όπως Πελάτες, Είδη, Έξοδα. Ένας Τύπος δεν εκτελεί κάποιες συγκεκριμένες λειτουργίες αλλά χρησιμοποιείται σαν βάση για την δημιουργία άλλων οντοτήτων όπως οι Δομές.

Εισαγωγή

Ένα από τα βασικά προβλήματα, που αντιμετωπίζει κάθε σύγχρονος οργανισμός είναι ο προσδιορισμός της στρατηγικής του και του βαθμού υλοποίησης των στρατηγικών στόχων που έχει θέσει η διοίκηση. Προκειμένου να υλοποιηθεί η στρατηγική ενός οργανισμού, απαιτείται να βρεθεί ένας τρόπος επικοινωνίας και παρουσίασης της στρατηγικής στους εργαζόμενους, προκειμένου αυτοί να δουλέψουν τόσο για την επίτευξη των προσωπικών στόχων όσο και για των στόχων του οργανισμού.

Ο όρος **Management Control – Έλεγχος Διοίκησης** αναφέρεται στην διαδικασία σύμφωνα με την οποία οι διευθυντές μιας επιχείρησης επηρεάζουν τα άλλα μέλη του οργανισμού έτσι ώστε να αναπτύξουν τις στρατηγικές του οργανισμού. Έλεγχος Διοίκησης είναι όλες εκείνες οι διαδικασίες και οι λειτουργίες που χρησιμοποιούν οι διευθυντές έτσι ώστε να πετύχουν την υλοποίηση των στόχων τους και ταυτόχρονα τους στόχους του οργανισμού. Σκοπός του Ελέγχου Διοίκησης είναι να λάβουν οι διευθυντές του οργανισμού τις κατάλληλες αποφάσεις για τον οργανισμό, να διαχειριστούν τους πόρους του οργανισμού αποδοτικά έτσι ώστε να επιτευχθούν οι στόχοι που έχουν τεθεί. Έλεγχος Διοίκησης επομένως είναι ένα σύστημα ελέγχου της υλοποίησης της προκαθορισμένης στρατηγικής ενός οργανισμού.

Πέρα από το πρόβλημα του Ελέγχου Διοίκησης, ένα άλλο πρόβλημα που αντιμετωπίζουν οι σύγχρονοι οργανισμοί είναι αυτό της ανάπτυξης εφαρμογών για την υποστήριξη των εσωτερικών λειτουργιών τους. Στο περιβάλλον δραστηριοποίησης των σύγχρονων επιχειρήσεων το περιβάλλον αλλάζει ολοένα και πιο γρήγορα και οι απαιτήσεις του ανταγωνισμού για την διατήρηση των επιχειρήσεων στην αγορά είναι όλο και πιο μεγάλες. Οι επιχειρήσεις πρέπει να λαμβάνουν υπόψη τους κάθε φορά τις απαιτήσεις των χρηστών και να προβαίνουν στις απαραίτητες αλλαγές στα συστήματα τους.

Υπάρχει λοιπόν η απαίτηση για ευέλικτα συστήματα, ώστε να διαφοροποιούνται στον μικρότερο δυνατό χρόνο. Οι εφαρμογές πρέπει να είναι σχεδιασμένες με τέτοιο τρόπο ώστε να μπορούν εύκολα να επεκταθούν. Μια λύση που προτείνεται στο πρόβλημα αυτό, μέσα από την διεθνής βιβλιογραφία και τις διάφορες τεχνικές ανάπτυξης εφαρμογών είναι η ανάπτυξη αντικειμενοστραφών εφαρμογών βασισμένες στην **N-tier Αρχιτεκτονική**. Η Αρχιτεκτονική N-tier αναφέρεται στην σχεδίαση της αρχιτεκτονικής μιας εφαρμογής ενώ ο **Αντικειμενοστραφής προγραμματισμός** αναφέρεται σε μια τεχνική ανάπτυξης του κώδικα της εφαρμογής με την χρήση αντικειμένων (κλάσεων).

Απαραίτητη προϋπόθεση για την επιτυχία του Αντικειμενοστραφούς Προγραμματισμού και του σχεδιασμού της Αρχιτεκτονικής της εφαρμογής, είναι η δημιουργία του **μοντέλου της εφαρμογής**. Ένα μοντέλο είναι το σχέδιο της εφαρμογής, που εκτός των άλλων δείχνει στους εμπλεκομένους την χρήση των αντικειμένων, που χρησιμοποιούνται στην εφαρμογή και την αλληλεπίδραση τους, περιγράφει με οπτικό και συνοπτικό τρόπο τις βασικές λειτουργίες της εφαρμογής και παρουσιάζει τον τρόπο με τον οποίο είναι οργανωμένη η εφαρμογή. Με την κατασκευή ενός μοντέλου της εφαρμογής, συγκεντρώνεται η εργασία των αναλυτών και των σχεδιαστών της εφαρμογής, οι οποίοι καταγράφουν τις απαιτήσεις του οργανισμού και αναπαρίσταται οι απαιτήσεις αυτές με ένα σχέδιο. Αυτή η διαδικασία ονομάζεται **Μοντελοποίηση**

Ένα μοντέλο της εφαρμογής είναι μια αφαιρετική παρουσίαση των κυρίων τμημάτων της εφαρμογής, μια προσεγγιστική αναπαράσταση της εφαρμογής, προτού αυτή αρχίσει να αναπτύσσεται, γεγονός που μειώνει την πολυπλοκότητα της ανάπτυξης της και βελτιώνεται η επικοινωνία, ο σχεδιασμός και η αξιολόγηση της εφαρμογής.

Η έννοια της **Οπτικής Μοντελοποίησης (Visual Modeling)** σχετίζεται με την διαδικασία της αναπαράστασης της πληροφορίας, που παρέχεται από το μοντέλο και από την αρχιτεκτονική της εφαρμογής, με γραφικό τρόπο χρησιμοποιώντας ένα τυποποιημένο σύνολο γραφικών στοιχείων. Η τυποποίηση είναι ένας κρίσιμος παράγοντας επιτυχίας της οπτικής Μοντελοποίησης, γιατί καθιερώνεται ένας κοινός τρόπος ανταλλαγής πληροφοριών. Θα ήταν μεγάλο πλεονέκτημα για ένα οργανισμό να μπορούσε να χρησιμοποιήσει ένα τυποποιημένο μηχανισμό καταγραφής και διάδοσης των απαιτήσεων και της ανάλυσης εφαρμογών έτσι ώστε να μπορέσει να αυξήσει την ποιότητα και να μειώσει τον χρόνο ανάπτυξης των εφαρμογών. Ο μηχανισμός αυτός είναι η οπτική μοντελοποίηση των αναπτυσσόμενων εφαρμογών. Ένας τέτοιος μηχανισμός οπτικής μοντελοποίησης είναι η **UML (Unified Modeling Language – Ενοποιημένη Γλώσσα Μοντελοποίησης**.

Τα παραπάνω προβλήματα αντιμετωπίζει και η βιομηχανία παραγωγής ορών ΒΙΟΣΕΡ, η οποία είναι μια. Το πρόβλημα που αντιμετωπίζει η εταιρεία εστιάζεται σε δύο ζητήματα. Πρώτον, η επίλυση του **προβλήματος του εσωτερικού ελέγχου** του οργανισμού και η παρουσίαση μιας διαδικασίας ελέγχου διοίκησης μέσω ενός μοντέλου ελέγχου. Δεύτερον, η δημιουργία μιας **μεθοδολογίας ανάλυσης και κατασκευής αντικειμενοστραφών εφαρμογών**. Στόχος είναι να επωφεληθεί ο οργανισμός από τα πλεονεκτήματα, που προσφέρουν οι Αντικειμενοστραφείς εφαρμογές καθώς και η κατασκευή εφαρμογών στηριζόμενες στην N-tier Αρχιτεκτονική.

Η παρούσα εργασία αποτελείται από δύο βασικά μέρη. Το **Πρώτο Μέρος**, είναι το θεωρητικό υπόβαθρο πάνω στο οποίο θα στηριχθεί το πρόβλημα και η επίλυση του.

Στο **Κεφάλαιο 1** παρουσιάζονται ορισμένες πληροφορίες σχετικά με την εταιρεία ΒΙΟΣΕΡ.

Στο **Κεφάλαιο 2** παρουσιάζονται τα θέματα σχετικά με τον Αντικειμενοστραφή Προγραμματισμό, όπως η σύγκριση των δύο τύπων ανάπτυξης εφαρμογών, του Δομημένου και του Αντικειμενοστραφούς Προγραμματισμού καθώς και οι βασικές αρχές του Αντικειμενοστραφούς Προγραμματισμού.

Στο **Κεφάλαιο 3** παρουσιάζονται τα θέματα σχετικά με την Αρχιτεκτονική Εφαρμογών. Παρουσιάζονται οι προηγούμενες αρχιτεκτονικές εφαρμογών που υπήρχαν τα περασμένα έτη (1-tier και 2-tier Αρχιτεκτονικές) και η N-tier Αρχιτεκτονική, με τα πλεονεκτήματα που προσφέρει η αρχιτεκτονική αυτή στην ανάπτυξη σύγχρονων εφαρμογών. Επίσης παρουσιάζεται η σχέση ανάμεσα στον Αντικειμενοστραφή προγραμματισμό και την N-tier Αρχιτεκτονική καθώς επίσης αναφέρονται τα πλεονεκτήματα της Μοντελοποίησης των εφαρμογών.

Ένα εργαλείο μοντελοποίησης και ανάπτυξης εφαρμογών είναι η Unified Modeling Language (UML), που παρουσιάζεται στο **Κεφάλαιο 4**. Στο κεφάλαιο αυτό παρουσιάζονται αναλυτικά τα διαγράμματα της UML, η σειρά χρήσης και οι σχέσεις μεταξύ των διαγραμμάτων αυτών καθώς και τα πλεονεκτήματα που παρέχονται από την χρήση της UML στην ανάπτυξη αντικειμενοστραφών εφαρμογών. Το κεφάλαιο αυτό κλείνει και το πρώτο μέρος της παρουσίασης της θεωρίας.

Το **Δεύτερο Μέρος**, που ασχολείται με την επίλυση του προβλήματος, περιλαμβάνει τα παρακάτω κεφάλαια.

Το **Κεφάλαιο 5** παρουσιάζει το πρόβλημα που αντιμετωπίζει η εταιρεία ΒΙΟΣΕΡ και τον τρόπο, με τον οποίο θα αναλυθεί και θα αντιμετωπιστεί.

Στο **Κεφάλαιο 6** παρουσιάζονται κάποια εισαγωγικά τα θέματα σχετικά με τα Συστήματα Ελέγχου Διοίκησης (Management Control Systems). Αναφέρονται οι γενικές έννοιες και τα

χαρακτηριστικά των συστημάτων αυτών. Στην συνέχεια παρουσιάζεται το περιβάλλον του ελέγχου διοίκησης, που περιλαμβάνει την παρουσίαση της στρατηγικής των επιχειρήσεων, την συμπεριφορά των επιχειρήσεων (συνταύτιση στόχων και διαδικασία ελέγχου – Formal Control Process) καθώς και τα είδη των Κέντρων Ευθύνης σε ένα οργανισμό (Κέντρα Εσόδων, Εξόδων, Κέρδους και Επένδυσης). Τέλος, παρουσιάζεται η Διαδικασία Ελέγχου Διοίκησης, που περιλαμβάνει τον Στρατηγικό Προγραμματισμό, την χρήση Προϋπολογισμών, και την Μέτρηση και Αποτίμηση Απόδοσης του οργανισμού. Επίσης γίνεται αναφορά στη θεωρία αναφορικά με την Balanced Scorecard.

Το **Κεφάλαιο 7** παρουσιάζει την επίλυση του προβλήματος. Η επίλυση του προβλήματος αναλύεται σε δύο ζητήματα. Το πρώτο παρουσιάζει μια Διαδικασία Ελέγχου Διοίκησης. Η διαδικασία αυτή περιλαμβάνει την παρουσίαση του Στρατηγικού Χάρτη, των στρατηγικών στόχων και έργων, την μοντελοποίηση του οργανισμού, τον προϋπολογισμό και τον έλεγχο και αποτίμηση της λειτουργίας των Κέντρων Ευθύνης στην ΒΙΟΣΕΡ. Αποτέλεσμα της διαδικασίας είναι ένα Θεωρητικό Μοντέλο, που περιλαμβάνει εφαρμογές, οι οποίες επιλύουν το πρόβλημα του Ελέγχου Διοίκησης. Το δεύτερο ζήτημα παρουσιάζει μια διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών (χαρακτηριστικά και μοντέλο διαδικασίας). Επίσης παρουσιάζεται ένα παράδειγμα χρήσης της διαδικασίας με την UML (υλοποίηση εφαρμογής).

Στο τελευταίο **Κεφάλαιο 8** παρουσιάζονται τα συμπεράσματα, που προκύπτουν από την παρούσα εργασία καθώς επίσης αναφέρονται τυχών προτάσεις για περαιτέρω έρευνα.

Σκοπός της παρούσας εργασίας είναι η παρουσίαση μερικών προβλημάτων, που αντιμετωπίζουν οι σύγχρονοι οργανισμοί, όπως η εφαρμογή μιας διαδικασίας ελέγχου διοίκησης για τον εσωτερικό έλεγχο του οργανισμού και την υλοποίηση της στρατηγικής για την επίτευξη των στρατηγικών στόχων του οργανισμού. Η εφαρμογή μιας διαδικασίας, όμως σε οποιονδήποτε οργανισμό συνεπάγεται την ανάπτυξη μιας αντίστοιχης εφαρμογής, η οποία θα υποστηρίζει την εκάστοτε διαδικασία. Έτσι, αναφέρονται οι τεχνικές υλοποίησης εφαρμογών (όπως είναι ο Αντικειμενοστραφής Προγραμματισμός βασισμένος στην N-tier Αρχιτεκτονική) καθώς επίσης και τα πλεονεκτήματα που παρέχει η μοντελοποίηση εφαρμογών με την χρήση ενός σύγχρονου προτύπου, όπως είναι η UML.

Η καταγραφή της θεωρίας έγινε μέσα από την ανάγνωση της ξένης, κυρίως βιβλιογραφίας αναφορικά με τα παραπάνω θέματα και έγινε προσπάθεια καταγραφής των βασικότερων εννοιών. Η παρουσίαση των εννοιών έγινε με τέτοιο τρόπο, ώστε να φαίνεται η πρακτική χρήση τους και να μπορέσει ο αναγνώστης να δει τις τεχνικές ανάλυσης και ανάπτυξης εφαρμογών βασισμένες υπαρκτά προβλήματα της εταιρείας ΒΙΟΣΕΡ, στην οποία εργάζομαι. Στην διαδικασία χρήσης της UML παρουσιάζεται ένα πραγματικό πρόβλημα ανάπτυξης εφαρμογής, που αντιμετωπίσαμε στο Τμήμα Πληροφορικής της εταιρείας, εγώ και οι συνάδελφοι μου με τους οποίους έχω τη τιμή να συνεργάζομαι. Ελπίζω το τελικό αποτέλεσμα της εργασίας να ικανοποιεί τον αναγνώστη.

Αθήνα, Ιανουάριος 2006

Μέρος Α. Θεωρητικό Μέρος

Κεφάλαιο 1. Στοιχεία ΒΙΟΣΕΡ Α.Ε.

Ιστορία – Οργανωτική Δομή

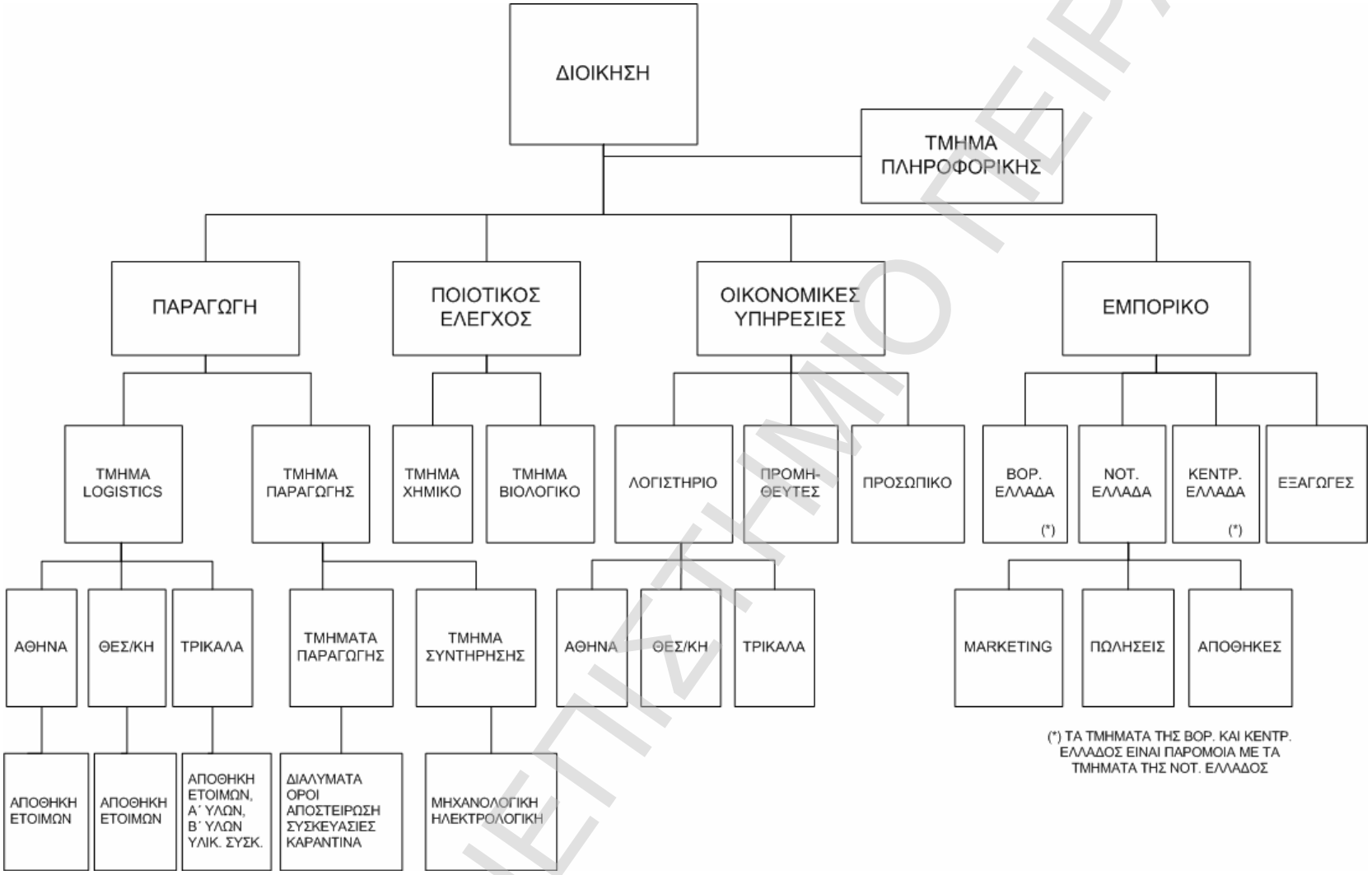
Η ΒΙΟΣΕΡ Α.Ε. – Βιομηχανία Παρασκευής Ορών ιδρύθηκε το 1980 στα Τρίκαλα από μια ομάδα νέων επιστημόνων με στόχο τη δημιουργία μιας πρότυπης Ελληνικής Βιομηχανίας φαρμάκων. Οι ιδιόκτητες εγκαταστάσεις της ολοκληρώθηκαν το 1983 (μια σύγχρονη παραγωγική μονάδα και αποθήκη σε ένα χώρο 36,000 m^2, 10 km περίπου έξω από τα Τρίκαλα, στην Κεντρική Ελλάδα). Η λειτουργία της εταιρείας ξεκινά τον Ιανουάριο του 1984.

Η εξαγωγική της δραστηριότητα αρχίζει το 1986, δύο χρόνια μετά τη λειτουργία της, η οποία αφορά κυρίως σε πωλήσεις ορών προς τη Γερμανία και την Κύπρο. Σήμερα οι πωλήσεις σε Ευρωπαϊκές χώρες και στην Κύπρο απορροφούν πάνω από το 30% της παραγωγής της.

Η ΒΙΟΣΕΡ Α.Ε. θεωρείται για τον κλάδο της πρωτοπόρος στην εισαγωγή νέας τεχνολογίας παρασκευής ορών στην Ελλάδα. Είναι πρωτοπόρος στην παραγωγή αποστειρωμένων ιατρικών βοηθημάτων μιας χρήσης κυρίως συσκευών έγχυσης ορών και μεταγγίσεως αίματος. Μέχρι σήμερα η εταιρεία κατέχει το μεγαλύτερο μερίδιο στην εσωτερική αγορά των παρεντερικών διαλυμάτων και συσκευών έκχυσης και είναι η μοναδική επιχείρηση στην Ελλάδα που παράγει διαλύματα περιτοναϊκής κάθαρσης (CAPD).

Η ΒΙΟΣΕΡ Α.Ε. είναι μια πολυμετοχική εταιρεία με τους ιδρυτές της να διατηρούν τον έλεγχο της εταιρείας. Το Διοικητικό Συμβούλιο απαρτίζεται από παλιούς και νέους μετόχους και εκπροσωπούν της πλειοψηφία του Μετοχικού Κεφαλαίου της εταιρείας.

Η οργανωτική δομή της εταιρείας παρουσιάζεται στο παρακάτω διάγραμμα.



Διάγραμμα 1: Οργανόγραμμα της ΒΙΟΣΕΡ Α.Ε.

Λειτουργίες εταιρείας ΒΙΟΣΕΡ

Στη συνέχεια παρουσιάζονται οι κύριες λειτουργίες της εταιρείας ΒΙΟΣΕΡ.

Παρασκευή & Ποιοτικός Έλεγχος

Η ΒΙΟΣΕΡ Α.Ε. είναι μια σύγχρονη φαρμακευτική παραγωγική μονάδα, εγκατεστημένη σε ένα χώρο 36.000m^2, 10km έξω από τα Τρίκαλα, στην Κεντρική Ελλάδα.

Η παραγωγή της βασίζεται στην υψηλών προδιαγραφών τεχνογνωσία και τεχνολογία ενός παγκόσμια αναγνωρισμένου φαρμακευτικού Γερμανικού οίκου. Παράγει παρεντερικά διαλύματα μικρού και μεγάλου όγκου, διαλύματα περιτοναϊκής κάθαρσης, ουρολογικά, έκπλυσης και συσκευές ενδοφλέβιας έγχυσης και μετάγγισης αίματος. Τα προϊόντα της παρασκευάζονται σύμφωνα με τους ισχύοντες κανόνες καλής παρασκευής (GMP) και η ποιότητά τους ανταποκρίνεται πλήρως στις προδιαγραφές του Παγκόσμιου Οργανισμού Υγείας.

Η παραγωγή των παρεντερικών διαλυμάτων μεγάλου όγκου, γίνεται με την τεχνολογία blow-fill-seal, μέσω του αυτοματοποιημένου συστήματος Bottlepack που εξασφαλίζει: άριστες συνθήκες παραγωγής – μηδενικός βαθμός σωματιδιακής και μικροβιακής επιμόλυνσης. Η ικανοποίηση του πελάτη στην εγχώρια και διεθνή αγορά, είναι για τη ΒΙΟΣΕΡ Α.Ε. στοιχείο κλειδί για τον καθορισμό του πρωταρχικού της στόχου "παραγωγή ασφαλών προϊόντων, υψηλής και σταθερής ποιότητας με το χαμηλότερο δυνατό κόστος".

Για την επίτευξη αυτού του στόχου:

- Το σύστημα διασφάλισης ποιότητας που εφαρμόζει η ΒΙΟΣΕΡ Α.Ε. ενεργοποιείται συνεχώς, βάσει γραπτών διαδικασιών καθορισμένων σύμφωνα με τις απαιτήσεις των κανόνων GMP και των προτύπων ISO 9001-9004.
- Ο έλεγχος για την επιβεβαίωση αξιοπιστίας των εγκαταστάσεων, συστημάτων, διαδικασιών και μεθόδου (Qualification / Validation) αποτελεί για τη ΒΙΟΣΕΡ Α.Ε. αναγκαιότητα και εκτελείται σε κάθε περίπτωση

Για την παραγωγή, τον έλεγχο και τη διακίνηση των προϊόντων της η ΒΙΟΣΕΡ Α.Ε., κατέχει τα πιστοποιητικά ποιότητας EN ISO 9002, EN 46002. Για τα ιατρικά και τεχνολογικά προϊόντα της, κατέχει τη σήμανση CE και πιστοποιητικό συμμόρφωσης με την Υπουργική απόφαση Ε3/833/899.

Τεχνολογία & Ανάπτυξη

Η ΒΙΟΣΕΡ ΑΕ επικεντρώνει τις προσπάθειές της τόσο στην παρασκευή και εξαγωγή προϊόντων που η ίδια παράγει με το σύγχρονο Γερμανικό know-how αλλά και στην δημιουργία Τμήματος Έρευνας.

Έχει χρηματοδοτήσει μέχρι σήμερα ερευνητικά προγράμματα στο Πανεπιστήμιο Κρήτης με στόχο την βελτίωση των προϊόντων της, την προώθηση νέων τεχνολογιών και την ανάπτυξη πρωτοποριακών και εξειδικευμένων διαλυμάτων. Έχει παράγει επιτυχώς καρδιοπληγικά διαλύματα για τις ανάγκες ενός από τα μεγαλύτερα καρδιοχειρουργικά κέντρα της Ελλάδας.

Η στενή συνεργασία με καταξιωμένες στον κλάδο της εταιρίες του εξωτερικού, την κάνει να πρωτοστατεί σε τεχνολογικές εφαρμογές και επενδύει σε ήδη δοκιμασμένη μεθοδολογία και πείρα. Η περαιτέρω αυτόνομη ανάπτυξή της είναι η απόδειξη της σημασίας που δίνει σε

μελέτες ακόμη και ανεξάρτητων ερευνητών και επιστημόνων, και η προσπάθεια για μελλοντικές συνεργασίες στον τομέα της έρευνας και ανάπτυξης βοηθά στον κύριο στόχο της.

Η ΒΙΟΣΕΡ Α.Ε. βασίζεται στο εκπαιδευμένο και υπεύθυνο προσωπικό της, στις υπερσύγχρονες εγκαταστάσεις της, στην πλήρως εξοπλισμένη παραγωγική μονάδα και στην συνεργασία της με σημαντικά νοσοκομεία και ιατρικά κέντρα.

Προϊόντα

Τα προϊόντα της ΒΙΟΣΕΡ είναι τα παρακάτω:

- Διαλύματα Ενδοφλέβιας Έγχυσης (παρεντερικά διαλύματα για ενδοφλέβια χρήση).
- Διαλύματα έκπλυσης (διαλύματα έκπλυσης για εξωτερική χρήση).
- Ιατρικά συστήματα έγχυσης υγρών (ιατρικά συστήματα μιας χρήσης για ενδοφλέβια έγχυση διαλυμάτων και μετάγγιση αίματος).
- Ουρολογικά διαλύματα έκπλυσης (για διεγχειρητική και μετεγχειρητική χρήση).
- Ουρολογικά συστήματα (για ουρολογικές εγχειρήσεις).
- CAPD (Διαλύματα Συνεχούς Φορητής Περιτοναϊκής Κάθαρσης).
- Προϊόντα συνεργαζόμενων εταιρειών. Η ΒΙΟΣΕΡ λειτουργεί ως αποκλειστική αντιπρόσωπος των παρακάτω εταιρειών και είναι υπεύθυνη για την διακίνηση των προϊόντων τους:
    - B|BRAUN Melsungen AG
    - ICU Medical Inc.
    - VENETEC International.
    - MÖLNLYCKE Health Care AB
    - BARRIER

Τα προϊόντα της ΒΙΟΣΕΡ χωρίζονται σε 2 κατηγορίες. Τα **παραγόμενα προϊόντα** (παραγωγή τους στο εργοστάσιο της εταιρείας στα Τρίκαλα) και τα **εισαγόμενα προϊόντα** (τα προϊόντα των προαναφερθέντων συνεργαζόμενων εταιρειών.)

Τα παραγόμενα προϊόντα χωρίζονται σε υποκατηγορίες ανάλογα με την φύση τους, δηλαδή σε Διαλύματα, Ορούς, Φύσιγγες, Συσκευές, Σάκοι, Εξαρτήματα κλπ.

Logistics & Δίκτυο Διανομής

Ένας από τους λόγους που έκαναν την ΒΙΟΣΕΡ ΑΕ να ξεχωρίσει από το ξεκίνημά της και να καταφέρει σήμερα να κατέχει το μεγαλύτερο μερίδιο της αγοράς σε παρεντερικά διαλύματα στον Ελλαδικό χώρο αλλά και να συνάπτει συνεργασίες με διεθνείς φαρμακευτικές εταιρείες είναι το γεγονός ότι παρέχει υπηρεσίες υψηλής ποιότητας με υπευθυνότητα και συνέπεια. Από την αρχή δημιούργησε ένα πολύ καλό δίκτυο διανομής με αποθήκες σε θέσεις κλειδιά δίνοντας πάντα έμφαση στην άμεση εξυπηρέτηση της πελατείας της.

Με την πάροδο του χρόνου και ακολουθώντας πάντα τις ανάγκες της η ΒΙΟΣΕΡ ΑΕ εκσυγχρονίζει συνεχώς αποθήκες, μηχανήματα, πληροφοριακά συστήματα και ανθρώπινο δυναμικό. Σήμερα η ΒΙΟΣΕΡ ΑΕ με την υποστήριξη υψηλού τεχνολογικά Logistic συστήματος και με αποθηκευτικούς χώρους στα Τρίκαλα (κεντρική αποθήκη 3700 παλέτες - 3500 m^2), Αθήνα και Θεσσαλονίκη (περιφερειακές 300 & 250 παλέτες αντίστοιχα - 600 & 350 m^2) είναι ικανή να καλύψει όλα τα νοσοκομεία της Ελληνικής επικράτειας στο μικρότερο δυνατό χρόνο παράδοσης, 24 ώρες για την ηπειρώτικη Ελλάδα και 48 ώρες για τα νησιά.

Οι πολύ καλές σχέσεις που έχουμε αναπτύξει με τους πελάτες είναι για εμάς η απόδειξη αλλά και ταυτόχρονα επιβράβευση της προσπάθειάς μας ώστε να συνεχίσουμε απερίσπαστα το έργο μας.

Τμήμα Πωλήσεων και Marketing

Το τμήμα Marketing και Πωλήσεων της ΒΙΟΣΕΡ Α.Ε. είναι χωρισμένο σε 3 τμήματα, το καθένα από τα οποία έχει αναλάβει πελάτες σε διάφορες τοποθεσίες. Συγκεκριμένα έχουμε:

1. Τμήμα Πωλήσεων Βόρειας Ελλάδας, με έδρα το υποκατάστημα Θεσσαλονίκης, που έχει αναλάβει όλους τους πελάτες της Β. Ελλάδας.
2. Τμήμα Πωλήσεων Κεντρικής Ελλάδας, με έδρα το κεντρικό γραφείο των Τρικάλων, που έχει αναλάβει τους πελάτες της Θεσσαλίας και των νησιών των Σποράδων.
3. Τμήμα Πωλήσεων Νοτίου Ελλάδας, με έδρα το υποκατάστημα Αθηνών, που έχει αναλάβει και το μεγαλύτερο μέρος των πελατών της εταιρείας. Συγκεκριμένα το τμήμα έχει αναλάβει τους πελάτες του Λεκανοπεδίου Αττικής, της Στερεάς Ελλάδας, της Πελοποννήσου, της Κρήτης και των νησιών Ιουνίου και Αιγαίου κλπ.
4. Τέλος, οι πελάτες εξωτερικού εξυπηρετούνται από τα κεντρικά γραφεία στα Τρίκαλα.

Στόχος του τμήματος Πωλήσεων, πέρα φυσικά από τις πωλήσεις, που είναι ο αυτοσκοπός του, είναι η παροχή πληροφοριών σχετικά την πρόβλεψη της ζήτησης, πληροφορίες, που όπως θα δούμε, είναι εξαιρετικά σημαντικές για τον σχεδιασμό παραγωγής της εταιρείας.

Η πρόβλεψη της ζήτησης μπορεί να γίνει βάσει των παρακάτω στοιχείων:

1. Στατιστικά στοιχεία από προηγούμενες πωλήσεις.
2. Έρευνα αγοράς για παράδειγμα ανάγκη αγοράς κάποιων προϊόντων από νέα νοσοκομεία, νέες κλινικές, νέες φαρμακαποθήκες, νέα φαρμακεία κλπ.
3. Διαδικασία διείσδυσης στην αγορά, με την χρήση διαφόρων τρόπων όπως συνέδρια, επισκέψεις σε νοσοκομεία, διαφημιστικά φυλλάδια κα.
4. Παρακολούθηση επικαιρότητας και πληροφόρηση ανταγωνισμού (για παράδειγμα προβλήματα ανταγωνιστών, σχέδιά τους κλπ.)

Για την κάλυψη των αναγκών του τμήματος Πωλήσεων και Marketing, η ΒΙΟΣΕΡ διαθέτει ένα οργανωμένο και εκτεταμένο δίκτυο από αξιόλογους και έμπειρους πωλητές, με άρτια εκπαίδευση και κατάρτιση στον τομέα των πωλήσεων ιατρικών προϊόντων. Οι κινήσεις, οι στόχοι και τα στοιχεία των πωλητών είναι αναγκαία τόσο για τις ίδιες τις πωλήσεις της

εταιρείας αλλά και για τον σχεδιασμό της παραγωγής και την βέλτιστη διοίκηση των αποθεμάτων της εταιρείας.

Εξαγωγές

Η ΒΙΟΣΕΡ Α.Ε. αναπτύσσεται όχι μόνο στην εγχώρια αλλά και στη διεθνή αγορά. Η εξαγωγική δραστηριότητα της ΒΙΟΣΕΡ Α.Ε. αρχίζει το 1986, μόλις δηλαδή δύο χρόνια μετά την έναρξη της λειτουργίας της, η οποία αφορά κυρίως σε πωλήσεις παρεντερικών διαλυμάτων.

Η τεχνολογία αιχμής και η πιστοποίηση της εταιρείας μας σύμφωνα με τα διεθνή standards μας έδωσαν τη δυνατότητα να εισχωρήσουμε πολύ γρήγορα σε μεγάλες ξένες αγορές, είτε μέσω απευθείας registration των προϊόντων μας (Ρουμανία, Κύπρο, Αλβανία κ.λ.π.), είτε μέσω συμφωνιών για παραγωγή Under Contract Manufacturing (Γερμανία, Σκοτία κ.λ.π.).

Στους στρατηγικούς στόχους της εταιρείας μας, είναι η ανάπτυξη της εξαγωγικής μας δραστηριότητας με την εξεύρεση νέων πελατών στη διεθνή αγορά.

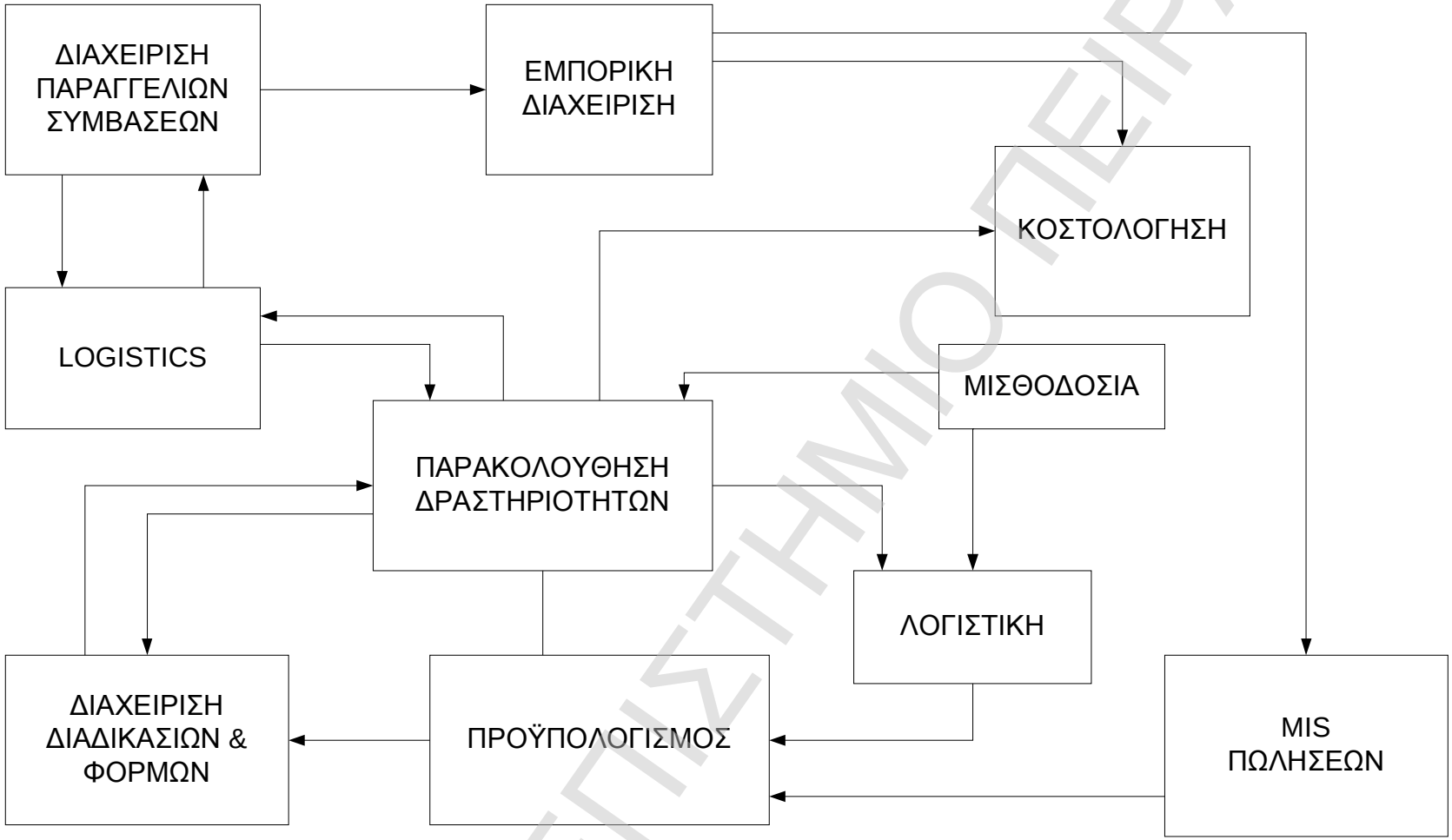
Πληροφοριακό Σύστημα Εταιρείας – Τμήμα Πληροφορικής

Η ΒΙΟΣΕΡ Α.Ε. ανταποκρινόμενη στις ανάγκες και τις απαιτήσεις για Νέες Τεχνολογίες που πρέπει να έχει μια σύγχρονη βιομηχανία, έχει επενδύσει στον τομέα της Πληροφορικής, με στόχο την ανάπτυξη ενός Ολοκληρωμένου Πληροφοριακού Συστήματος, που θα ικανοποιεί τις ανάγκες της και θα υποστηρίζει πλήρως όλες τις δραστηριότητές της. Το τμήμα Πληροφορικής της εταιρείας κατέχει πρωτεύοντα ρόλο στην οργανωτική δομή της εταιρείας. Μέσα από συνεχής έρευνα και αξιοποίηση των Νέων Τεχνολογιών και των δυνατοτήτων που παρέχονται από την εξέλιξη της Πληροφορικής, το Τμήμα Πληροφορικής έχει μελετήσει, σχεδιάσει και αναπτύξει όλες τις εφαρμογές πληροφορικής της εταιρείας.

Λειτουργία του τμήματος Πληροφορικής της εταιρείας είναι η υποστήριξη όλων των χρηστών των συστημάτων και η συνεχής βελτίωση των εφαρμογών, με στόχο την άριστη υποστήριξη των διαδικασιών της εταιρείας.

Οι εφαρμογές πληροφορικής είναι ξεχωριστά προγράμματα, που υποστηρίζουν τις λειτουργίες της ΒΙΟΣΕΡ. Οι εφαρμογές αυτές συνεργάζονται αρμονικά η μια με την άλλη με αποτέλεσμα ένα ολοκληρωμένο πληροφοριακό σύστημα που μπορεί εύκολα να συγκριθεί σε απόδοση και δυνατότητες με τα πληροφοριακά συστήματα που κυκλοφορούν στην αγορά.

Το διάγραμμα του Πληροφοριακού Συστήματος της εταιρείας παρουσιάζεται παρακάτω.



Διάγραμμα 2: Συστήματα Πληροφορικής της ΒΙΟΣΕΡ Α.Ε.

Διεθνείς Συνεργασίες

Η εξαιρετική ποιότητα των προϊόντων της ΒΙΟΣΕΡ Α.Ε. σύμφωνα με τα διεθνή standards και η σημαντική θέση που κατέχει στην ελληνική αγορά (Leadership στο νοσοκομειακό χώρο), έχουν σαν αποτέλεσμα να διασφαλισθούν πολλές συνεργασίες με οίκους του εξωτερικού, που διακρίνονται σε:

- Συμφωνίες αποκλειστικής διανομής στην Ελληνική αγορά οίκων του εξωτερικού (Exclusive Distributorship Agreement):
    - **B|Braun Melsungen AG**
    - **ICU Medical Inc.**
    - **VENETEC International**
    - **MÖLNLYCKE Health Care AB**
    - **BARRIER**
- Συμφωνίες παραγωγής προϊόντων στη ΒΙΟΣΕΡ Α.Ε. (Contract manufacturing) κορυφαίων πολυεθνικών του κλάδου
- Εξαγωγές των προϊόντων της ΒΙΟΣΕΡ Α.Ε., μετά από Registration, σε διάφορες χώρες όπως:
    - Κύπρος
    - Ρουμανία
    - Αλβανία
    - Κόσσοβο

Η ΒΙΟΣΕΡ Α.Ε. είναι επικεφαλής στο Νοσοκομειακό τομέα στην Ελλάδα και μπορεί να θέσει στόχο της την περαιτέρω βελτίωση του διεθνούς της προφίλ.

Κεφάλαιο 2. Αντικειμενοστραφής Προγραμματισμός

Το περιβάλλον δραστηριοποίησης των σύγχρονων επιχειρήσεων αλλάζει ολοένα και πιο γρήγορα και οι απαιτήσεις του ανταγωνισμού για την διατήρηση των επιχειρήσεων στην αγορά είναι όλο και πιο μεγάλες. Οι επιχειρήσεις πρέπει να λαμβάνουν υπόψη τους κάθε φορά τις απαιτήσεις των χρηστών και να προβαίνουν στις απαραίτητες τροποποιήσεις στα συστήματα τους. Βέβαια αυτό δεν σημαίνει ότι πρέπει να αλλάζει ολόκληρο το πληροφοριακό σύστημα, που λειτουργεί σε ένα οργανισμό αλλά η αλλαγή συγκεκριμένων τμημάτων (modules) ενός υπό-συστήματος για να αντεπεξέλθει στις καινούργιες απαιτήσεις των χρηστών.

Υπάρχει λοιπόν η απαίτηση για ευέλικτα συστήματα, ώστε αυτά να τροποποιούνται στον μικρότερο δυνατό χρόνο. Οι εφαρμογές πρέπει να είναι σχεδιασμένες με τέτοιο τρόπο ώστε να μπορούν εύκολα να επεκταθούν. Στο παρόν κεφάλαιο θα παρουσιάσουμε δύο είδη προγραμματισμού εφαρμογών, τον Δομημένο και τον Αντικειμενοστραφή. Θα αναφέρουμε τα χαρακτηριστικά κάθε είδους και θα κάνουμε μια σύγκριση των δύο αυτών τρόπων ανάπτυξης εφαρμογών. Τέλος, θα αναφέρουμε τις βασικές αρχές του αντικειμενοστραφούς προγραμματισμού καθώς και τα πλεονεκτήματα που προσφέρει.

Δομημένος Προγραμματισμός

Ο **δομημένος προγραμματισμός** (**structured programming**) ή αλλιώς ο **διαδικαστικός προγραμματισμός** (**procedural programming**), ο προγραμματισμός δηλαδή με την χρήση διαδικασιών και συναρτήσεων ήταν ο κυρίαρχος τρόπος ανάπτυξης λογισμικού τα προηγούμενα χρόνια. Σύμφωνα με τον δομημένο προγραμματισμό, τα πολύπλοκα προβλήματα χωρίζονταν σε μικρά τμήματα κάθε ένα από τα οποία έλυνε ένα τμήμα του γενικού προβλήματος, μειώνοντας την πολυπλοκότητα του αρχικού προβλήματος.

Οι προγραμματιστές με τον τρόπο αυτό ανέπτυσσαν δομημένα προγράμματα με αρχή, μέση και τέλος. Τα κύρια χαρακτηριστικά των προγραμμάτων ήταν η κλήση διαδικασιών, οι οποίες εκτελούσαν συγκεκριμένες λειτουργίες καθώς και η χαρακτηριστική δομή των προγραμμάτων αυτών, η οποία έπρεπε να ήταν σαφώς προκαθορισμένη. Η σχεδίαση των εφαρμογών γινόταν από πάνω προς τα κάτω («top-down»), σχεδιάζονταν δηλαδή αρχικά η γενική δομή της εφαρμογής και στην συνέχεια αναπτύσσονταν τα κομμάτια του κώδικα [29]. Ένα όμως σημαντικό θέμα που έπρεπε να αντιμετωπιστεί ήταν η δυνατότητα επαναχρησιμοποίησης κώδικα.

Οι προγραμματιστές δημιουργούσαν τμήματα κώδικα που αναλάμβαναν συγκεκριμένες λειτουργίες της εφαρμογής, όπως εκτύπωση, επεξεργασία συγκεκριμένων δεδομένων, έκδοση αναφορών κ.λ.π. και τα αποθήκευαν με την μορφή συναρτήσεων σε συγκεκριμένα αρχεία (τα λεγόμενα modules), τα οποία μπορούσαν να χρησιμοποιηθούν από διάφορες εφαρμογές. Αυτό είχε σαν πλεονέκτημα την μείωση του χρόνου συγγραφής των συγκεκριμένων τμημάτων κώδικα. Η χρήση όμως των modules είχε και κάποια **μειονεκτήματα**, κυρίως για τους προγραμματιστές, οι οποίοι έπρεπε να αναπτύσσουν τις εφαρμογές του οργανισμού αλλά και σε όσους ήθελαν να καταλάβουν την λογική των modules σχετικά με τις εφαρμογές, οι οποίες τα χρησιμοποιούν:

- Τα modules περιείχαν πληθώρα συναρτήσεων για όλες τις εφαρμογές στις οποίες χρησιμοποιούνταν με αποτέλεσμα την δυσκολία συντήρησης και τεκμηρίωσης του κώδικα.

- Η λογική των συναρτήσεων των modules έπρεπε να ήταν πολύ γενική, ανεξάρτητη από την εφαρμογή που τις θα χρησιμοποιούσε με αποτέλεσμα την δυσκολία ανάγνωσης του κώδικα (για να καταλάβουμε τι κάνει μια συγκεκριμένη συνάρτηση).
- Τα modules δεν είχαν κάποια συγκεκριμένη εσωτερική κατάσταση (κάποιες μεταβλητές οι οποίες είναι σημαντικές για την εφαρμογή και παίρνουν κάποιες τιμές). Τα δεδομένα ήταν ανεξάρτητα από τις συναρτήσεις και έπρεπε να υπάρχει αυστηρός έλεγχος σε όλες τις συναρτήσεις, οι οποίες άλλαζαν την τιμή των μεταβλητών, έτσι ώστε να μην υπάρχει ασυνέπεια στα δεδομένα και να μην αλλάζει η τιμή τους από λάθος
- Τα modules ήταν ανεξάρτητα το ένα από το άλλο και δεν σχετίζονταν μεταξύ τους έτσι ώστε να δίνουν κάποια πληροφορία σχετικά με την εφαρμογή, στην οποία χρησιμοποιούνταν.

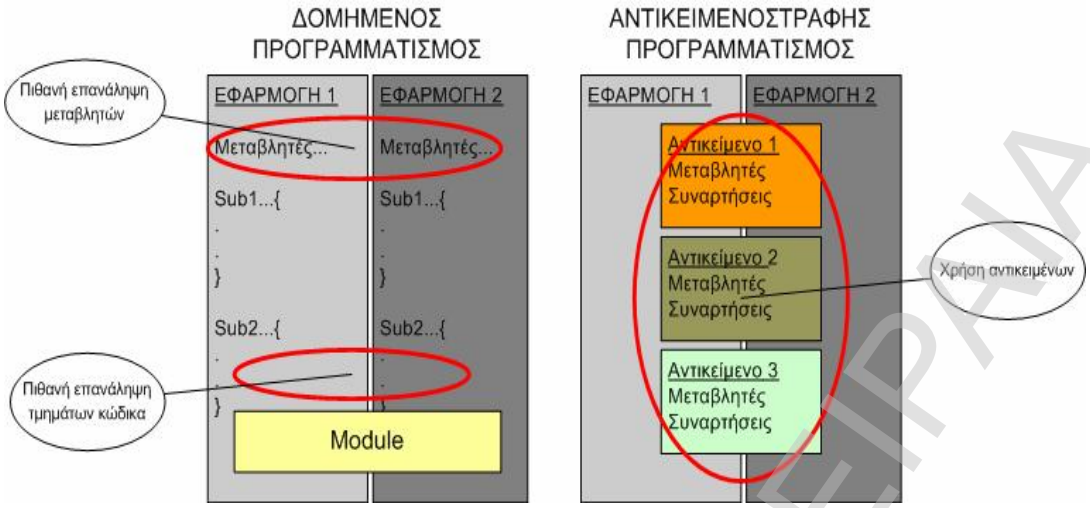
Πέρα από τα προβλήματα των modules, ο δομημένος προγραμματισμός παρουσίαζε και κάποια άλλα **προβλήματα**:

- Οι μεταβλητές ανήκαν στο επίπεδο της εφαρμογής και σε πολλές περιπτώσεις είχαμε την επανάληψη ίδιων μεταβλητών από διαφορετικές εφαρμογές (μεταβλητές που για τον ένα ή τον άλλο λόγο δεν μπορούσαν να τοποθετηθούν σε ένα module).
- Υπήρχαν συναρτήσεις οι οποίες γράφονταν σε περισσότερες από μια εφαρμογές για να εξυπηρετήσουν συγκεκριμένους σκοπούς. Οι συναρτήσεις αυτές δεν μπορούσαν να τοποθετηθούν σε ένα module και έπρεπε επομένως να ξαναγραφτούν σε όλες τις εφαρμογές.

Τα προβλήματα αυτά έρχεται να λύσει σε μεγάλο βαθμό ο **Αντικειμενοστραφής** (**Object Oriented – OO**) **Προγραμματισμός**.

Αντικειμενοστραφής Προγραμματισμός

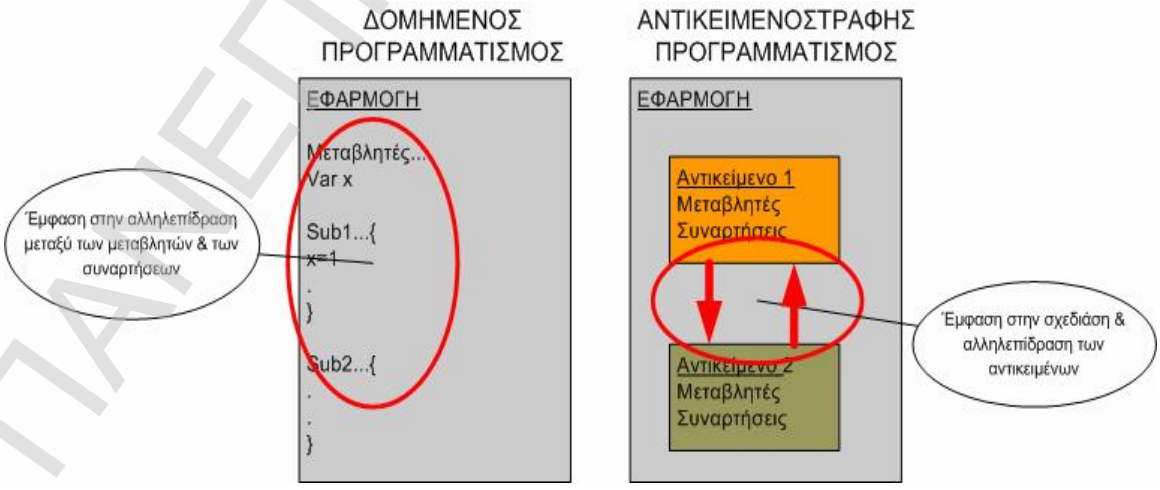
Στον Αντικειμενοστραφή Προγραμματισμό οι προγραμματιστές δημιουργούν ενότητες κώδικα με κοινά χαρακτηριστικά, που ονομάζονται αντικείμενα (objects). Με την χρήση των αντικειμένων γίνεται προσπάθεια αντιστοίχησης των εννοιών του πραγματικού κόσμου (για παράδειγμα πελάτες, προμηθευτές, είδη, τιμολόγια κλπ.) σε τμήματα κώδικα, που χρησιμοποιούνται στην εφαρμογή. Τα αντικείμενα που χρησιμοποιούνται έχουν το ιδιαίτερο χαρακτηριστικό ότι είναι ένα σύνολο από μεταβλητές και από συναρτήσεις, οι οποίες διαχειρίζονται τις μεταβλητές, κάτω από ένα κοινό όνομα.



Διάγραμμα 3: Δομημένος – Αντικειμενοστραφής Προγραμματισμός

Στον **δομημένο προγραμματισμό** δίνονταν έμφαση στην επικοινωνία μεταξύ των δεδομένων και των συναρτήσεων, που διαχειρίζονταν τα δεδομένα αυτά. Το σύστημα θεωρείται σαν ένα σύνολο από συναρτήσεις, οι οποίες μοιράζονται και αλλάζουν τα ίδια δεδομένα. Έπρεπε επομένως οι προγραμματιστές να ήταν πολύ προσεκτικοί στην διαχείριση των δεδομένων έτσι ώστε τα προγράμματα να λειτουργούν σωστά. Όσο όμως μεγάλωναν οι εφαρμογές, η αλληλεπίδραση μεταξύ των δεδομένων και των συναρτήσεων γινόταν ολοένα και πιο πολύπλοκη και ο έλεγχος πιο δύσκολος.

Στον **αντικειμενοστραφή προγραμματισμό** αντίθετα, δίνεται έμφαση στον σχεδιασμό των αντικειμένων και στην επικοινωνία ανάμεσα στα αντικείμενα, με την μορφή ανταλλαγής μηνυμάτων. Το σύστημα θεωρείται σαν ένα σύνολο αντικειμένων, που επικοινωνούν μέσω μηνυμάτων και εκτελούν συγκεκριμένες λειτουργίες. Τα αντικείμενα που ενσωματώνουν μέσα τους δεδομένα και συναρτήσεις θεωρούνται κρίσιμα στοιχεία της εφαρμογής και πρέπει να ελέγχεται η σχεδίαση τους και η επικοινωνία τους με τα άλλα αντικείμενα της εφαρμογής [26].



Διάγραμμα 4: Δομημένος – Αντικειμενοστραφής Προγραμματισμός (2)

Τα αντικείμενα που δημιουργούνται χρησιμοποιούνται σε κάθε εφαρμογή που αναπτύσσεται σε ένα οργανισμό. Ο αντικειμενοστραφής προγραμματισμός έχει κάποιες συγκεκριμένες αρχές, οι οποίες παρουσιάζονται στη συνέχεια.

Αρχές Αντικειμενοστραφούς Προγραμματισμού

Οι βασικές αρχές του Αντικειμενοστραφούς Προγραμματισμού είναι [2, 12, 16]:

- **Η Αφαίρεση (Abstraction)**: η αφαιρετική παρουσίαση μιας έννοιας του πραγματικού κόσμου παρουσιάζοντας μόνο τις λεπτομέρειες που δείχνουν πως θα χρησιμοποιήσουμε το αντικείμενο.
- **Η Ενθυλάκωση (Encapsulation) – ή Απόκρυψη Δεδομένων (Data Hiding)**: η ενσωμάτωση των χαρακτηριστικών (μεταβλητών) και τον μεθόδων (συναρτήσεων) σε κάθε αντικείμενο έτσι ώστε να εμφανίζονται μόνο οι πληροφορίες χρήσης του αντικειμένου και να κρύβονται οι πληροφορίες λειτουργίας του αντικειμένου.
- **Η Κληρονομικότητα (Inheritance)**: η ιδιότητα των αντικειμένων να κληρονομούν χαρακτηριστικά και μεθόδους από άλλα αντικείμενα.
- **Ο Πολυμορφισμός (Polymorphism)**: η ιδιότητα των αντικειμένων να υλοποιούν με διαφορετικό τρόπο τις μεθόδους τους, που έχουν ίδιο όνομα με μεθόδους άλλων αντικειμένων.

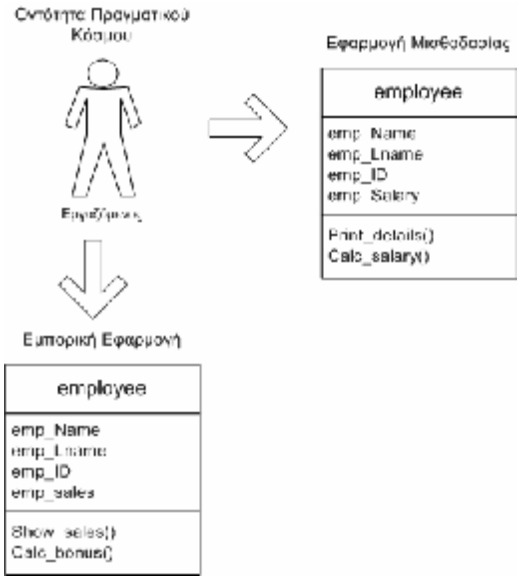
Στη συνέχεια παρουσιάζονται αναλυτικά οι βασικές αρχές του αντικειμενοστραφούς προγραμματισμού.

Αφαίρεση (Abstraction)

Αφαίρεση είναι η παρουσίαση μιας οντότητας του «πραγματικού κόσμου» εμφανίζοντας μόνο τις πληροφορίες που σχετίζονται με το πώς θα χρησιμοποιήσουμε το αντικείμενο αυτό. Η ποιότητα και η αξία της αφαίρεσης σχετίζεται με το πόσο καλά παρουσιάζουμε τα αντικείμενα του πραγματικού κόσμου σε σχέση με:

- τις πληροφορίες που συντηρούμε για το αντικείμενο αυτό (τι θέλουμε να ξέρουμε για το αντικείμενο αυτό) και
- τι θέλουμε να κάνουμε με αυτό το αντικείμενο.

Σε μια εφαρμογή ενός οργανισμού μια σωστή αφαίρεση θα πρέπει να περιέχει μόνο τις απαραίτητες πληροφορίες για την οντότητα, η οποία θα εξυπηρετήσει τις ανάγκες τις εφαρμογής. Έτσι για μια οντότητα δεν παρουσιάζονται περιττές πληροφορίες, που μπορεί να μην χρησιμοποιηθούν ποτέ, αλλά μόνο αυτές που απαιτούνται από την συγκεκριμένη εφαρμογή στην οποία χρησιμοποιείται το αντικείμενο.



Διάγραμμα 5: Παράδειγμα Αφαίρεσης

Για παράδειγμα, για μια οντότητα Εργαζόμενος, οι πληροφορίες που εμφανίζονται σε μια εφαρμογή Μισθοδοσίας θα είναι το ονοματεπώνυμο και ο μισθός του εργαζόμενου καθώς και κάποιες μέθοδοι, που διαχειρίζονται τον μισθό του. Αντίθετα, σε μια Εμπορική εφαρμογή πιθανόν να μην χρειάζεται να εμφανίζεται ο μισθός του εργαζόμενου αλλά άλλες πληροφορίες, όπως για παράδειγμα τα στοιχεία πωλήσεων του εργαζόμενου και άλλες μέθοδοι.

Ενθυλάκωση (Encapsulation) – Απόκρυψη Δεδομένων (Data Hiding)

Η **Ενθυλάκωση** δηλώνει ότι όταν σχεδιάζουμε ένα αντικείμενο πρέπει να ξεχωρίσουμε τι πρέπει να ξέρουμε για ένα αντικείμενο σύμφωνα με δύο έννοιες:

- την ελάχιστη πληροφορία, που χρειάζεται για να χρησιμοποιήσουμε ένα αντικείμενο
- την ελάχιστη πληροφορία, που απαιτείται για να κάνουμε το αντικείμενο να λειτουργήσει σωστά.

Πιθανόν να θεωρεί κάποιος ότι η έννοια της Ενθυλάκωσης είναι παρόμοια με την έννοια της Αφαίρεσης. Στην πραγματικότητα η ιδιότητα της Αφαίρεσης κρύβει τις πληροφορίες του αντικειμένου του πραγματικού κόσμου, που δεν απαιτείται να παρουσιάζονται σε μια εφαρμογή. Αντίθετα, η Ενθυλάκωση δηλώνει ότι ενσωματώνουμε τις ιδιότητες και τις μεθόδους ενός αντικειμένου μέσα στο ίδιο το αντικείμενο, παρουσιάζοντας μόνο εκείνα που χρειάζονται για να λειτουργήσει σωστά το αντικείμενο.

Σύμφωνα με την έννοια της ενθυλάκωσης από ένα αντικείμενο κρύβουμε τις πληροφορίες που σχετίζονται με την υλοποίηση του αντικειμένου (πληροφορίες για την εσωτερική λειτουργία του αντικειμένου) και εμφανίζουμε μόνο τις πληροφορίες που απαιτούνται για να χρησιμοποιηθεί το αντικείμενο. Οι πληροφορίες που εμφανίζονται στους χρήστες και σχετίζονται με την χρήση του ονομάζονται διεπαφή (interface) του αντικειμένου. Επομένως για να χρησιμοποιήσουμε ένα αντικείμενο παρουσιάζουμε την διεπαφή του στους χρήστες.

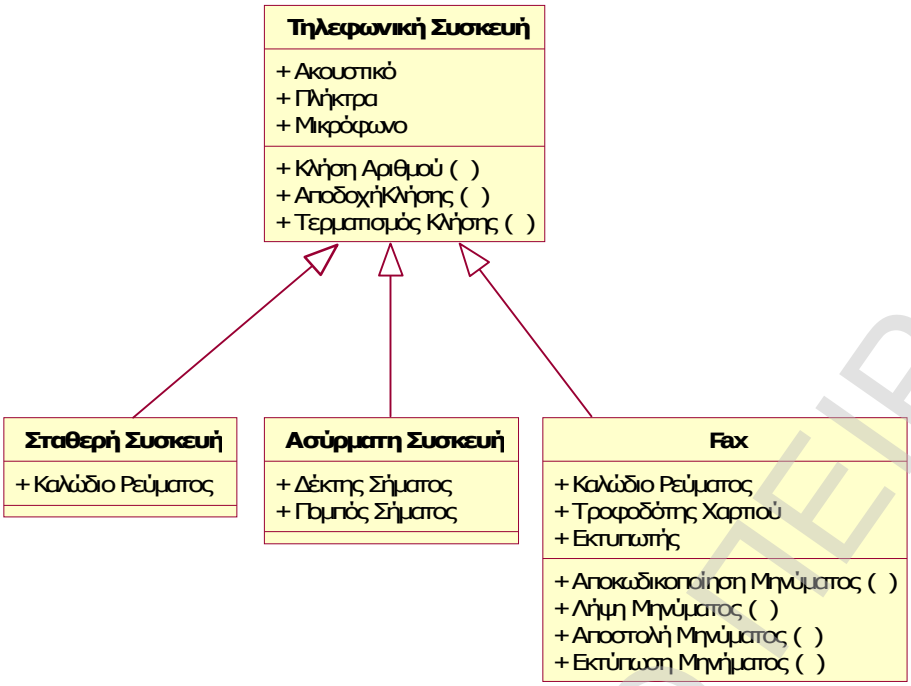
Μια διεπαφή ενός αντικειμένου από μόνη της δεν κάνει και πολλά πράγματα. Για να λειτουργήσει ένα αντικείμενο πρέπει να παρέχουμε στους χρήστες την απαραίτητη πληροφορία χρήσης του αντικειμένου. Αυτή η πληροφορία είναι οι μέθοδοι, οι συναρτήσεις δηλαδή, του αντικειμένου που χρησιμοποιούνται για να υλοποιηθεί η διεπαφή του. Σύμφωνα με τον ορισμό της ενθυλάκωσης μέσα σε ένα αντικείμενο ενθυλακώνουμε, τοποθετούμε δηλαδή, χαρακτηριστικά και λειτουργίες αποκρύπτοντας τα από τον «έξω κόσμο». Η επικοινωνία με τους χρήστες γίνεται μέσω της διεπαφής του αντικειμένου.

Η ενθυλάκωση λειτουργεί και για ένα επιπλέον σκοπό. Αποκρύπτουμε πληροφορίες υλοποίησης και δεδομένα από τους χρήστες ώστε αυτά να μην μπορούν να αλλάξουν χωρίς να απαιτείται και έτσι αλλοιώσουν την λειτουργία του αντικειμένου. Για το λόγο αυτό η ενθυλάκωση αναφέρεται και σαν **Απόκρυψη δεδομένων (Data hiding)**. Πρέπει να κρύβεται η πληροφορία μέσα στα αντικείμενα για να αποφεύγεται μη εξουσιοδοτημένη αλλαγή της γεγονός που μπορεί να οδηγήσει σε σφάλμα της λειτουργίας της εφαρμογής.

Κληρονομικότητα (Inheritance)

Η κληρονομικότητα, όπως γίνεται εύκολα αντιληπτό, αναφέρεται στον μηχανισμό που επιτρέπει την δημιουργία καινούργιων αντικειμένων βάσει έτοιμων αντικειμένων. Ένα αντικείμενο *παιδί* κληρονομεί τα χαρακτηριστικά ενός αντικειμένου *πατέρα*. Στον αντικειμενοστραφή προγραμματισμό αρχικά δημιουργούνται τα αντικείμενα πατέρας με συγκεκριμένα χαρακτηριστικά και συμπεριφορά (μεθόδους) και στη συνέχεια όπου απαιτείται δημιουργούνται τα αντικείμενα-παιδιά, που κληρονομούν τις ιδιότητες και την συμπεριφορά του αντικειμένου-πατέρας. Στην συνέχεια δημιουργούνται οι ιδιότητες και η συμπεριφορά που χαρακτηρίζει μοναδικά κάθε παιδί.

Ένα παράδειγμα κληρονομικότητας είναι το αντικείμενο Τηλεφωνική Συσκευή που έχει συγκεκριμένα χαρακτηριστικά, όπως Ακουστικό, Μικρόφωνο, Πλήκτρα κλπ. και εκτελεί συγκεκριμένες λειτουργίες, όπως Κλήση Αριθμού, Αποδοχή Κλήσης, Τερματισμός Κλήσης. Το αντικείμενο Τηλεφωνική Συσκευή στη συνέχεια κληρονομείται από άλλα αντικείμενα, όπως τα αντικείμενα Σταθερή Συσκευή, Ασύρματη Συσκευή, Fax. Κάθε αντικείμενο με τη σειρά του έχει και επιπλέον χαρακτηριστικά, που δεν έχουν τα άλλα αντικείμενα-παιδιά (για παράδειγμα η Σταθερή Συσκευή έχει καλώδιο τηλεφώνου, η Ασύρματη Συσκευή έχει δέκτη και πομπό σήματος, το Fax έχει τροφοδότη χαρτιού, εκτυπωτή και εκτελεί επιπλέον λειτουργίες όπως η αποκωδικοποίηση του μηνύματος, η εκτύπωση του μηνύματος, η αποστολή μηνύματος κλπ.).



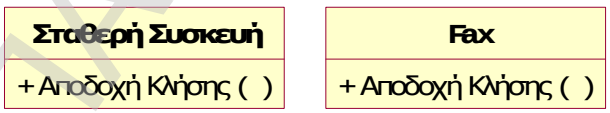
Διάγραμμα 6: Παράδειγμα Κληρονομικότητας

Το βασικό πλεονέκτημα της κληρονομικότητας είναι η ευκολία συντήρησης της εφαρμογής. Όταν απαιτείται η αλλαγή σε συγκεκριμένα αντικείμενα, που έχουν το ίδιο αντικείμενο-πατέρα, η αλλαγή γίνεται στον πατέρα και κληρονομείται με την σειρά της σε όλα τα αντικείμενα-παιδιά.

Πολυμορφισμός (Polymorphism)

Η έννοια του Πολυμορφισμού αναφέρεται στην δυνατότητα των αντικειμένων να έχουν λειτουργίες με ίδιο όνομα με άλλα αντικείμενα και το κάθε αντικείμενο να υλοποιεί την λειτουργία αυτή διαφορετικά ανάλογα με τον τύπο του. Πολυμορφισμός είναι δηλαδή η υλοποίηση της ίδιας μεθόδου/συνάρτησης με πολλούς διαφορετικούς τρόπους.

Για παράδειγμα, για τα δύο αντικείμενα Σταθερή Συσκευή τηλεφώνου και Fax, έχουμε την ίδια συνάρτηση Αποδοχή Κλήσης, που διαχειρίζεται την εισερχόμενη κλήση. Όταν θέλουμε να χρησιμοποιήσουμε την συνάρτηση αυτή το κάθε αντικείμενο θα υλοποιήσει την συνάρτηση διαφορετικά ανάλογα με τον τύπο του. Η Σταθερή Συσκευή απλώς θα δεχτεί την εισερχόμενη κλήση, ενώ το Fax μπορεί είτε να δεχτεί την εισερχόμενη κλήση σαν απλό τηλέφωνο είτε σαν μήνυμα και αντιστοίχως να τυπώσει το εισερχόμενο κείμενο.



Διάγραμμα 7: Παράδειγμα Πολυμορφισμού

Η έννοια του πολυμορφισμού σχετίζεται με τις έννοιες της ενθυλάκωσης και της κληρονομικότητας. Η ενθυλάκωση, όπως αναφέραμε, αναφέρεται στην απόκρυψη των δεδομένων που σχετίζονται με την υλοποίηση των διαφόρων λειτουργιών των

αντικειμένων. Η επικοινωνία με τους χρήστες γίνεται με την χρήση της διεπαφής, η οποία καλεί την αντίστοιχη λειτουργία του αντικειμένου. Επειδή η υλοποίηση της λειτουργίας του αντικειμένου είναι ανεξάρτητη από την παρουσίαση της μέσω της διεπαφής κάθε μπορεί αντικείμενο να καλεί από την ίδια διεπαφή διαφορετικές συναρτήσεις με το ίδιο όνομα κάτω από συγκεκριμένες συνθήκες.

Επίσης, η ιδιότητα της κληρονομικότητας επιτρέπει στα αντικείμενα-πατέρας να μοιράζουν στα αντικείμενα-παιδιά συναρτήσεις με το ίδιο όνομα, όπου κάθε αντικείμενο-παιδί υλοποιεί διαφορετικά την κάθε συνάρτηση ανάλογα με τον τύπου του.

Αυτό έχει σαν αποτέλεσμα την χρήση προκαθορισμένων ονομάτων και περιγραφών για τα διάφορα αντικείμενα μειώνοντας έτσι την πολυπλοκότητα. Ένα από τα πλεονεκτήματα του πολυμορφισμού, όπως και με τα άλλα χαρακτηριστικά του αντικειμενοστραφούς προγραμματισμού, είναι η ευκολία συντήρησης της εφαρμογής.

Πλεονεκτήματα Αντικειμενοστραφούς Προγραμματισμού

Από την παρουσίαση των βασικών αρχών του αντικειμενοστραφούς προγραμματισμού καταλαβαίνουμε ότι αυτός ο τρόπος ανάπτυξης εφαρμογών έχει αρκετά πλεονεκτήματα. Στη συνέχεια αναφέρουμε τα βασικά **πλεονεκτήματα** του αντικειμενοστραφούς προγραμματισμού [26].

- **Απλότητα**: οι έννοιες του πραγματικού κόσμου μεταφέρονται σε όρους που αντιλαμβάνονται όσοι συμμετέχουν στην ανάπτυξη της εφαρμογής. Έτσι μειώνεται η πολυπλοκότητα και η δομή των προγραμμάτων είναι πιο σαφής.
- **Επέκταση, αλλαγή και συντήρηση εφαρμογών**: με την χρήση των αντικειμένων οι εφαρμογές είναι πιο εύκολο να επεκταθούν (με την προσθήκη νέων αντικειμένων) ή να αλλαχθούν (με την τροποποίηση υπαρχόντων αντικειμένων) χωρίς να επηρεαστεί η λειτουργικότητα των υπολοίπων αντικειμένων αφού αυτά είναι ανεξάρτητα το ένα από το άλλο.
- **Επαναχρησιμοποίηση κώδικα**: οι προγραμματιστές μπορούν να χρησιμοποιήσουν έτοιμα αντικείμενα σε περισσότερες από μια εφαρμογές. Επίσης, η δυνατότητα της κληρονομικότητας μειώνει τον χρόνο συγγραφής κώδικα για νέα αντικείμενα.
- **Προστασία κώδικα**: η ενθυλάκωση των μεταβλητών και των συναρτήσεων μέσα σε ένα αντικείμενο παρέχει υψηλό επίπεδο προστασίας των δεδομένων από λανθασμένες τροποποιήσεις. Εξασφαλίζεται έτσι η ακεραιότητα και η ακρίβεια των δεδομένων.
- **Παροχή ενός ξεκάθαρου πλαισίου (framework) ανάπτυξης εφαρμογών**: οι προγραμματιστές στηρίζονται σε μια σαφώς καθορισμένη βάση ανάπτυξης εφαρμογών και αντιλαμβάνονται καλύτερα την έννοια των αντικειμένων.

Κεφάλαιο 3. Αρχιτεκτονική Εφαρμογών

Η έννοια της Αρχιτεκτονικής των εφαρμογών σχετίζεται με τον τρόπο, με τον οποίο είναι δομημένες οι εφαρμογές, που αναπτύσσονται σε ένα οργανισμό. Είναι ο τρόπος με τον οποίο κατασκευάζονται οι εφαρμογές σε ένα οργανισμό. Μια δημοφιλής αρχιτεκτονική είναι αυτή της αρχιτεκτονικής πολλών επιπέδων (N-tier Architecture[1]). Ο όρος αυτός σχετίζεται με τον διαχωρισμό των εφαρμογών ενός οργανισμού σε ξεχωριστά, ανεξάρτητα τμήματα. Κάθε τμήμα (ή διαφορετικά κάθε επίπεδο) είναι σαφώς ορισμένο και εκφράζει μια γενική λειτουργία ή σύνολο λειτουργιών που σχετίζονται με την εφαρμογή. Ο διαχωρισμός αυτός των επιπέδων μπορεί να είναι φυσικός, δηλαδή τα διάφορα επίπεδα λειτουργούν σε διαφορετικούς υπολογιστές ή μπορεί να είναι λογικός με την έννοια ότι η εφαρμογή «τρέχει» στον ίδιο υπολογιστή αλλά έχει γίνει λογικός διαχωρισμός των επιπέδων της εφαρμογής.

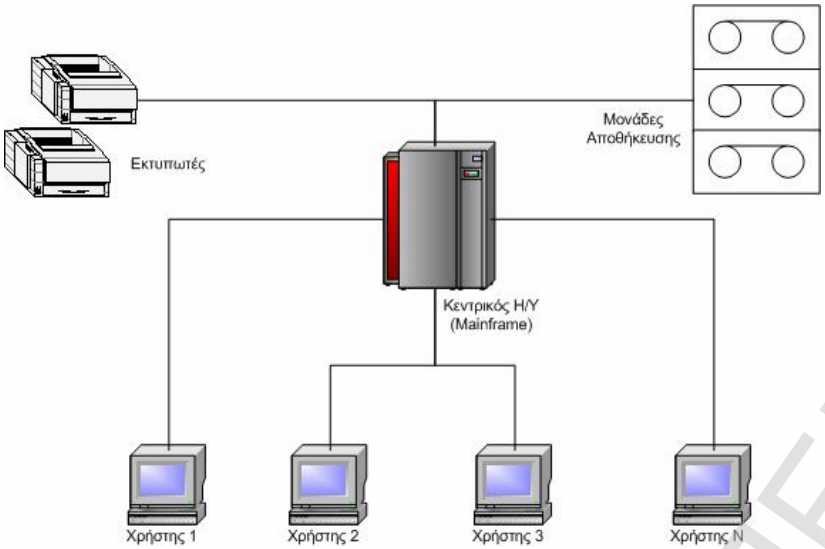
Τι σημαίνει όμως Αρχιτεκτονική N-tier και τι λειτουργίες κάνει κάθε επίπεδο; Συνοπτικά, η αρχιτεκτονική N-tier δείχνει τον διαχωρισμό των τμημάτων κώδικα και των χρησιμοποιούμενων βιβλιοθηκών που χρησιμοποιούνται στις εφαρμογές ενός οργανισμού σε ένα αριθμό επιπέδων (3 ή περισσότερα επίπεδα) κάθε ένα από τα οποία εκτελεί συγκεκριμένες λειτουργίες. Είναι ο τρόπος με τον οποίο σχεδιάζονται οι εφαρμογές ενός οργανισμού. Στο κεφάλαιο αυτό θα παρουσιάσουμε την N-tier Αρχιτεκτονική σύμφωνα με την οποία αναπτύσσονται οι εφαρμογές σε ένα οργανισμό.

Πριν όμως παρουσιάσουμε την αρχιτεκτονική N-tier για να καταλάβουμε την χρησιμότητα της και τα πλεονεκτήματα της καλό είναι να δούμε τα προηγούμενα στάδια, που προϋπήρξαν στην διαδικασία ανάπτυξης εφαρμογών και απασχόλησαν τον κόσμο της πληροφορικής τα προηγούμενα χρόνια [24].

Αρχιτεκτονική 1-tier

Στα πρώτα χρόνια χρήσης των ηλεκτρονικών υπολογιστών στις επιχειρήσεις (δεκαετίες ’70 και ’80) όταν ξεκίνησαν να αναπτύσσονται και οι πρώτες εμπορικές εφαρμογές υπολογιστών για την κάλυψη των αναγκών των επιχειρήσεων, γινόταν χρήση ενός κεντρικού υπολογιστή (γνωστού ως **Mainframe**). Ο κεντρικός υπολογιστής, ο οποίος συνήθως ήταν αρκετά μεγάλος σε μέγεθος, βρισκόταν σε ένα κεντρικό σημείο στο κτίριο του οργανισμού. Όλη η επεξεργασία των δεδομένων γίνονταν στον κεντρικό υπολογιστή, στον οποίο «έτρεχαν» και οι εφαρμογές του οργανισμού.

[1] Στην συνέχεια της παρουσίασης για την παρουσίαση του όρου *Επίπεδα Αρχιτεκτονικής* για λόγους ευκολίας θα χρησιμοποιείται ο αγγλικός όρος tier (π.χ. N-tier, 3-tier κλπ.). Επίσης οι όροι ***tier*** και ***layer*** είναι ταυτόσημοι επομένως μπορούμε να χρησιμοποιούμε και τους δύο όρους.



Διάγραμμα 8: Αρχιτεκτονική 1-tier

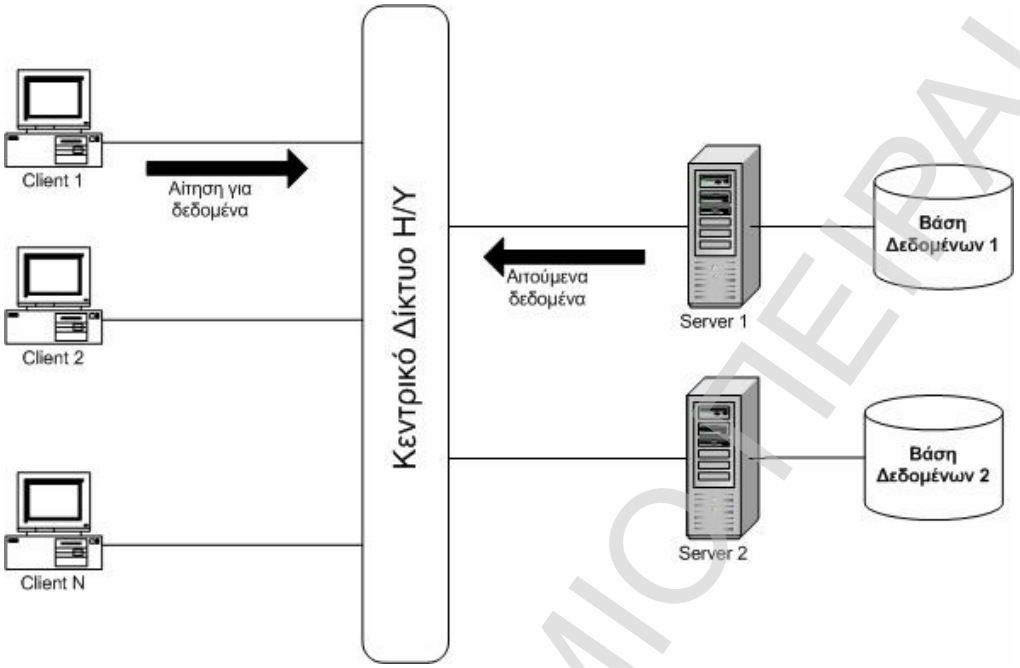
Οι χρήστες των εφαρμογών επικοινωνούσαν με τον κεντρικό υπολογιστή μέσω απλών τερματικών (γνωστά και ως dumb terminals – χαζά τερματικά) τα οποία λάμβαναν τα αποτελέσματα της επεξεργασίας από τον κεντρικό υπολογιστή και τα εμφάνιζαν στις οθόνες τους. Οι περιφερειακές συσκευές (εκτυπωτές, μονάδες αποθήκευσης) βρίσκονταν πάνω στον κεντρικό υπολογιστή και μέσω αυτού είχαν πρόσβαση σε αυτές οι τελικοί χρήστες. Η αρχιτεκτονική αυτή του ενός επιπέδου είχε σαν πλεονέκτημα την απλότητα της χρήσης και της αποτελεσματικότητας, κυρίως όσον αφορά την αξιοπιστία των δεδομένων και την συντήρηση αλλά είχε σαν μεγάλο μειονέκτημα το τεράστιο κόστος αγοράς του κεντρικού υπολογιστή, που το έκανε απαγορευτικό για επιχειρήσεις, κυρίως μικρού και μεσαίου μεγέθους. Επίσης σε κάθε τερματικό έπρεπε να «κατεβαίνουν» μέσω του δικτύου όλες οι πληροφορίες των εφαρμογών, τα απαιτούμενα γραφικά καθώς και οι επεξεργασμένες πληροφορίες για την παρουσίαση στους τελικούς χρήστες, επιβαρύνοντας έτσι την λειτουργία τόσο του δικτύου όσο και των τερματικών.

Αρχιτεκτονική 2-tier

Στην δεκαετία του 1980 σημειώθηκε μια επανάσταση στον χώρο της Πληροφορικής: **ο Προσωπικός Υπολογιστής (Personal Computer – PC)**. Το PC έγινε ο καθιερωμένος εξοπλισμός για μια επιχείρηση λόγω της υπολογιστικής ισχύος που διέθετε και ήταν κατά πολύ μεγαλύτερη από αυτή των τερματικών που χρησιμοποιούνταν έως τώρα. Η χρήση σκληρών δίσκων και μνήμης επέτρεπε στα PC να φιλοξενούν τις εφαρμογές του οργανισμού ενώ η ανάπτυξη των εφαρμογών λογισμικού ήταν ραγδαία. Η ανάγκη χρήσης προσωπικών εφαρμογών που θα έτρεχαν σε κάθε υπολογιστή οδήγησε στην ανάπτυξη ενός καινούργιου μοντέλου αρχιτεκτονικής υπολογιστών, όπου θα επέτρεπε στον οργανισμό να μοιράζει δεδομένα στους υπολογιστές των χρηστών. Το νέο αυτό μοντέλο έγινε γνωστό σαν **Μοντέλο Πελάτη/Εξυπηρετητή (Client/Server)**.

Το μοντέλο Client/Server, που είναι γνωστό και σαν **Αρχιτεκτονική 2-tier** λειτουργεί ως εξής: Ο Client στον προσωπικό υπολογιστή συνδέεται στο Κεντρικό Δίκτυο Υπολογιστών και ζητά δεδομένα από τον Server. Ο Server για να εξυπηρετήσει τις αιτήσεις που δέχεται συνδέεται σε μια Βάση Δεδομένων και ανακτά τα δεδομένα που απαιτούνται. Η Βάση Δεδομένων μπορεί να βρίσκεται σε διαφορετικό ή στον ίδιο Server. Μέσω του δικτύου

αποστέλλει τα δεδομένα στον Client εξυπηρετώντας έτσι τις αιτήσεις των πελατών του. Η επεξεργασία των δεδομένων γίνεται στον υπολογιστή του Client και τα αποτελέσματα παρουσιάζονται στην οθόνη του τελικού χρήστη.



Διάγραμμα 9: Αρχιτεκτονική 2-tier

Το μοντέλο αυτό έχει σαν πλεονέκτημα το ότι δεν μεταφέρονται τα πολύπλοκα γραφικά των εφαρμογών από τον Server στους υπολογιστές, όπως συνέβαινε με τον μοντέλο του Κεντρικού Υπολογιστή αλλά μόνο οι πληροφορίες που είναι απαραίτητες. Η επεξεργασία και η παρουσίαση των δεδομένων γίνεται στους υπολογιστές των Clients. Επίσης το μοντέλο αυτό είναι εύκολο να υλοποιηθεί. Χρειάζονται μόνο οι Servers για την εξυπηρέτηση των Clients και πιθανόν οι υπολογιστές που θα φιλοξενούν τις Βάσεις Δεδομένων καθώς και οι υπολογιστές των Clients. Σε κάθε περίπτωση πάντως, το κόστος αγοράς και συντήρησης των μηχανημάτων αυτών είναι μικρότερο από το κόστος του Κεντρικού Υπολογιστή.

Ωστόσο το μοντέλο αυτό παρουσιάζει ορισμένα προβλήματα:

- Οι συνδέσεις των δικτύων είναι ακριβές. Απαιτούν αρκετό χρόνο για να επιτευχθούν και απαιτούν πολύ μνήμη στον Server. Βέβαια η ραγδαία ανάπτυξη της τεχνολογίας των υπολογιστών έχει καταστήσει δυνατή την αγορά φτηνών και δυνατών μηχανημάτων, παρόλα αυτά όμως το κόστος του δικτύου είναι ένας παράγοντας που δεν μπορεί να αγνοηθεί
- Μπορεί να συνδεθεί ένα συγκεκριμένος αριθμός Clients σε ένα Server έτσι ώστε να μπορεί ο τελευταίος να εξυπηρετήσει τις αιτήσεις σε ικανοποιητικό βαθμό απόκρισης. Έτσι ο ρυθμός εξυπηρέτησης των αιτήσεων στην Βάση Δεδομένων μειώνεται όσο αυξάνονται οι αιτήσεις για ανάκτηση δεδομένων. Επίσης η αλλαγή της Βάσης Δεδομένων απαιτεί την αλλαγή ολόκληρων των εφαρμογών αφού οι εντολές σύνδεσης με την Βάση υπάρχουν στους Clients.

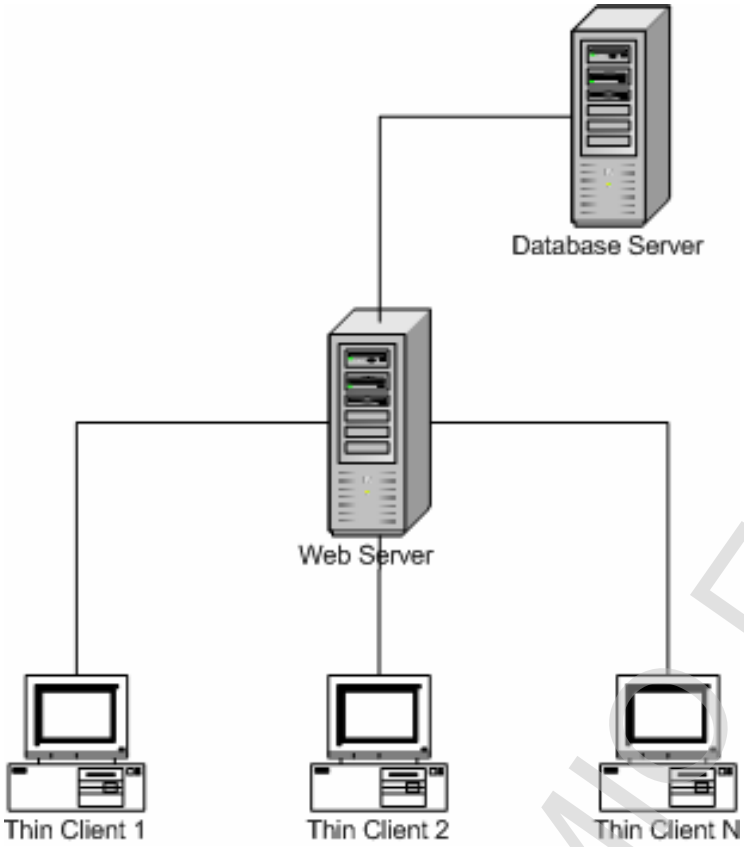
- Η μέθοδος αυτή είναι αναποτελεσματική όσον αφορά τον βαθμό χρήσης των πόρων του δικτύου. Έχει υπολογιστεί ότι πολλοί χρήστες χρησιμοποιούν την σύνδεση του δικτύου μόνο κατά 2-3% για να ανακτήσουν τα δεδομένα. Την άλλη ώρα είτε επεξεργάζονται τα δεδομένα είτε ασχολούνται με κάποια άλλη εργασία αφήνοντας την σύνδεση ανοιχτή καταναλώνοντας πόρους στον Server.
- Το κύριο πρόβλημα είναι ότι σε κάθε Client υπολογιστή βρίσκονται οι επιχειρησιακοί κανόνες για την επεξεργασία των δεδομένων, οι εντολές σύνδεσης με την Βάση Δεδομένων καθώς και οι οθόνες παρουσίασης των δεδομένων στους τελικούς χρήστες (User Interface). Αυτό έχει σαν αποτέλεσμα την δυσκολία συντήρησης των εφαρμογών (κάθε αλλαγή σε μια εφαρμογή συνεπάγεται την αλλαγή σε όλους τους Clients) αλλά και την ανάγκη δημιουργίας «βαρέων» υπολογιστών (*Fat Clients*), με την έννοια της πολυπλοκότητας της επεξεργασίας και παρουσίασης των δεδομένων.

Τα παραπάνω προβλήματα της Αρχιτεκτονικής 2-tier δημιούργησαν την απαίτηση δημιουργίας ενός διαφορετικού μοντέλου. Η ανάγκη μεγάλης υπολογιστικής δύναμης των Clients έπρεπε να μειωθεί και να υπάρχουν περισσότεροι «ελαφριοί» υπολογιστές (*Thin Clients*) που θα ασχολούνται μόνο με την παρουσίαση των δεδομένων. Ο κορμός των εφαρμογών του οργανισμού και οι επιχειρησιακοί κανόνες, βάσει των οποίων γίνεται η επεξεργασία των δεδομένων, πρέπει να υπάρχουν τοπικά σε ένα Server για να υπάρχει η ευκολία συντήρησης των εφαρμογών. Τέλος, η Βάση Δεδομένων πρέπει να χωρίζεται (λογικά και φυσικά) από όλες τις εφαρμογές διευκολύνοντας έτσι την αλλαγή της όταν αυτή κάποια στιγμή απαιτηθεί.

Όλες αυτές οι απαιτήσεις οδήγησαν στην δημιουργία του μοντέλου της **Αρχιτεκτονικής N-tier** (όπου N θεωρείται ο αριθμός των ξεχωριστών επιπέδων που χωρίζεται η αρχιτεκτονική συνήθως 3 ή περισσότερα επίπεδα).

Αρχιτεκτονική N-tier

Σύμφωνα με το μοντέλο της αρχιτεκτονικής N-tier έχουμε διαφορετικά επίπεδα το κάθε ένα από τα οποία είναι ξεχωριστό και σαφώς ορισμένο από τα άλλα επίπεδα. Η διαφορά με το προηγούμενο μοντέλο 2-tier (Client/Server) είναι ότι η λειτουργία της επεξεργασίας των δεδομένων έχει φύγει από τους Clients και έχει μεταφερθεί στον Server αφήνοντας στους Clients μόνο τις απαραίτητες λειτουργίες της παρουσίασης των δεδομένων στους τελικούς χρήστες (για τον λόγο αυτό οι Clients ονομάζονται *Thin Clients*).



Διάγραμμα 10: Φυσική υλοποίηση Αρχιτεκτονικής N-tier

Η επεξεργασία των δεδομένων σύμφωνα με συγκεκριμένους επιχειρησιακούς κανόνες και λογική έχει μεταφερθεί στον Web Server ενώ υπάρχει και το επίπεδο της Βάση Δεδομένων (ή οποιαδήποτε άλλη πηγή δεδομένων χρησιμοποιείται) το οποίο είναι ανεξάρτητο από τα άλλα επίπεδα. Να τονίσουμε στο σημείο ότι το μοντέλο της Αρχιτεκτονικής N-tier δεν διαφέρει κατά πολύ από το μοντέλο 2-tier ενώ κάποιος μπορεί να παρατηρήσει ότι έχει ομοιότητες και με την Αρχιτεκτονική 1-tier. Η πραγματικότητα είναι ότι μοντέλο της Αρχιτεκτονικής N-tier έχει πολλές ομοιότητες και με τις προηγούμενες αρχιτεκτονικές που αναφέραμε.

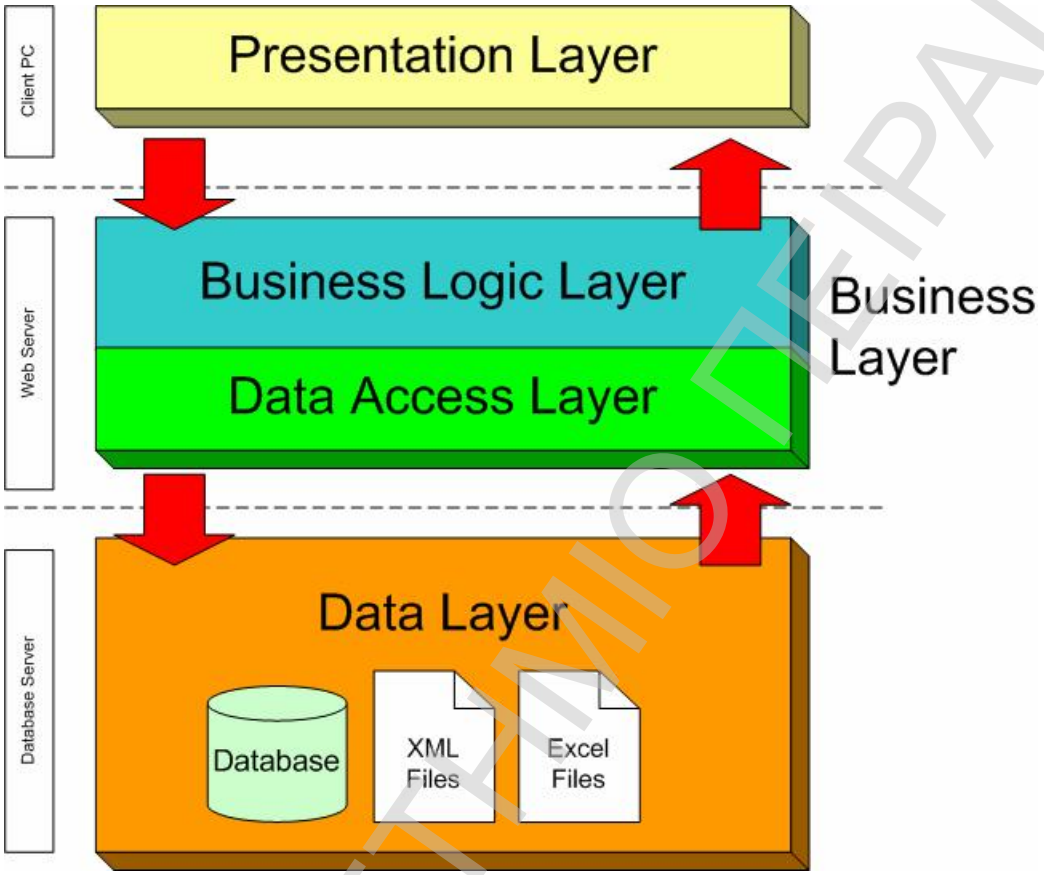
Επίπεδα N-tier Αρχιτεκτονικής

Στην N-tier Αρχιτεκτονική συνήθως υπάρχουν τα παρακάτω επίπεδα:

- Presentation Layer (Παρουσίαση Δεδομένων)
- Business layer (Ανάκτηση Δεδομένων και Κανόνες Επιχείρησης)
    - Business Logic Layer (BLL) (Κανόνες και Λογική Επιχείρησης)
    - Data Access Layer (DAL) (Ανάκτηση Δεδομένων)
- Data Layer (Επίπεδο Αποθήκευσης Δεδομένων)

Το κάθε επίπεδο, όπως φαίνεται στην παρακάτω εικόνα, είναι σαφώς ορισμένο και ανεξάρτητο από τα άλλα επίπεδα της αρχιτεκτονικής. Και αυτός ακριβώς είναι ο λόγος της

επιτυχίας της αρχιτεκτονικής αυτής. Η απομόνωση του κάθε επιπέδου διευκολύνει την συντήρηση των εφαρμογών και την εύκολη μετάβαση σε άλλες τεχνολογίες (για παράδειγμα καινούργια Βάση Δεδομένων) χωρίς να επηρεάζεται σε μεγάλο βαθμό η λειτουργία των υπολοίπων επιπέδων.



Διάγραμμα 11: Λογική υλοποίηση Αρχιτεκτονικής N-tier

Presentation Layer

Το Presentation layer είναι υπεύθυνο για την παρουσίαση των δεδομένων και την επικοινωνία με τον τελικό χρήστη της εφαρμογής. Στο επίπεδο αυτό ανήκουν οι οθόνες της εφαρμογής (windows forms αν μιλάμε για windows application) καθώς και οι σελίδες (web pages αν μιλάμε για web application). Οποιοδήποτε λειτουργία που ανήκει σε κάποια οθόνη ή σε κάποια web page θα έχει κλήσεις σε κάποια συγκεκριμένη λειτουργία των κατωτέρων επιπέδων για την ανάκτηση των δεδομένων και την παρουσίαση τους στην οθόνη.

Στο επίπεδο αυτό μιλώντας και για αντικειμενοστραφή προγραμματισμό όπως παρουσιάστηκε προηγουμένως, ανήκουν οι κλάσεις (αντικείμενα) που χρησιμοποιούνται για την παρουσίαση των δεδομένων καθώς και για την δημιουργία του interface των οθονών και των σελίδων των εφαρμογών. Το επίπεδο αυτό δεν «γνωρίζει» από πού ήρθαν τα δεδομένα αλλά ασχολείται μόνο με την παρουσίαση τους στους τελικούς χρήστες.

Το επίπεδο αυτό εκφράζει κάθε φορά την εφαρμογή, που χρησιμοποιείται. Έχουμε δηλαδή ένα επίπεδο παρουσίασης για κάθε εφαρμογή που χτίζεται.

Business Layer

Το Business Layer χρησιμοποιείται για την ανάκτηση των δεδομένων από το κατώτερο επίπεδο και την προώθηση τους στο επίπεδο της παρουσίασης προς τους τελικούς χρήστες της εφαρμογής. Στο επίπεδο αυτό γίνεται η απαραίτητη επεξεργασία των δεδομένων (βάσει συγκεκριμένων επιχειρησιακών κανόνων όπως θα δούμε παρακάτω) πριν αυτά προωθηθούν στο ανώτερο επίπεδο.

Το Business layer λειτουργεί σαν ένας συνδετικός κρίκος ανάμεσα στο επίπεδο, στο οποίο είναι αποθηκευμένα τα δεδομένα και του επιπέδου παρουσίασης των δεδομένων. Είναι το πλέον σημαντικό επίπεδο της αρχιτεκτονικής καθώς είναι το επίπεδο που εκφράζει την λογική και τους κανόνες της επιχείρησης, που χρησιμοποιούνται για την επεξεργασία των δεδομένων.

Στόχος του επιπέδου αυτού είναι να απομονώσουμε την πολυπλοκότητα της επιχειρησιακής λογικής από την εργασία της σχεδίασης της εφαρμογής (από την μία σχεδιασμός οθονών και σελίδων για την παρουσίαση των δεδομένων και από την άλλη η επίτευξη της επικοινωνίας με την πηγή δεδομένων – όπου πηγή δεδομένων θεωρούμε μια Βάση Δεδομένων ή XML ή Excel αρχεία για ανάκτηση δεδομένων). Έτσι αν αλλάξει η επιχειρησιακή λογική και οι επιχειρησιακοί κανόνες οι αλλαγές θα γίνουν μόνο σε αυτό το επίπεδο χωρίς να επηρεάζονται τα υπόλοιπα επίπεδα της εφαρμογής σε μεγάλο βαθμό.

Το επίπεδο αυτό από την στιγμή που εκφράζει την επιχειρησιακή λογική και τους κανόνες που χρησιμοποιούνται για την επεξεργασία των δεδομένων είναι κοινό για όλες τις εφαρμογές του οργανισμού μειώνοντας σε μεγάλο βαθμό την πολυπλοκότητα και το κόστος ανάπτυξης τους.

Το επίπεδο αυτό χωρίζεται σε δύο επίπεδα. Το Business Rules Layer (που εκφράζει την επιχειρησιακή λογική και κανόνες) και το Data Access Layer (που χρησιμοποιείται για την ανάκτηση των δεδομένων).

Business Logic Layer (BLL)

Το Business Logic Layer χρησιμοποιείται για την επεξεργασία των δεδομένων που έρχονται από τα κατώτερα επίπεδα και την προετοιμασία για την αποστολή τους στο ανώτερο επίπεδο. Η επεξεργασία των δεδομένων γίνεται βάσει συγκεκριμένων επιχειρησιακών κανόνων που ισχύουν για τις εφαρμογές της επιχείρησης. Στο επίπεδο αυτό υπάρχουν οι κλάσεις που αντιστοιχούν σε συγκεκριμένα αντικείμενα και εκφράζουν της λογική της εφαρμογής.

Το επίπεδο αυτό δεν γνωρίζει από πού έρχονται τα δεδομένα (από ποια πηγή) ούτε πως θα παρουσιαστούν στους τελικούς χρήστες της εφαρμογής. Αυτές είναι λειτουργίες των άλλων επιπέδων.

Data Access Layer (DAL)

Το Data Access Layer χρησιμοποιείται για την ανάκτηση των δεδομένων από το κατώτερο επίπεδο (από την πηγή δεδομένων). Στο επίπεδο αυτό γίνονται οι κύριες λειτουργίες που γίνονται σε μία Βάση Δεδομένων: Ανάκτηση (Select), Καταχώρηση (Insert), Διαγραφή (Delete), Ενημέρωση (Update) από και προς την πηγή των δεδομένων. Το επίπεδο αυτό προωθεί τα δεδομένα στο ανώτερο επίπεδο (στο BLL) για την επεξεργασία τους σύμφωνα με τους κανόνες και την λογική της επιχείρησης.

Στο επίπεδο αυτό υπάρχουν οι κλάσεις που αντιστοιχούν σε συγκεκριμένα αντικείμενα της επιχειρησιακής λογικής όπως έχουν καθοριστεί στο ανώτερο επίπεδο BLL και αναλαμβάνουν την επικοινωνία με την πηγή δεδομένων. Το επίπεδο αυτό δεν γνωρίζει πώς θα γίνει η επεξεργασία στα δεδομένα ούτε πώς θα παρουσιαστούν αυτά στους τελικούς χρήστες της εφαρμογής.

Data Layer

Το Data Layer είναι το κατώτερο επίπεδο στο οποίο βρίσκονται αποθηκευμένα τα δεδομένα της εφαρμογής. Στο επίπεδο αυτό ανήκει για παράδειγμα η Βάση Δεδομένων (SQL Server, Oracle, Access κλπ.), XML, Excel αρχεία ή οποιοδήποτε αρχείο χρησιμοποιείται για την αποθήκευση των δεδομένων. Το επίπεδο αυτό είναι «απομονωμένο» από τα άλλα επίπεδα με την έννοια ότι δεν γνωρίζει που θα πάνε τα δεδομένα. Έτσι εάν αλλάξει η Βάση Δεδομένων ή ο τρόπος με τον οποίο αποθηκεύονται τα δεδομένα οι αλλαγές περιορίζονται στο επίπεδο αυτό χωρίς να επηρεάσουν τα ανώτερα επίπεδα.

Το επίπεδο αυτό είναι κοινό για όλες τις εφαρμογές του οργανισμού.

Πλεονεκτήματα N-tier Αρχιτεκτονικής

Υπάρχουν πολλά πλεονεκτήματα από την χρήση της Αρχιτεκτονικής N-tier για την ανάπτυξη των εφαρμογών ενός οργανισμού.

- **Μικρότερο κόστος ανάπτυξης**: χτίζοντας εφαρμογές πολλών επιπέδων μειώνεται το κόστος ανάπτυξης αφού πολλά συστατικά μέρη χρησιμοποιούνται πολλές φορές (για παράδειγμα το Business και το Data Layer που είναι κοινά για όλες τις εφαρμογές του οργανισμού).
- **Μικρότερο κόστος συντήρησης**: οι κανόνες της επιχείρησης (που εκφράζουν τον τρόπο επεξεργασίας των δεδομένων) βρίσκονται απομονωμένοι σε ένα επίπεδο πράγμα που σημαίνει ότι μπορούν εύκολα να συντηρηθούν. Επίσης η αλλαγή της Βάσης Δεδομένων θα επιφέρει αλλαγές μόνο στο τελευταίο επίπεδο (Data Layer) και ίσως και στο επίπεδο ανάκτησης δεδομένων (Data Access Layer).
- **Τυποποίηση των επιχειρηματικών κανόνων**: οι επιχειρηματικοί κανόνες εκφράζονται με την μορφή αντικειμένων και συναρτήσεων σε ένα συγκεκριμένο επίπεδο που μπορούν να χρησιμοποιηθούν από άλλα σημεία της εφαρμογής.
- **Καλύτερη ποιότητα εφαρμογών**: η χρήση ξεχωριστών και μεμονωμένων επιπέδων στις εφαρμογές δίνει την δυνατότητα καλύτερου ελέγχου συγκεκριμένων τμημάτων της εφαρμογής και διόρθωσης των διαφόρων προβλημάτων.
- **Επαναχρησιμοποίηση κώδικα**: από την φύση της η αρχιτεκτονική πολλών επιπέδων προωθεί την χρησιμοποίηση έτοιμων τμημάτων κώδικα με την ενσωμάτωση συγκεκριμένων επιπέδων σε όλες τις εφαρμογές του οργανισμού.
- **Χρήση διαφορετικών εργαλείων ανάπτυξης**: η ανάπτυξη πολλών επιπέδων δίνει την δυνατότητα στους προγραμματιστές να χρησιμοποιήσουν διαφορετικά εργαλεία ανάπτυξης για το κάθε επίπεδο.
- **Χρήση νέων τεχνολογιών**: σε κάποιο επίπεδο της εφαρμογής που είναι απομονωμένο από τα άλλα επίπεδα μπορεί να γίνει χρήση καινούργιας τεχνολογίας

χωρίς να επηρεάσει σε μεγάλο βαθμό το σύνολο της εφαρμογής (για παράδειγμα η αλλαγή της Βάσης Δεδομένων).

Αντικειμενοστραφής Προγραμματισμός και N-tier Αρχιτεκτονική

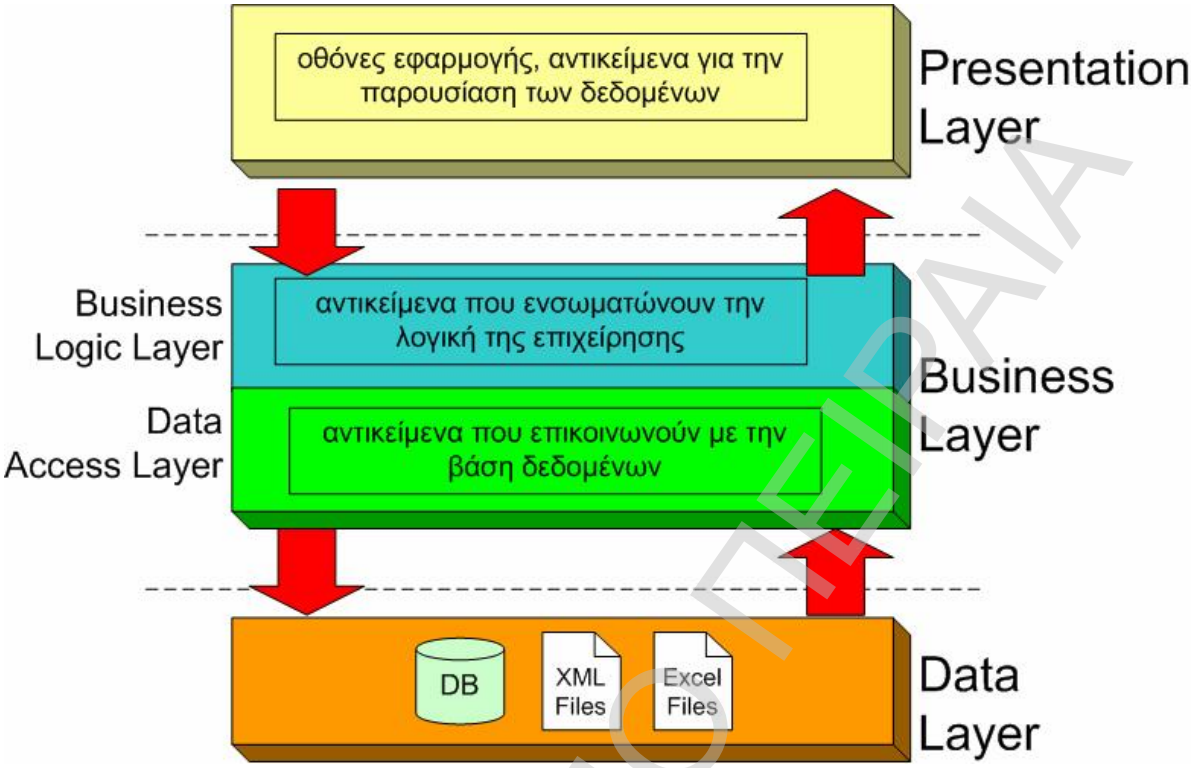
Βασικό χαρακτηριστικό της Αρχιτεκτονικής N-tier είναι ο φυσικός ή/και λογικός διαχωρισμός των διαφόρων επιπέδων της εφαρμογής. Κάθε επίπεδο εκτελεί συγκεκριμένες λειτουργίες και επικοινωνεί με τα υπόλοιπα επίπεδα για επιτευχθεί ο τελικός σκοπός που είναι η σωστή λειτουργία της εφαρμογής. Όπως στην περίπτωση του Αντικειμενοστραφούς προγραμματισμού, έχουμε μεμονωμένες οντότητες – τα αντικείμενα – τα οποία περιέχουν μέσα τους τις μεταβλητές και τις συναρτήσεις και εκτελούν συγκεκριμένες λειτουργίες, έτσι και στην περίπτωση της Αρχιτεκτονικής N-tier έχουμε μεμονωμένα επίπεδα που εκτελούν συγκεκριμένες λειτουργίες.

Παρατηρούμε ότι οι έννοιες του Αντικειμενοστραφούς Προγραμματισμού σχετίζονται με τις έννοιες της Αρχιτεκτονικής n-tier αν και ασχολούνται με διαφορετικά θέματα. Η Αρχιτεκτονική N-tier αναφέρεται στην σχεδίαση της αρχιτεκτονικής μιας εφαρμογής ενώ ο Αντικειμενοστραφής προγραμματισμός αναφέρεται σε μια τεχνική ανάπτυξης του κώδικα της εφαρμογής. Εντούτοις, είναι πιο εύκολο να αναπτύξουμε μια εφαρμογή βασισμένη στην αρχιτεκτονική N-tier με όρους αντικειμενοστραφούς προγραμματισμού παρά εάν αναπτύσσαμε την εφαρμογή με άλλη τεχνική, όπως για παράδειγμα με δομημένο προγραμματισμό.

Ο δομημένος προγραμματισμός είναι μια τεχνική ανάπτυξης προγραμμάτων όπου χωρίζουμε την εφαρμογή σε συναρτήσεις οι οποίες εκτελούν κάποιες λειτουργίες. Το συνολικό πρόγραμμα παρόλα αυτά είναι ενιαίο έχει δηλαδή μια αρχή, μέση και ένα τέλος ενώ οι συναρτήσεις επικοινωνούν μεταξύ τους για να πετύχουν το τελικό αποτέλεσμα. Είναι δύσκολο να χωρίσουμε λογικά τα διάφορα επίπεδα της Αρχιτεκτονικής N-tier εάν έχουμε ένα μεγάλο πρόγραμμα το οποίο καλεί συναρτήσεις από διαφορετικά σημεία της εφαρμογής και χρησιμοποιεί ένα πλήθος διαφορετικών μεταβλητών.

Σε αντίθεση με τον δομημένο προγραμματισμό, στην περίπτωση του αντικειμενοστραφούς προγραμματισμού τα πράγματα είναι πιο ξεκάθαρα. Στο σύστημα δημιουργούνται διάφορα αντικείμενα, ανεξάρτητα το ένα από το άλλο, που εκτελούν συγκεκριμένες λειτουργίες. Τα αντικείμενα επικοινωνούν μεταξύ τους μέσω μηνυμάτων. Μπορούμε λοιπόν να ομαδοποιήσουμε διάφορα αντικείμενα που εκτελούν συγκεκριμένες λειτουργίες και μέσα από την ομαδοποίηση αυτή να ορίσουμε τα αντίστοιχα επίπεδα της αρχιτεκτονικής. Για παράδειγμα, μπορούμε να ομαδοποιήσουμε τα αντικείμενα που επικοινωνούν με την Βάση Δεδομένων σε ένα πακέτο (package)[2] και να το ονομάσουμε Data Layer. Ομοίως δημιουργούμε και πακέτα για τα υπόλοιπα επίπεδα της αρχιτεκτονικής.

[2] Package (πακέτο) είναι ένας μηχανισμός ομαδοποίησης αντικειμένων για την οργάνωση τους. Τα αντικείμενα παραμένουν ανεξάρτητα το ένα από το άλλο, βρίσκονται όμως κάτω από ένα κοινό όνομα για την εύκολη μεταφορά τους από εφαρμογή σε εφαρμογή. Τα πακέτα ονομάζονται και Βιβλιοθήκες (libraries).



Διάγραμμα 12: Αντικειμενοστραφής Προγραμματισμός & Αρχιτεκτονική N-tier

Κάθε επίπεδο της αρχιτεκτονικής περιέχει ένα ή περισσότερα πακέτα αντικειμένων ή μεμονωμένα αντικείμενα που εκτελούν συγκεκριμένες λειτουργίες και σχετίζονται με το επίπεδο στο οποίο ανήκουν.

Απαραίτητη προϋπόθεση για την επιτυχία του αντικειμενοστραφούς προγραμματισμού και του σχεδιασμού της αρχιτεκτονικής της εφαρμογής είναι η δημιουργία του **μοντέλου της εφαρμογής**. Ένα μοντέλο είναι το σχέδιο της εφαρμογής που εκτός των άλλων δείχνει στους εμπλεκομένους την χρήση των αντικειμένων που χρησιμοποιούνται στην εφαρμογή και την αλληλεπίδραση τους, περιγράφει με οπτικό και συνοπτικό τρόπο τις βασικές λειτουργίες της εφαρμογής και παρουσιάζει τον τρόπο με τον οποίο είναι οργανωμένη η αρχιτεκτονική της εφαρμογής.

Αξία της Μοντελοποίησης [2, 4]

Το μοντέλο είναι ένα σχέδιο (blueprint) του συστήματος που αναπτύσσεται και προσπαθεί να δώσει μια οπτική περιγραφή του. Όπως το αρχιτεκτονικό σχέδιο δείχνει στον αρχιτέκτονα και σε κάθε εμπλεκόμενο την όψη του υπό ανάπτυξη κτιρίου και πώς τα διάφορα «αντικείμενα» που χρησιμοποιούνται σε αυτό σχετίζονται το ένα με το άλλο (για παράδειγμα η μορφή που θα έχει το κάθε δωμάτιο, οι υδραυλικές εγκαταστάσεις σε κάθε όροφο, το δίκτυο του ρεύματος στους τοίχους του οικοδομήματος κλπ.) έτσι και ένα μοντέλο δείχνει την εφαρμογή προτού αυτή αρχίσει να αναπτύσσεται. Έτσι οι εμπλεκόμενοι μπορούν να βεβαιωθούν για την εφαρμογή που πρόκειται να αναπτύξουν στη συνέχεια καταγράφονται οι απαιτήσεις των χρηστών και παρέχεται ένας τρόπος αντιμετώπισης των μελλοντικών αλλαγών στις απαιτήσεις τις εφαρμογής.

Με την κατασκευή ενός μοντέλου της εφαρμογής συγκεντρώνεται η εργασία των αναλυτών και των σχεδιαστών της εφαρμογής, οι οποίοι καταγράφουν τις απαιτήσεις του οργανισμού

και αναπαρίσταται οι απαιτήσεις αυτές με ένα σχέδιο. Αυτή η διαδικασία ονομάζεται **Μοντελοποίηση**.

Ένα μοντέλο της εφαρμογής είναι μια αφαιρετική παρουσίαση των κυρίων τμημάτων της εφαρμογής, μια προσεγγιστική αναπαράσταση της εφαρμογής προτού αυτή αρχίσει να αναπτύσσεται γεγονός που μειώνει την πολυπλοκότητα της ανάπτυξης της και βελτιώνεται η επικοινωνία, ο σχεδιασμός και η αξιολόγηση της εφαρμογής.

Η χρήση ενός μοντέλου για την παρουσίαση των προδιαγραφών της εφαρμογής, η σχεδίαση του συστήματος πάνω στο μοντέλο και η κωδικοποίηση και συνένωση των μερών (components) της εφαρμογής έχει τα παρακάτω **πλεονεκτήματα**:

- Παρέχεται μια σαφής δομή για την εύρεση της λύσης στο πρόβλημα.
- Η χρήση αντικειμένων ανταποκρίνεται πιο πολύ στην εννοιολογική αντίληψη των πραγμάτων από τον σχεδιαστή και τον χρήστη της εφαρμογής.
- Παρέχεται μια αφαίρεση της γενικότητας της εφαρμογής για να ελεγχθεί η πολυπλοκότητα.
- Οι απαιτούμενες αλλαγές στο σύστημα επικεντρώνονται σε συγκεκριμένα τμήματα της εφαρμογής.
- Η κληρονομικότητα και ο πολυμορφισμός (χαρακτηριστικά στοιχεία του ΟΟ προγραμματισμού) κάνουν τα συστήματα εύκολα επεκτάσιμα συμβάλλοντας στην γρήγορη ανάπτυξη εφαρμογών.
- Ελαττώνεται το κόστος ανάπτυξης της εφαρμογής.
- Βελτιώνεται η επικοινωνία μεταξύ των εμπλεκομένων στην ανάπτυξη της εφαρμογής (σχεδιαστές, αναλυτές, προγραμματιστές, πελάτες κλπ.).

Οπτική Μοντελοποίηση [17]

Η έννοια της Οπτικής Μοντελοποίησης (Visual Modeling) σχετίζεται με την διαδικασία της αναπαράστασης της πληροφορίας που παρέχεται από το μοντέλο και από την αρχιτεκτονική της εφαρμογής με γραφικό τρόπο χρησιμοποιώντας ένα **τυποποιημένο σύνολο γραφικών στοιχείων**. Η τυποποίηση είναι ένας κρίσιμος παράγοντας επιτυχίας της οπτικής Μοντελοποίησης γιατί καθιερώνεται ένας κοινός τρόπος ανταλλαγής πληροφοριών με αποτέλεσμα να βελτιστοποιείται η **επικοινωνία**. Η επικοινωνία μεταξύ των χρηστών, των προγραμματιστών, των αναλυτών, των δοκιμαστών (testers), των διευθυντών και οποιοδήποτε άλλων εμπλεκομένων στην διάρκεια ανάπτυξης μιας εφαρμογής είναι ο βασικός στόχος της μοντελοποίησης που προσπαθεί να επιτύχει την διάδοση της γνώσης, που έχει αποκτηθεί σε ολόκληρο τον οργανισμό.

Στην σημερινή παγκόσμια αγορά που ο ανταγωνισμός εντείνεται ολοένα και περισσότερο η αναπαράσταση της γνώσης, που αποκτάται καθημερινά σε ένα οργανισμό και η δυνατότητα διάδοσης της σε κάθε ενδιαφερόμενο είναι ένα κρίσιμος παράγοντας επιτυχίας του οργανισμού. Στην διάρκεια ανάπτυξης των εφαρμογών για την επίλυση των προβλημάτων ενός οργανισμού «δημιουργείται» γνώση που πολλές φορές «χάνεται» και δεν υπάρχει δυνατότητα να χρησιμοποιηθεί ξανά. Λέγοντας γνώση αναφερόμαστε σε κάποιες μεθοδολογίες και τεχνικές ανάπτυξης εφαρμογών που σπάνια καταγράφονται από κάποιον υπεύθυνο με αποτέλεσμα η εργασία αυτή απλώς να εκτελείται και μετά να χάνεται ή απλά να μην υπάρχει η δυνατότητα να χρησιμοποιηθεί στο μέλλον.

Σε πολλές περιπτώσεις έχει παρατηρηθεί ότι όταν αναπτύσσεται μια νέα εφαρμογή πολλά βήματα γίνονται από την αρχή, όπως για παράδειγμα κάποιες επιχειρηματικές λειτουργίες ή η ίδια η επιχειρησιακή λογική (business logic) του οργανισμού. Τα βήματα αυτά δεν καταγράφονται στην αρχή ως τυποποιημένες διαδικασίες ή σαν έτοιμες συναρτήσεις μέσα σε μια βιβλιοθήκη κώδικα με αποτέλεσμα κάθε φορά να αποτυπώνονται με την συγγραφή του ίδιου κώδικα που πιθανόν να έχει χρησιμοποιηθεί και σε κάποια άλλη εφαρμογή. Για παράδειγμα, μια διαδικασία ενός οργανισμού είναι η συμπεριφορά της επιχείρησης προς κάποιους συγκεκριμένους πελάτες (π.χ. κάποιες εκπτώσεις στην τελική τιμή που αποτυπώνεται με την συγγραφή της αντίστοιχης συνάρτησης σε μια εμπορική εφαρμογή για τον υπολογισμό της μείωσης). Όταν θα γραφεί στο μέλλον μια άλλη εφαρμογή η ίδια συνάρτηση ίσως θα πρέπει να ξαναγραφεί από την αρχή για να ενσωματωθεί στην καινούργια εφαρμογή. Έτσι η γνώση που έχει αποκτηθεί από την δημιουργία μιας εφαρμογής τις περισσότερες φορές δεν καταγράφεται και χάνεται με αποτέλεσμα σε μια καινούργια εφαρμογή κάποια βήματα να γίνονται ξανά από την αρχή.

Θα ήταν μεγάλο πλεονέκτημα για ένα οργανισμό να μπορούσε να χρησιμοποιήσει ένα τυποποιημένο μηχανισμό καταγραφής και διάδοσης της γνώσης αυτής έτσι ώστε να μπορέσει να αυξήσει την ποιότητα και να μειώσει τον χρόνο ανάπτυξης των εφαρμογών. Ο μηχανισμός αυτός είναι η οπτική μοντελοποίηση των αναπτυσσόμενων εφαρμογών. Ένας τέτοιος μηχανισμός οπτικής μοντελοποίησης είναι η **UML (Unified Modeling Language – Ενοποιημένη Γλώσσα Μοντελοποίησης**.

Κεφάλαιο 4. Unified Modeling Language (UML)

Γενικά

Η Unified Modeling Language (UML[3]) είναι ένα πρότυπο, ένας καλά καθορισμένος και κοινός τρόπος επικοινωνίας μεταξύ των εμπλεκομένων μερών (αναλυτές, σχεδιαστές, προγραμματιστές, διευθυντές, πελάτες κλπ.) στην διάρκεια ανάπτυξης συστημάτων. Ο οργανισμός OMG, ο οποίος ανέπτυξε και καθιέρωσε την UML σαν γενικό πρότυπο, αναφέρει την UML σαν...

«...μια γραφική γλώσσα η οποία συμβάλλει στην οπτικοποίηση (visualizing), προσδιορισμό (specifying), κατασκευή (constructing) και τεκμηρίωση (documenting) συστημάτων λογισμικού...» [20]

Η UML αποτελεί μια σαφής «γλώσσα» για προσδιορισμό, απεικόνιση, κατασκευή και τεκμηρίωση των μερών των πληροφοριακών συστημάτων που αναπτύσσονται σε ένα οργανισμό καθώς επίσης και του σχεδιασμού του επιχειρηματικού μοντέλου (business model) της εφαρμογής. Η UML είναι η συλλογή σε ένα καθορισμένο πρότυπο (standard) των καλύτερων μερών των μεθοδολογιών για Αντικειμενοστραφή ανάλυση και σχεδιασμό, που προϋπήρξαν στις δεκαετίες του ‘80 και στα τέλη της δεκαετίας του ‘90 και περιλαμβάνει όλα τα πλεονεκτήματα τους που οδήγησαν στην μοντελοποίηση και κατασκευή μεγάλων και σύνθετων υπολογιστικών συστημάτων.

Συγκεκριμένα η UML καθιερώθηκε σαν πρότυπο τον Νοέμβριο του 1997 από την Object Management Group (OMG), ένα ανοιχτό consortium εταιρειών πληροφορικής και αποτελεί συνένωση των μεθοδολογιών **Object Modeling Technique (OMT)** του Jim Rumbaugh, **Booch** του Grady Booch και της μεθοδολογίας **Objectory** του Ivar Jacobson (οι οποίοι είναι γνωστοί στην διεθνή βιβλιογραφία σαν «***the Three Amigos***»). Στόχος του οργανισμού OMG ήταν η συνένωση των καλύτερων τμημάτων των παραπάνω μεθοδολογιών και η παρουσίαση ενός γενικού και κοινώς αποδεκτού προτύπου σχεδιασμού και ανάπτυξης εφαρμογών.

Η UML είναι ένα σύνολο γραφικών σχημάτων, μια σημειογραφία που προσφέρει την δυνατότητα δημιουργίας των σχεδίων (blueprints[4]) του συστήματος. Η UML παρέχει στους εμπλεκομένους στην ανάπτυξη συστημάτων ένα ευρύ σύνολο γραφικών στοιχείων, που σχετίζονται με την τεχνολογία της πληροφορικής, όπως για παράδειγμα κλάσεις (classes), συστατικά (components), αντικείμενα (objects), κόμβους (nodes), δραστηριότητες (activities), use cases, καταστάσεις (states) και τις συσχετίσεις (relationships, associations) μεταξύ των παραπάνω στοιχείων.

Ένα παράδειγμα που χρησιμοποιείται ευρέως για την περιγραφή και την χρήση της UML (και γενικά για την χρήση ενός μοντέλου για μια αφηρημένη παρουσίαση της εφαρμογής) είναι αυτό της κατασκευής ενός οικοδομήματος. Ο αρχιτέκτονας (ο αναλυτής - σχεδιαστής) χρησιμοποιεί το αρχιτεκτονικό σχέδιο του οικοδομήματος για να μπορέσει να επικοινωνήσει με τον μηχανικό (προγραμματιστή) που έχει αναλάβει το χτίσιμο του οικοδομήματος. Εάν ο αρχιτέκτονας δεν είχε το σχέδιο και προσπαθούσε να επικοινωνήσει με τον μηχανικό μόνο με τα λόγια ή με κάποιον άλλο τρόπο για το πώς θα χτιστεί το

[3] Στη συνέχεια της εργασίας θα αναφερόμαστε στην Unified Modeling Language με τον όρο UML

[4] Ο όρος Blueprint προέρχεται από τον χώρο της αρχιτεκτονικής, όπου χρησιμοποιούνται μπλε χαρτιά με άσπρες γραμμές για να παρουσιαστεί το σχέδιο μια κατασκευής. Στην αρχιτεκτονική των εφαρμογών ο όρος blueprint δηλώνει το σχέδιο της εφαρμογής.

οικοδόμημα είναι πολύ πιθανό το τελικό αποτέλεσμα να είναι αμφιβόλου ποιότητας και εμπιστοσύνης. Έτσι και η UML παρέχει όλα εκείνα τα στοιχεία που θα χρησιμοποιηθούν για την υλοποίηση του σχεδίου της εφαρμογής που θα δείξει ο αρχιτέκτονας της εφαρμογής (ο αναλυτής) στον μηχανικό (στον προγραμματιστή).

Η UML παρέχει στους χρήστες ένα πλήθος από διαγράμματα κάθε ένα από τα οποία δείχνουν μια διαφορετική πλευρά του συστήματος. Οι διαφορετικές αυτές πλευρές του συστήματος ονομάζονται όψεις (views) και ουσιαστικά παρουσιάζουν το σύστημα από διαφορετικές οπτικές γωνίες όπως θα δούμε στη συνέχεια: την στατική, την λειτουργική και την δυναμική όψη του συστήματος.

Όψεις (Views) της UML

Μία όψη της UML είναι ένα σύνολο διαγραμμάτων της UML και παρουσιάζει μια πλευρά του συστήματος. Ένα σύστημα μπορεί να έχει πολλές πλευρές, όπως για παράδειγμα η πλευρά της λειτουργικότητας (functionality) του συστήματος (τι κάνει το σύστημα), της στατικής (static) δομής του και των δυναμικών (dynamic) αλληλεπιδράσεων των στοιχείων του. Από την παρατήρηση όλων των πλευρών του συστήματος μέσα από τις όψεις της UML θα μπορέσουμε τελικά έχουμε μια ολοκληρωμένη εικόνα του συστήματος.

Οι βασικές όψεις της UML είναι:

- **Static (Στατική)**: δείχνει μια στατική εικόνα του συστήματος παρουσιάζοντας τα συστατικά στοιχεία (components) που χρησιμοποιούνται στην δημιουργία του συστήματος.
- **Functional (Λειτουργική)**: δείχνει τις βασικές λειτουργίες του συστήματος και τον τρόπο με τον οποίο υλοποιούνται αυτές.
- **Dynamic (Δυναμική)**: δείχνει τις δυναμικές αλληλεπιδράσεις (επικοινωνία) μεταξύ των συστατικών στοιχείων του συστήματος.

Στη συνέχεια παρουσιάζονται αναλυτικά οι όψεις της UML.

Static (Στατική)

Η όψη αυτή παρουσιάζει μια στατική εικόνα του συστήματος παρουσιάζοντας τα βασικά συστατικά στοιχεία που χρησιμοποιούνται στην δημιουργία του συστήματος. Η στατική εικόνα του συστήματος αναφέρεται στην απλή παρουσίαση των στοιχείων του συστήματος (κλάσεις, δομικά στοιχεία του συστήματος – αρχεία, κώδικας κλπ.) καθώς και στις συσχετίσεις μεταξύ των στοιχείων αυτών. Στην στατική όψη δεν παρουσιάζεται πως συμπεριφέρονται τα στοιχεία αυτά (οι λειτουργίες που έχουν και πώς επικοινωνούν δυναμικά) αλλά η όψη αυτή χρησιμοποιείται απλώς σαν ένα σχέδιο (blueprint) της εφαρμογής. Για το λόγο αυτό αναφέρεται σαν στατική όψη και περιλαμβάνει τα εξής διαγράμματα της UML:

- **Class διαγράμματα**: παρουσιάζουν τις κλάσεις του συστήματος, τα βασικά αντικείμενα από τα οποία αποτελείται το σύστημα.
- **Component διαγράμματα**: παρουσιάζουν τα δομικά στοιχεία του κώδικα του συστήματος, όπως ο εκτελέσιμος κώδικας, αρχεία κλπ.

- **Deployment διαγράμματα**: παρουσιάζουν τα μέσα που χρησιμοποιούνται για την φυσική υλοποίηση του συστήματος όπως οι υπολογιστές, εκτυπωτές, μέσα αποθήκευσης και άλλα μηχανήματα.

Functional (Λειτουργική)

Η όψη αυτή παρουσιάζει την λειτουργική πλευρά του συστήματος δηλαδή πώς λειτουργεί το σύστημα (βασικές λειτουργίες του συστήματος). Η λειτουργικότητα του συστήματος οδηγεί στην καταγραφή των απαιτήσεων για το σύστημα αφού μέσα από την όψη αυτή φαίνεται τι θέλουμε να κάνει το σύστημα. Τα διαγράμματα της UML που ανήκουν σε αυτή την όψη είναι:

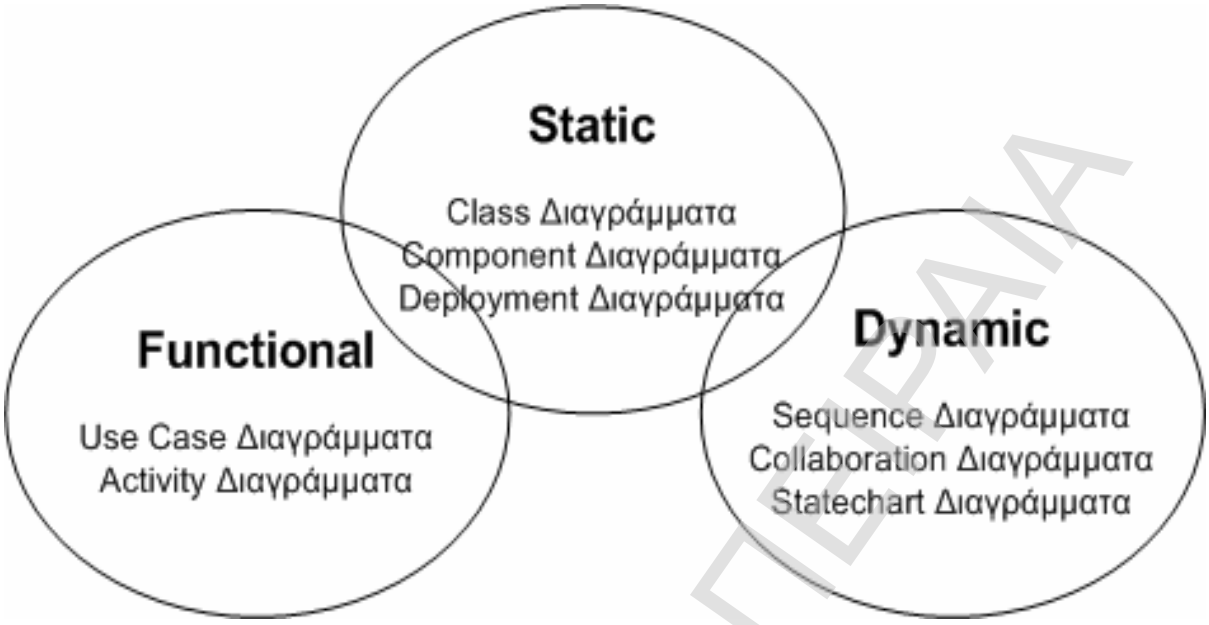
- **Use Case διαγράμματα**: παρουσιάζουν τις βασικές λειτουργίες του συστήματος (τι θέλουμε να κάνει το σύστημα). Χρησιμοποιούνται για την καταγραφή των απαιτήσεων του συστήματος.
- **Activity διαγράμματα**: παρουσιάζουν πως υλοποιούνται οι βασικές λειτουργίες του συστήματος.

Dynamic (Δυναμική)

Η όψη αυτή παρουσιάζει την συμπεριφορά των βασικών συστατικών στοιχείων του συστήματος, που είναι οι κλάσεις του συστήματος. Λέγοντας συμπεριφορά εννοούμε τον τρόπο με τον οποίο επικοινωνούν και αλληλεπιδρούν οι κλάσεις προκειμένου να υλοποιηθούν οι λειτουργίες του συστήματος. Τα διαγράμματα της UML που ανήκουν στην όψη αυτή είναι:

- **Sequence διαγράμματα**: παρουσιάζουν την αλληλεπίδραση των αντικειμένων μέσω της ανταλλαγής μηνυμάτων μεταξύ τους στην διάρκεια του χρόνου.
- **Collaboration διαγράμματα**: παρουσιάζουν την αλληλεπίδραση των αντικειμένων μέσω της ανταλλαγής μηνυμάτων σε αύξουσα σειρά.
- **Statechart διαγράμματα**: παρουσιάζουν τις διάφορες καταστάσεις, που έχουν οι βασικές κλάσεις της εφαρμογής όταν εκτελείται μια συγκεκριμένη λειτουργία.

Οι όψεις της UML όπως παρουσιάστηκαν παραπάνω δεν λειτουργούν ανεξάρτητα η μια από την άλλη. Προκειμένου να έχουμε μια ολοκληρωμένη εικόνα του συστήματος πρέπει να μελετήσουμε και να αναλύσουμε και τις τρεις όψεις του συστήματος και να τις χρησιμοποιήσουμε ταυτόχρονα για να έχουμε μια εμφάνιση του συστήματος από πολλές διαφορετικές οπτικές γωνίες.



Διάγραμμα 13: Όψεις της UML

Όπως βλέπουμε από το παραπάνω διάγραμμα οι όψεις της UML συνδυάζονται μεταξύ τους για να παρουσιάσουν όλες τις πλευρές του συστήματος. Η χρήση κάθε όψης συνεπάγεται την δημιουργία κάποιων διαγραμμάτων της UML που ανήκουν στην συγκεκριμένη όψη. Αυτό σημαίνει ότι πρέπει να δημιουργήσουμε διάφορα διαγράμματα της UML και να τα χρησιμοποιήσουμε σε συνδυασμό με τα υπόλοιπα διαγράμματα για να έχουμε μια ολοκληρωμένη εικόνα του συστήματος.

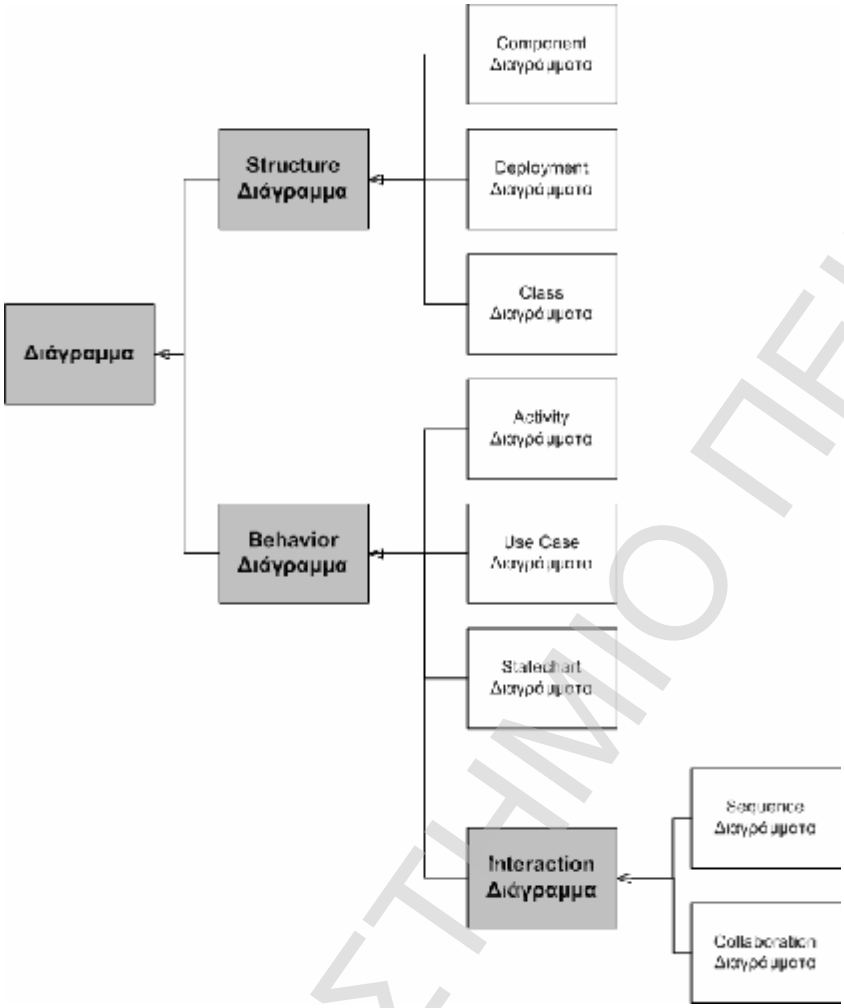
Διαγράμματα της UML

Τα βασικά διαγράμματα που παρέχονται από την UML είναι τα εξής:

- Use case
- Activity
- Class
- Sequence
- Collaboration
- Statechart
- Component
- Deployment

Τα διαγράμματα της UML χρησιμοποιούνται από όλους τους εμπλεκόμενους στην ανάπτυξη του συστήματος. Κάθε ένα από τα διαγράμματα, όπως και οι όψεις της UML, παρουσιάζουν ένα συγκεκριμένο χαρακτηριστικό του συστήματος (δομή του συστήματος, συμπεριφορά των στοιχείων που ανήκουν στο σύστημα και επικοινωνία μεταξύ αυτών των στοιχείων).

Τα διαγράμματα της UML ανάλογα με το περιεχόμενο τους ανήκουν σε συγκεκριμένες ομάδες διαγραμμάτων, όπως φαίνεται στο παρακάτω διάγραμμα:



Διάγραμμα 14: Ομαδοποίηση διαγραμμάτων UML

Interaction: δείχνουν την επικοινωνία μεταξύ των βασικών οντοτήτων του συστήματος.

Behavior: δείχνουν την συμπεριφορά των βασικών οντοτήτων του συστήματος.

Structure: δείχνουν την δομή του συστήματος.

Στην συνέχεια παρουσιάζονται τα κυριότερα διαγράμματα της UML.

Use Case Διαγράμματα

Τα Use Case διαγράμματα είναι από τα σημαντικότερα διαγράμματα της UML και χρησιμοποιούνται πάντα κατά την διάρκεια ανάλυσης του συστήματος. Το Use Case διάγραμμα είναι ένα διάγραμμα που δείχνει μια γενική όψη του συστήματος από την πλευρά του χρήστη, δείχνει δηλαδή τις λειτουργίες του συστήματος όπως τις αντιλαμβάνεται ένας εξωτερικός παρατηρητής του συστήματος. Με τα Use Case διαγράμματα παρουσιάζονται οι απαιτήσεις (requirements) των χρηστών από το σύστημα και αναπαριστάται η λειτουργικότητα του συστήματος (functionality).

Τα βασικά στοιχεία (elements) του Use Case διαγράμματος είναι:

- **Actors**

  Οι actors αναπαριστούν τους χρήστες του συστήματος που είναι έξω από αυτό και επικοινωνούν με το σύστημα. Οι actors αφού είναι έξω από το σύστημα διαμορφώνουν την περιφέρεια του συστήματος και ορίζουν την εμβέλεια (scope) του συστήματος. Οι actors καθορίζουν το εξωτερικό περιβάλλον και διαχωρίζουν το σύστημα από αυτό . Οι actors αναπαρίστανται σαν φιγούρες (stick figures).

- **Use Case**

  Το use case χρησιμοποιείται για να αναπαραστήσει τμήματα της λειτουργικότητας που προσφέρεται από το σύστημα προς τον κάθε actor του συστήματος. Δείχνουν τις βασικές λειτουργίες (key features) του συστήματος. Ένας εύκολος τρόπος για να βρούμε τα use cases είναι να δούμε τους actors του συστήματος και να σκεφτούμε **τι θέλουν να κάνουν με το σύστημα**. Συνήθως τα use cases ανακαλύπτονται μέσα από συνεντεύξεις και συνομιλίες με τους χρήστες του συστήματος, οι οποίοι αναφέρουν τι θέλουν από το σύστημα χωρίς να τους ενδιαφέρει πώς θα υλοποιούνται οι λειτουργίες που θα προσφέρονται από αυτό.

  Με ένα use case **δεν περιγράφουμε πως** λειτουργεί το σύστημα αλλά το **τι είναι ικανό** να κάνει το σύστημα. Χρησιμοποιείται για να καθορίσει την συμπεριφορά του συστήματος χωρίς να αποκαλύπτει την εσωτερική δομή του. Κάθε use case στο διάγραμμα δείχνει τις λειτουργίες που η οντότητα εκτελεί καθώς αλληλεπιδρά με τους χρήστες (actors) της.

  Τα use cases αναπαρίστανται με ένα οβάλ σχήμα που περιέχει μια μικρή έκφραση με τη λειτουργία του use case αυτού.

Τα use cases χρησιμοποιούνται για την ανάλυση του συστήματος και βοηθάει πολύ στη καταγραφή των απαιτήσεων του συστήματος και στο σωστό σχεδιασμό του αφού πρέπει να έχουμε μια ολοκληρωμένη εικόνα για το τι θέλουμε και τι πρέπει να κάνει το σύστημα.

Έτσι κατά την αρχικό σχεδιασμό του συστήματος γίνεται προσπάθεια να καταγραφούν οι περισσότερες από τις λειτουργίες με την χρήση των use cases αλλά καθώς η ανάπτυξη προχωρά νέες λειτουργίες μπορεί να εμφανιστούν και να καταγραφούν στα Use Case διαγράμματα, που έχουν δημιουργηθεί.

- **Use case σενάρια**

Μαζί με ένα Use Case διάγραμμα συνήθως χρησιμοποιείται και ένας αριθμός από **use case σενάρια** που αναπαριστούν κάποιες περιπτώσεις (σενάρια) οι οποίες είναι πολύ πιθανό να συμβούν κατά τη διάρκεια εκτέλεσης του συστήματος. Με τον τρόπο αυτό μπορούμε να καταγράψουμε την συμπεριφορά του συστήματος σε διαφορετικές περιπτώσεις και να λάβουμε πολύτιμα συμπεράσματα σχετικά με την πορεία του σχεδιασμού του συστήματος. Τα use case σενάρια χρησιμοποιούνται για την δημιουργία άλλων διαγραμμάτων της UML (για παράδειγμα Activity διαγράμματα) καθώς επίσης και για τον τελικό έλεγχο του συστήματος αναφορικά με την σωστή συμπεριφορά του σε περιπτώσεις, που αναφέρονται στα σενάρια αυτά.

Οι συσχετίσεις (relationships) που είναι διαθέσιμες στα Use Case διαγράμματα είναι οι παρακάτω:

- **Association relationship**

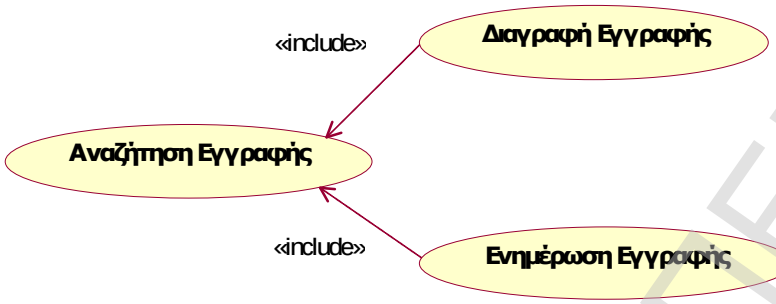
Χρησιμοποιείται για να δείξει ότι ένας actor επικοινωνεί με ένα use case. Αναπαρίσταται με μια γραμμή από τον actor προς το use case με το οποίο επικοινωνεί.

- **Include relationship**

Σε πολλές περιπτώσεις κάποιο use case περιλαμβάνει κάποια βήματα, τα οποία επαναλαμβάνονται και σε άλλα use cases. Προκειμένου να εξαλείψουμε αυτή την επανάληψη τα επαναλαμβανόμενα βήματα τα αναπαριστούμε με ένα use case και τα υπόλοιπα use cases συσχετίζονται με αυτό. Η συσχέτιση include χρησιμοποιείται για να δείξει ότι κάποια use cases χρησιμοποιούν κάποια λειτουργία (functionality) ενός άλλου use case. Αναπαρίσταται με μια διακεκομμένη γραμμή με την λέξη <<*include*>> με τη μορφή ανοικτού βέλους, με φορά από τα use cases που χρησιμοποιούν την κοινή λειτουργία προς το use case που περιέχει την κοινή λειτουργία.

Για παράδειγμα, έχουμε μια λειτουργία του συστήματος που ονομάζεται Διαγραφή Εγγραφής και χρησιμοποιείται για την διαγραφή μιας συγκεκριμένης εγγραφής από το σύστημα. Επίσης έχουμε μια λειτουργία που ονομάζεται Ενημέρωση Εγγραφής, η οποία χρησιμοποιείται για να ενημερώσει την εγγραφή. Οι δύο αυτές λειτουργίες για να

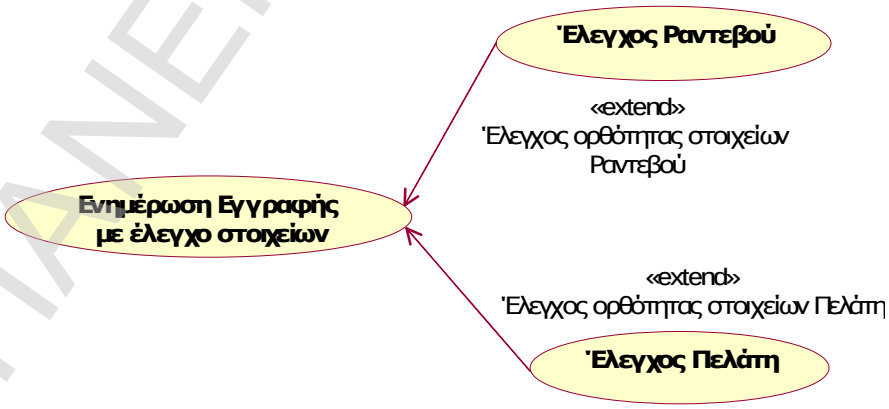
λειτουργήσουν πρέπει πρώτα να βρουν τη σωστή εγγραφή, δηλαδή πρέπει να χρησιμοποιήσουν την λειτουργία του συστήματος Εύρεση Εγγραφής. Σε μορφή κώδικα αυτή η συσχέτιση παρουσιάζεται με την κλήση της συνάρτησης Αναζήτηση Εγγραφής από δύο σημεία (από το σημείο της διαγραφής και από το σημείο την ενημέρωσης). Το χαρακτηριστικό της συσχέτισης αυτής είναι ότι η λειτουργία Ενημέρωση Εγγραφής θα χρησιμοποιεί **πάντα** την λειτουργία Αναζήτηση, που κάνει include.



- **Extend relationship**

Η extend συσχέτιση δείχνει ότι ένα use case χρησιμοποιεί την λειτουργικότητα ενός άλλου use case με το να προσθέτει (ή καλύτερα να επεκτείνει – extend) τις λειτουργίες του. Η συσχέτιση μεταξύ των δυο use cases δημιουργείται κάτω από μια συγκεκριμένη συνθήκη, δηλαδή πρέπει να ικανοποιείται κάποιο κριτήριο έτσι ώστε το ένα use case να χρησιμοποιήσει τις λειτουργίες του άλλου use case. Αναπαρίσταται με ένα διακεκομμένο βέλος, με φορά από το use case που παρέχει την λειτουργία προς το use case που δέχεται τις καινούργιες λειτουργίες, με την λέξη <<*extend*>>.

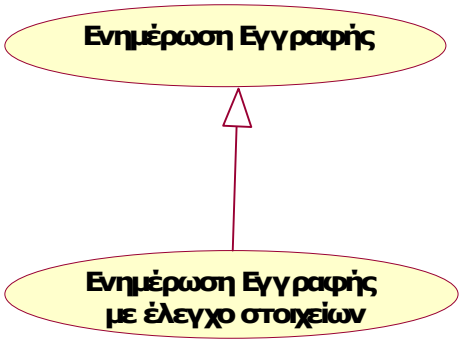
Για παράδειγμα, έχουμε την λειτουργία Ενημέρωση Εγγραφής, η οποία ενημερώνει με καινούργια στοιχεία μια καταχωρημένη εγγραφή. Η ενημέρωση της εγγραφής εκτός των άλλων ενημερώνει στοιχεία για ένα Πελάτη και για ένα Ραντεβού που έχει ο πελάτης. Η λειτουργία στην περίπτωση που αλλάζει στοιχεία Πελατών ή Ραντεβού, για να εξασφαλιστεί η ακεραιότητα των στοιχείων, θα πρέπει να χρησιμοποιήσει τις λειτουργίες Έλεγχος Πελάτη και Έλεγχος Ραντεβού για τον έλεγχο την ορθότητας των προτού κάνει την ενημέρωση της συνολικής εγγραφής.



- **Generalization relationship**

Χρησιμοποιείται για να αναπαραστήσει την κληρονομικότητα (inheritance) μεταξύ δυο use cases. Κληρονομικότητα μεταξύ use cases σημαίνει ότι ορισμένα use cases-παιδιά κληρονομούν κάποιες λειτουργίες από ένα use case-πατέρα. Αναπαρίσταται με μια γραμμή με ένα κλειστό βέλος, που δείχνει πάντα προς το use case πατέρας.

Για παράδειγμα, έχουμε το use case Ενημέρωση Εγγραφής και το use case Ενημέρωση εγγραφής με έλεγχο στοιχείων, όπως παρουσιάστηκε παραπάνω. Το δεύτερο use case κληρονομεί τα στοιχεία του πρώτου και προσθέτει επιπλέον λειτουργίες (έλεγχο στοιχείων πελατών και ραντεβού).



Επίσης generalization relationship (κληρονομικότητα) μπορεί να υπάρξει και μεταξύ των actors.

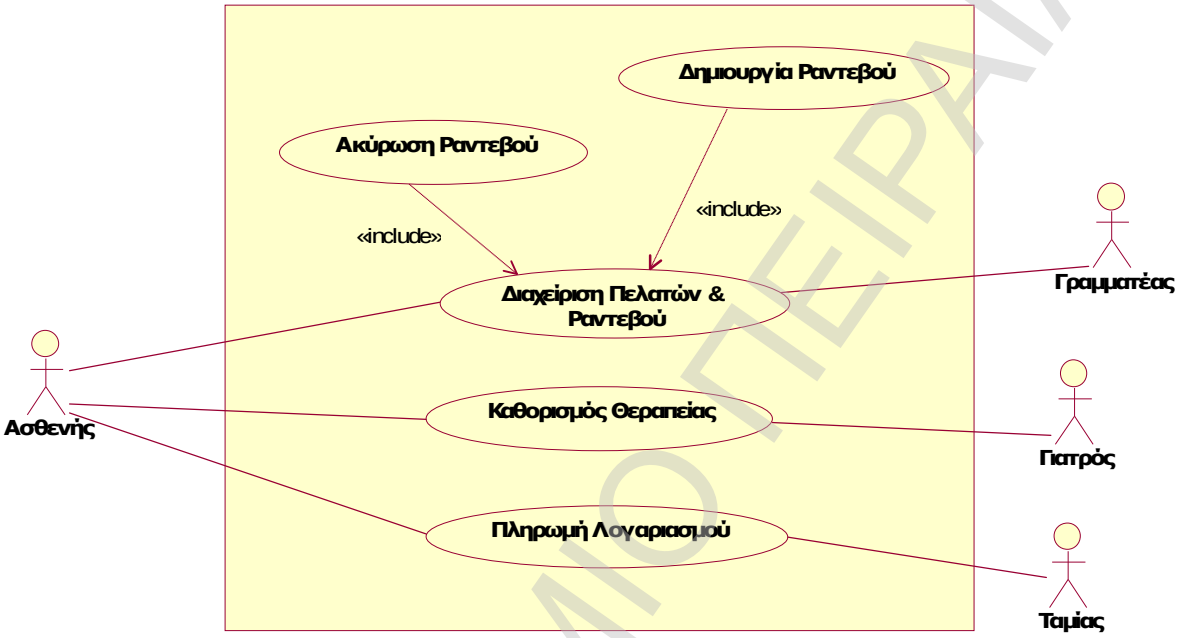
Χρήση των Use Case διαγραμμάτων

Τα Use Case διαγράμματα διευκολύνουν στην ανάλυση, σχεδίαση και ανάπτυξη του συστήματος. Ένα Use Case διάγραμμα περιγράφει την λειτουργικότητα του συστήματος, δηλαδή τις βασικές λειτουργίες του συστήματος όπως φαίνονται από ένα εξωτερικό παρατηρητή. Με τα Use Case διαγράμματα παρουσιάζονται οι απαιτήσεις των χρηστών από το σύστημα.

Η προβολή της ανταλλαγής των μηνυμάτων και των συσχετίσεων μεταξύ των actors του συστήματος μπορεί να βοηθήσει στην κατασκευή του συστήματος και στον έλεγχο των λειτουργιών του συστήματος με την χρήση διαφόρων use case σεναρίων (τα οποία αναπαρίστανται στη συνέχεια με την χρήση των Sequence και των Activity διαγραμμάτων). Τέλος, μέσα από τα Use Case διαγράμματα μπορεί να παραχθεί η τεκμηρίωση (documentation) του συστήματος αφού ουσιαστικά φαίνεται πώς οι χρήστες θα χρησιμοποιήσουν το σύστημα.

Παράδειγμα Use Case διαγράμματος

Στη συνέχεια παρουσιάζεται ένα παράδειγμα Use Case διαγράμματος, που παρουσιάζει ένα σύστημα διαχείρισης ενός ιδιωτικού ιατρείου.



Διάγραμμα 15: Παράδειγμα Use Case διαγράμματος

Παράδειγμα use case σεναρίου

Για να περιγράψουμε την λειτουργία ενός use case χρησιμοποιούμε ένα σενάριο που περιλαμβάνει μια σειρά από βήματα που γίνονται από την αρχή ως το τέλος της λειτουργίας του use case. Στα βήματα αυτά περιλαμβάνονται όλες οι περιπτώσεις που μπορούν να συμβούν κατά την διάρκεια της εκτέλεσης της λειτουργίας. Όλα τα διαφορετικά μονοπάτια από την αρχή ως το τέλος αποτελούν ένα διαφορετικό σενάριο. Στη συνέχεια παρουσιάζεται ένα σενάριο για την λειτουργία του συστήματος διαχείρισης ραντεβού των ασθενών.

Use Case: Διαχείριση Ραντεβού Ασθενών (Βασικό Σενάριο)

1. Εισαγωγή της γραμματέας στο σύστημα (έλεγχος στοιχείων χρήστη)
2. Λήψη στοιχείων ασθενούς
3. Έλεγχος στοιχείων ασθενούς
4. Δημιουργία νέου ραντεβού
   - 4.1. Καινούργιος ασθενής
     - 4.1.1. Καταχώρηση στοιχείων ασθενούς
     - 4.1.2. Δημιουργία καινούργιας καρτέλας ασθενούς
     - 4.1.3. Εύρεση διαθέσιμης ημερομηνίας ραντεβού
     - 4.1.4. Καταχώρηση στοιχείων Ραντεβού
     - 4.1.5. Ενημέρωση καρτέλας ασθενούς
   - 4.2. Παλιός ασθενής
     - 4.2.1. Άνοιγμα καρτέλας ασθενούς
     - 4.2.2. Εύρεση διαθέσιμης ημερομηνίας ραντεβού
     - 4.2.3. Καταχώρηση στοιχείων Ραντεβού
     - 4.2.4. Ενημέρωση καρτέλας ασθενούς
5. Τροποποίηση υπάρχοντος ραντεβού
   - 5.1. Καινούργιος ασθενής
     - 5.1.1. Έξοδος από το σύστημα
   - 5.2. Παλιός ασθενής
     - 5.2.1. Άνοιγμα καρτέλας ασθενούς
     - 5.2.2. Τροποποίηση στοιχείων Ραντεβού
     - 5.2.3. Ενημέρωση καρτέλας ασθενούς
6. Έξοδος από το σύστημα

Όπως βλέπουμε από το παραπάνω βασικό σενάριο μπορούμε να έχουμε παραπάνω από ένα μονοπάτια από την αρχή ως το τέλος της λειτουργίας. Κάθε ένα από τα μονοπάτια αυτά αποτελεί και ένα εναλλακτικό σενάριο εκτέλεσης της λειτουργίας.

Activity Διαγράμματα

Όπως ένα Use Case διάγραμμα δείχνει τον σκοπό (goal) μιας λειτουργίας του συστήματος, δηλαδή **τι** κάνει το σύστημα, τα Activity διαγράμματα δείχνουν τον τρόπο με τον οποίο θα επιτευχθεί η λειτουργία αυτή, δηλαδή **πως** θα εκτελεστεί η λειτουργία αυτή. Με τα Activity διαγράμματα μπορούμε να αναπαραστήσουμε πως υλοποιούνται τα διάφορα workflows στο σύστημα (η ροή δηλαδή της εργασίας στη λειτουργία αυτή), ποια μονοπάτια απόφασης (decision paths) πρέπει να ακολουθηθούν, οι διάφορες περιπτώσεις που υπάρχουν για να ολοκληρωθεί η λειτουργία που περιγράφεται και πως ολοκληρώνεται το workflow.

Συνήθως τα Activity διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν κάποιο σενάριο εκτέλεσης μιας λειτουργίας του συστήματος, που έχει δημιουργηθεί για την περίπτωση ενός Use Case διαγράμματος.

Τα βασικά στοιχεία (elements) ενός Activity διαγράμματος είναι:

- **Activities (Δραστηριότητες)**

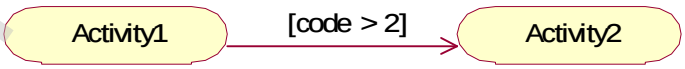
  Activity (δραστηριότητα) είναι ένα βήμα της συνολικής διαδικασίας στο οποίο πρέπει να εκτελεστεί η λειτουργία για να προχωρήσουμε σε ένα άλλο activity. Το activity αναπαρίσταται με ένα ορθογώνιο που έχει στρογγυλές άκρες και περιέχει το όνομα της δραστηριότητας.

- **Transition (Μετάβαση)**

  Transition (μετάβαση) είναι η αναπαράσταση του επόμενου βήματος, η μετάβαση δηλαδή σε ένα άλλο activity. Αναπαρίσταται με ένα βέλος, που δείχνει προς το επόμενο activity της διαδικασίας.

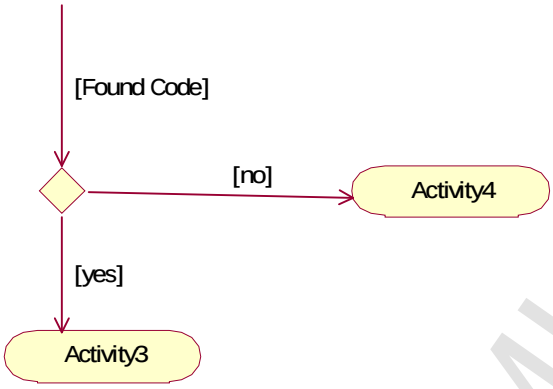
- **Guard Condition (Συνθήκες Ελέγχου)**

  Για να μεταβούμε από το ένα activity σε ένα άλλο πολλές φορές πρέπει κάποια συγκεκριμένη συνθήκη να είναι αληθής. Η guard condition (συνθήκη ελέγχου) αναπαρίσταται με το κείμενο της συνθήκης μέσα σε αγκύλες([..]), το οποίο τοποθετείται πάνω από το βέλος της μετάβασης (transition).



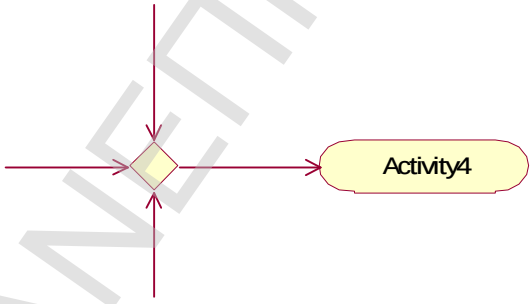
- **Decision (Σημεία Απόφασης)**

Decision είναι ένα σημείο απόφασης στο οποίο καταλήγουν κάποια transitions και πρέπει να ακολουθήσουν ένα συγκεκριμένο μονοπάτι ανάλογα με κάποια συνθήκη. Η κάθε έξοδος του decision έχει μια guard condition, μια συνθήκη δηλαδή που πρέπει να είναι αληθής έτσι ώστε να επιλεγεί η συγκεκριμένη έξοδος. Οι συνθήκες στις εξόδους του decision είναι αμοιβαίως αποκλειόμενες, δηλαδή η αληθής τιμή της μιας εξόδου σημαίνει ψευδής τιμή των υπολοίπων εξόδων έτσι ώστε να υπάρχει κάθε φορά μια μόνο έξοδος από το σημείο απόφασης. Το decision αναπαρίσταται με ένα ρόμβο.



- **Merge Point (Σημεία Συγχώνευσης)**

Merge point είναι το σημείο στο οποίο καταλήγουν δύο ή περισσότερα μονοπάτια και από το οποίο φεύγει ένα transition προς κάποιο activity. Τα μονοπάτια τα οποία καταλήγουν στο merge point πρέπει να είναι αμοιβαίως αποκλειόμενα, δηλαδή να είναι ανεξάρτητα το ένα από το άλλο. Το merge point αναπαρίσταται με ένα ρόμβο.



- **Start και End (Αρχή – Τέλος διαγράμματος)**

Μια μαύρη κουκκίδα δείχνει την αρχή του activity diagram 

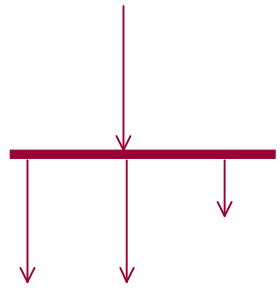
ενώ μια μικρή μαύρη κουκκίδα μέσα σε να κύκλο δείχνει το τέλος του activity diagram. 

Σε ένα activity diagram υπάρχει πάντα **μια αρχή** ενώ μπορεί να υπάρχουν **ένα ή περισσότερα τέλη**.

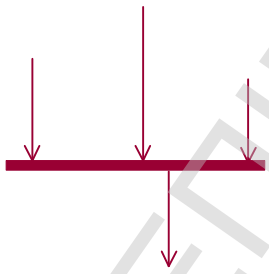
- **Concurrency (Ταυτόχρονη Επεξεργασία)**

Η ταυτόχρονη επεξεργασία (concurrency) σε ένα activity diagram αναπαρίσταται με μια χοντρή μαύρη γραμμή. Ταυτόχρονη επεξεργασία σημαίνει ότι εκτελούνται ταυτόχρονα διαφορετικές λειτουργίες οι οποίες καταλήγουν σε ένα κοινό σημείο. Η διαφορά με το σημείο συγχώνευσης (merge point) που παρουσιάσαμε προηγουμένως είναι ότι εδώ δεν έχουμε αμοιβαίως αποκλειόμενες λειτουργίες που καταλήγουν σε ένα σημείο αλλά έχουμε ένα συγχρονισμό λειτουργιών που εκτελούνται παράλληλα. Έχουμε δυο τύπους ταυτόχρονης επεξεργασίας:

Fork: ταυτόχρονη επεξεργασία όπου μια δραστηριότητα ξεκινάει πολλές ταυτόχρονες παράλληλες δραστηριότητες (αναπαράσταση **fork)**



Synchronization: ταυτόχρονη επεξεργασία όπου πολλαπλές παράλληλες δραστηριότητες συγχρονίζονται σε μια (αναπαράσταση **synchronization)**

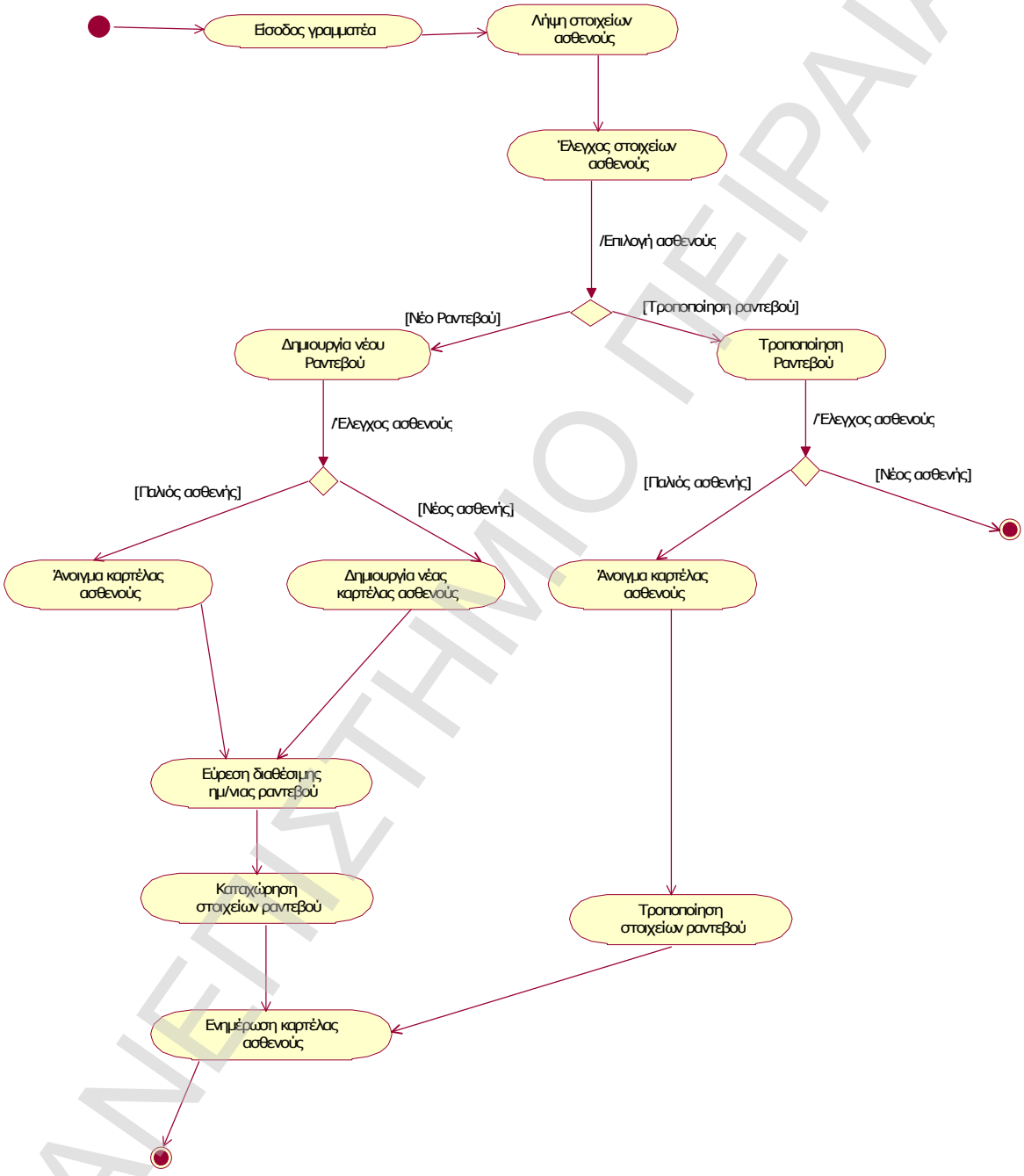


Χρήση Activity διαγραμμάτων

Τα Activity διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν πως εκτελείται μια λειτουργία του συστήματος, παρουσιάζονται τα βήματα δηλαδή που απαιτούνται για την εκτέλεση της λειτουργίας. Με τα Activity διαγράμματα ο χρήστης μπορεί να δει την ροή της εργασίας (workflow) γεγονός που τον βοηθάει να ελέγξει όλες τις συνθήκες που πρέπει να συμπεριλάβει στην εφαρμογή.

Παράδειγμα Activity διαγράμματος

Στη συνέχεια παρουσιάζεται ένα Activity διάγραμμα, που περιγράφει το σενάριο που παρουσιάστηκε παραπάνω για την λειτουργία της διαχείρισης των ραντεβού των ασθενών.



Διάγραμμα 16: Παράδειγμα χρήσης Activity διαγράμματος

Class Διαγράμματα

Τα Class Διαγράμματα είναι ο βασικός κορμός κάθε μοντέλου που περιλαμβάνει μια αντικειμενοστραφής ανάλυση. Ένα Class διάγραμμα περιγράφει την στατική δομή της εφαρμογής και περιλαμβάνει τις οντότητες (τα αντικείμενα – κλάσεις[5]) που θα χρησιμοποιηθούν σε αυτή. Συνήθως η δημιουργία ενός Class διαγράμματος έχει προέλθει μέσα από συνεντεύξεις με τους πελάτες της εφαρμογής για να ανακαλυφθούν οι κύριες οντότητες της εφαρμογής καθώς και από την ανάλυση των υπολοίπων διαγραμμάτων, που έχουν ήδη δημιουργηθεί όπως για παράδειγμα τα Use Case και τα Activity διαγράμματα.

Τα Use Case διαγράμματα δείχνουν τις απαιτήσεις σε αντικείμενα (objects) που χρειάζονται για να επιτευχθούν οι στόχοι (goals) του συστήματος, δηλαδή μέσα από την ανάλυση των διαγραμμάτων αυτών ανακαλύπτονται οι κύριες οντότητες της εφαρμογής. Τα Activity διαγράμματα χρησιμεύουν για τον καθορισμό της συμπεριφοράς (behavior) των αντικειμένων δηλαδή πως θα λειτουργούν τα αντικείμενα ποια θα είναι η ροή της εργασία καθ' όλη τη διάρκεια της εφαρμογής για να επιτευχθεί ο σκοπός της εφαρμογής.

Ένα Class διάγραμμα περιγράφει την κατασκευαστική δομή (structural view) του συστήματος, δηλαδή τους τύπους των αντικειμένων που θα χρησιμοποιηθούν στο σύστημα, παρουσιάζει την στατική δομή των αντικειμένων αυτών και περιγράφει το είδος των συσχετίσεων που υπάρχουν ανάμεσα στα αντικείμενα αυτά. Μέσα από την ανάλυση του Class διαγράμματος βλέπουμε τις κλάσεις που απαιτούνται για την υλοποίηση της εφαρμογής. Τα βασικά δομικά στοιχεία των Class διαγραμμάτων είναι:

- Οι κλάσεις (δομή και συμπεριφορά τους – μέθοδοι κλάσεων).
- Οι συσχετίσεις μεταξύ των κλάσεων.
- Η πολλαπλότητα (multiplicity) και navigation (ροή μέσα στις κλάσεις).

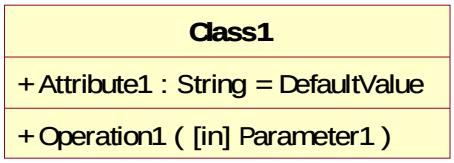
Class (Κλάση)

Μια κλάση αντιπροσωπεύει ένα σύνολο αντικειμένων με παρόμοια δομή, συμπεριφορά και συσχετίσεις. Είναι το πρότυπο (pattern) από το οποίο τα διάφορα αντικείμενα παράγονται κατά την διάρκεια της εκτέλεσης της εφαρμογής. Τα αντικείμενα που παράγονται από μια κλάση αναφέρονται σαν **στιγμιότυπα** της κλάση αυτής. Για παράδειγμα μπορούμε να έχουμε την κλάση Λογαριασμός Τράπεζας και από την κλάση αυτή να έχουμε τα στιγμιότυπα Λογαριασμός Μισθοδοσία, Λογαριασμός Ταμιευτηρίου, Λογαριασμός Όψεως κλπ.

Η κλάση είναι το βασικό δομικό στοιχείο του αντικειμενοστραφούς σχεδιασμού και ανάλυσης όπου γίνεται προσπάθεια να δημιουργηθεί το μοντέλο του συστήματος αποτελούμενο από κλάσεις, οι οποίες περιέχουν τα χαρακτηριστικά (attributes) και τις μεθόδους (operations) της κλάσης.

Η αναπαράσταση μιας κλάσης στην UML γίνεται με ένα ορθογώνιο, το οποίο αποτελείται από 3 τμήματα.

[5] Κλάση (class) σε όρους μιας γλώσσας προγραμματισμού είναι ένα πρότυπο που περιέχει χαρακτηριστικά και μεθόδους και από το πρότυπο αυτό δημιουργούνται τα στιγμιότυπα της κλάσης, που ονομάζονται αντικείμενα (objects). Η κλάση δηλαδή είναι μια αφηρημένη έννοια που αναφέρεται σε ένα όρο του πραγματικού κόσμου ενώ το αντικείμενο είναι ένα στιγμιότυπο της κλάσης αυτής. Συχνά όμως παρατηρείται στην βιβλιογραφία ότι δεν γίνεται διαχωρισμός μεταξύ του αντικειμένου και της κλάσης. Για λόγους ευκολίας, στην συνέχεια της εργασίας θα θεωρούμε τους δυο όρους αυτούς ταυτόσημους και θα χρησιμοποιούμε είτε τον όρο αντικείμενο είτε τον όρο κλάση χωρίς να γίνεται κάποιος συγκεκριμένος διαχωρισμός.



Τα τμήματα της κλάσης είναι:

- **Τμήμα Ονόματος Κλάσης**: στο τμήμα αυτό γράφεται το όνομα της κλάσης.
- **Τμήμα Χαρακτηριστικών Κλάσης**: στο τμήμα αυτό δηλώνονται τα χαρακτηριστικά της κλάσης, οι μεταβλητές δηλαδή που ανήκουν στην κλάση. Υποχρεωτικά κάθε χαρακτηριστικό ορίζεται με το όνομα και τον τύπο του (π.χ. Integer, String, Date κλπ.). Προαιρετικά για κάθε χαρακτηριστικό ορίζονται και τα εξής:

  Visibility: ορίζεται η ορατότητα του χαρακτηριστικού της κλάσης, σε ποια σημεία του προγράμματος είναι διαθέσιμο το συγκεκριμένο χαρακτηριστικό, Ορίζει επομένως ποιες άλλες κλάσεις μπορούν να «δουν» το χαρακτηριστικό αυτό. Οι διαθέσιμες επιλογές που έχει ο χρήστης για να ορίσει το visibility του χαρακτηριστικού είναι:

Συμβολισμός	**Ονομασία**	**Περιγραφή**
+	Public:	Ορατό από άλλες κλάσεις σε ολόκληρο το πρόγραμμα
-	Private:	Ορατό μόνο μέσα στην ίδια την κλάση
#	Protected:	Ορατό μόνο από υπό-κλάσεις (κλάσεις-παιδιά) της κλάσης
~	Package:	Ορατό από άλλες κλάσεις του ίδιου package της κλάσης

Πίνακας 1: Επιλογές ορατότητας (visibility)

  Αρχική Τιμή: δηλώνεται η αρχική τιμή του χαρακτηριστικού της κλάσης.

  Η γενική μορφή δήλωσης ενός attribute είναι:

  Ορατότητα Όνομα_Χαρακτηριστικού: Τύπος = Αρχική_Τιμή

  Για παράδειγμα: *+ myAge: Integer = 28*

- **Τμήμα Λειτουργιών Κλάσης**: στο τμήμα αυτό δηλώνονται οι λειτουργίες της κλάσης, δηλαδή οι συναρτήσεις της κλάσης. Όπως και με τα χαρακτηριστικά έτσι και στο τμήμα αυτό δηλώνεται το όνομα της μεθόδου, ο επιστρεφόμενος τύπος της, τυχόν ορίσματα (parameter list) που έχει η μέθοδος καθώς και το visibility.

  Η γενική μορφή δήλωσης μιας μεθόδου είναι:

  Ορατότητα Όνομα_Μεθόδου(Όνομα_Ορίσματος: Τύπος_Ορίσματος, …): Τύπος

  Για παράδειγμα: + CalculateBonus(misthos: double, bonus: double): double

Relationships (Συσχετίσεις)

Κάθε κλάση μπορεί να επικοινωνεί και να συσχετίζεται με κάποια άλλη κλάση. Υπάρχουν διάφοροι τύποι συσχετίσεων μεταξύ των κλάσεων, που παρουσιάζονται στη συνέχεια:

- **Association**

  Δείχνει ότι μια κλάση σχετίζεται με κάποια άλλη κλάση. Η συσχέτιση αυτή μπορεί να είναι δυο κατευθύνσεων (σχέση ανάμεσα και στις δύο κλάσεις) ή μιας κατεύθυνσης (δηλαδή μόνο η μία κλάση «γνωρίζει» την σχέση της με την άλλη κλάση). Στην περίπτωση της μονής κατεύθυνσης η γραμμή που συνδέει τις κλάσεις έχει ένα βέλος, που δείχνει προς την κλάση που δεν γνωρίζει την σχέση.

  Ένα παράδειγμα μια μονής σχέσης μεταξύ δύο κλάσεων είναι η σχέση μεταξύ των κλάσεων Χρήστης και Computer. Η σχέση μεταξύ των δύο αυτών κλάσεων είναι ότι η κλάση Χρήστης χρησιμοποιεί την κλάση Computer (όχι όμως και το αντίστροφο).

  Τα χαρακτηριστικά που ορίζονται για κάθε association είναι:

  Όνομα συσχέτισης: χρησιμοποιείται μια φράση με ρήμα η οποία δηλώνει με ποιο τρόπο η κλάση σχετίζεται με μια άλλη. Για παράδειγμα ο Χρήστης *χρησιμοποιεί* το Computer.

  Πολλαπλότητα (Multiplicity) συσχέτισης: Πολλαπλότητα είναι ο αριθμός των εμφανίσεων (instance) μιας κλάσης που συσχετίζεται με μια άλλη κλάση. Πολλαπλότητα είναι μια ένδειξη για το πόσα αντικείμενα συμμετέχουν σε κάθε συσχέτιση μεταξύ των κλάσεων. Για παράδειγμα ένας Χρήστης μπορεί να έχει από 1..* Computer (ένα ή περισσότερα computers).

  Οι περιπτώσεις πολλαπλής συσχέτισης μεταξύ των κλάσεων δίνεται από τον παρακάτω πίνακα:

Πολλαπλότητα	Επεξήγηση
0..1	Καμία ή μια εμφάνιση
0..* ή *	Κανένα όριο στον αριθμό των εμφανίσεων
1	Ακριβώς μία εμφάνιση
1..*	Τουλάχιστον μια συσχέτιση

Πίνακας 2: Επιλογές πολλαπλότητας (multiplicity)

- **Aggregation**

  Είναι μια ειδική μορφή συσχέτισης και δείχνει ότι οι κλάσεις σχετίζονται μεταξύ με τέτοιο τρόπο έτσι ώστε να ενώνονται μεταξύ τους για να δημιουργήσουν μια νέα κλάση (για παράδειγμα μια κλάση Βιβλίο σχετίζεται με την κλάση Συλλογή Βιβλίων με την

έννοια ότι πολλά Βιβλία δημιουργούν μια Συλλογή Βιβλίων). Υπάρχουν 2 ειδών aggregations:

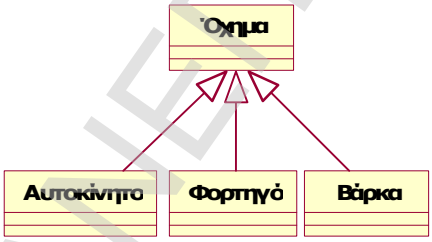
Basic Aggregation: δείχνει ότι μια κλάση είναι υφιστάμενη μιας άλλης κλάσης αλλά η κλάση-μέρος μπορεί να υπάρξει και μόνη της (π.χ. η κλάση Βιβλίο μπορεί να υπάρξει και μόνη της και μπορεί να δημιουργήσει την κλάση Συλλογή Βιβλίων). Αναπαρίσταται με μια γραμμή με ένα άσπρο ρόμβο προς την πλευρά της κλάσης-σύνολο.

Composition Aggregation: δείχνει την συσχέτιση μέρος-σύνολο αλλά σημαίνει ότι η κλάση-μέρος δεν μπορεί να υπάρξει μόνη της (π.χ. η κλάση Βιβλίο αποτελείται από πολλές κλάσεις Κεφάλαια, αλλά η κλάση Κεφάλαιο από μόνη της δεν μπορεί να υπάρχει). Αναπαρίσταται με μια γραμμή με ένα μαύρο ρόμβο προς την πλευρά της κλάσης σύνολο.

- **Generalization**

  Είναι ο τρόπος αναπαράστασης της κληρονομικότητας (inheritance) μεταξύ των κλάσεων (συσχέτιση πατέρα-παιδιού μεταξύ των κλάσεων). Στην συσχέτιση αυτή ο πατέρας κληρονομεί τα χαρακτηριστικά και τις μεθόδους του στα παιδιά.

  Η αναπαράσταση της κληρονομικότητας γίνεται με μια γραμμή με ένα κλειστό βέλος, που δείχνει από την κλάση-παιδί προς την κλάση-πατέρας. Για παράδειγμα έχουμε την κλάση Όχημα που κληρονομείται από τις κλάσεις Αυτοκίνητο, Φορτηγό και Βάρκα.



Χρήση των Class διαγραμμάτων

Τα Class διαγράμματα είναι ο κορμός της ανάλυσης του συστήματος και χρησιμοποιούνται για να παραστήσουν την στατική δομή της εφαρμογής. Παρουσιάζουν τα αντικείμενα, από τα οποία αποτελείται η εφαρμογή καθώς και τις συσχετίσεις μεταξύ αυτών. Μέσα από τα Class διαγράμματα θα μπορέσουν οι προγραμματιστές της εφαρμογής να δουν ποια αντικείμενα θα χρησιμοποιηθούν και έτσι θα μπορέσουν να αρχίσουν να δημιουργούν τις κλάσεις του συστήματος.

Sequence Διαγράμματα

Τα Sequence διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν την αλληλεπίδραση των αντικειμένων του συστήματος στη διάρκεια του χρόνου για μια συγκεκριμένη δραστηριότητα του συστήματος. Συνήθως χρησιμοποιείται ένα Sequence διάγραμμα για να περιγράψει μια λειτουργία του συστήματος που έχει περιγραφεί ήδη με ένα Use Case διάγραμμα. Ένα Sequence διάγραμμα δείχνει την συμμετοχή των αντικειμένων στην λειτουργία που περιγράφεται και την σειρά με την οποία ανταλλάσσονται τα μηνύματα που χρησιμεύουν για την επικοινωνία των αντικειμένων. Τα αντικείμενα που παρουσιάζονται σε ένα Sequence διάγραμμα έχουν προέλθει από την ανάλυση των Class διαγραμμάτων.

Με την χρήση των Sequence διαγραμμάτων μπορούμε να δούμε την διάρκεια ζωής των διαφόρων αντικειμένων σε μία συγκεκριμένη λειτουργία του συστήματος και την επικοινωνία μεταξύ τους με την προβολή των μηνυμάτων που ανταλλάσσονται. Λέγοντας διάρκεια ζωής εννοούμε την χρονική διάρκεια κατά την οποία τα αντικείμενα είναι ενεργά και ανταλλάζουν μηνύματα. Η έννοια του χρόνου αναπαρίσταται με την σειρά αποστολής των μηνυμάτων, η οποία καθορίζεται διαβάζοντας το διάγραμμα από πάνω προς τα κάτω.

Ένα Sequence διάγραμμα έχει δύο διαστάσεις. Η κάθετη διάσταση αντιπροσωπεύει τον χρόνο ζωής του κάθε αντικειμένου και η οριζόντια διάσταση περιέχει τα αντικείμενα που συμμετέχουν στην λειτουργία που περιγράφεται με το διάγραμμα.

Τα βασικά στοιχεία (elements) που χρησιμοποιούνται στα Sequence διαγράμματα είναι:

- **Objects (Αντικείμενα)**

  Είναι τα αντικείμενα τα οποία συμμετέχουν στην λειτουργία που περιγράφει το διάγραμμα και που επικοινωνούν το ένα με το άλλο. Τα αντικείμενα αναπαρίστανται με ένα τετράγωνο κουτί με το όνομα του αντικειμένου.

- **Objects Lifelines (Γραμμές Ζωής Αντικειμένων)**

  Είναι κάθετες γραμμές, που αναπαριστούν την διάρκεια ζωής των αντικειμένων. Μια **κάθετη διακεκομμένη γραμμή** δείχνει ότι το αντικείμενο είναι απενεργοποιημένο (αλλά όχι διαγραμμένο, συμμετέχει δηλαδή ακόμα στην λειτουργία απλώς ακόμα δεν έχει διαγραφεί) και περιμένει να δεχθεί κάποιο μήνυμα.

  Ένα **κάθετο στενό ορθογώνιο** δείχνει ότι το αντικείμενο είναι ενεργό και έχει στείλει κάποιο μήνυμα και πιθανόν να περιμένει απάντηση από το μήνυμα αυτό.

  Όταν ένα αντικείμενο έχει καταστραφεί (διαγραφεί) η γραμμή αυτή κόβεται και τοποθετείται ένα μεγάλο **X** πάνω στη γραμμή, που δείχνει ότι το αντικείμενο αυτό παύει πλέον να συμμετέχει στη λειτουργία που περιγράφεται (είναι αντίστοιχο με την εντολή *destroy* που χρησιμοποιείται για να διαγραφεί το αντικείμενο)

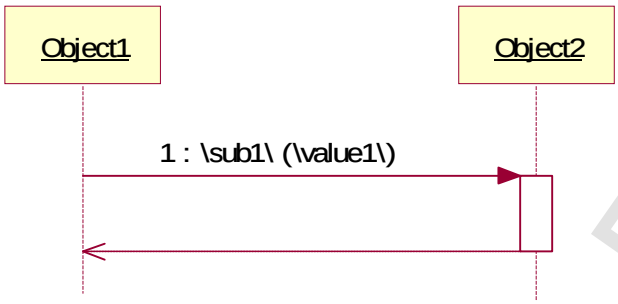
- **Messages (Μηνύματα)**

  Είναι τα διάφορα μηνύματα που αποστέλλονται ανάμεσα στα διάφορα αντικείμενα του διαγράμματος. Ένα μήνυμα αναπαρίσταται με ένα βέλος με κατεύθυνση στο αντικείμενο παραλήπτης και είναι αυτά που προκαλούν την έναρξη κάποιας λειτουργίας στο αντικείμενο στο οποίο απευθύνονται. Τα μηνύματα ουσιαστικά αναπαριστούν τις συναρτήσεις οι οποίες αποστέλλονται ανάμεσα στα διάφορα αντικείμενα της

εφαρμογής. Στα μηνύματα μπορούμε να ορίσουμε όνομα, ορίσματα (arguments) που τυχόν έχουν καθώς και τύπο (return type).

Τα μηνύματα σε ένα Sequence διάγραμμα είναι δυο ειδών:

Synchronous: μηνύματα που αποστέλλονται σε ένα αντικείμενο και απαιτούν απάντηση (response) από αυτό ότι το μήνυμα έχει παραδοθεί ορθά. Η απάντηση στο μήνυμα αναπαρίσταται στο διάγραμμα με μια διακεκομμένη γραμμή από τον παραλήπτη προς τον αποστολέα.



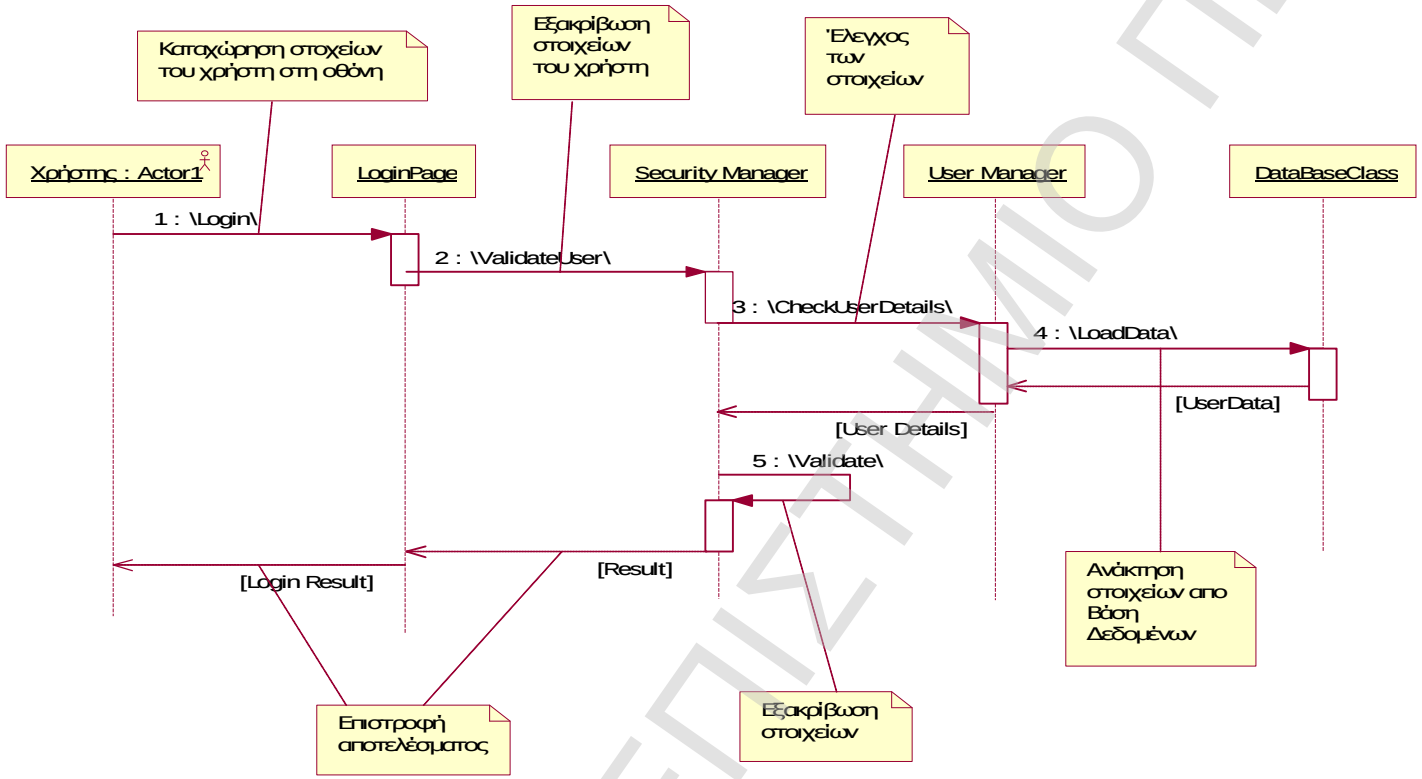
Asynchronous: μηνύματα που δεν απαιτούν απάντηση, απλώς είναι ένα απλό σήμα (signal) από ένα αντικείμενο που ενεργοποιεί ένα άλλο αντικείμενο.

Χρήση των Sequence διαγραμμάτων

Τα Sequence διαγράμματα χρησιμοποιούνται για να περιγράψουν μια λειτουργία του συστήματος στην διάρκεια του χρόνου. Η περιγραφή αυτή γίνεται μέσα από την παρουσίαση των αντικειμένων που ανήκουν στην λειτουργία αυτή, την παρουσίαση των μηνυμάτων (συναρτήσεων, μεθόδων) που ανταλλάσσονται μεταξύ των αντικειμένων καθώς και την παρουσίαση της διάρκειας ζωής των αντικειμένων, που αναπαριστάται από την σειρά των μηνυμάτων από πάνω προς τα κάτω.

Παράδειγμα Sequence διαγράμματος

Στη συνέχεια παρουσιάζεται ένα παράδειγμα Sequence διαγράμματος ελέγχου των στοιχείων των χρηστών, όταν αυτός συνδέεται σε μια login οθόνη για να εισέλθει σε μια εφαρμογή.



Διάγραμμα 17: Παράδειγμα Sequence διαγράμματος

Collaboration Διαγράμματα

Ένα Collaboration διάγραμμα είναι μια παραλλαγή των Sequence διαγραμμάτων στα οποία παρουσιάζονται για μια συγκεκριμένη λειτουργία του συστήματος τα αντικείμενα και η επικοινωνία μεταξύ τους. Η διαφορά είναι ότι δεν εμφανίζονται τα μηνύματα που αποστέλλονται μεταξύ των αντικειμένων στην διάρκεια του χρόνου (από πάνω προς τα κάτω) αλλά εμφανίζονται με αύξουσα αρίθμηση ανάλογα με την στιγμή της αποστολής τους.

Τα αντικείμενα που χρησιμοποιούνται είναι ίδια με τα αντικείμενα που χρησιμοποιήθηκαν στα Sequence διαγράμματα και αναπαρίστανται με ένα ορθογώνιο, που περιέχει το όνομα του αντικειμένου.

Τα μηνύματα αναπαρίστανται ως εξής:

 # αριθμός μηνύματος : [συνθήκη] όνομα_μηνύματος (ορίσματα)

όπου ο αύξων αριθμός του μηνύματος δείχνει την σειρά αποστολής των μηνυμάτων.

Όπως στα Sequence διαγράμματα η σειρά αποστολής των μηνυμάτων φαινόταν εάν τα διάβαζε κανείς από πάνω προς τα κάτω, στα Collaboration διαγράμματα η σειρά αποστολής των μηνυμάτων φαίνεται από τον αύξοντα αριθμό που υπάρχει στον τίτλο του μηνύματος.

Η μετάδοση των μηνυμάτων αναπαρίσταται με βέλη και όπως στην περίπτωση των Sequence διαγραμμάτων.

Χρήστη των Collaboration διαγραμμάτων

Τα Collaboration διαγράμματα έχουν την ίδια ακριβώς χρησιμότητα με τα Sequence διαγράμματα. Δείχνουν δηλαδή την λειτουργία ενός τμήματος του συστήματος παριστάνοντας τα διάφορα αντικείμενα και τα μηνύματα που ανταλλάσσονται μεταξύ τους. Η διαφορά με τα Sequence διαγράμματα είναι η παρουσίαση του χρόνου γίνεται όχι διαβάζοντας τα μηνύματα από πάνω προς τα κάτω αλλά η παρουσίαση της χρονικής διάρκειας παρουσιάζεται με ένα αύξοντα αριθμό που υπάρχει σε κάθε μήνυμα.

Statechart Διαγράμματα

Τα Statechart διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν τον κύκλο ζωής των διαφόρων αντικειμένων (objects) του συστήματος όπως αυτά έχουν δημιουργηθεί από τα Use Case, τα Sequence και τα Class διαγράμματα. Ο κύκλος ζωής των αντικειμένων αναφέρεται στις διάφορες καταστάσεις (states) τις οποίες έχουν αυτά τα αντικείμενα καθώς και στα γεγονότα (events) τα οποία δημιουργούν αυτές τις καταστάσεις. Περιγράφουν όλες τις πιθανές καταστάσεις στις οποίες μπορεί να βρεθεί ένα αντικείμενο όταν αντίστοιχα γεγονότα, που τα επηρεάζουν, συμβούν.

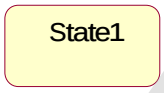
Μέσα από το διάγραμμα φαίνονται οι διάφορες λειτουργίες (activities) τις οποίες εκτελεί ένα αντικείμενο όταν βρίσκεται σε μια συγκεκριμένη κατάσταση. Τα Statechart διαγράμματα σε συνεργασία με τα Use Case διαγράμματα δείχνουν την συμπεριφορά κάποιων αντικειμένων, που χρησιμοποιούνται σε διάφορα use cases που έχουν δημιουργηθεί. Χρησιμοποιούνται για αντικείμενα των οποίων την συμπεριφορά είναι απαραίτητο να κατανοήσουμε μέσα σε ολόκληρο το σύστημα.

Τα Statechart διαγράμματα δεν χρησιμοποιούνται για την περιγραφή όλων των αντικειμένων του συστήματος, αλλά **μόνο για τα αντικείμενα, που θεωρούνται σημαντικά** και πρέπει να παρατηρείται η συμπεριφορά του και συνδυάζονται με άλλα διαγράμματα όπως τα Activity διαγράμματα για να δείξουν την ροή της εργασία σε κάποιο συγκεκριμένο σημείο της εφαρμογής και την συμπεριφορά των διαφόρων αντικειμένων που χρησιμοποιούνται.

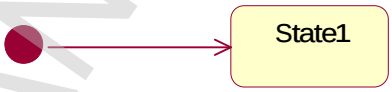
Τα βασικά στοιχεία των Statechart διαγραμμάτων παρουσιάζονται στη συνέχεια:

- **State (Κατάσταση)**

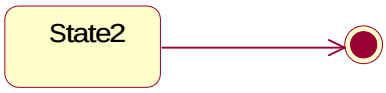
  Η κατάσταση (state) στην οποία επισέρχεται κάποιο αντικείμενο απεικονίζεται με ένα ορθογώνιο με στρογγυλές άκρες το οποίο περιέχει το όνομα της κατάστασης.



  Η αρχική κατάσταση του αντικειμένου απεικονίζεται με ένα μαύρο κύκλο, με ένα βέλος που δείχνει προς την αρχική κατάσταση.



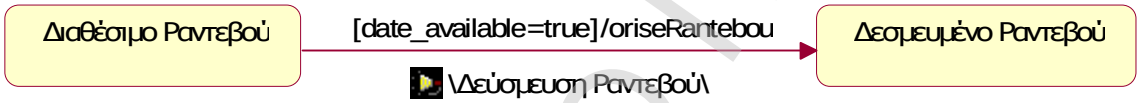
  Στο διάγραμμα παρουσιάζεται η τελική κατάσταση, στην οποία επισέρχεται το αντικείμενο και απεικονίζεται με ένα κύκλο που περιέχει ένα μικρότερο μαύρο κύκλο. Σε ένα Statechart διάγραμμα μπορούμε να έχουμε παραπάνω από μια τελικές καταστάσεις.



- **Events (Γεγονότα)**

  Event είναι το γεγονός, το οποίο προκαλεί την μετάβαση από την μια κατάσταση σε μια άλλη. Η μετάβαση (transition) από την μια κατάσταση στην άλλη απεικονίζεται με ένα βέλος το οποίο περιέχει το όνομα του event που προκαλεί την κατάσταση. Επίσης στο transition παρουσιάζεται όπου απαιτείται, μια συνθήκη ελέγχου (guard condition) η οποία πρέπει να είναι αληθής για να πραγματοποιηθεί η μετάβαση.

  Για παράδειγμα, για ένα αντικείμενο Ραντεβού αρχικά είναι διαθέσιμο να δοθεί σε ένα ασθενή. Όταν ζητηθεί ένα ραντεβού από ένα ασθενή και γίνει έλεγχος διαθέσιμης ημερομηνίας η κατάσταση του ραντεβού μετατρέπεται σε δεσμευμένο ραντεβού.



  Υπάρχουν 3 τύποι events:

  Call event: κλήση μιας συγκεκριμένης λειτουργίας από ένα αντικείμενο

  Change event: αλλαγή κατάστασης κάτω από μια συγκεκριμένη συνθήκη

  Time Event: εκτέλεση λειτουργίας μετά την έλευση συγκεκριμένου χρόνου

- **Actions (Λειτουργίες)**

  Action είναι η λειτουργία που καλείται όταν συμβεί ένα γεγονός (events) και είναι στην πραγματικότητα η λειτουργία που προκαλεί την μετάβαση στην επόμενη κατάσταση. Η αναπαράσταση του action απεικονίζεται με το όνομα του event, που προκαλεί την μετάβαση, το σύμβολο «/» και το όνομα του action, που εκτελείται για να πραγματοποιηθεί η μετάβαση.

  Στο προηγούμενο παράδειγμα, είχαμε ένα *call event*, την κλήση δηλαδή μιας λειτουργίας (action) και συγκεκριμένα της λειτουργίας *placeProduct*.

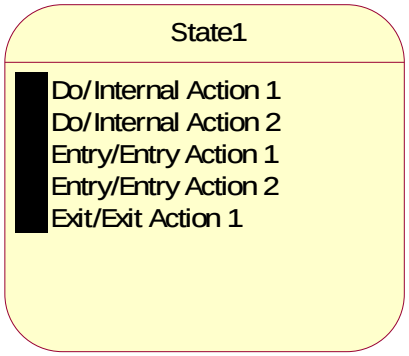
- **Internal Actions (Εσωτερικές Λειτουργίες)**

  Internal actions είναι οι εσωτερικές λειτουργίες οι οποίες μπορεί να εκτελούνται από ένα αντικείμενο όταν βρίσκονται σε κάποια συγκεκριμένη κατάσταση. Οι εσωτερικές λειτουργίες αναπαρίστανται μέσα στο ορθογώνιο της κατάστασης. Οι τύποι των εσωτερικών λειτουργιών ενός αντικειμένου είναι οι παρακάτω:

  Entry: εσωτερικές λειτουργίες που εκτελούνται όταν το αντικείμενο έρθει στην συγκεκριμένη κατάσταση. Αναπαρίσταται με τη λέξη *Entry*, το «/» και το όνομα της εσωτερικής λειτουργίας.

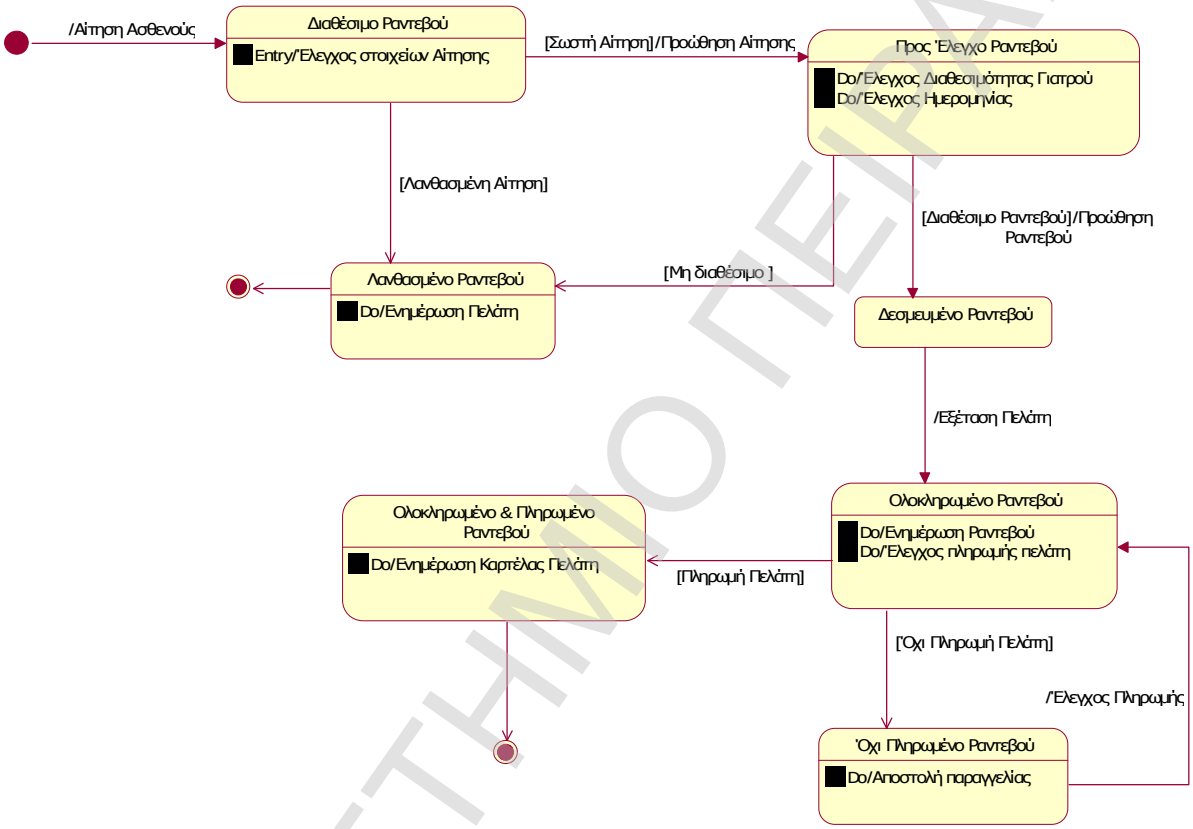
Do: εσωτερικές λειτουργίες που εκτελούνται μέσα σε μια κατάσταση. Αναπαρίστανται με τη λέξη *Do*, το «/» και το όνομα της εσωτερικής λειτουργίας, που θα εκτελεστεί μέσα στο αντικείμενο.

Exit: εσωτερικές λειτουργίες που θα εκτελεστούν όταν το αντικείμενο πάψει να έχει την συγκεκριμένη κατάσταση (πριν δηλαδή αλλάξει κατάσταση). Αναπαρίστανται με τη λέξη *Exit*, το «/» και το όνομα της εσωτερικής λειτουργίας.



Παράδειγμα Statechart διαγράμματος

Στη συνέχεια παρουσιάζεται ένα παράδειγμα Statechart διαγράμματος για την περίπτωση της εφαρμογής διαχείρισης ραντεβού ασθενών. Στο παράδειγμα παρουσιάζονται οι καταστάσεις του αντικειμένου Ραντεβού και τα διάφορα γεγονότα που αλλάζουν τις καταστάσεις του.



Διάγραμμα 18: Παράδειγμα Statechart διαγράμματος

Component Diagrams

Το Component διάγραμμα χρησιμοποιείται για να περιγράψει την φυσική υλοποίηση τμημάτων λογισμικού του συστήματος. Όπως τα Class διαγράμματα χρησιμοποιούνται για να περιγράψουν τον λογικό σχεδιασμό της εφαρμογής (logical design) με την παρουσίαση των κλάσεων που είναι η εννοιολογική παρουσίαση των αντικειμένων που αποτελούν την εφαρμογή έτσι και τα Component διαγράμματα χρησιμοποιούνται για να περιγράψουν την **φυσική υλοποίηση** (physical implementation) της εφαρμογής παρουσιάζοντας τα τμήματα πηγαίου κώδικα (source code) και τα εκτελέσιμα αρχεία (executable files) που χρησιμοποιούνται στην εφαρμογή.

Σκοπός του Component διαγράμματος είναι η παρουσίαση των τμημάτων του κώδικα της εφαρμογής (τμήματα λογισμικού – software components) και των συσχετίσεων που υπάρχουν μεταξύ τους. Τα components (συστατικά μέρη) που μπορεί να υπάρξουν σε ένα σύστημα του ομαδοποιούνται σε τρεις ομάδες:

Deployment components: συστατικά μέρη τα οποία χρησιμεύουν για να τρέξει η εφαρμογή.

Work product components: στοιχεία που περιλαμβάνουν μοντέλα, πηγαίο κώδικα και αρχεία δεδομένων.

Execution components: στοιχεία που δημιουργούνται κατά την εκτέλεση της εφαρμογής.

Για μια επιχείρηση τα κομμάτια του λογισμικού που αναφέρθηκαν παραπάνω μπορεί επίσης να αντιπροσωπεύουν επιχειρηματικές διαδικασίες και επιχειρηματικά έγγραφα. Τα συστατικά μέρη (components) της εφαρμογής μπορεί να δημιουργηθούν από τις κλάσεις του συστήματος, που έχουν ήδη οριστεί στα Class διαγράμματα, είτε να σχεδιαστούν από την αρχή από τους προγραμματιστές της εφαρμογής είτε να παραληφθούν από κάποιον third party vendor λογισμικού (για παράδειγμα κάποιο προϊόν λογισμικού τρίτης εταιρείας που έχει αγοραστεί από τον οργανισμό).

Τα βασικά στοιχεία (elements) των Component διαγραμμάτων είναι τα παρακάτω:

- **Component**

  Το component είναι ένα δομικό στοιχείο της εφαρμογής μας. Γραφικά αναπαρίσταται με ένα ορθογώνιο με δυο μικρά ορθογώνια στην αριστερή πλευρά του και μπορεί να είναι τμήμα κώδικα, κάποιο εκτελέσιμο αρχείο, ένα αρχείο βιβλιοθήκης εφαρμογών του οργανισμού (π.χ. ένα αρχείο .dll) ή οτιδήποτε άλλο ανήκει και αποτελεί τμήμα της εφαρμογής .

- **Component Stereotypes**

  Stereotypes είναι δηλώσεις που δείχνουν τον τύπο του component και δηλώνονται πάνω από το όνομα του. Τα διαθέσιμα stereotypes που μπορούν να χρησιμοποιηθούν σε ένα component είναι τα παρακάτω:

  <<*executable*>>: δήλωση εκτελέσιμου αρχείου.

  <<*library*>>: δήλωση κάποιας βιβλιοθήκης που χρησιμοποιείται από ένα εκτελέσιμο αρχείο κατά την εκτέλεση της εφαρμογής.

  <<*table*>>: πίνακας ή άλλο στοιχεία μιας Βάσης Δεδομένων.

  <<*file*>>: δήλωση κάποιου αρχείου δεδομένων ή κάποιου αρχείου με κώδικα.

  <<*document*>>: δήλωση κάποιου αρχείου (π.χ. μια web σελίδα ή κάποιο έγγραφο που χρησιμοποιείται στην εφαρμογή).

- **Component Dependencies**

  Dependencies είναι οι σχέσεις που υπάρχουν μεταξύ των διαφόρων components. Οι εξαρτήσεις μεταξύ των components δείχνουν πώς αυτά σχετίζονται μεταξύ τους και πως οι αλλαγές σε ένα component μπορεί να επηρεάσουν άλλα components του συστήματος. Οι σχέσεις αυτές μεταξύ των components αναπαρίστανται με μια διακεκομμένη γραμμή, με ένα ανοιχτό βέλος.

Deployment Diagrams

Τα Deployment διαγράμματα χρησιμοποιούνται για να αναπαραστήσουν την αρχιτεκτονική του hardware του συστήματος καθώς επίσης και τον προσδιορισμό των τμημάτων του συστήματος στα οποία γίνονται οι διάφορες εργασίες, όπως για παράδειγμα κάποιοι συγκεκριμένοι υπολογιστές, servers, εκτυπωτές, θέσεις αποθήκευσης κλπ. Σκοπός των διαγραμμάτων αυτών είναι να παρουσιαστεί η φυσική υλοποίηση του συστήματος με την παρουσίαση του hardware της εφαρμογής και συνήθως χρησιμοποιούνται σε συνεργασία με τα Component διαγράμματα για μια ολοκληρωμένη εικόνα του συστήματος. Παρουσιάζεται ο φυσικός σχεδιασμός (στοιχεία κώδικα) της εφαρμογής και τα μηχανήματα στα οποία τοποθετούνται τα στοιχεία αυτά.

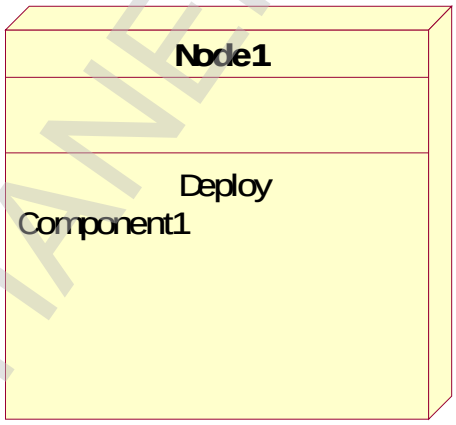
Τα βασικά στοιχεία των deployment diagrams παρουσιάζονται παρακάτω:

- **Nodes (Κόμβοι)**

  Κάθε κόμβος (node) είναι μια ανεξάρτητη μονάδα του συστήματος (υλικό ή μονάδα επεξεργασίας) στο οποίο γίνονται οι διάφορες εργασίες κατά την εκτέλεση της εφαρμογής. Οι κόμβοι περιλαμβάνουν συσκευές επεξεργασίας (ηλεκτρονικοί υπολογιστές, εκτυπωτές και άλλες μηχανές επεξεργασίας) αλλά επίσης και ανθρώπινους πόρους (human resources) ή μηχανικούς πόρους επεξεργασίας (mechanical process resources).

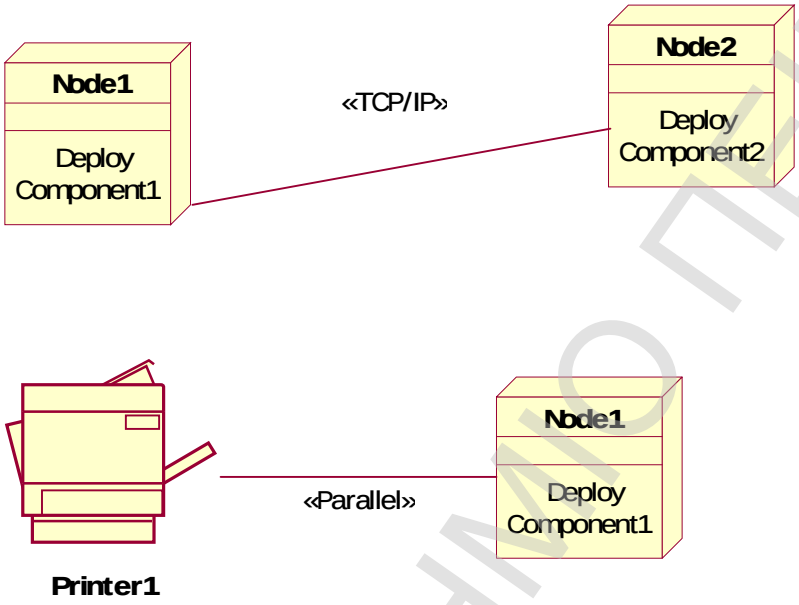
  Μέσα σε ένα κόμβο μπορούν να οριστούν και components (τμήματα κώδικα) κάποιου συγκεκριμένου υπολογιστή που χρησιμοποιούνται κατά την εκτέλεση του προγράμματος για να υπάρχει καλύτερη ομαδοποίηση τόσο του λογισμικού όσο και του υλικού της εφαρμογής.

  Το Deployment διάγραμμα υποδεικνύει το πώς τα components από τα οποία αποτελείται το σύστημα κατανέμονται στις φυσικές μονάδες του συστήματος. Τα Deployment διαγράμματα χρησιμοποιούνται σε συνεργασία με τα Component διαγράμματα για να αναπαραστήσουν τα δομικά στοιχεία της εφαρμογής (λογισμικό) μαζί με τα απαραίτητα μηχανήματα στα οποία βρίσκονται τα προϊόντα λογισμικού. Οι κόμβοι αναπαρίστανται με ένα τρισδιάστατο ορθογώνιο, το οποίο μπορεί να αποτελεί έναν υπολογιστή ή κάποιο άλλο hardware



- **Association**

  Η επικοινωνία (association) μεταξύ των κόμβων αναπαρίσταται με μια γραμμή από τον ένα κόμβο στον άλλο. Στην γραμμή επικοινωνίας μπορεί να χρησιμοποιηθεί και μια περιγραφή για να αναπαραστήσει τον τρόπο επικοινωνίας μεταξύ των υλικών του συστήματος. Για παράδειγμα μπορεί να χρησιμοποιηθεί η λέξη <<TCP/IP>> για να δείξει ότι η επικοινωνία γίνεται μέσω ενός τοπικού δικτύου (intranet) ή η λέξη <<parallel>> για να δείξει τον τρόπο επικοινωνίας ενός Η/Υ με έναν εκτυπωτή.



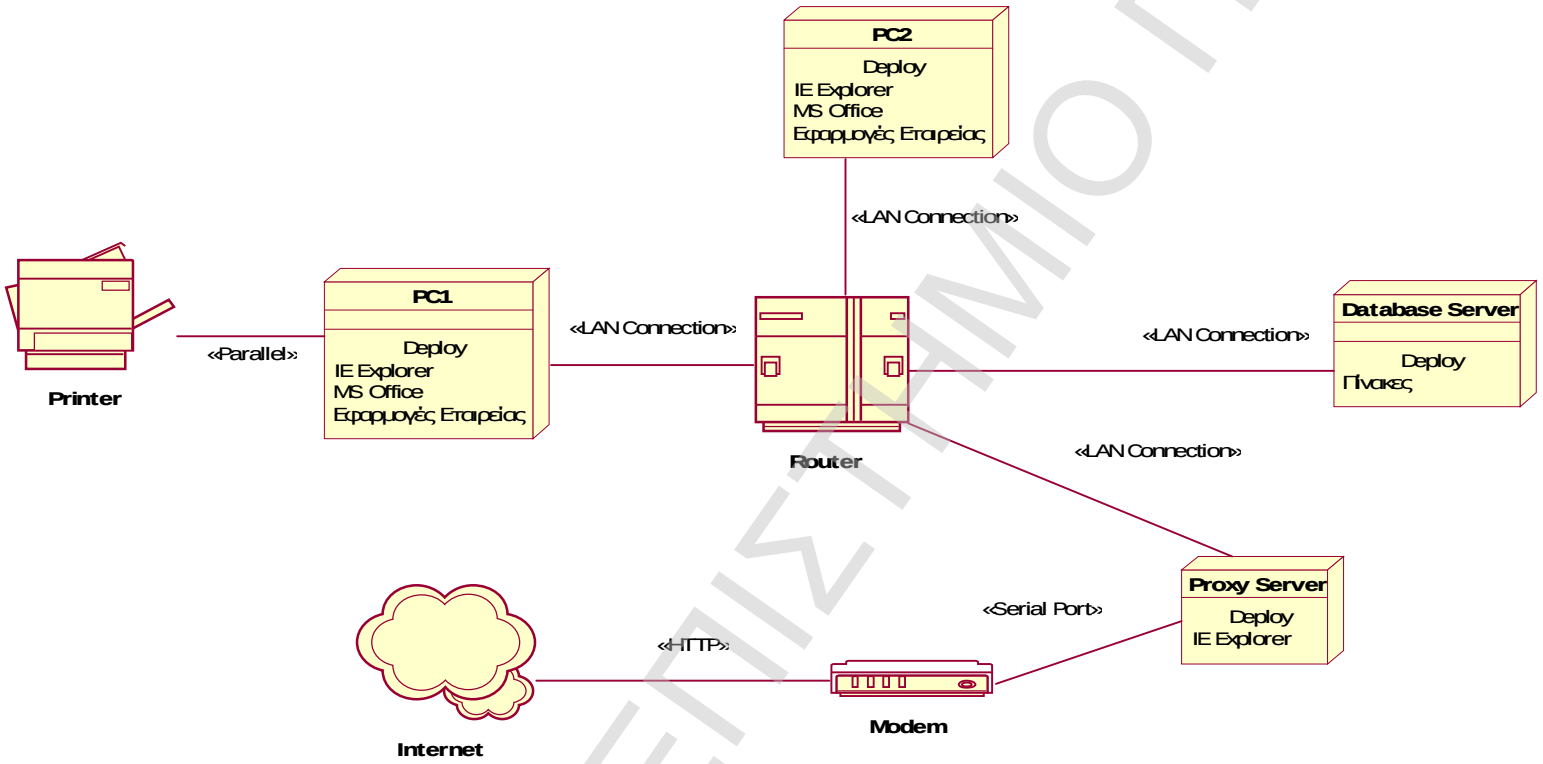
Χρήση των Deployment και των Component διαγραμμάτων

Ο συνδυασμός των δύο παραπάνω διαγραμμάτων των Deployment και των Component διαγραμμάτων, παρέχει μια υψηλού επιπέδου φυσική περιγραφή του ολοκληρωμένου συστήματος παρουσιάζοντας το πώς τα δομικά στοιχεία του συστήματος (components, nodes) κατανέμονται, επικοινωνούν και συνδυάζονται.

Μέσα από τα Component διαγράμματα μπορούμε να δούμε μια φυσική περιγραφή του λογισμικού (software) του συστήματος, μέσα από τα Deployment διαγράμματα μπορούμε να δούμε την διάταξη του hardware του συστήματος ενώ πολλές φορές γίνεται συνδυασμός και των δυο διαγραμμάτων για να έχουμε μια ολοκληρωμένη εικόνα του συστήματος.

Παράδειγμα Deployment διαγράμματος

Στη συνέχεια παρουσιάζεται ένα παράδειγμα χρήσης ενός Deployment διαγράμματος, όπου παρουσιάζεται η αρχιτεκτονική μιας μικρής εταιρείας. Συγκεκριμένα φαίνονται οι υπολογιστές, που περιέχουν τα components (εφαρμογές εταιρείας, MS Office, IE Explorer), τα άλλα μηχανήματα που χρησιμοποιούνται καθώς και οι συσχετίσεις μεταξύ του hardware.



Διάγραμμα 19: Παράδειγμα χρήσης Deployment διαγράμματος

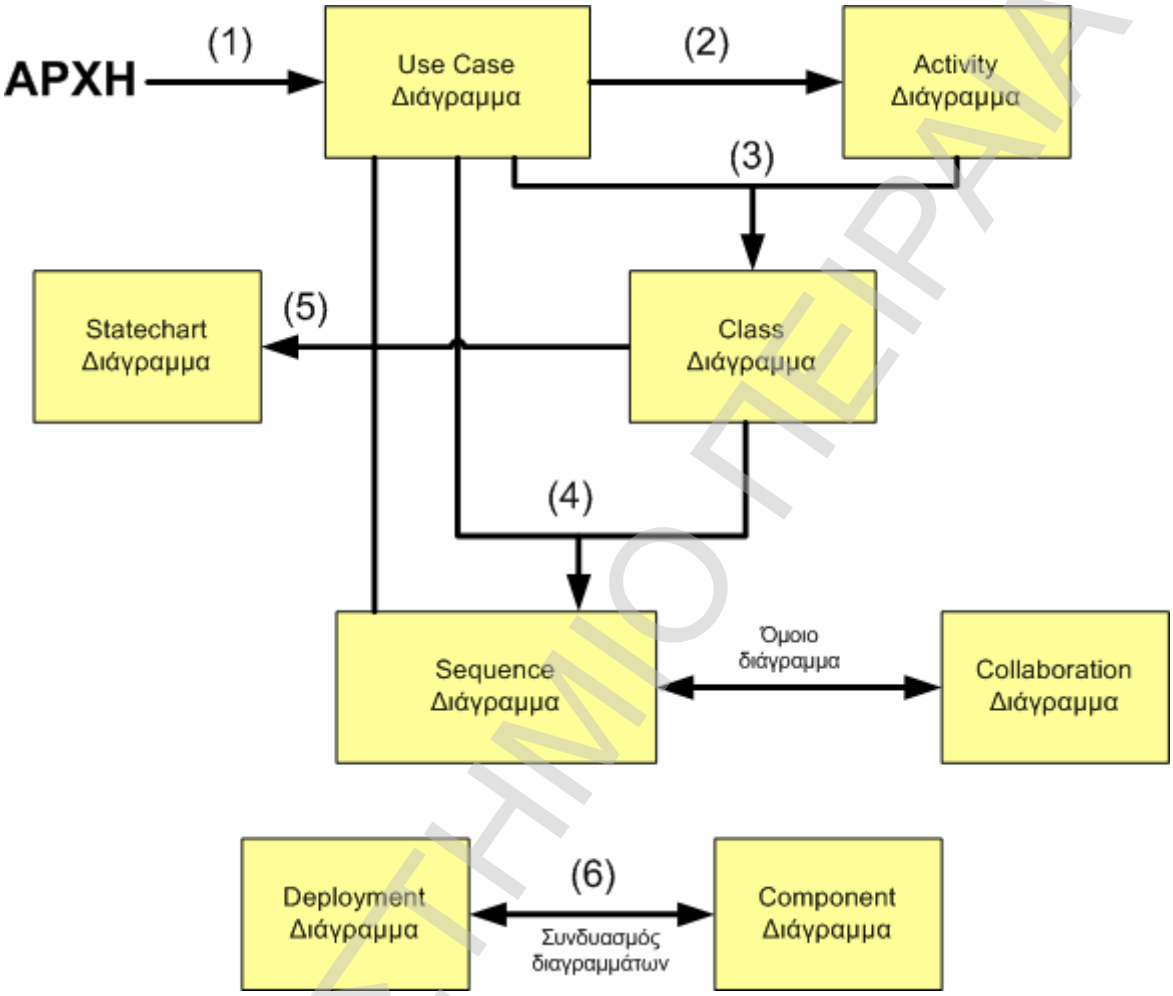
Χρήση των διαγραμμάτων της UML

Στον παρακάτω πίνακα, παρουσιάζονται τα διαγράμματα της UML, που παρουσιάσαμε καθώς και η χρήση τους κατά την διάρκεια της ανάπτυξης της εφαρμογής.

Διάγραμμα	**Χρήση**
Use Case	- Περιγραφή βασικών λειτουργιών του συστήματος - Απαιτήσεις χρηστών - Τεκμηρίωση συστήματος - Χρήση σεναρίων για έλεγχο των λειτουργιών
Activity	- Αναπαράσταση εκτέλεσης των λειτουργιών - Προβολή workflow σε διάφορα σημεία της εφαρμογής - Υλοποίηση των use case σεναρίων που έχουν δημιουργηθεί
Class	- Δημιουργούνται ύστερα από ανάλυση των Use Case (εύρεση αντικειμένων) και των Activity (εύρεση συσχετίσεων) διαγραμμάτων - Παρουσίαση της στατικής δομής της εφαρμογής - Παρουσίαση των αντικειμένων της εφαρμογής και συσχετίσεων μεταξύ τους - Αποτελεί βάση για τους προγραμματιστές της εφαρμογής να υλοποιήσουν τις κλάσεις της εφαρμογής
Sequence	- Προβολή της αλληλεπίδρασης των αντικειμένων στην διάρκεια του χρόνου - Προβολή ανταλλαγής μηνυμάτων μεταξύ των αντικειμένων (από πάνω προς τα κάτω) - Περιγράφουν συγκεκριμένες λειτουργίες του συστήματος (από τα Use Case διαγράμματα) - Τα αντικείμενα προέρχονται από τα Class διαγράμματα
Collaboration	- Ίδια χρήση με τα Sequence διαγράμματα - Η σειρά ανταλλαγής των μηνυμάτων αναπαρίσταται με αύξουσα αρίθμηση
Statechart	- Αναπαριστούν την διάρκεια ζωής των αντικειμένων - Παρουσιάζονται οι καταστάσεις (states) των βασικών αντικειμένων της εφαρμογής - Παρουσιάζονται τα γεγονότα (events) που αλλάζουν μια κατάσταση - Παρουσιάζονται οι εσωτερικές λειτουργίες ενός αντικειμένου σε μία κατάσταση - Τα αντικείμενα προέρχονται από τα Use Case, Sequence και Class διαγράμματα
Component	- Αναπαριστούν την φυσική υλοποίηση των τμημάτων λογισμικού της εφαρμογής - Παρουσιάζονται οι συσχετίσεις μεταξύ τους
Deployment	- Αναπαριστούν την φυσική υλοποίηση του hardware της εφαρμογής - Συνδυάζονται με τα Component διαγράμματα για μια ολοκληρωμένη εικόνα της εφαρμογής

Πίνακας 3: Συνοπτική παρουσίαση χρήσης των διαγραμμάτων της UML

Στο παρακάτω διάγραμμα παρουσιάζεται η σειρά με την οποία χρησιμοποιούνται τα διαγράμματα της UML.

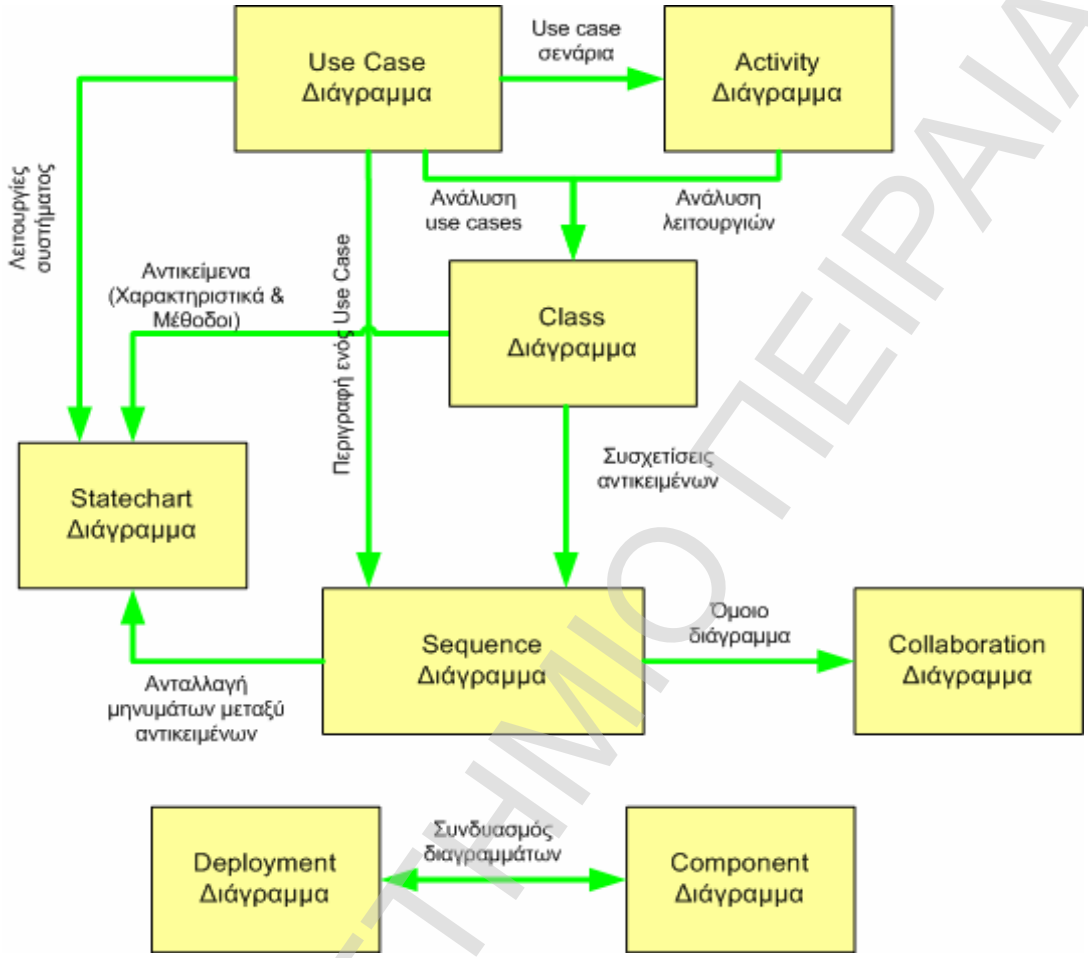


Διάγραμμα 20: Σειρά δημιουργίας των διαγραμμάτων της UML

Όπως παρατηρούμε από το διάγραμμα αρχικά δημιουργούνται τα Use Case διαγράμματα. Στη συνέχεια δημιουργούνται τα Activity διαγράμματα, τα οποία με την μαζί με τα Use Case βοηθούν στην δημιουργία των Class διαγραμμάτων. Συνεχίζοντας, τα Use Case μαζί με τα Class διαγράμματα βοηθούν στην δημιουργία των Sequence διαγραμμάτων, ενώ τα Class μαζί με τα Use Case και τα Sequence διαγράμματα βοηθούν στην δημιουργία των Statechart διαγραμμάτων. Να τονίσουμε ότι τα Collaboration διαγράμματα είναι όμοια με τα Sequence, επομένως μπορούμε να πούμε ότι δημιουργούνται με τον ίδιο τρόπο.

Τέλος, τα Deployment και τα Component διαγράμματα είναι αυτόνομα κατά κάποιο τρόπο με την έννοια ότι δεν εξαρτώνται από τα υπόλοιπα διαγράμματα αλλά συνδυάζονται μόνο μεταξύ τους.

Το παρακάτω διάγραμμα παρουσιάζει τον τρόπο με τον οποίο τα διαγράμματα της UML σχετίζονται μεταξύ τους.



Διάγραμμα 21: Σχέσεις μεταξύ των διαγραμμάτων της UML

Όπως βλέπουμε από το διάγραμμα, τα Use Case διαγράμματα σχετίζονται με τα Activity διαγράμματα μέσω των use case σεναρίων, που έχουν κατασκευαστεί δηλαδή δημιουργείται ένα Activity διάγραμμα για να παρουσιάσει την υλοποίηση ενός σεναρίου. Μέσα από την ανάλυση των use cases (Use Case διαγράμματα) και των λειτουργιών (Activity διαγράμματα) υποστηρίζεται η δημιουργία των Class διαγραμμάτων. Μέσα από την περιγραφή των Use Case και τις συσχετίσεις μεταξύ των αντικειμένων των Class διαγραμμάτων δημιουργούνται τα Sequence διαγράμματα. Τα Collaboration διαγράμματα είναι όμοια με τα Sequence επομένως δημιουργούνται με το ίδιο ακριβώς τρόπο.

Στη συνέχεια, τα Use Case διαγράμματα που παρουσιάζουν τις λειτουργίες του συστήματος, τα Class διαγράμματα που παρουσιάζουν τα χαρακτηριστικά και τις μεθόδους των αντικειμένων και τα Sequence διαγράμματα που δείχνουν την ανταλλαγή μηνυμάτων μεταξύ των αντικειμένων βοηθούν στην δημιουργία των Statechart διαγραμμάτων.

Τέλος, τα Deployment και τα Component διαγράμματα δημιουργούνται ανεξάρτητα από τα υπόλοιπα διαγράμματα και συνδυάζονται μεταξύ τους.

Στόχοι και πλεονεκτήματα της UML

Η UML έχει σχεδιαστεί με σκοπό να εκπληρώσει συγκεκριμένους στόχους έτσι ώστε να μπορέσει πραγματικά να γίνει ένα πρότυπο και να εξυπηρετήσει τις πρακτικές ανάγκες στην διαδικασία ανάπτυξης εφαρμογών. Οι **στόχοι** που προσπαθεί να εξυπηρετήσει η UML, όπως τους έχουν θέσει οι δημιουργοί της, είναι παρακάτω:

- Παροχή στους σχεδιαστές μία έτοιμη-προς-χρήση, εκφραστική και οπτική γλώσσα μοντελοποίησης για την παραγωγή και ανταλλαγή μοντέλων εφαρμογών.
- Υποστήριξη προδιαγραφών που είναι ανεξάρτητες από συγκεκριμένες γλώσσες προγραμματισμού και διαδικασίες ανάπτυξης εφαρμογών.
- Παροχή μιας τυποποιημένης βάσης για την κατανόηση της γλώσσας μοντελοποίησης.
- Ενθάρρυνση της αγοράς εργαλείων μοντελοποίησης. Λόγω του ότι η UML αποτελεί ένα καθιερωμένο πρότυπο ενθαρρύνεται η ανάπτυξη εργαλείων μοντελοποίησης που υποστηρίζουν την UML και παράλληλα περιέχουν επιπλέον δυνατότητες, όπως ολοκλήρωση με κώδικα, έλεγχος σύνταξης, διαχείριση βάσεων δεδομένων κλπ.
- Υποστήριξη εννοιών υψηλότερου επιπέδου όπως τα συστατικά στοιχεία των εφαρμογών (components), συνεργασίες μεταξύ συστατικών στοιχείων (collaborations) κλπ.

Η UML, μέσα από την εκπλήρωση των στόχων της προσφέρει πολλά **πλεονεκτήματα** στην διάρκεια ανάπτυξης εφαρμογών:

- Παρέχει μια εύκολη και κατανοητή γλώσσα για οπτική μοντελοποίηση και ανάπτυξη του συστήματος.
- Παρέχει την δυνατότητα ενοποίησης των υπαρχόντων μεθοδολογιών ανάπτυξης αντικειμενοστραφών εφαρμογών και χρήση των πλεονεκτημάτων τους.
- Ενσωματώνει τις στρατηγικές, τους κανόνες και τις πρακτικές της επιχείρησης στο σύστημα.
- Επικεντρώνεται σε σύγχρονα θέματα ανάπτυξης εφαρμογών όπως η κλιμάκωση (scale), η κατανομή των συστημάτων (distribution), οι ταυτόχρονες διαδικασίες (concurrency) κα.
- Υποστηρίζει προδιαγραφές που είναι ανεξάρτητες από κάποια συγκεκριμένη γλώσσα προγραμματισμού ή διαδικασία ανάπτυξης.

Μέρος Β. Επίλυση Προβλήματος

Κεφάλαιο 5. Παρουσίαση Προβλήματος

Στο πρώτο μέρος της εργασίας παρουσιάσαμε το θεωρητικό υπόβαθρο πάνω στο οποίο θα στηριχθούμε για να παρουσιάσουμε στη συνέχεια το πρόβλημα της εταιρείας ΒΙΟΣΕΡ. Συγκεκριμένα παρουσιάσαμε θέματα σχετικά με τον Αντικειμενοστραφή Προγραμματισμό, την Αρχιτεκτονική των εφαρμογών καθώς και μια αναλυτική παρουσίαση της UML και την χρησιμότητα της στην διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών.

Αφού μελετήσαμε την θεωρία γύρω από την διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών και της αρχιτεκτονικής εφαρμογών επόμενο βήμα είναι να εφαρμόσουμε τα παραπάνω θέματα σε ένα υπαρκτό πρόβλημα που αντιμετωπίζει η εταιρεία ΒΙΟΣΕΡ. Για το λόγο αυτό πραγματοποιήθηκε μια σειρά συναντήσεων με τους υπεύθυνους της εταιρείας, συγκεκριμένα με τον Γενικό Διευθυντή και τον διευθυντή Πληροφορικής της εταιρείας για να εντοπιστούν τα προβλήματα που αντιμετωπίζει ο οργανισμός σχετικά με τα παραπάνω θέματα. Μέσα από τις συναντήσεις και τις συζητήσεις έγινε προσπάθεια καταγραφής του προβλήματος της ΒΙΟΣΕΡ.

Το πρόβλημα της εταιρείας ΒΙΟΣΕΡ εστιάζεται σε δύο ζητήματα. Πρώτον, στην επίλυση του προβλήματος του εσωτερικού ελέγχου του οργανισμού. Προκειμένου να παρουσιάσουμε την έννοια της Διαδικασίας Ελέγχου Διοίκησης θα αναφέρουμε την θεωρία σχετικά με τα Συστήματα Ελέγχου Διοίκησης καθώς και μια αναφορά στην μεθοδολογία Balanced Scorecard.

Για τις ανάγκες της ΒΙΟΣΕΡ πρέπει να εφαρμοστεί η θεωρία των συστημάτων ελέγχου διοίκησης για τον έλεγχο του οργανισμού, όσων αφορά την υλοποίηση της στρατηγικής του οργανισμού. Ο έλεγχος αυτός ξεκινάει από την δήλωση της στρατηγικής που θέλει να υλοποιήσει η ΒΙΟΣΕΡ, οι στόχοι δηλαδή που θέλει να επιτύχει ο οργανισμός και να βρεθεί ένας τρόπος υλοποίησης των στόχων αυτών. Ένα καλό εργαλείο που θα μπορούσαμε να χρησιμοποιήσουμε είναι η Balanced Scorecard μέσω του οποίου θα μπορέσουμε να λύσουμε το πρόβλημα του Ελέγχου Διοίκησης του οργανισμού. Η Balanced Scorecard παρέχει μια συλλογή από 4 προοπτικές, τα Χρηματοοικονομικά, του Πελάτες, τις Εσωτερικές Λειτουργίες και την Μάθηση και Ανάπτυξη του οργανισμού και με την χρήση των προοπτικών αυτών η Balanced Scorecard λειτουργεί σαν ένα στρατηγικό εργαλείο ελέγχου και διοίκησης.

Ο έλεγχος σχετίζεται με ένα μηχανισμό αξιολόγησης της απόδοσης οργανισμού και με τον τρόπο σύμφωνα με τον οποίο θα γίνονται οι απαραίτητες αλλαγές προκειμένου να επιτευχθούν οι στρατηγικοί στόχοι του οργανισμού. Το πρόβλημα της ΒΙΟΣΕΡ σχετίζεται επίσης και με την παρουσίαση ενός συνόλου εφαρμογών οι οποίες θα επιλύουν το πρόβλημα του ελέγχου.

Δεύτερον, αφού επιλυθεί το πρώτο θέμα και προκύψει ένα μοντέλο επίλυσης του προβλήματος του ελέγχου διοίκησης του οργανισμού θα πρέπει να υλοποιηθεί το μοντέλο αυτό (ή μέρος αυτού) με την βοήθεια μιας αντικειμενοστραφούς μεθοδολογίας ανάπτυξης εφαρμογών. Πρέπει να κατασκευαστεί μια εφαρμογή που θα υλοποιεί το πρώτο ζήτημα της ανάπτυξης μιας διαδικασίας ελέγχου διοίκησης της εταιρείας ΒΙΟΣΕΡ.

Επιπρόσθετα είναι ανάγκη να καθοριστεί μια μεθοδολογία ανάλυσης και κατασκευής αντικειμενοστραφών εφαρμογών. Στόχος είναι να επωφεληθεί ο οργανισμός από τα πλεονεκτήματα που προσφέρουν οι αντικειμενοστραφείς εφαρμογές καθώς και η κατασκευή εφαρμογών στηριζόμενες στην N-tier Αρχιτεκτονική. Πέρα από την ανάπτυξη μιας εφαρμογής που θα επιλύει το πρόβλημα του ελέγχου διοίκησης του οργανισμού η

ΒΙΟΣΕΡ θα έχει στην διάθεσή της μια διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών για να μπορεί να την εφαρμόσει στην κατασκευή εφαρμογών, για την αντιμετώπιση μελλοντικών αναγκών του οργανισμού. Μια μεθοδολογία ανάπτυξης αντικειμενοστραφών εφαρμογών, θα μπορούσε να στηριχθεί πάνω στην UML, για την κατασκευή και τεκμηρίωση των τμημάτων της εφαρμογής.

Παρακάτω παρουσιάζονται συνοπτικά τα ζητήματα που θέλουμε να λύσουμε και τα θέματα, που σχετίζονται με κάθε ζήτημα.

Σύστημα Ελέγχου Διοίκησης	• Έλεγχος Διοίκησης οργανισμού • Εφαρμογή Balanced Scorecard • Αξιολόγηση Απόδοσης • Σύνολο εφαρμογών

ΤΟ ΠΡΟΒΛΗΜΑ

Μεθοδολογία ανάπτυξης αντικειμενοστραφών εφαρμογών	• Ανάπτυξη εφαρμογής για επίλυση πρώτου ζητήματος • Διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών • N-tier Αρχιτεκτονική • Χρήση UML

Πίνακας 4: Παρουσίαση Προβλήματος

Πριν αναφέρουμε την μέθοδο επίλυσης του προβλήματος θα πρέπει να αναφέρουμε την θεωρία αναφορικά με τα Συστήματα Ελέγχου Διοίκησης καθώς και την θεωρία σχετικά με την Balanced Scorecard. Τα θέματα αυτά παρουσιάζονται στο επόμενο κεφάλαιο.

Κεφάλαιο 6. Management Control Systems

Βασικές Έννοιες

Ο όρος Management Control – Έλεγχος Διοίκησης[6] αναφέρεται στην διαδικασία σύμφωνα με την οποία οι διευθυντές μιας επιχείρησης επηρεάζουν άλλα μέλη του οργανισμού έτσι ώστε να αναπτύξουν τις στρατηγικές του οργανισμού [1]. Έλεγχος Διοίκησης είναι όλες εκείνες οι διαδικασίες και οι λειτουργίες που χρησιμοποιούν οι διευθυντές ενός οργανισμού έτσι ώστε να πετύχουν την υλοποίηση των στόχων τους και ταυτόχρονα τους στόχους του οργανισμού. Σκοπός του Ελέγχου Διοίκησης είναι να λάβουν οι διευθυντές του οργανισμού τις κατάλληλες αποφάσεις για τον οργανισμό και να διαχειριστούν τους πόρους του οργανισμού αποδοτικά και αποτελεσματικά έτσι ώστε να επιτευχθούν οι στόχοι που έχουν τεθεί.

Προκειμένου να καταλάβουμε την λειτουργία ενός Συστήματος Ελέγχου Διοίκησης (Management Control System) πρέπει να αναλύσουμε τα συστατικά μέρη του συστήματος αυτού, δηλαδή πρέπει να αναλύσουμε πως δουλεύει ένα Σύστημα Ελέγχου και πως λειτουργεί η διαδικασία Ελέγχου Διοίκησης.

Ένα **Σύστημα Ελέγχου** λειτουργίας ενός οργανισμού αποτελείται συνήθως από τα εξής στοιχεία:

- Από ένα **μηχανισμό ανίχνευσης** (Detector) του περιβάλλοντος για την παροχή πληροφοριών σχετικά με το τι συμβαίνει στο περιβάλλον του υπό εξέταση οργανισμού.
- Από ένα **μηχανισμό σύγκριση**ς (Assessor) των πληροφοριών που προέρχονται από τον προηγούμενο μηχανισμό ανίχνευσης με κάποιες σταθερές, που χαρακτηρίζουν την ομαλή λειτουργία του οργανισμού.
- Από ένα **μηχανισμό αλλαγής** (Effector) της συμπεριφοράς του οργανισμού, εάν απαιτηθεί αυτό από τον προηγούμενο μηχανισμό σύγκρισης έτσι ώστε να εξασφαλιστεί η ομαλή λειτουργία του οργανισμού.
- Από ένα **δίκτυο εσωτερικής επικοινωνίας** (Communications Network) για την μετάδοση πληροφοριών ανάμεσα στον μηχανισμό ανίχνευσης και στον μηχανισμό σύγκρισης και ανάμεσα στον μηχανισμό σύγκρισης και στον μηχανισμό αλλαγής.

Η διαδικασία **Ελέγχου Διοίκησης** είναι η διαδικασία κατά την οποία σε όλα τα επίπεδα διοίκησης ενός οργανισμού οι διευθυντές επιβεβαιώνουν ότι τα άτομα που επιβλέπουν υλοποιούν τις προκαθορισμένες στρατηγικές του οργανισμού. Είναι ένα σύστημα ελέγχου της υλοποίησης της προκαθορισμένης στρατηγικής ενός οργανισμού.

Σύμφωνα με το σύστημα ελέγχου, που παρουσιάστηκε παραπάνω, στην διαδικασία ελέγχου διοίκησης ισχύουν τα παρακάτω:

- Ο μηχανισμός ανίχνευσης αναφέρει τι συμβαίνει σε ολόκληρο τον οργανισμό.

[6] Στη συνέχεια της εργασίας θα αναφερόμαστε στον όρο Management Control με τον όρο Έλεγχος Διοίκησης. Στην περίπτωση που χρησιμοποιούνται ορισμοί από την ξένη βιβλιογραφία θα γίνεται μετάφραση των ορισμών αυτών, όπου είναι δυνατό και διευκολύνει τον αναγνώστη.

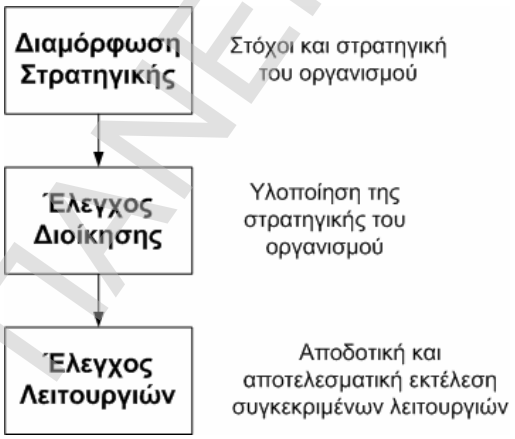
- Οι μηχανισμοί σύγκρισης συγκρίνουν τις πληροφορίες αυτές με την επιθυμητή κατάσταση του οργανισμού (στόχοι που έχουν τεθεί από την διοίκηση του οργανισμού).
- Οι μηχανισμοί αλλαγής αναλαμβάνουν διορθωτικές κινήσεις όταν υπάρχει σημαντική διαφορά ανάμεσα στην πραγματική κατάσταση του οργανισμού και στην επιθυμητή κατάσταση, που θα πρέπει να είναι ο οργανισμός.
- Με τα συστήματα εσωτερικής επικοινωνίας μεταφέρονται οι πληροφορίες ανάμεσα στους διάφορους μηχανισμούς του συστήματος ελέγχου διοίκησης.

Ένα Σύστημα Ελέγχου Διοίκησης είναι ένα σύστημα που χρησιμοποιεί συγκεκριμένους μηχανισμούς ελέγχου για να επιτευχθεί ο σκοπός του συστήματος που είναι η υλοποίηση της στρατηγικής του οργανισμού. Η διαδικασία Ελέγχου Διοίκησης δεν είναι μια αυτοματοποιημένη διαδικασία ή ένα αυτοματοποιημένο σύστημα με την έννοια ότι αποτελεί μια συγκεκριμένη εφαρμογή υποστήριξης της διοίκησης αλλά είναι περισσότερο ένα σύνολο δραστηριοτήτων. Για να επιτευχθεί ο τελικός σκοπός του Ελέγχου Διοίκησης απαιτείται συντονισμός πολλών ατόμων που υπάρχουν σε ένα οργανισμό (διευθυντών και εργαζομένων).

Χαρακτηριστικά του Ελέγχου Διοίκησης

Η διαδικασία του Ελέγχου Διοίκησης διαχωρίζεται από άλλες δύο διαδικασίες που εκτελούνται σε ένα οργανισμό. Την διαδικασία της Διαμόρφωσης Στρατηγικής (Strategy Formulation) και του Ελέγχου Λειτουργιών (Task Control). Η διαδικασία της **Διαμόρφωσης Στρατηγικής** αναφέρεται στον σχεδιασμό των στόχων που θα πρέπει να επιτύχει ο οργανισμός μακροπρόθεσμα και την στρατηγική που θα ακολουθήσει για την επίτευξη των στόχων αυτών. Η διαδικασία του **Ελέγχου Λειτουργιών** αναφέρεται στον έλεγχο συγκεκριμένων λειτουργιών που εκτελούνται καθημερινά σε ένα οργανισμό για την αποδοτική και αποτελεσματική εκτέλεση τους.

Η διαδικασία του Ελέγχου Διοίκησης σε ένα οργανισμό τοποθετείται ανάμεσα στις διαδικασίες της Διαμόρφωσης Στρατηγικής και του Ελέγχου Λειτουργιών, όπως φαίνεται στο παρακάτω διάγραμμα.

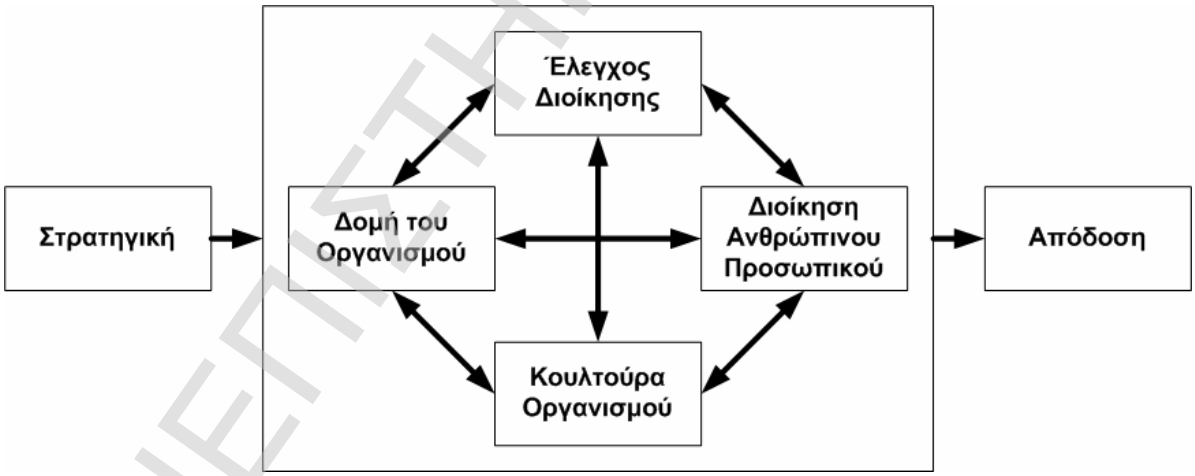


Διάγραμμα 22: Τοποθέτηση του Ελέγχου Διοίκησης

Ο Έλεγχος Διοίκησης ελέγχει την υλοποίηση της στρατηγικής που έχει διαμορφωθεί στην διαδικασία της Διαμόρφωσης Στρατηγικής. Επίσης η διαδικασία Ελέγχου Διοίκησης αναφέρεται στις γενικά καθορισμένες δραστηριότητες των διευθυντών, οι οποίοι πρέπει να αποφασίσουν και να κρίνουν το τι πρέπει να γίνει κάτω από τους γενικούς περιορισμούς που επιβάλλονται από την στρατηγική του οργανισμού. Αντίθετα, ο Έλεγχος Δραστηριοτήτων αναφέρεται σε συγκεκριμένες δραστηριότητες του οργανισμού πολλές από τις οποίες χρειάζονται λίγη κρίση από πλευράς των διευθυντών του οργανισμού (καθημερινές λειτουργίες ενός οργανισμού).

Η διαδικασία του Ελέγχου Διοίκησης εστιάζεται κυρίως στην εκτέλεση μιας καθορισμένης στρατηγικής. Βέβαια η διαδικασία του Ελέγχου Διοίκησης δεν είναι ο μόνος τρόπος για τον έλεγχο της υλοποίησης της στρατηγικής που έχει διαμορφωθεί σε ένα οργανισμό αλλά είναι ένα από τους διάφορους μηχανισμούς που έχει στην διάθεση του ένας οργανισμός για να υλοποιήσει την σχεδιασμένη στρατηγική. Οι μηχανισμοί που έχει στην διάθεσή του ένας οργανισμός για την υλοποίηση της στρατηγικής είναι οι παρακάτω:

- Έλεγχος Διοίκησης του Οργανισμού
- Δομή του Οργανισμού (ρόλοι στον οργανισμό, αναφορές που ανταλλάσσονται μεταξύ τμημάτων και διαχωρισμός των αρμοδιοτήτων στον οργανισμό)
- Κουλτούρα του Οργανισμού (κοινές αξίες, συμπεριφορά και κανόνες μέσα στον οργανισμό)
- Διοίκηση Ανθρώπινου Δυναμικού του Οργανισμού (επιλογή, εκπαίδευση και έλεγχος υπαλλήλων σε ένα οργανισμό)



Διάγραμμα 23: Μηχανισμοί υλοποίησης στρατηγικής

Ο Έλεγχος Διοίκησης είναι η διαδικασία σύμφωνα με την οποία οι διευθυντές επηρεάζουν τα άλλα μέλη ενός οργανισμού να αναπτύξουν την καθορισμένη στρατηγική του οργανισμού. Οι κύριες δραστηριότητες του Έλεγχου Διοίκησης είναι οι παρακάτω:

- **Σχεδιασμός** του τι πρέπει να κάνει ο οργανισμός.
- **Συντονισμός** των δραστηριοτήτων των διάφορων μερών ενός οργανισμού.
- **Επικοινωνία** διάδοση. δηλαδή της απαιτούμενης πληροφορίας.
- **Αποτίμηση** της πληροφορίας αυτής.

- **Λήψη απόφασης** για το ποιες ενέργειες πρέπει να γίνουν, εάν αυτό απαιτείται.
- **Επηρεασμός** διαφόρων ατόμων στον οργανισμό, να αλλάξουν την συμπεριφορά τους.

Στη συνέχεια της εργασίας θα χωρίσουμε σε δυο τμήματα την παρουσίαση του Ελέγχου Διοίκησης, όπως φαίνεται παρακάτω[1]:

- **Περιβάλλον του Ελέγχου Διοίκησης**
  - Στρατηγική Επιχειρήσεων
  - Χαρακτηριστικά της Συμπεριφοράς των Επιχειρήσεων
  - Κέντρα Ευθύνης (Responsibility Centers)
- **Διαδικασία Ελέγχου Διοίκησης**
  - Στρατηγικός Προγραμματισμός
  - Προϋπολογισμός (Budget)
  - Μέτρηση Απόδοσης
    - Balanced Scorecard

Περιβάλλον του Ελέγχου Διοίκησης

Στρατηγική Επιχειρήσεων

Ο Έλεγχος Διοίκησης είναι ένας μηχανισμός υλοποίησης της στρατηγικής που έχει καθοριστεί σε ένα οργανισμό. Προκειμένου όμως να καταλάβουμε τον ρόλο του Ελέγχου Διοίκησης πρέπει να καταλάβουμε τι είναι στρατηγική. Στην διεθνής βιβλιογραφία υπάρχουν πολλοί ορισμοί της έννοιας Στρατηγική.

Για παράδειγμα, στρατηγική είναι «*ο καθορισμός των βασικών μακροχρόνιων στόχων και σκοπών μιας επιχείρησης και η υιοθεσία μιας σειράς πράξεων και ο προσδιορισμός των αναγκαίων μέσων για την επίτευξη αυτών των στόχων*» [5].

Ένας άλλος ορισμός που έχει δοθεί είναι ότι «*στρατηγική είναι η κατεύθυνση και το εύρος των δραστηριοτήτων μιας επιχείρησης μακροπρόθεσμα, η οποία εξασφαλίζει ανταγωνιστικό πλεονέκτημα για την επιχείρηση μέσω της διάταξης των πόρων της μέσα σε ένα μεταβαλλόμενο περιβάλλον σε στόχο να ανταποκριθεί στις ανάγκες των αγορών και να ικανοποιήσει τις προσδοκίες των βασικών ομάδων ενδιαφερομένων (stakeholders)*» [8]

Η στρατηγική είναι ένα σχέδιο σύμφωνα με το οποίο ο οργανισμός θα επιτύχει τους στόχους, που έχει θέσει.

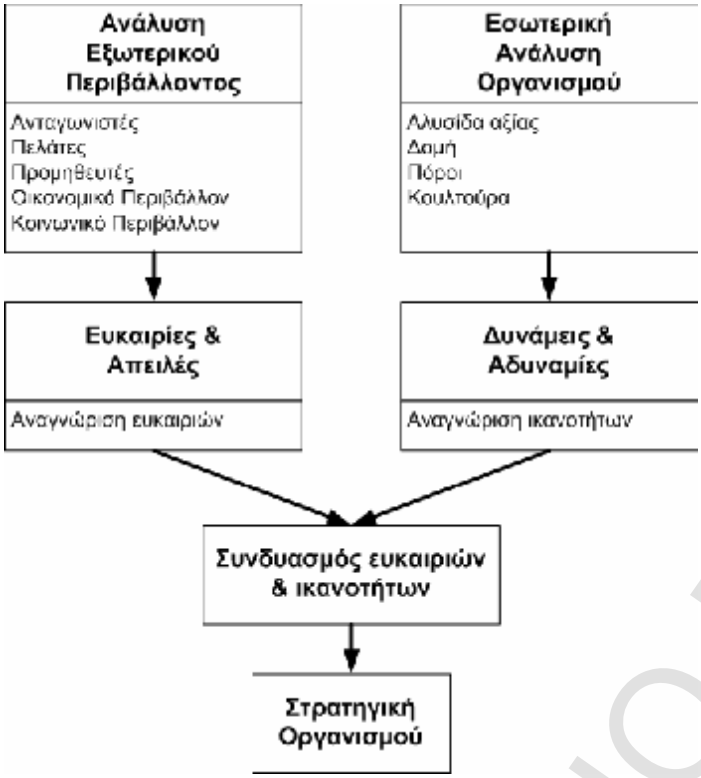
Σχεδόν σε κάθε οργανισμό οι στόχοι οι οποίοι τίθενται από τους διευθυντές του είναι:

- Η κερδοφορία του οργανισμού
- Η μεγιστοποίηση της αξίας και η ικανοποίηση των προσδοκιών των βασικών ομάδων ενδιαφερομένων (Stakeholders)
- Η μείωση του ρίσκου

Η στρατηγική περιγράφει την γενική κατεύθυνση στην οποία πρέπει να βαδίσει ο οργανισμός μακροπρόθεσμα για να επιτύχει τους στόχους που έχει θέσει. Ένας οργανισμός αναπτύσσει την στρατηγική του μέσα από την **ανάλυση του εξωτερικού περιβάλλοντος** του (ανάλυση του κλάδου που ανήκει ο οργανισμός και των εξωτερικών παραγόντων που επηρεάζουν τον οργανισμό όπως οι ανταγωνιστές, οι πελάτες, οι προμηθευτές καθώς και το ευρύτερο οικονομικό και κοινωνικό περιβάλλον που ανήκει ο οργανισμός) για την εύρεση των Ευκαιριών (Opportunities) και των Απειλών (Threats) από το περιβάλλον του.

Επίσης, ο οργανισμός προχωράει στην **εσωτερική ανάλυση του ίδιου του οργανισμού** (μελέτη της αλυσίδας αξίας, της δομής, των πόρων και της κουλτούρας του οργανισμού) προκειμένου να προσδιορισθούν οι Δυνάμεις (Strengths) και οι Αδυναμίες (Weaknesses) του ίδιου του οργανισμού.

Στη συνέχεια οι εξωτερικές ευκαιρίες από τον εξωτερική ανάλυση του περιβάλλοντος και οι εσωτερικές ικανότητες του οργανισμού από την εσωτερική ανάλυση του συνδυάζονται για να καθοριστεί η στρατηγική που θα ακολουθήσει ο οργανισμός.



Διάγραμμα 24: Διαμόρφωση στρατηγικής

Η στρατηγική κάθε επιχείρησης θα μπορούσε να μελετηθεί σε δύο επίπεδα:

- Στρατηγική σε **επιχειρηματικό επίπεδο (corporate level strategy)** που αναφέρεται στο σύνολο του οργανισμού και πραγματεύεται θέματα όπως η αποστολή (mission) και το όραμα (vision) της επιχείρησης, στους τομείς δραστηριοτήτων στους οποίους θα επιθυμούσε να δραστηριοποιηθεί η επιχείρηση, στην κατανομή των πόρων κλπ.
- Στρατηγική κάθε μιας από της **στρατηγικές επιχειρηματικές μονάδες της επιχείρησης** (**business level strategy**) που είναι η στρατηγική που ακολουθεί κάθε μια από τις στρατηγικές επιχειρηματικές μονάδες (Strategic Business Units – SBUs) στις οποίες είναι οργανωμένη η επιχείρηση.

Χαρακτηριστικά της Συμπεριφοράς των Επιχειρήσεων

Ένα καλά σχεδιασμένο Σύστημα Ελέγχου Διοίκησης επηρεάζει την συμπεριφορά των ανθρώπων σε ένα οργανισμό με την έννοια της **συνταύτισης στόχων**, την εξασφάλιση δηλαδή ότι οι ενέργειες που θα γίνουν μεμονωμένα από τους εργαζόμενους σε ένα οργανισμό θα βοηθήσουν τον οργανισμό να εκπληρώσει και τους δικούς του στόχους. Η συνταύτιση στόχων σε ένα οργανισμό **επηρεάζεται** από διάφορους **παράγοντες**, επίσημους και ανεπίσημους.

Στην συνέχεια παρουσιάζονται η έννοια της συνταύτισης των στόχων σε ένα οργανισμό καθώς και οι παράγοντες που επηρεάζουν την συνταύτιση αυτή μεταξύ του οργανισμού και των εργαζόμενων σε αυτόν.

Συνταύτιση Στόχων (Goal Congruence)

Ένα βασικό χαρακτηριστικό των διευθυντικών στελεχών των οργανισμών είναι η θέσπιση στόχων για τον οργανισμό και η προσπάθεια επίτευξης τους. Από την άλλη πλευρά όμως κάθε εργαζόμενος σε ένα οργανισμό έχει και δικούς του προσωπικούς στόχους (για παράδειγμα αύξηση μισθού, προώθηση σε ανώτερες θέσεις στον οργανισμό κλπ.) που δεν είναι απαραίτητο να συμβαδίζουν με τους στόχους του οργανισμού στον οποίο ανήκουν. Το κεντρικό σημείο ενός Συστήματος Ελέγχου Διοίκησης είναι η εξασφάλιση της **συνταύτισης στόχων** μεταξύ του οργανισμού και των εμπλεκομένων σε αυτόν (διεύθυνση, εργαζόμενοι κλπ.). Αυτό σημαίνει ότι οι ενέργειες που κάνουν τα άτομα για να επιτύχουν τους προσωπικούς τους στόχους συμφωνούν με τις ενέργειες που γίνονται από πλευράς του οργανισμού για την επίτευξη των στόχων του οργανισμού.

Βέβαια απόλυτη συνταύτιση στόχων μεταξύ ατόμων και οργανισμού είναι πολύ δύσκολο να επιτευχθεί ή πολλές φορές είναι αδύνατο, δεδομένου ότι οι στόχοι των ατόμων προσβλέπουν στο προσωπικό τους συμφέρον και δεν είναι απαραίτητο να συμβαδίζουν με τους στόχους του οργανισμού. Για παράδειγμα κάθε εργαζόμενος επιθυμεί αύξηση μισθού αλλά αυτό πολλές φορές αντίκειται στον στόχο του οργανισμού για μείωση του κόστους. Ωστόσο, ένα ικανοποιητικό Σύστημα Ελέγχου τουλάχιστον δεν θα ενθαρρύνει τα άτομα να ενεργήσουν ενάντια στους στόχους του οργανισμού.

Έτσι μέσα από το Σύστημα Ελέγχου Διοίκησης οι διευθυντές ενός οργανισμού μπορούν να ενθαρρύνουν τους εργαζόμενους να σκεφτούν τους στόχους του οργανισμού, στον οποίο ανήκουν και να προσπαθήσουν να οδηγήσουν στην συνταύτιση των προσωπικών στόχων τους με αυτών του οργανισμού.

Παράγοντες που επηρεάζουν την συνταύτιση στόχων

Σε κάθε οργανισμό υπάρχουν πολλοί παράγοντες, οι οποίο επηρεάζουν την συνταύτιση των στόχων ανάμεσα στον οργανισμό και των εμπλεκομένων σε αυτόν. Οι παράγοντες αυτοί επηρεάζουν άμεσα ή έμμεσα τους εργαζόμενους στον οργανισμό έτσι ώστε να τους ενθαρρύνουν να εκπληρώσουν τους στόχους του οργανισμού μέσα από την εκπλήρωση των δικών τους προσωπικών στόχων.

Οι παράγοντες επιρροής των εργαζόμενων μπορεί να είναι **ανεπίσημοι** (**informal**) με την έννοια ότι δεν είναι κάπου καταγεγραμμένοι αλλά ακολουθούνται σε κάθε οργανισμό είτε εμπειρικά είτε αποτελούν μέρος της κουλτούρας του οργανισμού και περνάνε από

εργαζόμενο σε εργαζόμενο. Υπάρχουν όμως και παράγοντες οι οποίο είναι **επίσημοι** (**formal**) με την έννοια ότι είναι καταγεγραμμένοι, αποτελούν τμήμα κάποιου συστήματος ελέγχου και είναι γνωστοί σε όλους τους εργαζόμενους. Οι παράγοντες επιρροής των εργαζομένων παρουσιάζονται στη συνέχεια:

Ανεπίσημοι Παράγοντες

Εξωτερικοί παράγοντες: οι παράγοντες αυτοί υφίστανται στο κοινωνικό περιβάλλον, στο οποίο δραστηριοποιείται ο οργανισμός. Αυτοί οι παράγοντες αναφέρονται συχνά και σαν εργασιακή ηθική (work ethic). Οι εξωτερικοί παράγοντες επηρεάζονται από την τοπική κοινωνία στην οποία δραστηριοποιείται ο οργανισμός (για παράδειγμα μια μικρή πόλη), από τον βιομηχανικό κλάδο στον οποίο ανήκει ο οργανισμός, επηρεάζονται, τέλος, από την χώρα στην οποία ανήκει ο οργανισμός (αυτό είναι παράδειγμα πολυεθνικών επιχειρήσεων, που οι εξωτερικοί παράγοντες επιρροής των εργαζόμενων διαφέρουν από χώρα σε χώρα).

Εσωτερικοί παράγοντες: οι παράγοντες αυτοί υπάρχουν μέσα στον ίδιο τον οργανισμό και ακολουθούνται συνήθως από όλους τους εργαζομένους. Τέτοιοι παράγοντες είναι:

- *Η κουλτούρα της επιχείρησης*: κοινές αξίες, κοινά πιστεύω μέσα στον οργανισμό και άτυποι κανόνες συμπεριφοράς που είναι γνωστοί σε όλους τους εργαζόμενους και τηρούνται από αυτούς.
- *Ο τρόπος διοίκησης των διευθυντών*: έχει την ισχυρότερη επιρροή μέσα σε ένα οργανισμό και εξαρτάται από τις ικανότητες και τα χαρίσματα κάθε διευθυντή σε ένα οργανισμό.
- *Η δομή του οργανισμού*: ο τρόπος που είναι καθορισμένες οι αρμοδιότητες και οι ευθύνες κάθε διευθυντικού στελέχους. Είναι πολύ σημαντικό να είναι καθορισμένος ο τρόπος με τον οποίο θα ενημερώνει ένας διευθυντής του υπολοίπους (προϊστάμενους και υφιστάμενους) στον οργανισμό.
- *Ο τρόπος επικοινωνίας μέσα στην επιχείρηση*: οι πληροφορίες που θα δέχονται οι διευθυντές στον οργανισμό προκειμένου να δουλέψουν για τους στόχους του οργανισμού.

Επίσημοι Παράγοντες

Οι επίσημοι παράγοντες επηρεάζουν την συνταύτιση των στόχων μεταξύ των εργαζομένων και του οργανισμού σε μεγαλύτερο βαθμό. Οι επίσημοι κανόνες επιρροής παρουσιάζονται είτε με την μορφή καταγεγραμμένων εσωτερικών κανόνων μέσα στον ίδιο τον οργανισμό είτε με την μορφή κάποιας διαδικασίας ελέγχου, που στην προκειμένη περίπτωση είναι το Σύστημα Ελέγχου Διοίκησης.

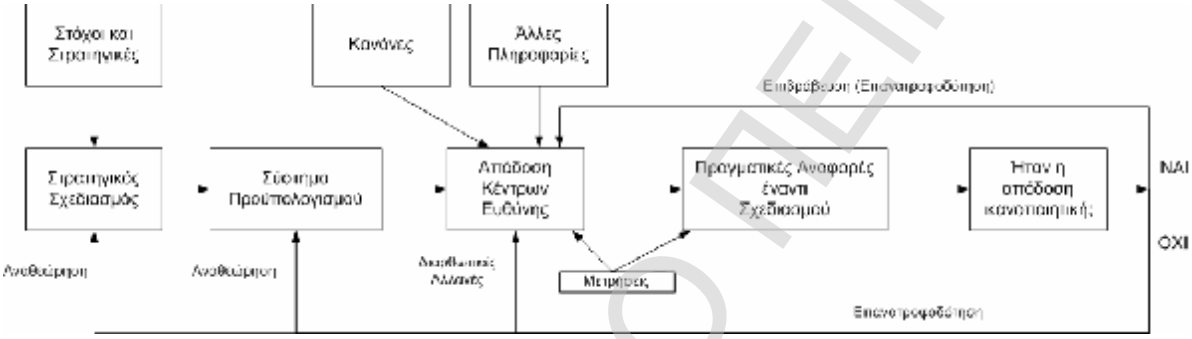
Κανόνες: οι κανόνες περιλαμβάνουν όλους τους τύπους επίσημων οδηγιών και ελέγχων που είναι καταγεγραμμένοι και γνωστοί σε όλους τους εργαζόμενους σε ένα οργανισμό. Οι κανόνες αυτοί περιλαμβάνουν συγκεκριμένες οδηγίες προς τα διάφορα τμήματα που οργανισμού, περιγραφές εργασίας κάθε εργαζομένου ή ομάδας εργαζομένων (job description) δηλαδή περιγράφουν την εργασία που κάνει κάθε υπάλληλος σε ένα οργανισμό.

Επίσης οι κανόνες περιλαμβάνουν συγκεκριμένες διαδικασίες λειτουργίας του οργανισμού (standard operating procedures), οποιασδήποτε μορφής κατευθυντήριες οδηγίες προς τους

εργαζομένους και κάθε μορφής άλλους μηχανισμούς ελέγχου που μπορεί να υπάρχουν σε ένα οργανισμό.

Διαδικασία Ελέγχου (Formal Control Process)

Η διαδικασία ελέγχου είναι το ίδιο το Σύστημα Ελέγχου Διοίκησης και αναφέρεται επίσης και σαν **Formal Control Process** δηλαδή σαν μια επίσημη διαδικασία ελέγχου. Το παρακάτω διάγραμμα παρουσιάζει συνοπτικά τα βήματα που ακολουθούνται σε ένα Σύστημα Ελέγχου Διοίκησης



Διάγραμμα 25: Formal Control Process[7]

Αρχικά δημιουργείται το **στρατηγικό πλάνο** για να υλοποιήσει τους στόχους και την στρατηγική που θα ακολουθήσει ο οργανισμός. Στη συνέχεια το στρατηγικό πλάνο μετατρέπεται σε ένα ετήσιο **προϋπολογισμό** ο οποίος εστιάζεται στα προβλεπόμενα έσοδα και έξοδα κάθε ξεχωριστού **Κέντρου Ευθύνης** (στα οποία έχει χωριστεί ο οργανισμός). Για να μετρηθεί η **απόδοση** κάθε κέντρου ευθύνης χρησιμοποιούνται συγκεκριμένοι κανόνες του οργανισμού καθώς επίσης και κάθε άλλη διαθέσιμη πληροφορία.

Στη συνέχεια οι πραγματικές αναφορές με τα αποτελέσματα της απόδοσης των κέντρων ευθύνης **συγκρίνονται** με τους στόχους που έχουν τεθεί μέσα από τον προϋπολογισμό για να προσδιοριστεί εάν η **απόδοση** των κέντρων ευθύνης ήταν **ικανοποιητική**. Στην περίπτωση που η απόδοση είναι ικανοποιητική υπάρχει **επανατροφοδότηση** προς τα κέντρα ευθύνης με την μορφή κάποιας **επιβράβευσης** για την επίτευξη των στόχων του οργανισμού. Στην περίπτωση που δεν είναι ικανοποιητική η απόδοση υπάρχει επανατροφοδότηση έτσι ώστε να **αναθεωρηθεί** είτε το αρχικό στρατηγικό πλάνο είτε οι στόχοι του προϋπολογισμού είτε να γίνουν διορθωτικές αλλαγές στα κέντρα ευθύνης του οργανισμού.

[7] Πηγή «*Management Control Systems*», Anthony, Govindarajan, 11th Edition

Κέντρα Ευθύνης (Responsibility Centers)

Σύμφωνα με την διαδικασία του Ελέγχου Διοίκησης ο υπό μελέτη οργανισμός για την καλύτερη παρακολούθηση του χωρίζεται σε συγκεκριμένες μονάδες που ονομάζονται Κέντρα Ευθύνης (Responsibility Centers). Ένα κέντρο ευθύνης είναι μια επιχειρησιακή μονάδα (organizational unit) η οποία διοικείται από ένα διευθυντή, ο οποίος είναι υπεύθυνος για τις λειτουργίες του κέντρου αυτού. Ο οργανισμός είναι ένα σύνολο από κέντρα ευθύνης κάθε ένα από τα οποία μπορεί να αναπαρασταθεί σαν ένα κουτί στο οργανόγραμμα του οργανισμού.

Ένα κέντρο ευθύνης υφίσταται στον οργανισμό για να επιτελέσει ένα ή περισσότερους σκοπούς, οι οποίοι αναφέρονται σαν στόχοι (objectives) του κέντρου αυτού. Οι στόχοι κάθε κέντρου ευθύνης σχετίζονται με τους στόχους του οργανισμού που έχει θέσει για να υλοποιήσει την στρατηγική του. Συγκεκριμένα, η επίτευξη των μεμονωμένων στόχων όλων των κέντρου ευθύνης του οργανισμού αντανακλά και την επίτευξη των στόχων του οργανισμού, αφού ο οργανισμός θεωρείται το σύνολο των κέντρων ευθύνης.

Για να εκτελέσει την λειτουργία του ένα κέντρο ευθύνης δέχεται σαν είσοδο συγκεκριμένα στοιχεία όπως υλικά, εργασία και υπηρεσίες τα οποία αποτιμώνται με τη μορφή κόστους. Χρησιμοποιώντας κεφάλαια κίνησης, εξοπλισμό και άλλους πόρους (ανάλογα με την φύση του) το κέντρο εκτελεί τις λειτουργίες του προκειμένου να μετατρέψει τα στοιχεία εισόδου (inputs) σε στοιχεία εξόδου (outputs) ή αποτελέσματα. Τα αποτελέσματα της λειτουργίας του κέντρου ευθύνης μπορεί να είναι συγκεκριμένα προϊόντα ή υπηρεσίες τα οποία θεωρούνται συγκεκριμένα αγαθά (για παράδειγμα το αποτέλεσμα του τμήματος παραγωγής του οργανισμού) ή μπορεί να είναι στοιχεία εξόδου προς άλλα κέντρα ευθύνης του οργανισμού.

Διάγραμμα 26: Λειτουργία Κέντρου Ευθύνης

Σχέση ανάμεσα Εισόδου και Εξόδου

Η διοίκηση ενός κέντρου ευθύνης είναι υπεύθυνη για την διασφάλιση της βέλτιστης σχέσης ανάμεσα στα στοιχεία εισόδου και στα στοιχεία εξόδου. Σε μερικά κέντρα ευθύνης η σχέση αυτή είναι ευθύς, για παράδειγμα η σχέση ανάμεσα στα στοιχεία εισόδου και εξόδου σε ένα κέντρο παραγωγής, όπου το πλήθος των πόρων που χρησιμοποιούνται για την παραγωγή επηρεάζουν τα παραγόμενα του κέντρου ευθύνης «Κέντρο Παραγωγής».

Υπάρχουν όμως και πολλές περιπτώσεις τα στοιχεία εισόδου δεν σχετίζονται άμεσα με τα στοιχεία εξόδου. Για παράδειγμα, το κόστος διαφήμισης (είσοδος) δεν σχετίζονται άμεσα με τα έσοδα των πωλήσεων (έξοδος) σε ένα τμήμα πωλήσεων, τα έξοδα Έρευνας και Ανάπτυξης (R&D – Research and Development) δεν σχετίζονται άμεσα με κάποιο παραγόμενο προϊόν αλλά η σχέση αυτή είναι μακροπρόθεσμη δηλαδή τα αποτελέσματα της έρευνας επηρεάζουν μακροπρόθεσμα ένα προϊόν κ.α.

Μέτρηση στοιχείων Εισόδου – Εξόδου

Το κόστος σε όρους χρήματος είναι ένας συνήθης τρόπος μέτρησης των πόρων που χρησιμοποιούνται σαν είσοδος σε ένα κέντρο ευθύνης. Έτσι, εάν ξέρουμε την αξία σε χρηματικούς όρους κάθε μονάδας των πόρων που χρησιμοποιούμε, πολλαπλασιάζουμε την αξία αυτή με την ποσότητα των χρησιμοποιούμενων πόρων για να υπολογίσουμε το τελικό κόστος των μονάδων εισόδου για ένα κέντρο ευθύνης.

Οι έννοιες της εισόδου, εξόδου και του κόστους μπορούν να χρησιμοποιηθούν για να εξηγήσουν τις έννοιες της αποδοτικότητας και της αποτελεσματικότητας για ένα κέντρο ευθύνης.

- **Αποδοτικότητα (efficiency)**: ο λόγος των στοιχείων εξόδου ως προς τα στοιχεία εισόδου ή αλλιώς η ποσότητα των στοιχείων εξόδου που δημιουργούνται από μια μονάδα εισόδου. Η έννοια της αποδοτικότητας χρησιμοποιείται για να απαντήσει στην ερώτηση: «*κάνουμε τα πράγματα σωστά;*»
- **Αποτελεσματικότητα (effectiveness)**: προσδιορίζεται με την σχέση ανάμεσα στα στοιχεία εξόδου ενός κέντρου ευθύνης και των στόχων του κέντρου, δηλαδή κατά πόσο τα στοιχεία εξόδου ενός κέντρου ευθύνης σχετίζονται με τους στόχους που έχει τεθεί για το κέντρο αυτό. Η έννοια της αποτελεσματικότητας χρησιμοποιείται για να απαντήσει στην ερώτηση: «*κάνουμε τα σωστά πράγματα;*»

Οι έννοιες της αποδοτικότητας και της αποτελεσματικότητας δεν είναι αμοιβαίως αποκλειόμενες έννοιες για ένα κέντρο ευθύνης αλλά πρέπει να συνδυάζονται μεταξύ τους. Με λίγα λόγια ένα κέντρο ευθύνης πρέπει να είναι ταυτόχρονα και αποδοτικό και αποτελεσματικό. Αυτό σημαίνει ότι ένα κέντρο ευθύνης πρέπει να πετυχαίνει τους στόχους του με τον βέλτιστο τρόπο. Οι βασικοί στόχοι που έχουν τεθεί για ένα κέντρο ευθύνης, σχετίζονται με την έννοια της μεγιστοποίησης του κέρδους. Επομένως, το κέρδος είναι ένα μέτρο της αποδοτικότητας και της αποτελεσματικότητας για ένα κέντρο ευθύνης.

Τύποι Κέντρων Ευθύνης

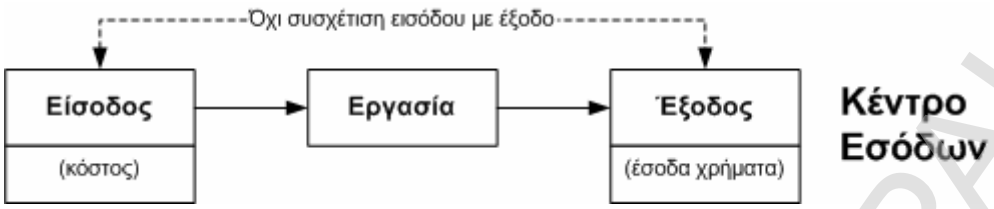
Σε ένα οργανισμό υπάρχουν τέσσερις τύποι κέντρων ευθύνης ανάλογα με τον τρόπο με τον οποίο μετράμε την είσοδο και την έξοδο σε χρηματικούς όρους για το κάθε κέντρο:

- **Κέντρα εσόδων** (Revenue centers)
- **Κέντρα εξόδων** (Expense centers)
- **Κέντρα κέρδους** (Profit centers)
- **Κέντρα επένδυσης** (Investment centers)

Κέντρα Εσόδων (Revenue Centers)

Κέντρο εσόδων θεωρείται το κέντρο ευθύνης εκείνο, το οποίο η έξοδος του (έσοδα) αποτιμώνται σε χρηματικούς όρους και δεν γίνεται προσπάθεια να συσχετισθεί η έξοδος με την είσοδο. Ένα παράδειγμα τέτοιων κέντρων εσόδων είναι τα τμήματα marketing και πωλήσεων, τα οποία δεν έχουν την δυνατότητα να θέσουν την τιμή των προϊόντων που

πωλούν ή δεν έχουν σχέση με το κόστος παραγωγής τους (αυτές είναι ευθύνες των κέντρων που σχετίζονται με την παραγωγή των προϊόντων).

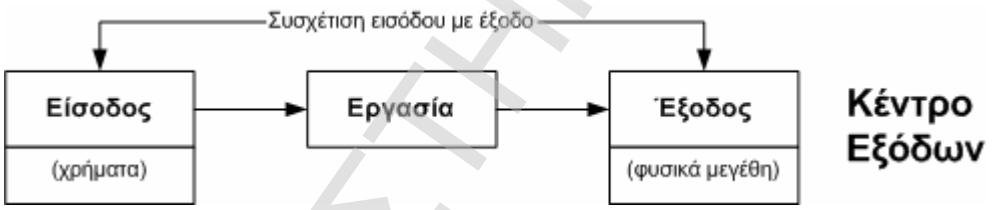


Κέντρο Εσόδων

Κέντρα Εξόδων (Expense Centers)

Τα κέντρα εξόδων είναι κέντρα ευθύνης, των οποίων τα στοιχεία εισόδου αποτιμώνται σε χρηματικούς όρους ενώ τα στοιχεία εξόδου δεν αποτιμώνται. Υπάρχουν δύο τύποι κέντρων εξόδων σε ένα οργανισμό.

Τα κέντρα εξόδων[8], των οποίων η είσοδος μπορεί να μετρηθεί με χρηματικούς όρους ενώ η έξοδος υπολογίζεται με φυσικούς όρους. Τέτοια κέντρα σχετίζονται με τις λειτουργίες της αποθήκευσης, μεταφοράς και διανομής των οποίων τα αποτελέσματα είναι μετρήσιμα αλλά δεν αποτιμώνται σε χρηματικούς όρους. Σε αυτά τα κέντρα εξόδων οι μονάδες εξόδου πολλαπλασιάζονται με το κόστος κάθε μονάδας που παράγεται για να υπολογισθεί το τελικό κόστος που θα έπρεπε να είχε το κέντρο εξόδων αυτό.



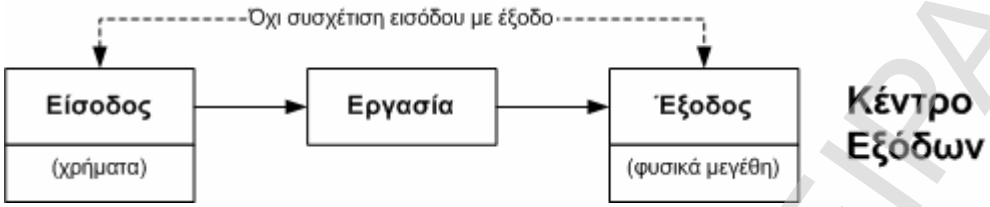
Κέντρο Εξόδων

Σε ένα οργανισμό υπάρχουν και άλλα είδη κέντρων εξόδων[9] των οποίων το τελικό κόστος δεν είναι εύκολο να υπολογισθεί και τις περισσότερες φορές απαιτείται η κρίση και η αντίληψη του διευθυντή που τα διοικεί. Τέτοια κέντρα εξόδων είναι τα τμήματα ενός οργανισμού που σχετίζονται με λειτουργίες:

- **Διαχείρισης και υποστήριξης**: είναι δύσκολο να μετρηθεί το τελικό αποτέλεσμα της λειτουργίας του κέντρου αυτού.
- **Έρευνας και ανάπτυξης**: είναι πολύ δύσκολο να συσχετισθεί άμεσα το αποτέλεσμα της λειτουργίας του κέντρου αυτού με τα στοιχεία εισόδου. Για παράδειγμα τα χρήματα που δαπανώνται για τις λειτουργίες της έρευνας και της ανάπτυξης σε ένα οργανισμό δεν μπορούν να συσχετισθούν άμεσα με κάποιο παραγόμενο προϊόν.

[8] Στην βιβλιογραφία αναφέρονται σαν Engineered Expense Centers, δηλαδή το κόστος υπολογίζεται με ένα «μηχανικό» τρόπο.
[9] Στην βιβλιογραφία τα κέντρα αυτά αναφέρονται σαν Discretionary Expense Centers.

- **Λειτουργίες marketing**: τα κέντρα αυτά περιλαμβάνουν τις λειτουργίες logistics (οι λειτουργίες που γίνονται από την στιγμή της παραγωγής ενός προϊόντος μέχρι να φτάσει αυτό στον τελικό πελάτη) και τις «καθαρές» λειτουργίες marketing (οι λειτουργίες που εκτελούνται για να πραγματοποιηθούν παραγγελίες για τα προϊόντα του οργανισμού).

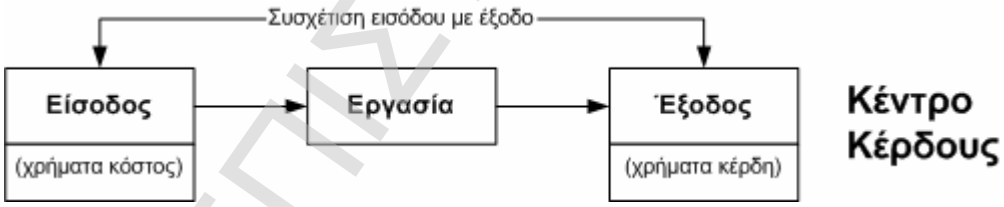


Κέντρο Εξόδων

Τα στοιχεία εξόδου των κέντρων αυτών δεν μπορούν να αποτιμηθούν με χρηματικούς όρους. Σε αυτά τα κέντρα εξόδων η διαφορά ανάμεσα στα προϋπολογιστικά στοιχεία και τις πραγματικές δαπάνες δεν μπορεί να θεωρηθεί σαν ένα μέτρο αποδοτικότητας του κέντρου.

Κέντρα Κέρδους (Profit Centers)

Στην περίπτωση που η οικονομική απόδοση ενός κέντρου ευθύνης αποτιμάται σε όρους κέρδους (η διαφορά δηλαδή ανάμεσα στα έσοδα και τα έξοδα) τότε το κέντρο αυτό ονομάζεται κέντρο κέρδους. Η έννοια του κέρδους είναι ένας πολύ χρήσιμος δείκτης της απόδοσης ενός κέντρου ευθύνης γιατί οι διευθυντές του οργανισμού χρησιμοποιούν ένα μετρητή κατανοητό που είναι προτιμότερος από την χρήση διαφόρων άλλων δεικτών απόδοσης.



Κέντρο Κέρδους

Η ευθύνη δημιουργίας των κέντρων κέρδους μεταφέρεται στους διευθυντές χαμηλότερων επιπέδων. Προκειμένου όμως να δημιουργηθούν οι κατάλληλες συνθήκες έτσι ώστε να δημιουργηθούν τα κέντρα κέρδους σε έναν οργανισμό πρέπει να ισχύουν οι παρακάτω δύο συνθήκες:

- Οι διευθυντές των χαμηλότερων επιπέδων του οργανισμού πρέπει να έχουν πρόσβαση στις σχετικές πληροφορίες που απαιτούνται για να ληφθεί η απόφαση για αύξηση των εξόδων με την προσμονή για μεγαλύτερη αύξηση των εσόδων από τις πωλήσεις.
- Επίσης πρέπει να υπάρχει ένας τρόπος μέτρησης της αποτελεσματικότητας της λήψης απόφασης που θα κάνει ο διευθυντής του κέντρου σχετικά με το ύψος των εξόδων έναντι των εσόδων που περιμένει από το κέντρο.

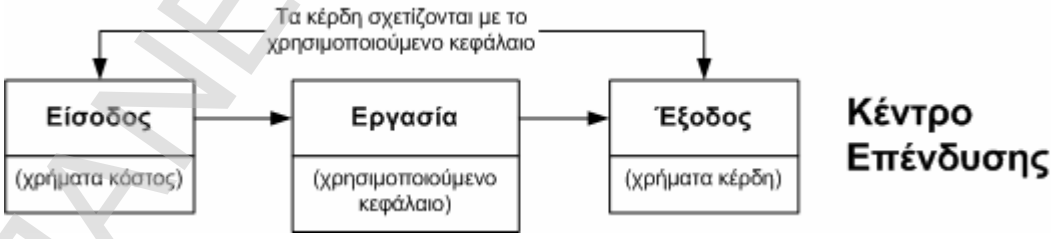
Ο διαμερισμός του οργανισμού σε κέντρα κέρδους μπορεί να προσφέρει πολλά **πλεονεκτήματα**. Βελτιώνεται η ταχύτητα λήψης αποφάσεων αφού οι διευθυντές των κέντρων κέρδους που είναι σε χαμηλότερα επίπεδα δεν είναι υποχρεωμένοι να αναφέρονται σε ανώτερα επίπεδα. Έτσι βελτιώνεται η ποιότητα των αποφάσεων, δίνεται μεγαλύτερο βάρος στην κερδοφορία των κέντρων και προσφέρεται ένας καλύτερος τρόπος μέτρησης της απόδοσης των κέντρων.

Οι επιχειρησιακές μονάδες (Business Units) ενός οργανισμού πολλές φορές, θεωρούνται σαν κέντρα κέρδους αφού οι διευθυντές των κέντρων μπορούν να ελέγξουν τους πόρους που απαιτούνται για τις λειτουργίες των κέντρων αυτών. Συνήθως οι επιχειρησιακές μονάδες σχετίζονται με λειτουργίες όπως marketing, ανάπτυξη προϊόντων, υπηρεσίες και υποστήριξη πελατών. Οι διευθυντές αυτών των κέντρων είναι σε θέση να επηρεάσουν τα έσοδα και το κόστος των κέντρων που διοικούν αν και σε πολλές περιπτώσεις επιβάλλονται περιορισμοί στην αυτονομία αυτών των επιχειρηματικών μονάδων από την Γενική Διεύθυνση του οργανισμού.

Αναφορικά με την μέτρηση της αποδοτικότητας των κέντρων αυτών υπάρχουν δύο τύποι μέτρησης της κερδοφορίας τους. Υπάρχει το μέτρο της **απόδοσης της διοίκησης**, το οποίο χρησιμοποιείται για να προσδιορίσει το πόσο καλά κάνει την δουλειά του ο διευθυντής. Αυτό το μέτρο χρησιμοποιείται για τον σχεδιασμό, τον συντονισμό και τον έλεγχο των καθημερινών λειτουργιών του κέντρου κέρδους. Επιπλέον υπάρχει και το μέτρο της **οικονομικής απόδοσης** του κέντρου, το οποίο χρησιμοποιείται για να προσδιορίσει πόσο καλά κάνει την δουλειά του το κέντρο κέρδους ως οικονομική οντότητα.

Κέντρα Επένδυσης (Investment Centers)

Υπάρχουν διάφορα κέντρα ευθύνης, στα οποία το κέρδος (έξοδος των κέντρων αυτών) συγκρίνεται με τα κεφάλαια (assets) τα οποία χρησιμοποιούνται στα κέντρα αυτά για να εκτελέσουν τις διάφορες εργασίες τους. Τέτοια κέντρα ευθύνης ονομάζονται κέντρα επένδυσης. Πολλές φορές τέτοια κέντρα ευθύνης λόγω του ότι γίνεται αναφορά στο κέρδος σαν έξοδος τους θεωρούνται και σαν κέντρα κέρδους παρά σαν ξεχωριστά κέντρα επένδυσης. Υπάρχουν όμως περιπτώσεις που είναι αρκετά δύσκολο να μετρηθούν τα κεφάλαια, που χρησιμοποιούνται για την λειτουργία τέτοιων κέντρων έτσι ώστε να θεωρούνται τα κέντρα αυτά σαν ξεχωριστή ομάδα κέντρων επένδυσης.



Κέντρο Επένδυσης

Ο σκοπός της συσχέτισης των χρησιμοποιούμενων κεφαλαίων με τα κέρδη επιτελεί δυο σκοπούς. Πρώτον, παρέχεται χρήσιμη πληροφορία σχετικά με τα κεφάλαια που χρησιμοποιούνται σε ένα οργανισμό προς τους διευθυντές για την λήψη αποφάσεων που θα

είναι οι καλύτερες για τον οργανισμό. Δεύτερον, δίνεται η δυνατότητα μέτρησης της απόδοσης των κέντρων αυτών σαν οικονομικές οντότητες.

Η συσχέτιση των κερδών ενός κέντρου επένδυσης με το κεφάλαιο, που έχει επενδυθεί σε αυτό είναι, είναι ένας χρήσιμος δείκτης που βοηθάει στον έλεγχο του κέντρου. Αυτό ισχύει σε περιπτώσεις που σε ένα κέντρο επένδυσης χρησιμοποιούνται διαφορετικές ποσότητες κεφαλαίων σε σχέση με ένα άλλο κέντρο επένδυσης. Έτσι, μπορεί για παράδειγμα δύο κέντρα επένδυσης να έχουν τα ίδια κέρδη αλλά για το πρώτο κέντρο να έχουν επενδυθεί τριπλάσια κεφάλαια σε σχέση με το δεύτερο κέντρο. Έτσι καταλαβαίνουμε ότι το δεύτερο κέντρο επένδυσης έχει καλύτερη απόδοση σε σχέση με το πρώτο.

Σε μια επιχειρησιακή μονάδα το άτομο που την διευθύνει έχει δύο στόχους σχετικά με την απόδοση της. Πρώτον, πρέπει η μονάδα να παράγει ικανοποιητικό κέρδος σε σχέση με τους πόρους που υπάρχουν στην διάθεσή της. Δεύτερον, πρέπει να γίνονται επενδύσεις στη μονάδα αυτή μόνο όταν οι επενδύσεις αυτές θα μπορέσουν να προσφέρουν κάποιο κέρδος. Η συσχέτιση των κερδών με τα κεφάλαια, που επενδύονται σε ένα κέντρο ευθύνης είναι σε θέση να υποστηρίξει τους διευθυντές στην επίτευξη των παραπάνω στόχων τους.

Διαδικασία Ελέγχου Διοίκησης

Η διαδικασία του Ελέγχου Διοίκησης (Management Control Process) περιλαμβάνει διάφορους τρόπους επικοινωνίας μεταξύ των διευθυντών και των υπαλλήλων του οργανισμού για την επίτευξη των στόχων του οργανισμού. Η επικοινωνία αυτή λαμβάνει χώρα σε ένα συγκεκριμένο σύστημα ελέγχου, το οποίο περιλαμβάνει τις ακόλουθες δραστηριότητες: 1) **Στρατηγικός Προγραμματισμός** (Strategic Planning), 2) **Προϋπολογισμός** (Budget), 3) **Εκτέλεση** και 4) **Μέτρηση και Αποτίμηση Απόδοσης** (Performance Measurement).

Στη συνέχεια παρουσιάζονται οι δραστηριότητες αυτές.

Στρατηγικός Προγραμματισμός

Στρατηγικό πρόγραμμα (Strategic Plan) είναι η επίσημη καταγραφή των πλάνων που θα ακολουθήσει ένας οργανισμός στην διάρκεια του χρόνου για να επιτύχει τους στόχους του. Μπορούμε να ορίσουμε σαν στρατηγικό προγραμματισμό την δραστηριότητα στην οποία αποφασίζονται τα κύρια βήματα που θα ακολουθηθούν καθώς και το προβλεπόμενο ύψος των πόρων που θα χρησιμοποιηθούν για να υλοποιηθεί η στρατηγική, που έχει καθοριστεί για τον οργανισμό.

Ο στρατηγικός προγραμματισμός διαφέρει από την διαμόρφωση της στρατηγικής σε ένα οργανισμό. Διαμόρφωση στρατηγικής είναι η διαδικασία της επιλογής νέων στρατηγικών, που θα ακολουθηθούν από τον οργανισμό ενώ στρατηγικός προγραμματισμός είναι η διαδικασία της επιλογής των τρόπων σύμφωνα με τους οποίους θα υλοποιηθεί η στρατηγική που έχει καθοριστεί για τον οργανισμό. Στην διαδικασία διαμόρφωσης της στρατηγικής τίθενται οι στόχοι του οργανισμού και στην συνέχεια δημιουργείται η στρατηγική, που θα υλοποιήσει του στόχους του. Στη συνέχεια η διαδικασία του στρατηγικού προγραμματισμού παίρνει τους στόχους και τις στρατηγικές που έχουν καθοριστεί και προσπαθεί να δημιουργήσει τα προγράμματα εκείνα που θα εκτελέσουν τις στρατηγικές και θα επιτύχει του στόχους αποδοτικά και αποτελεσματικά.

Σε πολλές εταιρείες οι στόχοι και οι στρατηγικές τους δεν είναι επαρκώς καθορισμένες έτσι ώστε οι διευθυντές να έχουν ένα ξεκάθαρο πλαίσιο βάσει του οποίου θα στηρίξουν την επιλογή των αποφάσεών τους, για το μέλλον της εταιρείας. Για το λόγο αυτό, η διαδικασία του στρατηγικού προγραμματισμού είναι μια σημαντική διαδικασία δεδομένο του ότι παρέχεται ένας επίσημος τρόπος καταγραφής των λεπτομερειών σχετικά με τους στόχους και την στρατηγική του οργανισμού.

Πλεονεκτήματα του Στρατηγικού Προγραμματισμού

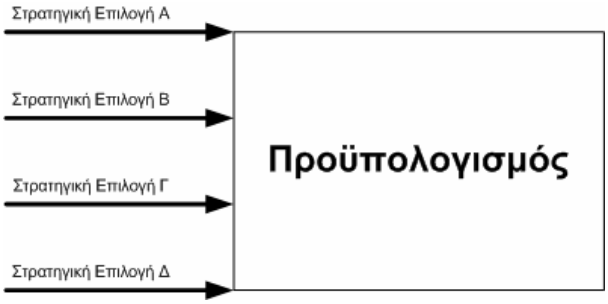
Η διαδικασία του στρατηγικού προγραμματισμού μπορεί να προσφέρει στον οργανισμό πολλά πλεονεκτήματα. Συγκεκριμένα, ο στρατηγικός προγραμματισμός μπορεί να θεωρηθεί σαν:

- **Πλαίσιο ανάπτυξης του προϋπολογισμού**

  Ένα σημαντικό πλεονέκτημα της διαδικασίας του στρατηγικού προγραμματισμού είναι η παροχή ενός πλαισίου ανάπτυξης του προϋπολογισμού. Η διοίκηση του οργανισμού μπορεί να προγραμματίσει την διάθεση των πόρων του οργανισμού μέσα από τον

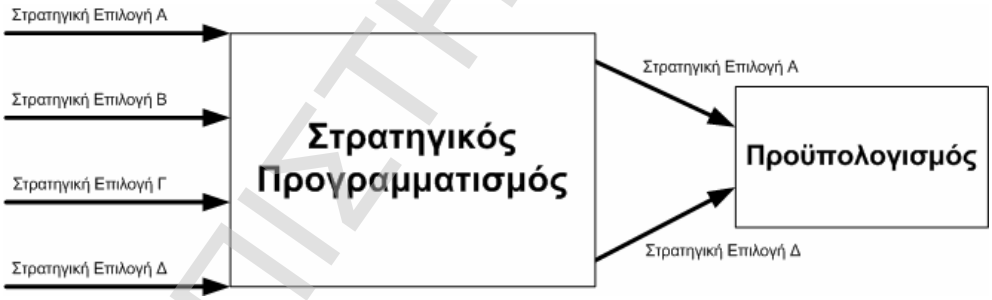
προϋπολογισμό έχοντας μια ξεκάθαρη εικόνα για το που θα κινηθεί ο οργανισμός τα επόμενα χρόνια μέσα από την δήλωση των στόχων και της στρατηγικής του.

Ένας οργανισμός χωρίς μια συγκεκριμένη διαδικασία στρατηγικού προγραμματισμού έχει να αντιμετωπίσει πολλές επιλογές και μια πληθώρα πληροφοριών στην κατάρτιση του προϋπολογισμού και πιθανόν να οδηγηθεί σε μη ικανοποιητικές επιλογές και την παράβλεψη πιθανών εναλλακτικών λύσεων.



Σχεδιασμός προϋπολογισμού χωρίς την χρήση στρατηγικού προγραμματισμού

Αντίθετα, με την χρήση μιας διαδικασίας στρατηγικού προγραμματισμού εξετάζονται όλες οι στρατηγικές που έχει στην διάθεση του ο υπεύθυνος δημιουργίας του προϋπολογισμού. Έτσι μικραίνει το εύρος των επιλογών και γίνεται επιλογή των βέλτιστων στρατηγικών επιλογών και στην συνέχεια με την χρήση των επιλογών αυτών σχεδιάζεται ο προϋπολογισμός.



Σχεδιασμός προϋπολογισμού με την χρήση στρατηγικού προγραμματισμού

- **Εργαλείο διοίκησης**

  Ο στρατηγικός προγραμματισμός είναι ένα πολύ καλό εργαλείο για διοικητική εκπαίδευση και εκμάθηση. Οι διευθυντές του οργανισμού έχουν στην διάθεση τους το εργαλείο αυτό για να κατανοήσουν την στρατηγική του οργανισμού και τον τρόπο, με τον οποίο θα εφαρμοστεί αυτή.

- **Μηχανισμός πίεσης των διευθυντών να σκέφτονται μακροπρόθεσμα και να στηρίζουν την στρατηγική του οργανισμού**

  Σε ένα οργανισμό συνήθως οι διευθυντές ασχολούνται περισσότερο με διαδικαστικά θέματα, και σκέφτονται κυρίως για το παρόν του οργανισμού, για καθημερινές λειτουργίες παρά για το μέλλον του οργανισμού. Ο στρατηγικός προγραμματισμός

μπορεί να θεωρηθεί σαν ένας μηχανισμός πίεσης των διευθυντών του οργανισμού να περνούν περισσότερο χρόνο για να σκέφτονται σοβαρότερα μακροπρόθεσμα προβλήματα.

Επίσης, μέσα από τις συζητήσεις και τις διαπραγματεύσεις που γίνονται κατά την διάρκεια του στρατηγικού προγραμματισμού γίνονται πιο ξεκάθαροι οι στόχοι και η στρατηγική του οργανισμού στους διευθυντές με αποτέλεσμα να σχετίζονται οι προσωπικοί στόχοι κάθε διευθυντή με τους στόχους του οργανισμού. Η προετοιμασία του στρατηγικού προγραμματισμού μπορεί να φανερώσει ότι οι προσωπικοί στόχοι κάθε διευθυντή να μην συμβαδίζουν με τους στόχους του οργανισμού με αποτέλεσμα οι πρώτοι να πρέπει να καθοριστούν από την αρχή.

Διαδικασία Στρατηγικού Προγραμματισμού

Σε ένα οργανισμό, που λειτουργεί βάσει του ημερολογιακού έτους συνήθως ξεκινάει η διαδικασία του στρατηγικού σχεδιασμού γύρω στην άνοιξη και την ολοκληρώνεται το φθινόπωρο πριν από την ετοιμασία του ετήσιου προϋπολογισμού. Η διαδικασία του στρατηγικού προγραμματισμού είναι μια διαδικασία, συνήθως βασισμένη σε ένα πλάνο πέντε ετών. Η διαδικασία του στρατηγικού προγραμματισμού περιλαμβάνει τα παρακάτω βήματα:

- **Αναθεώρηση και ενημέρωση του στρατηγικού προγράμματος, του περασμένου έτους.**

  Κατά την διάρκεια του έτους ενδέχεται να έχουν γίνει αλλαγές και αναθεωρήσεις στο στρατηγικό πλάνο (του περασμένου έτους) στην περίπτωση που αυτό είχε κριθεί αναγκαίο. Το πρώτο βήμα της διαδικασίας του στρατηγικού προγραμματισμού είναι η αναθεώρηση του προγράμματος του περασμένου έτους προκειμένου να ανακαλυφθούν πιθανές αλλαγές και να γίνουν οι απαραίτητες ενημερώσεις στο πρόγραμμα, του τρέχοντος έτους που σχεδιάζεται.

- **Εξέταση υποθέσεων και οδηγιών**

  Το αναθεωρημένο στρατηγικό πρόγραμμα βασίζεται σε συγκεκριμένες υποθέσεις όπως είναι η εξέλιξη των τιμών των πρώτων υλών, οι μισθοί και οι τιμές της αγοράς, η ανάπτυξη του ΑΕΠ της χώρας που δραστηριοποιείται ο οργανισμός, οι συνθήκες της αγοράς κλπ. Αυτές οι υποθέσεις πρέπει να εξετασθούν ξανά για να ανακαλυφθούν τυχόν αλλαγές, που θα επηρεάσουν το στρατηγικό πρόγραμμα.

  Επίσης από την διοίκηση του οργανισμού παρέχονται κάποιες οδηγίες για εσωτερικά θέματα του οργανισμού, που επηρεάζουν το στρατηγικό σχέδιο. Τέτοια θέματα σχετίζονται με τους μισθούς και το ποσοστό αυξήσεων τους, με υπάρχουσες και νέες γραμμές παραγωγής προϊόντων, με τιμές πώλησης κλπ. Όλες οι παραπάνω οδηγίες εξετάζονται από τους διευθυντές μέσα από συναντήσεις για να αποφασιστεί κατά πόσο ισχύουν και εάν πρέπει να αλλάξουν σε κάποια σημεία.

- **Πρώτη επανάληψη του νέου στρατηγικού προγράμματος**

  Χρησιμοποιώντας τις οδηγίες, τις υποθέσεις και τους στόχους του οργανισμού τα διάφορα τμήματα του οργανισμού (κάθε επιχειρηματική μονάδα – business unit του οργανισμού) παρέχουν μία πρώτη έκδοση του στρατηγικού προγράμματος που θα ακολουθήσουν μαζί με την κατάλληλη τεκμηρίωση (τους λόγους δηλαδή) για το πρόγραμμα αυτό. Τα προγράμματα αυτά για κάθε επιχειρηματική μονάδα σχετίζονται

με θέματα όπως τα έσοδα και τα έξοδα του τμήματος, οι πιθανές πωλήσεις και η παραγωγή, ο αριθμός των εργαζομένων κάθε τμήματος κλπ.

- **Ανάλυση δεδομένων**

  Η διοίκηση του οργανισμού παίρνει την πρώτη έκδοση από κάθε επιχειρηματική μονάδα και προσπαθεί να τις ομαδοποιήσει σε ένα συγκεντρωτικό στρατηγικό πρόγραμμα. Οι υπεύθυνοι του στρατηγικού προγραμματισμού αναλύουν το πρόγραμμα από κάθε τμήμα διεξοδικά και σε βάθος. Ελέγχεται η συνέπεια κάθε προγράμματος σχετικά με το κατά πόσο ένα πρόγραμμα μιας επιχειρηματικής μονάδας του οργανισμού μπορεί να επηρεάσει το πρόγραμμα μιας άλλης επιχειρηματικής μονάδας.

  Επίσης καθορίζονται θέματα που σχετίζονται με την επικοινωνία μεταξύ των διευθυντών του οργανισμού και την μορφή των αναφορών, που θα χρησιμοποιούνται για το σκοπό αυτό. Όλα τα παραπάνω θέματα διευθετούνται μέσα από συναντήσεις και συζητήσεις ανάμεσα στους διευθυντές του οργανισμού και τους υπευθύνους κάθε επιχειρηματικής μονάδας του οργανισμού.

- **Δεύτερη επανάληψη του νέου στρατηγικού προγράμματος**

  Η ανάλυση των πρώτων προγραμμάτων κάθε επιχειρηματικής μονάδας του οργανισμού μπορεί να φανερώσει κάποια προγράμματα στα οποία να απαιτείται διόρθωση σε κάποια θέματα. Είναι όμως πολύ πιθανό η αλλαγή στο πρόγραμμα ενός τμήματος να επηρεάζει το πρόγραμμα άλλων τμημάτων του οργανισμού. Έτσι γίνονται αλλαγές σε όλο το στρατηγικό πρόγραμμα του οργανισμού, όπου βέβαια αυτό απαιτείται. Οι αλλαγές βέβαια γίνονται σε συγκεκριμένα σημεία των προγραμμάτων και σαφώς είναι ευκολότερες από την αρχική εργασία δημιουργίας των προγραμμάτων.

- **Τελική αναθεώρηση και έγκριση**

  Στο τελευταίο βήμα της διαδικασίας του στρατηγικού προγραμματισμού γίνεται η τελική συνάντηση των διευθυντών του οργανισμού για να γίνει η τελική αναθεώρηση του στρατηγικού προγράμματος του οργανισμού. Ο γενικός διευθυντής του οργανισμού δίνει και την τελική έγκριση του προγράμματος. Η έγκριση του τελικού προγράμματος πρέπει να γίνει πριν από την εκκίνηση της προετοιμασίας του προϋπολογισμού του οργανισμού. Αυτό είναι αναγκαία γιατί το στρατηγικό πρόγραμμα είναι ένα σημαντικό στοιχείο εισόδου στην διαδικασία δημιουργίας του προϋπολογισμού.

Προϋπολογισμός (Budget)

Οι προϋπολογισμοί είναι ένα χρήσιμο εργαλείο για βραχυπρόθεσμο προγραμματισμό και έλεγχο σε ένα οργανισμό. Ένας λειτουργικός προϋπολογισμός (operational budget) συνήθως αφορά στο τρέχον ημερολογιακό έτος και παρουσιάζει των προγραμματισμό των εσόδων και των εξόδων για το έτος αυτό. Ο προϋπολογισμός έχει κάποια χαρακτηριστικά όπως:

- Γίνεται πρόβλεψη κερδών για τον οργανισμό.
- Δηλώνεται σε χρηματικούς όρους ενώ κάποιοι χρηματικοί όροι μπορούν να εκφραστούν με μη-χρηματικούς όρους όπως π.χ. ποσότητες προϊόντων που θα πωληθούν.
- Η διοίκηση του οργανισμού συμφωνεί για να αναλάβει ευθύνη για την υλοποίηση των στόχων το προϋπολογισμού.
- Ο προϋπολογισμός ελέγχεται και εγκρίνεται από ένα πρόσωπο ανώτερο στην ιεραρχία του οργανισμού από τα άτομα που τον συνέταξαν.
- Όταν εγκριθεί ο προϋπολογισμός μπορεί να αλλάξει μόνο κάτω από συγκεκριμένες προϋποθέσεις.
- Περιοδικά τα πραγματικά χρηματοοικονομικά αποτελέσματα συγκρίνονται με τα στοιχεία του προϋπολογισμού και οι διαφορές εντοπίζονται και ελέγχονται.

Χρήση του προϋπολογισμού

Η διαδικασία προετοιμασίας του προϋπολογισμού, που πραγματοποιείται σε ένα οργανισμό αποσκοπεί σε ορισμένους σκοπούς:

- **Διόρθωση του Στρατηγικού Προγράμματος**

  Ο προϋπολογισμός ο οποίος ολοκληρώνεται προς το τέλος του έτους παρέχει την δυνατότητα να χρησιμοποιηθούν όλες οι διαθέσιμες πληροφορίες που υπάρχουν μέχρι εκείνη την δεδομένη στιγμή καθώς και να χρησιμοποιηθεί η κρίση των διευθυντών από όλα τα επίπεδα του οργανισμού. Μέσα από τις πληροφορίες και την κρίση των διευθυντών παρέχεται η δυνατότητα να παρθούν αποφάσεις οι οποίες θα βελτιώσουν την απόδοση του οργανισμού.

- **Ενημέρωση**

  Ο προϋπολογισμός είναι μια τυπική έκφραση των μελλοντικών σχεδιών του οργανισμού καθώς και ένα μέσο γνωστοποίησης, διάχυσης και επικοινωνίας των βραχυπρόθεσμων στόχων του οργανισμού στο ανθρώπινο δυναμικό της. Μέσα από την διαδικασία κατάρτισης του προϋπολογισμού των διαφόρων επιχειρηματικών τμημάτων του οργανισμού οι διευθυντές κάθε τμήματος έχουν την δυνατότητα να γνωρίσουν τους στόχους που σχετίζονται με το τμήμα που διοικούν. Οι διευθυντές στη συνέχεια έχουν την δυνατότητα να ενημερώσουν για τους στόχους του τμήματος και τους υπόλοιπους εργαζόμενους (σε όσο βάθος κρίνει απαραίτητο ο κάθε διευθυντής).

- **Συντονισμός**

  Σε έναν οργανισμό οι διευθυντές των επιχειρηματικών μονάδων του οργανισμού συμμετέχουν στην διαδικασία σύνταξης του προϋπολογισμού της κάθε μονάδας. Στη συνέχεια, το υπεύθυνο προσωπικό συνενώνει τα διάφορα «κομμάτια» σε έναν ενιαίο προϋπολογισμό, στο οποίο είναι πολύ πιθανό να φανούν κάποιες ασυνέπειες μεταξύ των

στόχων των τμημάτων (πολλές φορές οι στόχοι ενός τμήματος του οργανισμού μπορεί να έρχονται σε αντίθεση με τους στόχους κάποιου άλλου τμήματος).

Μέσα από την διαδικασία προετοιμασίας του προϋπολογισμού τέτοια πιθανά λάθη εντοπίζονται και μέσα από την συνεργασία όλων των τμημάτων γίνεται προσπάθεια επίλυσης των προβλημάτων που θα παρουσιαστούν.

- **Ανάθεση Ευθύνης**

  Η έγκριση του προϋπολογισμού από τους διευθυντές κάθε τμήματος του οργανισμού σημαίνει αυτόματα ότι κάθε διευθυντής είναι υπεύθυνος. Ο διευθυντής του κάθε τμήματος έχει την εξουσιοδότηση να ξοδέψει μια συγκεκριμένη ποσότητα χρημάτων για συγκεκριμένους σκοπούς για το τμήμα του χωρίς να πρέπει να περιμένει να λάβει κάποια έγκριση από την ανωτέρα διοίκηση. Αυτό όμως σημαίνει και ότι ο διευθυντής έχει την ευθύνη για την οικονομική πορεία του τμήματος του οργανισμού που διοικεί και για την επίτευξη των στόχων, που έχουν τεθεί στον προϋπολογισμό.

- **Βάση για αποτίμηση απόδοσης**

  Ο προϋπολογισμός αναπαριστά κατά κάποιο τρόπο μια δέσμευση των ατόμων που τον συντάσσουν προς τους ανωτέρους τους όσον αφορά τους στόχους που αναφέρει ο προϋπολογισμός. Έτσι ο προϋπολογισμός μπορεί να θεωρηθεί σαν ένα μέσο αποτίμησης της απόδοσης κάθε τμήματος του οργανισμού. Η σύγκριση της απόδοσης γίνεται ανάμεσα στους στόχους που έχουν τεθεί στον προϋπολογισμό και στην πραγματική απόδοση του κάθε τμήματος στην διάρκεια του έτους. Επίσης είναι σημαντικό να θυμόμαστε ότι ο προϋπολογισμός αναθέτει ευθύνη στον υπεύθυνο που τον έχει συντάξει και εγκρίνει και για την ανωτέρα διοίκηση ο προϋπολογισμός κάθε τμήματος είναι ένα μέσο εκτίμησης της απόδοσης (κάθε τμήματος ξεχωριστά αλλά και ολόκληρου του οργανισμού γενικότερα).

Περιεχόμενα του Λειτουργικού Προϋπολογισμού

Ένας λειτουργικός προϋπολογισμός αναφέρεται στον οργανισμό σαν ολότητα αλλά μπορεί επίσης να αναφέρεται και σε κάθε κέντρο ευθύνης (επιχειρηματική μονάδα) του οργανισμού ξεχωριστά ανάλογα με τον βαθμό λεπτομέρειας που θέλουμε να παρουσιάσουμε. Ένας λειτουργικός προϋπολογισμός συνήθως περιλαμβάνει:

- **Έσοδα**: περιλαμβάνει τις προβλεπόμενες πωλήσεις επί την αναμενόμενη τιμή πώλησης. Το τμήμα αυτό είναι το πιο σημαντικό αλλά και με την μεγαλύτερη αβεβαιότητα αφού μιλάμε για προβλεπόμενα μεγέθη.
- **Κόστος παραγωγής και κόστος πωλήσεων**: παρουσιάζεται το κόστος υλικών και εργασίας για τις προγραμματισμένες ποσότητες παραγωγής. Οι διευθυντές παραγωγής καταστρώνουν πλάνα για την απόκτηση πρώτων υλών και εργασίας. Επίσης κατασκευάζουν προγράμματα παραγωγής έτσι ώστε να εξασφαλίσουν ότι οι απαιτούμενοι πόροι για την παραγωγή θα είναι διαθέσιμοι.
- **Έξοδα Marketing και έξοδα Logistics**: Ένα μεγάλο μέρος των εξόδων που θα χρησιμοποιηθούν για την προώθηση των πωλήσεων πρέπει να εξασφαλιστεί στον προϋπολογισμό. Σε μερικούς οργανισμούς τα έξοδα Logistics (κόστη που σχετίζονται με καταχώρηση παραγγελιών, αποθήκευση, ανάκτηση παραγγελιών, μεταφορές και παραδόσεις στους πελάτες) αναφέρονται σε διαφορετικό τμήμα στον προϋπολογισμό.
- **Γενικά έξοδα και έξοδα Διοίκησης**: έξοδα που σχετίζονται με το προσωπικό, για τις επιχειρηματικές μονάδες του οργανισμού.

- **Έξοδα Έρευνας και Ανάπτυξης**: τα έξοδα έρευνας και διοίκησης μπορεί να παρουσιάζονται σαν ένα συνολικό μέγεθος που προβλέπεται να χρησιμοποιηθεί για έρευνα και ανάπτυξη. Επίσης τα έξοδα αυτά μπορεί να παρουσιαστούν σαν ποσά για κάθε project έρευνας και ανάπτυξης που έχει αναλάβει το αντίστοιχο τμήμα του οργανισμού.

Χαρακτηριστικά επιτυχημένων προϋπολογισμών

Για να χαρακτηρισθεί ένας προϋπολογισμός επιτυχημένος πρέπει να έχει κάποια χαρακτηριστικά τα οποία παρουσιάζονται συνοπτικά παρακάτω:

- Ένας προϋπολογισμός είναι επιτυχημένος όταν υιοθετείται και υποστηρίζεται από όλους τους διευθυντές που έχουν την ευθύνη της υλοποίησης του. Τα στελέχη αυτά πρέπει να νιώθουν ότι είναι δικός τους ο προϋπολογισμός και ότι αυτοί είναι που θα συμβάλλουν στην επίτευξη των στόχων του προϋπολογισμού.
- Ένας προϋπολογισμός έχει πολλές πιθανότητες να επιτύχει όταν το ανθρώπινο δυναμικό στον οργανισμό τον εμπιστεύεται και τον θεωρεί σαν ένα όργανο συντονισμού και σχεδιασμού. Ο προϋπολογισμός πρέπει να θεωρείται σαν ένα μέσο που θα βοηθήσει τους εργαζόμενους να κάνουν καλύτερα την δουλειά τους και όχι σαν ένας μηχανισμός πίεσης και καταλογισμού ευθύνης από τους ανωτέρους.
- Ένας επιτυχημένος προϋπολογισμός είναι ένα μέσο παρακίνησης για την επίτευξη των στόχων και την διαρκή βελτίωση των επιδόσεων των διαφόρων επιχειρηματικών τμημάτων του οργανισμού και όχι ένα άλλοθι για την δικαιολόγηση αρνητικών αποκλίσεων, παραλήψεων ή αστοχιών.
- Ένας επιτυχημένος προϋπολογισμός πρέπει να είναι τεχνικά άρτιος και οι πληροφορίες του να είναι αξιόπιστες προκειμένου να κερδίσει την εκτίμηση των εργαζόμενων στον οργανισμό και να καταβάλουν αυτοί την απαραίτητη προσπάθεια για την επίτευξη των στόχων του.

Διαδικασία προετοιμασίας Προϋπολογισμού

Για την αποτελεσματικότερη ανάπτυξη και λειτουργία ενός προϋπολογισμού η διαδικασία ετοιμασίας του πρέπει να περιλαμβάνει τα παρακάτω βήματα:

<u>Οργανωτική Δομή</u>

Το έργο της προετοιμασίας του προϋπολογισμού συνήθως ανατίθεται σε ένα αρμόδιο τμήμα του οργανισμού, το οποίο εκτελεί τις παρακάτω λειτουργίες:

- Έκδοση και αποστολή των κατάλληλων εγχειριδίων, διαδικασιών και των σχετικών εντύπων για την προετοιμασία του προϋπολογισμού.
- Προετοιμασία των απαραίτητων πληροφοριών που σχετίζονται με την κατάρτιση του προϋπολογισμού και διάδοση τους σε όλο τον οργανισμό.
- Παροχή βοήθειας σε κάθε τμήμα ξεχωριστά για την σύνταξη του προϋπολογισμού.
- Έλεγχος και διόρθωση των προϋπολογισμών κάθε τμήματος του οργανισμού πριν από την τελική έκδοσή τους.
- Ανάλυση των προτεινόμενων σχεδίων προϋπολογισμών και ετοιμασία προτάσεων στους συντάκτες των προϋπολογισμών και στην ανώτερα διοίκηση.

- Αξιολόγηση και σύνθεση των προτάσεων προϋπολογισμών κάθε τμήματος ξεχωριστά.

Επίσης πρέπει να οριστεί η επιτροπή του οργανισμού εκείνη, η οποία θα αναλάβει τον έλεγχο, εκτίμηση και έγκριση του προϋπολογισμού. Η επιτροπή αυτή συνήθως αποτελείται από ανώτερα στελέχη του οργανισμού, όπως ο γενικός διευθυντής, ο οικονομικός διευθυντής του οργανισμού και άλλοι διευθυντές ανάλογα με την οργανωτική δομή του οργανισμού.

Έκδοση οδηγιών και συμβουλών

Αφού έχει καθοριστεί το αρμόδιο τμήμα του οργανισμού που θα υποστηρίξει την προετοιμασία του προϋπολογισμού και η επιτροπή προϋπολογισμού που τελικά θα τον εγκρίνει, το πρώτο βήμα της διαδικασίας προετοιμασία του προϋπολογισμού είναι η έκδοση των κατευθυντήριων οδηγιών για την σύνταξη του προϋπολογισμού και η διάδοση των οδηγιών στους διευθυντές όλων των τμημάτων του οργανισμού. Ο οδηγίες αυτές δείχνουν την κατεύθυνση και τον τόνο του προϋπολογισμού.

Όλα τα κέντρα ευθύνης του οργανισμού πρέπει να ακολουθούν τις οδηγίες αυτές κατά την διάρκεια της κατάρτισης του προϋπολογισμού. Τέτοιες οδηγίες σχετίζονται με γενικά θέματα όπως η γενική εικόνα της οικονομίας και της αγοράς, ο πληθωρισμός κλπ. Επίσης οι οδηγίες αφορούν συγκεκριμένα τον οργανισμό όπως οι στόχοι του οργανισμού, οι εκπτώσεις τιμών, οι αμοιβές προσωπικού, τα αναμενόμενα κέρδη από τις πωλήσεις κ.α.

Αρχική έκδοση του προϋπολογισμού

Με την χρήση των οδηγιών ο διευθυντής κάθε κέντρου ευθύνης του οργανισμού με την υποστήριξη του προσωπικού του τμήματος του κατασκευάζει μια αρχική πρόταση προϋπολογισμού για την επόμενη περίοδο. Για την κατάρτιση του πρώτου σχεδίου του προϋπολογισμού λαμβάνονται υπόψη στοιχεία που σχετίζονται με τις τρέχουσες συνθήκες στις οποίες λειτουργεί το τμήμα. Είναι σημαντικό όμως να ληφθούν υπόψη και στοιχεία που προκύπτουν από την ανάλυση του εσωτερικού και του εξωτερικού περιβάλλοντος του τμήματος του οργανισμού που είναι πολύ πιθανό να επηρεάσουν την τρέχουσα κατάσταση του τμήματος. Οι παράγοντες που μπορεί να επηρεάσουν την σύνταξη του προϋπολογισμού είναι οι παρακάτω:

Εξωτερικοί Παράγοντες

- αλλαγές στην οικονομική δραστηριότητα του τμήματος
- αλλαγές στις τιμές των πρώτων υλών και υπηρεσιών που αγοράζει το τμήμα
- αλλαγές στο εργατικό κόστος
- αλλαγές στο κόστος Έρευνας και Διοίκησης, Marketing και Διαχείρισης και Υποστήριξης
- αλλαγές στην τιμή πώλησης

Εσωτερικές Πολιτικές και Πρακτικές

- Αλλαγές στο κόστος παραγωγής που σχετίζεται με νέο εξοπλισμό και μεθόδους παραγωγής

- Αλλαγές στο μερίδιο αγοράς

Διαπραγματεύσεις

Ο διευθυντής του τμήματος που έχει συντάξει τον προϋπολογισμό συζητά τις προτάσεις του με τους ανώτερους του. Το βήμα αυτό είναι η καρδιά όλης της διαδικασίας. Στην φάση αυτή αξιολογείται κατά πόσο ο προτεινόμενος προϋπολογισμός κινείται εντός των ορίων των κατευθυντήριων οδηγιών, επιβεβαιώνεται κατά πόσο αναφέρεται σε ρεαλιστικά μεγέθη και εξετάζεται η συμβατότητα του προϋπολογισμού του τμήματος με τους προϋπολογισμούς των άλλων επιχειρηματικών τμημάτων του οργανισμού.

Έτσι εντοπίζονται οι αναγκαίες διορθώσεις που πρέπει να γίνουν στον αρχικό προϋπολογισμό. Οι διορθώσεις αυτές θα οριστικοποιηθούν μετά από το πέρας των διαπραγματεύσεων που θα λάβουν χώρα στο βήμα αυτό.

Αξιολόγηση και έγκριση προϋπολογισμού

Τα προτεινόμενα σχέδια προϋπολογισμών κάθε τμήματος περνάνε στην συνέχεια διαδοχικά από όλα τα επίπεδα του οργανισμού έως ότου φτάσουν στο ανώτατο επίπεδο της διοίκησης. Στο επίπεδο αυτό γίνεται αξιολόγηση των προϋπολογισμών και ο τελικός προϋπολογισμός του οργανισμού λαμβάνει την τελική του μορφή.

Η επιτροπή προϋπολογισμού που αποτελείται από τα ανώτερα στελέχη του οργανισμού αξιολογεί την συνεκτικότητα των επιμέρους προϋπολογισμών, τον βαθμό επίτευξης των απαιτούμενων στόχων και την χρονική περίοδο του προϋπολογισμού. Μετά την αξιολόγηση η επιτροπή προϋπολογισμού εγκρίνει τον προϋπολογισμό, ο οποίος μετά από την αντίστοιχη έγκριση και υπογραφή από τον γενικό διευθυντή του οργανισμού, εισάγεται στο διοικητικό συμβούλιο του οργανισμού για έγκριση και επικύρωση.

Αναθεώρηση προϋπολογισμού

Ένα από τα σημαντικότερα θέματα που πρέπει να απασχολεί την διοίκηση του οργανισμού είναι η διαδικασία αναθεώρησης του προϋπολογισμού αφού αυτός έχει εγκριθεί. Οι διαδικασίες αναθεώρησης του προϋπολογισμού διαφέρουν από οργανισμό σε οργανισμό. Συνήθως καθιερώνονται συγκεκριμένες διαδικασίες αναθεώρησης κάθε τετράμηνο, τρίμηνο κλπ. Οι διαδικασίες αυτές είναι συστηματικές. Υπάρχουν όμως και διαδικασίες αναθεώρησης του προϋπολογισμού οι οποίες υλοποιούνται κάτω από συγκεκριμένες προϋποθέσεις.

Οι αναθεωρήσεις του προϋπολογισμού πρέπει να γίνονται μόνο όταν οι συνθήκες είναι τέτοιες ώστε ο εγκεκριμένος προϋπολογισμός να είναι εντελώς αναληθής σε σχέση με τις τρέχουσες καταστάσεις στον οργανισμό και να μην αποτελεί πλέον ένα εργαλείο ελέγχου.

Οι οργανισμοί που λειτουργούν σε ένα δυναμικό και μεταβαλλόμενο περιβάλλον αποκτούν πλεονέκτημα από την περιοδική αναθεώρηση των στοιχείων του προϋπολογισμού και αυτό συμβαίνει γιατί ο αναθεωρημένος προϋπολογισμός παρέχει καλύτερη καθοδήγηση γιατί περιλαμβάνει και τις αλλαγές που έχουν γίνει.

Μέτρηση και Αποτίμηση Απόδοσης

Ένας τρόπος αποτίμησης της οικονομικής απόδοσης ενός οργανισμού είναι η χρήση χρηματοοικονομικών αναφορών όπως εκφράζονται αυτές μέσα από την χρήση των προϋπολογισμών. Οι αναφορές αυτές βασισμένες σε χρηματοοικονομικά κριτήρια και μέσα μέτρησης παρουσιάζουν την απόδοση του οργανισμού σε χρηματοοικονομικούς όρους, παρουσιάζουν δηλαδή στοιχεία εσόδων, εξόδων, κερδών κλπ. Ωστόσο, η χρηματοοικονομική απόδοση ενός οργανισμού είναι μόνο η μια πλευρά της συνολικής απόδοσής του. Ένα σύστημα μέτρησης και αποτίμησης της απόδοσης του οργανισμού πρέπει να περιέχει και μη-χρηματοοικονομικά στοιχεία για να παρέχει μια συνολική εικόνα της απόδοσης του οργανισμού.

Σκοπός ενός συστήματος μέτρησης της απόδοσης είναι η εφαρμογή της στρατηγικής του οργανισμού. Οι διευθυντές που θα χρησιμοποιήσουν τέτοια συστήματα πρέπει να επιλέξουν τα μέτρα μέτρησης εκείνα που αντιπροσωπεύουν καλύτερα την στρατηγική του οργανισμού για να γίνεται ο καλύτερος έλεγχος. Τα μέτρα μέτρησης χρησιμοποιούνται σαν κριτήρια για παρούσα και μελλοντική επιτυχία. Εάν κάποια σημαντικά μέτρα έχουν βελτιωθεί ο οργανισμός έχει υλοποιήσει την στρατηγική του.

Ένα σημαντικό λάθος που γίνεται σε πολλές περιπτώσεις είναι ότι χρησιμοποιούνται μόνο χρηματοοικονομικά μέτρα μέτρησης για να καθοριστεί η βελτίωση της απόδοσης του οργανισμού. Σε πολλές επιχειρήσεις βασικός στόχος είναι το βέλτιστο κέρδος των μετόχων τους (shareholder returns). Ωστόσο, η βραχυπρόθεσμη κερδοφορία μιας επιχείρησης δεν σημαίνει απαραίτητα και μακροπρόθεσμα κέρδη αφού η κερδοφορία των μετόχων σε μια επιχείρηση σημαίνει παρούσα αξία από μελλοντικά κέρδη, δηλαδή βραχυπρόθεσμα κέρδη.

Έτσι, η μέτρηση της απόδοσης ενός οργανισμού στηριζόμενη μόνο σε χρηματοοικονομικά δεδομένα (δείκτες) είναι επικίνδυνη και μπορεί να κρύβει πολλά προβλήματα για πολλούς λόγους [1, 22]:

- Μπορεί να ενθαρρύνονται βραχυπρόθεσμες ενέργειες που δεν είναι στα μακροπρόθεσμα σχέδια του οργανισμού.
- Οι χρηματοοικονομικοί δείκτες είναι στατικοί και δεν μπορούν να απεικονίσουν το σύγχρονο επιχειρηματικό περιβάλλον.
- Οι διευθυντές των επιχειρηματικών μονάδων του οργανισμού μπορεί μην παίρνουν χρήσιμες μακροχρόνιες αποφάσεις προκειμένου να έχουν βραχυπρόθεσμα κέρδη.
- Ο έλεγχος των οικονομικών αποτελεσμάτων ενός τμήματος του οργανισμού μπορεί να αποτελέσει κίνητρο για παραποίηση των οικονομικών δεδομένων.
- Οι χρηματοοικονομικοί δείκτες δεν αντιπροσωπεύουν την στρατηγική της επιχείρησης, τους στόχους των εργαζομένων και τις ανάγκες των πελατών.
- Οι χρηματοοικονομικοί δείκτες δείχνουν τις συνέπειες μόνο των δραστηριοτήτων του οργανισμού και όχι τις αιτίες των προβλημάτων.

Ο έλεγχος και η αποτίμηση μετρήσεων που βασίζονται μόνο σε χρηματοοικονομικά στοιχεία δεν είναι σίγουρο πως θα οδηγήσει στην επιτυχή υλοποίηση της στρατηγικής ενός οργανισμού. Η λύση είναι να γίνεται μέτρηση και αποτίμηση της απόδοσης των επιχειρηματικών μονάδων (business units) του οργανισμού βάσει χρηματοοικονομικών και μη-χρηματοοικονομικών μετρήσεων. Οι μη-χρηματοοικονομικές μονάδες μέτρησης, που

υποστηρίζουν την εφαρμογή της στρατηγικής αναφέρονται και σαν *βασικοί παράγοντες επιτυχίας (key success factors)* [1].

Βασικοί Παράγοντες Επιτυχίας

Οι βασικοί παράγοντες επιτυχίας είναι οι μη-χρηματοοικονομικοί παράγοντες βάσει των οποίων γίνεται η μέτρηση και αποτίμηση της απόδοσης ενός οργανισμού. Οι παράγοντες αυτοί είναι:

- **Παράγοντες σχετικοί με Πελάτες**. Οι παράγοντες αυτοί σχετίζονται με τους πελάτες του οργανισμού και περιλαμβάνουν:
    - Παραγγελίες: οι παραγγελίες τις οποίες «κλείνουν» οι πελάτες για κάποια συγκεκριμένα προϊόντα.
    - Παραγγελίες σε αναμονή (Back Orders): η διαφορά ανάμεσα στην παραγωγή και τις πωλήσεις.
    - Μερίδιο αγοράς.
    - Παραγγελίες μεγάλων πελατών.
    - Ικανοποίηση πελάτη.
    - Διατήρηση πελάτη: η διάρκεια της σχέσης με ένα συγκεκριμένο πελάτη.
    - Πίστη πελατών.
- **Παράγοντες σχετικοί με Εσωτερικές Επιχειρηματικές Λειτουργίες**. Οι παράγοντες αυτοί σχετίζονται με τις εσωτερικές επιχειρηματικές λειτουργίες του οργανισμού και περιλαμβάνουν:
    - Εκμετάλλευση δυναμικότητας
    - On-time παραδόσεις προϊόντων
    - Γύρισμα αποθέματος
    - Ποιότητα

Ένα σύστημα μέτρησης της απόδοσης είναι η **Balanced Scorecard** [9]. Σύμφωνα με το σύστημα αυτό σε κάθε επιχειρηματική μονάδα τίθενται συγκεκριμένοι στόχοι οι οποίοι αποτιμώνται βάσει 4 διαφορετικών συνιστωσών:

- Χρηματοοικονομικά στοιχεία
- Πελάτες
- Εσωτερικές επιχειρηματικές λειτουργίες
- Μάθηση και Ανάπτυξη

Το σύστημα προσπαθεί να φέρει μια ισορροπία ανάμεσα στις παραπάνω διαφορετικές μετρήσεις έτσι ώστε να επιτύχει την συνταύτιση των στόχων: οι λειτουργίες των υπαλλήλων να στοχεύουν και στην επίτευξη των στόχων του οργανισμού.

Balanced Scorecard

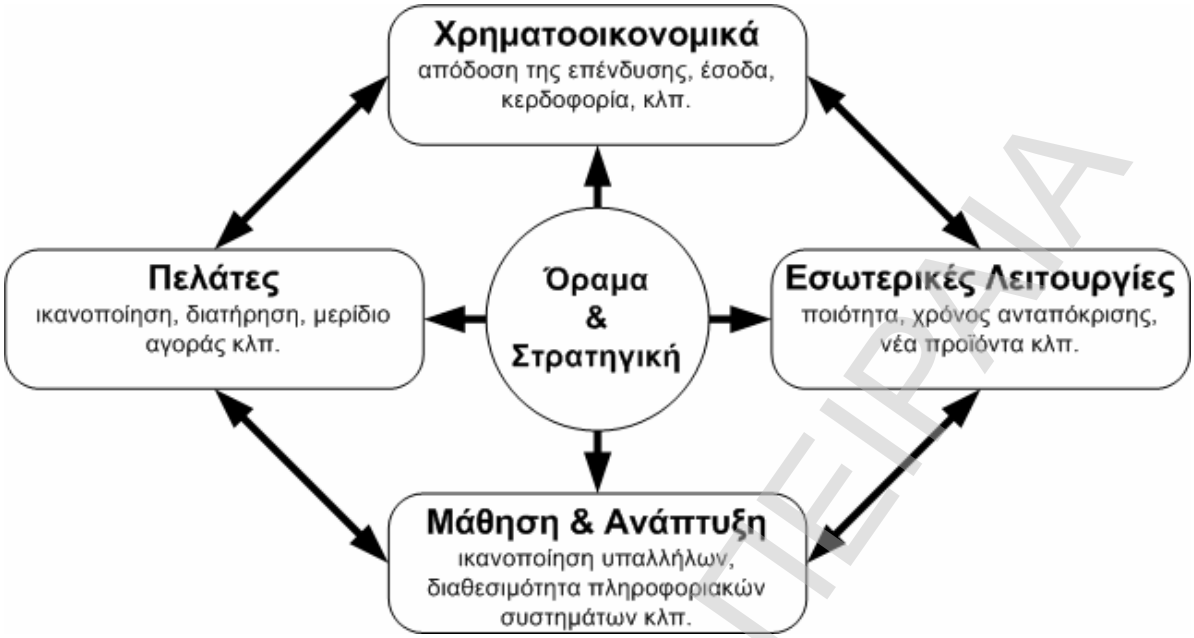
Ένα ικανοποιητικό σύστημα μέτρησης και αποτίμησης της απόδοσης ενός οργανισμού πρέπει να περιλαμβάνει στοιχεία που να στηρίζονται τόσο σε χρηματοοικονομικά όσο και σε μη-χρηματοοικονομικά δεδομένα. Πρέπει να υπάρχει ένα σύστημα μέτρησης της απόδοσης, το οποίο να ευθυγραμμίζει τους οργανισμούς με τις στρατηγικές τους. Ένα τέτοιο σύστημα πρέπει να μην είναι προσανατολισμένο μόνο σε βραχυπρόθεσμα χρηματοοικονομικά στοιχεία αλλά να ενσωματώνει και πιθανές ευκαιρίες για τον οργανισμό για παροχή προϊόντων και υπηρεσιών αξίας ικανοποιώντας τους πελάτες του οργανισμού. Επίσης ένα τέτοιο σύστημα πρέπει να παρουσιάζει και τις προοπτικές ανάπτυξης του ίδιου του οργανισμού μέσω της βελτίωσης των λειτουργιών του αλλά και την βελτίωση της σχέσης του με τους εργαζόμενους.

Η Balanced Scorecard (BSC) είναι ένα σύστημα μέτρησης της απόδοσης ενός οργανισμού, το οποίο μεταφράζει την αποστολή και την στρατηγική ενός οργανισμού σε ένα σύνολο μονάδων μέτρησης της απόδοσης παρέχοντας στον οργανισμό ένα πλαίσιο για ένα στρατηγικό σύστημα διοίκησης. Η BSC δίνει έμφαση στην επίτευξη των χρηματοοικονομικών στόχων του οργανισμού καθώς και στους οδηγούς (drivers) οι οποίοι χρησιμοποιούνται για την επίτευξη των στόχων αυτών. Η BSC δίνει την δυνατότητα για παρακολούθηση των χρηματοοικονομικών αποτελεσμάτων καθώς και για την παρακολούθηση των άυλων εκείνων μέσων που απαιτούνται για την ανάπτυξη του οργανισμού: τις εσωτερικές λειτουργίες, τους πελάτες και τους εργαζόμενους του οργανισμού.

Ένας οργανισμός πέρα από την επίτευξη των χρηματοοικονομικών στόχων του πρέπει να εστιάζει την προσοχή του και σε άυλα εκείνα κεφάλαια που διαθέτει. Τα άυλα κεφάλαια επιτρέπουν σε ένα οργανισμό:

- να αναπτύξει σωστές σχέσεις με τους πελάτες του
- να δημιουργήσει καινοτομικά προϊόντα και υπηρεσίες
- να παράγει προϊόντα κα υπηρεσίες υψηλής ποιότητας
- να ενθαρρύνει τους εργαζόμενους να αναπτύξουν τις ικανότητες τους.

Η BSC μετράει την απόδοση ενός οργανισμού χρησιμοποιώντας 4 διαφορετικές συνιστώσες τις οποίες προσπαθεί να ισορροπήσει: τα ***χρηματοοικονομικά στοιχεία***, τους ***πελάτες***, τις ***εσωτερικές επιχειρηματικές λειτουργίες*** και την ***μάθηση και ανάπτυξη*** των εργαζόμενων του οργανισμού. Είναι ένα μείγμα των παραδοσιακών χρηματοοικονομικών δεικτών για στοιχεία του παρελθόντος με μη-χρηματοοικονομικούς δείκτες (βασικοί παράγοντες επιτυχίας). Οι μη-χρηματοοικονομικοί δείκτες χρησιμοποιούνται σαν κριτήριο αξιολόγησης της μελλοντικής απόδοσης του οργανισμού, της πορείας του και της στρατηγικής του.



Διάγραμμα 27: Οι 4 συνιστώσες του συστήματος Balanced Scorecard

Η λογική του συστήματος Balanced Scorecard λειτουργεί ως εξής: ο οργανισμός προκειμένου να επιτύχει τα επιθυμητά ***χρηματοοικονομικά*** αποτελέσματα πρέπει να έχει δημιουργήσει την μέγιστη δυνατή αξία στα προϊόντα ή της υπηρεσίες του για τους ***πελάτες***. Αυτό όμως προϋποθέτει την καλύτερη δυνατή λειτουργία των ***εσωτερικών επιχειρηματικών λειτουργιών*** του οργανισμού. Για να συμβεί όμως αυτό, πρέπει να υπάρχει ***μάθηση και ανάπτυξη*** μεταξύ των εργαζομένων του οργανισμού [22].

Η BSC συνεχίζει να χρησιμοποιεί τα παραδοσιακά χρηματοοικονομικά στοιχεία ως παρουσίαση της απόδοσης των κρίσιμων επιχειρηματικών λειτουργιών του οργανισμού αλλά βελτιώνει την πληροφόρηση μέσα από ένα πιο γενικό και ολοκληρωμένο σύνολο μετρικών, που συσχετίζουν τους πελάτες, τις εσωτερικές λειτουργίες και τους εργαζόμενους με την μακροπρόθεσμη οικονομική απόδοση του οργανισμού. Η BSC παρέχει ένα πλαίσιο, ένα τρόπο επικοινωνίας του οράματος και της στρατηγικής του οργανισμού. Χρησιμοποιεί μετρικές για να ενημερώσει τους εργαζόμενους τους τρόπους παρούσας και μελλοντικής επιτυχίας του οργανισμού. Συσχετίζει τους τρόπους εξυπηρέτησης των αναγκών των πελατών με τα μέσα και τους πόρους που διαθέτει ο οργανισμός (εργαζόμενοι και εσωτερικές λειτουργίες).

Οι 4 συνιστώσες του συστήματος BSC επιτρέπουν την ισορροπία ανάμεσα στους βραχυπρόθεσμους και μακροπρόθεσμους στόχους, ανάμεσα στα επιθυμητά αποτελέσματα και τον τρόπο επίτευξης των αποτελεσμάτων αυτών, ανάμεσα τους στόχους και τις μετρικές τους. Οι συνιστώσες της BSC εκφράζουν μια σχέση αιτίας-αποτελέσματος με την έννοια ότι παρουσιάζονται οι στόχοι, που πρέπει να επιτύχει ο οργανισμός και οι τρόποι με τους οποίους θα επιτευχθούν οι στόχοι αυτοί. Έτσι κάθε ενέργεια (αιτία) που γίνεται επηρεάζει με τη σειρά της κάποιο στόχο (αποτέλεσμα) που έχει τεθεί από τον οργανισμό. Στη συνέχεια παρουσιάζονται συνοπτικά οι 4 συνιστώσες της BSC.

Συνιστώσες της Balanced Scorecard

- **Χρηματοοικονομικά**

  Η BSC χρησιμοποιεί τις χρηματοοικονομικές μετρικές αφού είναι ένας πολύτιμος τρόπος μέτρησης των χρηματοοικονομικών συνεπειών των ενεργειών που έχουν ήδη πραγματοποιηθεί. Οι χρηματοοικονομικές μετρικές παρουσιάζουν εάν η στρατηγική ενός οργανισμού και η υλοποίηση της μπορούν να συνεισφέρουν στην δημιουργία ικανοποιητικών οικονομικών αποτελεσμάτων για τον οργανισμό.

  Οι χρηματοοικονομικοί στόχοι που εκφράζονται με αυτή την συνιστώσα της BSC σχετίζονται με την κερδοφορία του οργανισμού, τα έσοδα, την επιστροφή κεφαλαίου που έχει επενδυθεί (ROCE – Return On Capital Employed), με την οικονομική ανάπτυξη του οργανισμού και άλλες σχετικές μετρικές.

- **Πελάτες**

  Η συνιστώσα αυτή εκφράζει τους πελάτες του οργανισμού. Συγκεκριμένα, η διοίκηση του οργανισμού προσδιορίζει τα τμήματα της αγοράς στα οποία κάθε επιχειρησιακή μονάδα (business unit) του οργανισμού θα δραστηριοποιείται και μετράει την απόδοση κάθε επιχειρησιακής μονάδας σε κάθε τμήμα της αγοράς.

  Οι μετρικές της απόδοσης κάθε επιχειρησιακής μονάδας του οργανισμού αναφορικά με τους πελάτες της σχετίζονται με έννοιες όπως η ικανοποίηση του πελάτη, η διατήρηση του, η απόκτηση νέων πελατών, η κερδοφορία κάθε πελάτη και το μερίδιο αγοράς που έχει αποκτήσει ο οργανισμός (ή η συγκεκριμένη επιχειρησιακή μονάδα του οργανισμού). Η συνιστώσα αυτή επίσης σχετίζεται και με την αξία την οποία προσφέρει η επιχειρησιακή μονάδα προς συγκεκριμένους πελάτες με στόχο την διατήρηση τους (για παράδειγμα γρήγορη παράδοση προϊόντων, παροχή καινοτόμων προϊόντων και υπηρεσιών κλπ.).

- **Εσωτερικές επιχειρηματικές λειτουργίες**

  Σύμφωνα με τη συνιστώσα αυτή οι διευθυντές του οργανισμού προσπαθούν να εντοπίσουν τις επιχειρηματικές λειτουργίες που θα πετύχουν την βέλτιστη εξυπηρέτηση των πελατών καθώς και αυτές που θα επιτύχουν τους χρηματοοικονομικούς στόχους του οργανισμού. Η συνιστώσα αυτή δεν σχετίζεται μόνο με υπάρχουσες λειτουργίες αλλά προτείνει και την δημιουργία νέων λειτουργιών όπου αυτό είναι αναγκαίο έτσι ώστε ο οργανισμός να πετύχει τους στόχους του και να εξυπηρετήσει τους πελάτες του.

  Επίσης η συνιστώσα αυτή σχετίζεται με λειτουργίες του οργανισμού που θα εξυπηρετήσουν υπάρχοντες πελάτες με προϊόντα και υπηρεσίες καθώς και με εντελώς καινούργια και καινοτομικά προϊόντα και υπηρεσίες που θα εξυπηρετήσουν υπάρχοντες και μελλοντικούς πελάτες.

- **Μάθηση και Ανάπτυξη**

  Η τέταρτη και τελευταία συνιστώσα της BSC, η μάθηση και ανάπτυξη υποδεικνύει την υποδομή που πρέπει να έχει ο οργανισμός έτσι ώστε να επιτύχει μακροπρόθεσμη ανάπτυξη και βελτίωση. Η μάθηση και ανάπτυξη του οργανισμού σχετίζεται με τρεις έννοιες: άνθρωποι, πληροφοριακά συστήματα και τρόπος λειτουργίας του οργανισμού. Οι προηγούμενες συνιστώσες της BSC, τα χρηματοοικονομικά, οι πελάτες και οι εσωτερικές λειτουργίες φανερώνουν τα κενά στην υποδομή που υπάρχουν σε έναν οργανισμό σχετικά με τις δυνατότητες των ανθρώπων, των πληροφοριακών συστημάτων και των διαδικασιών προκειμένου να υποστηρίξουν τις συνιστώσες αυτές.

Για να κλείσουν αυτά τα κενά η διοίκηση του οργανισμού πρέπει να επενδύσει στην μάθηση των εργαζομένων, στον εμπλουτισμό των πληροφοριακών συστημάτων και την διόρθωση του τρόπου λειτουργίας του οργανισμού.

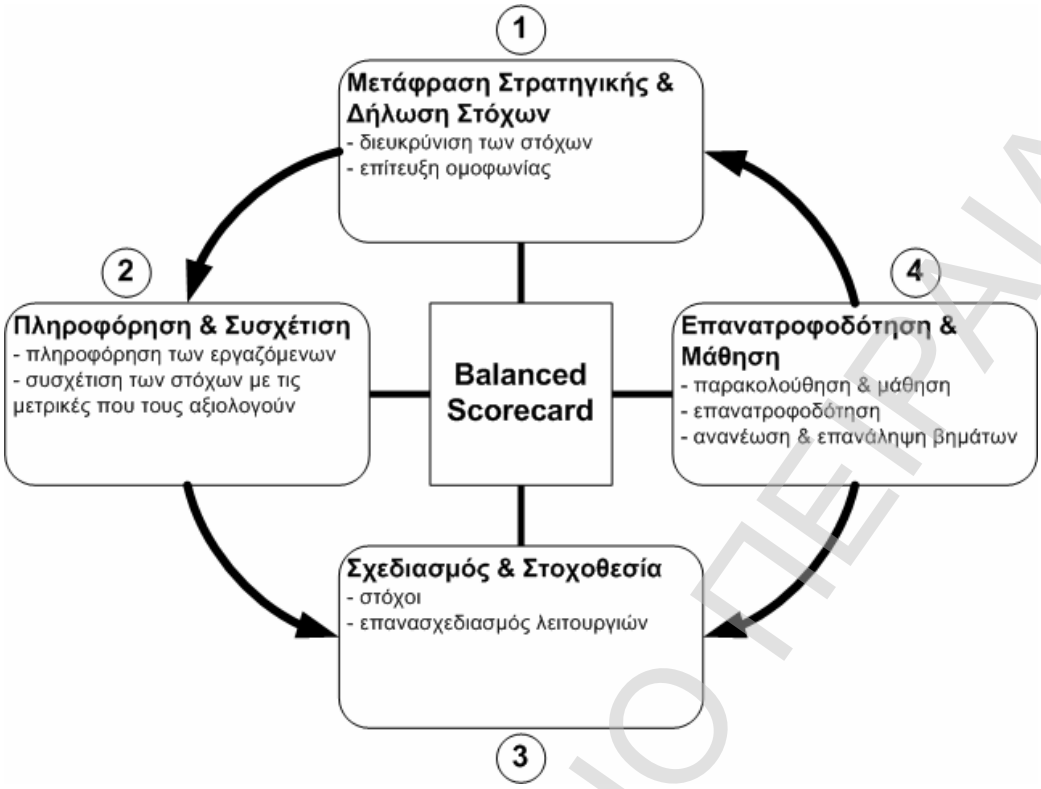
Η έννοια των εργαζόμενων σχετίζεται με θέματα όπως η ικανοποίηση των εργαζόμενων, η διατήρηση τους, η εκπαίδευση τους και η ανάπτυξη των δεξιοτήτων τους. Η έννοια των πληροφοριακών συστημάτων σχετίζεται με θέματα όπως άμεση διαθεσιμότητα των πληροφοριών, ενημέρωση για τους κρίσιμους πελάτες για την λήψη αποφάσεων γρήγορα και άμεσα. Τέλος τρόπος λειτουργίας του οργανισμού σχετίζεται με την ευθυγράμμιση των εργαζόμενων με τις λειτουργίες του οργανισμού και την παροχή των μετρικών εκείνων που δείχνουν την βελτίωση των κρίσιμων διαδικασιών του οργανισμού.

Χρήση της Balanced Scorecard σαν σύστημα Διοίκησης

Η BSC υποστηρίζει ότι οι χρηματοοικονομικές και μη-χρηματοοικονομικές μετρικές πρέπει να αποτελούν μέρος ενός γενικού συστήματος πληροφόρησης των εργαζομένων σε ολόκληρο τον οργανισμού. Οι στόχοι και οι μετρήσεις της BSC προέρχονται από την δήλωση των στόχων και της στρατηγικής του οργανισμού και σκοπός της BSC είναι να μεταφράσει την στρατηγική του οργανισμού σε κατανοητούς στόχους για όλους τους εργαζόμενους. Έτσι πρέπει να υπάρχει μια ισορροπία ανάμεσα στα εξωτερικά δεδομένα που εκφράζουν την στάση και την θέληση των πελατών με τα εσωτερικά δεδομένα του οργανισμού που εκφράζουν τις εσωτερικές λειτουργίες και την μάθηση και ανάπτυξη μέσα στην οργανισμό.

Προκειμένου να υποστηρίξει την στρατηγική του μακροπρόθεσμα ένας οργανισμός πρέπει να χρησιμοποιεί την BSC σαν ένα σύστημα διοίκησης. Ένα τέτοιο σύστημα αποτελείται από τα παρακάτω βήματα:

- μετάφραση της στρατηγικής του οργανισμού και δήλωση των στόχων
- πληροφόρηση και συσχέτιση ανάμεσα στους στόχους και τις μετρικές
- σχεδιασμός και στοχοθεσία
- επανατροφοδότηση και μάθηση



Διάγραμμα 28: Η Balanced Scorecard σαν σύστημα διοίκησης

- **Μετάφραση στρατηγικής και δήλωση των στόχων**

  Η διαδικασία της BSC ξεκινάει με την μετάφραση της στρατηγικής κάθε επιχειρηματικής μονάδας του οργανισμού σε συγκεκριμένους στρατηγικούς στόχους. Έτσι τίθενται συγκεκριμένοι Χρηματοοικονομικοί στόχοι που πρέπει να επιτύχει η επιχειρηματική μονάδα, καθώς και στόχοι που σχετίζονται με συγκεκριμένους Πελάτες ή τμήματα της αγοράς στα οποία δραστηριοποιείται ο οργανισμός. Μέσα από τους στόχους που τίθενται ο οργανισμός μπορεί να καθορίσει τις μετρικές που θα χρησιμοποιηθούν για να αξιολογήσουν τις Εσωτερικές Λειτουργίες του. Συγκεκριμένα καθορίζονται και αξιολογούνται οι λειτουργίες του οργανισμού, που θα επιτύχουν υψηλή απόδοση για τους πελάτες και για τα χρηματοοικονομικά αποτελέσματα του οργανισμού. Στη συνέχεια μέσα από την αξιολόγηση των εσωτερικών λειτουργιών του οργανισμού θα φανερωθούν οι απαιτήσεις για Μάθηση και Ανάπτυξη (που σχετίζονται με κατάρτιση και μάθηση των εργαζομένων, βελτίωση των πληροφοριακών συστημάτων και αλλαγή των εσωτερικών διαδικασιών.

  Μέσα από το πρώτο βήμα της διαδικασίας της BSC γίνεται προσπάθεια να διευκρινιστούν οι στόχοι και η στρατηγική του οργανισμού. Προκειμένου όμως να συνεχιστεί σωστά η διαδικασία, δεδομένου του ότι εμπλέκονται πολλά πρόσωπα – διευθυντές τμημάτων, υπεύθυνοι της BSC κλπ. – πρέπει να υπάρχει ομοφωνία στους στόχους που τίθενται έτσι ώστε στη συνέχεια όλοι οι εργαζόμενοι να ενημερωθούν με την σειρά τους για τους στόχους, που πρέπει να επιτύχουν.

- **Πληροφόρηση και συσχέτιση στόχων και μετρικών**

  Στο επόμενο βήμα της BSC όλοι οι εργαζόμενοι του οργανισμού πρέπει να ενημερωθούν για τους στόχους που έχουν καθοριστεί προκειμένου να επιτύχει η στρατηγική του οργανισμού. Οι εργαζόμενοι ενημερώνονται για τις μετρικές οι οποίες

θα χρησιμοποιηθούν για να αξιολογήσουν τους στόχους που θα πρέπει να επιτύχουν. Αφού κάθε υπάλληλος πληροφορηθεί για τους γενικούς στόχους του οργανισμού και για τις μετρικές που θα χρησιμοποιηθούν για να αξιολογήσουν την εργασία τους στη συνέχεια μπορούν να τεθούν συγκεκριμένοι στόχοι για κάθε επιχειρηματική μονάδα ξεχωριστά.

Στόχος του βήματος αυτού είναι κάθε εργαζόμενος του οργανισμού να έχει κατανοήσει τους στόχους του οργανισμού, την στρατηγική που θα χρησιμοποιηθεί για την επίτευξη των στόχων αυτών καθώς και τις μετρικές που θα χρησιμοποιηθούν για να αξιολογήσουν τους στόχους αυτούς.

- **Σχεδιασμός και στοχοθεσία**

  Το επόμενο βήμα της BSC σχετίζεται με τον καθορισμό των μακροπρόθεσμο στόχων (σε ένα πλάνο 3 με 5 ετών) που εάν επιτευχθούν θα σημαίνουν και την αλλαγή του οργανισμού. Για να επιτύχει τους χρηματοοικονομικούς στόχους του ο οργανισμός πρέπει να θέσει συγκεκριμένους στόχους για τους πελάτες του, τις εσωτερικές λειτουργίες του και για την μάθηση και ανάπτυξη του. Αφού έχουν τεθεί ο στόχοι γίνεται προσπάθεια συσχέτισης των λειτουργιών του οργανισμού με τους στόχους αυτούς και όπου απαιτείται οι λειτουργίες αυτές σχεδιάζονται από την αρχή (reengineering).

- **Επανατροφοδότηση και μάθηση**

  Το τελευταίο βήμα της BSC σχετίζεται με την δυνατότητα μάθησης της διοίκησης. Μέσα από ένα σύστημα επανατροφοδότησης η διοίκηση ελέγχει την πορεία της εφαρμογής της στρατηγικής του οργανισμού και όπου αυτό απαιτείται γίνονται αλλαγές στην στρατηγική.

  Στην περίπτωση που έχουν γίνει σημαντικές αλλαγές στην στρατηγική του οργανισμού τα τρία πρώτα βήματα της διαδικασίας της BSC επαναλαμβάνονται.

Πλεονεκτήματα της Balanced Scorecard

Τα πλεονεκτήματα της BSC, μέσα από την χρήση της σε διάφορους οργανισμούς, παρουσιάζονται στη συνέχεια:

- Αποσαφήνιση του οράματος και της στρατηγικής του οργανισμού
- Αναγνώριση των στρατηγικών στόχων και η συσχέτιση τους με τις μετρικές, που τους αξιολογούν
- Παροχή τρόπου αξιολόγησης της πορείας του οργανισμού
- Έλεγχος των επιχειρηματικών λειτουργιών και δυνατότητα επαναπληροφόρησης (feedback) και μάθησης (learning)
- Ευθυγράμμιση των προσωπικών στόχων των εργαζόμενων με τους στόχους του οργανισμού
- Μακροπρόθεσμος προσανατολισμός της στρατηγικής για την εξασφάλιση βιωσιμότητας και κερδοφορίας για τον οργανισμό

Κεφάλαιο 7. Επίλυση Προβλήματος

Στο κεφάλαιο αυτό θα παρουσιάσουμε την λύση που προτείνουμε για την αντιμετώπιση του προβλήματος της ΒΙΟΣΕΡ. Το πρόβλημα της εταιρείας εστιάζεται σε δύο ζητήματα. Το πρώτο ζήτημα αναφέρεται στην κατασκευή μιας Διαδικασίας Ελέγχου της διοίκησης του οργανισμού για τον εσωτερικό έλεγχο με την βοήθεια της Balanced Scorecard. Το δεύτερο ζήτημα αναφέρεται τόσο στην κατασκευή μιας εφαρμογής που να αντιμετωπίζει κάποια από τα θέματα του πρώτου ζητήματος αλλά και στην κατασκευή μια διαδικασίας ανάπτυξης αντικειμενοστραφών εφαρμογών με την βοήθεια της UML.

Στο πρώτο μέρος του κεφαλαίου θα παρουσιάσουμε ένα Θεωρητικό Μοντέλο που προτείνουμε για την επίλυση του προβλήματος της διαδικασίας ελέγχου της διοίκησης του οργανισμού. Το θεωρητικό μοντέλο αποτελείται από μια συγκεκριμένη διαδικασία ελέγχου καθώς και τα συγκεκριμένα πληροφοριακά συστήματα τα οποία πρέπει να αναπτυχθούν σε ένα οργανισμό για να αντιμετωπίσει το πρόβλημα του ελέγχου διοίκησης.

Στο δεύτερο μέρος του κεφαλαίου θα παρουσιάσουμε την διαδικασία ανάπτυξης μιας αντικειμενοστραφούς εφαρμογής η οποία υλοποιεί κάποιες από τις ανάγκες του θεωρητικού μοντέλου του πρώτου μέρους. Η διαδικασία ανάπτυξης της εφαρμογής είναι μια αντικειμενοστραφής διαδικασία και χρησιμοποιεί την UML για την κατασκευή του μοντέλου του συστήματος, την καταγραφή των απαιτήσεων και την ανάλυση του συστήματος.

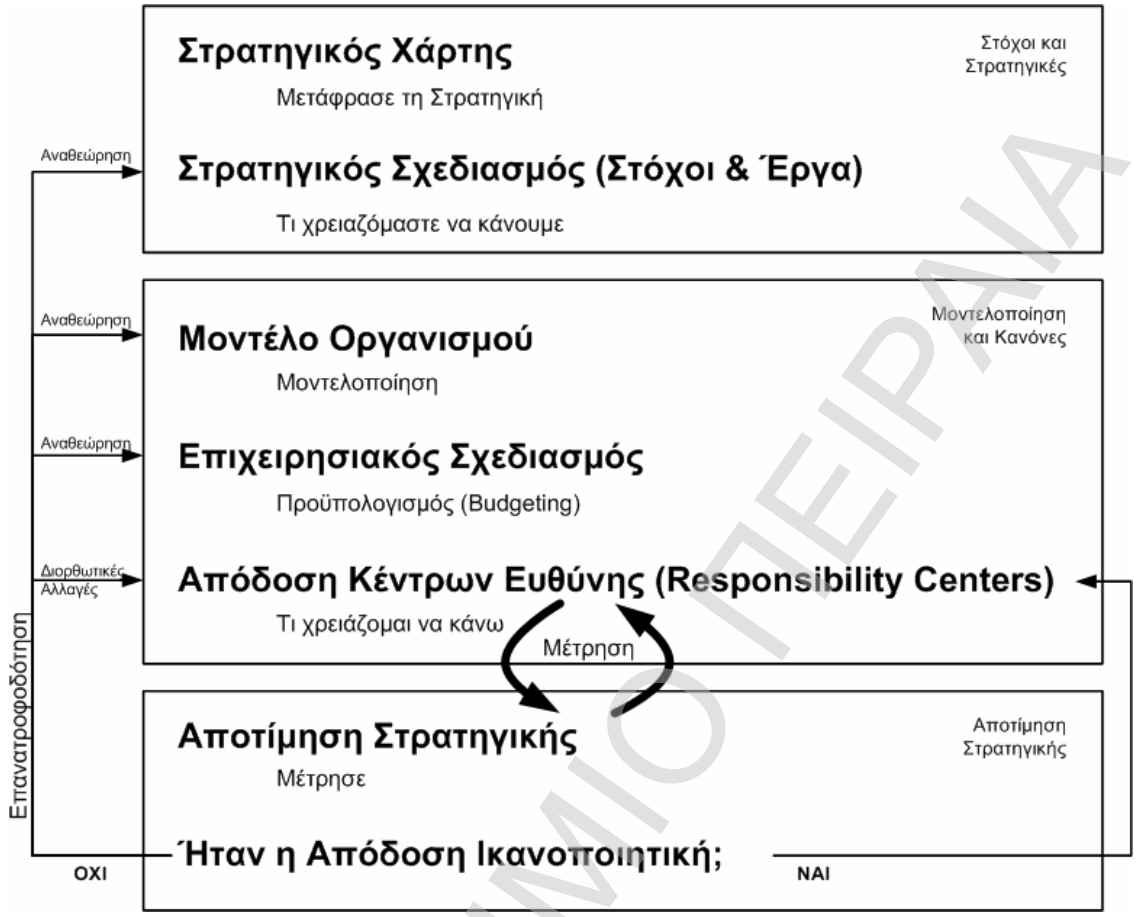
Θεωρητικό Μοντέλο Διαδικασίας Ελέγχου Διοίκησης

Για να αντιμετωπίσουμε το πρώτο ζήτημα του προβλήματος της κατασκευής ενός Συστήματος Ελέγχου Διοίκησης της ΒΙΟΣΕΡ προτείνουμε την παρακάτω διαδικασία ελέγχου διοίκησης που είναι ανάλογη με την διαδικασία ελέγχου (Formal Control Process) που παρουσιάστηκε στην παρουσίαση της θεωρίας αναφορικά με τα Συστήματα Ελέγχου Διοίκησης (Management Control Systems)[10].

Τα βήματα της διαδικασίας Ελέγχου Διοίκησης που προτείνουμε είναι τα παρακάτω:

1. Δημιουργία Στρατηγικού Χάρτη
2. Στρατηγικός Σχεδιασμός (Στόχοι και Έργα)
3. Μοντελοποίηση του Οργανισμού
4. Επιχειρησιακός Σχεδιασμός (Σύστημα Προϋπολογισμού)
5. Απόδοση Κέντρων Ευθύνης
6. Αποτίμηση Στρατηγικής – Έλεγχος Απόδοσης

[10] Βλέπε Κεφάλαιο 6 «Management Control Systems», σελίδα 72



Διάγραμμα 29: Προτεινόμενη Διαδικασία Ελέγχου Διοίκησης

Η διαδικασία ελέγχου αποτελείται από τρία ξεχωριστά επίπεδα. Τους Στόχους και τις Στρατηγικές, την Μοντελοποίηση και τους Κανόνες και την Αποτίμηση της Στρατηγικής. Η διαδικασία ελέγχου είναι μια επαναληπτική διαδικασία που έχει σαν στόχο τον ικανοποιητικό και αποδοτικό έλεγχο της απόδοσης του οργανισμού. Ουσιαστικά αποτελεί ένα εργαλείο εσωτερικού ελέγχου της απόδοσης του οργανισμού και της επίτευξης της στρατηγικής και των στόχων που έχουν τεθεί. Τα βήματα της διαδικασίας παρουσιάζονται στη συνέχεια.

Στρατηγικοί Χάρτες

Κάθε επιχείρηση έχει μια διαφορετική θεώρηση για το τι είναι στρατηγική. Άλλοι την συσχετίζουν με τα χρηματοοικονομικά πλάνα για έσοδα και αύξηση κερδών. Άλλοι με παραγωγή προϊόντων και υπηρεσιών με εστίαση σε συγκεκριμένες ομάδες πελατών, με την ποιότητα ή με την μάθηση και εκπαίδευση των άυλων αγαθών του οργανισμού – τους ανθρώπους. Βλέπουμε ότι δεν υπάρχει μια καθολική και κατανοητή παρουσίαση για το τι είναι πραγματικά η στρατηγική. Χωρίς μια ξεκάθαρη και κατανοητή περιγραφή της στρατηγικής είναι πολύ δύσκολο για τους διευθυντές ενός οργανισμού να μεταφέρουν τους στόχους και να προσπαθήσουν να παρουσιάσουν την στρατηγική στους ίδιους τους εργαζομένους τους.

Η Balanced Scorecard (BSC) αποτελεί ένα εργαλείο διοίκησης και επιτρέπει στον οργανισμό να αναγνωρίσει κρίσιμα άυλα κεφάλαια (άνθρωποι, πληροφορίες και

κουλτούρα/τρόπος διοίκησης) που ανήκουν σε αυτόν. Είναι ένα εργαλείο που επιτρέπει σε ένα οργανισμό να εφαρμόσει την στρατηγική του και παρέχει μια γλώσσα επικοινωνίας έτσι ώστε οι υπεύθυνοι σε ένα οργανισμό να μπορούν να συζητήσουν και να μεταφέρουν τις οδηγίες και τις κατευθύνσεις για τον οργανισμό. Οι στρατηγικές μετρικές που χρησιμοποιούνται (Χρηματοοικονομικά Στοιχεία, Πελάτες, Εσωτερικές Επιχειρηματικές Λειτουργίες και Καινοτομία και Μάθηση) δεν πρέπει μόνο να χρησιμοποιηθούν σαν μετρικές απόδοσης αλλά και σαν ένα σύνολο από δεσμούς αιτίας-αποτελέσματος μεταξύ των στόχων του οργανισμού και των συνιστωσών της BSC.

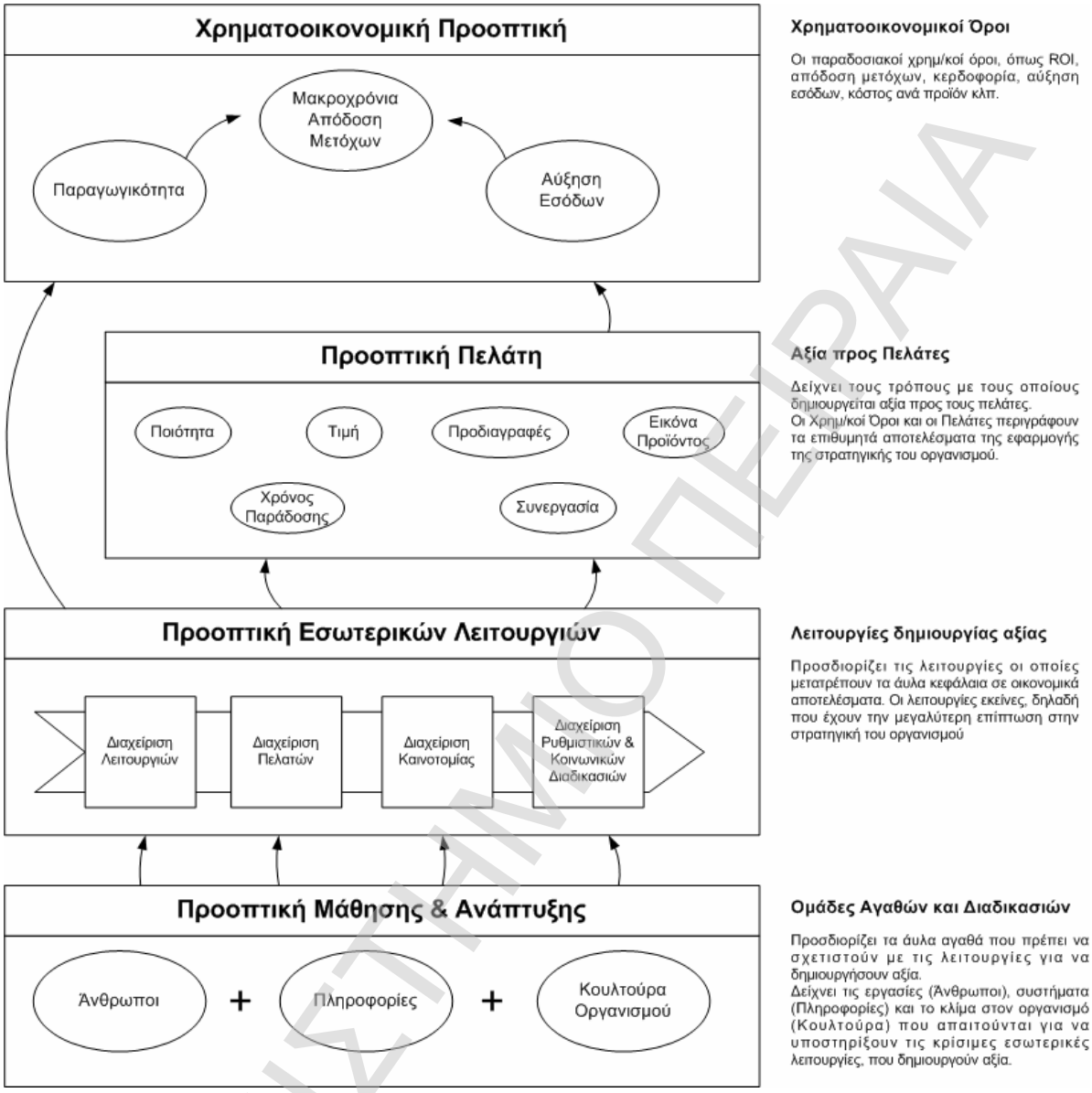
Ένας **Στρατηγικός Χάρτης** είναι μια γενική αναπαράσταση του τρόπου με τον οποίο συνδέονται οι συνιστώσες της BSC και οι στόχοι του οργανισμού [10]. Είναι μια οπτική αναπαράσταση των σχέσεων αιτίας-αποτελέσματος μεταξύ των τμημάτων της στρατηγικής του οργανισμού. Παρόλο που υπάρχουν πολύ τρόποι για να παρουσιαστεί η στρατηγική ενός οργανισμού, ένας Στρατηγικός Χάρτης είναι ένας ενιαίος και συνεπής τρόπος περιγραφής της στρατηγικής ενός οργανισμού έτσι ώστε οι στρατηγικοί στόχοι και οι μετρικές να καθορίζονται και να διαχειρίζονται.

Ο Στρατηγικός Χάρτης χρησιμοποιεί τις 4 Προοπτικές της Balanced Scorecard.

- Χρηματοοικονομικά
- Πελάτες
- Εσωτερικές Λειτουργίες
- Μάθηση και Ανάπτυξη

Ο στρατηγικός χάρτης παρέχει ένα πλαίσιο που δείχνει πως η στρατηγική ενώνει τα άυλα κεφάλαια με τις λειτουργίες που δημιουργούν αξία για τους πελάτες. Δείχνει τον τρόπο με τον οποίο συνδέονται οι τέσσερις συνιστώσες με τους στόχους του οργανισμού με σχέσεις αιτίας-αποτελέσματος.

Οι **Χρηματοοικονομικοί** στόχοι ενός οργανισμού είναι η κερδοφορία και η μακροχρόνια απόδοση των μετόχων του οργανισμού. Η κερδοφορία μπορεί να επιτευχθεί είτε μέσω της αύξησης της αύξησης των εσόδων είτε μέσω της παραγωγικότητας (μεγαλύτερη παραγωγή με μικρότερο κόστος). Η αύξηση εσόδων επιτυγχάνεται μέσω της σχέσης του οργανισμού με υπάρχοντες πελάτες και μέσω της πώλησης νέων προϊόντων. Παραγωγικότητα σημαίνει μείωση των άμεσων και έμμεσων εξόδων (μείωση κόστους) είτε με την βέλτιστη εκμετάλλευση του κεφαλαίου του οργανισμού. Στόχος της στρατηγικής του οργανισμού είναι να βρει μια ισορροπία ανάμεσα στα συχνά αντίθετες λειτουργίες της παραγωγικότητας και της αύξησης εσόδων.



Διάγραμμα 30: Στρατηγικός Χάρτης

Η προοπτική των **Πελατών** περιγράφει πως ο οργανισμός θα δημιουργήσει διαφοροποιημένη και διαρκής αξία σε συγκεκριμένους (targeted) πελάτες. Οι διευθυντές του οργανισμού πρέπει να αναγνωρίσουν τις συγκεκριμένες ομάδες πελατών που πρέπει να εξυπηρετήσουν καθώς και τις μετρικές που αξιολογούν την απόδοση του οργανισμού στην εξυπηρέτηση των πελατών. Οι μετρικές που προσδίδουν διαφοροποίηση στην εξυπηρέτηση των πελατών είναι η τιμή, η ποιότητα, η εικόνα και οι προδιαγραφές του προϊόντος, ο χρόνος παράδοσης και συνεργασία με τους πελάτες. Οι Χρηματοοικονομικοί όροι και οι Πελάτες δείχνουν τα επιθυμητά αποτελέσματα από την εφαρμογή της στρατηγικής του οργανισμού. Οι μετρικές αυτές δίνουν την δυνατότητα στους υπευθύνους του οργανισμού να μεταφράσουν την στρατηγική σε συγκεκριμένους στόχους τους οποίους οι εργαζόμενοι καταλαβαίνουν και δουλεύουν για την επίτευξή τους.

Οι στόχοι στις επόμενες δύο προοπτικές, Εσωτερικές Λειτουργίες και Μάθηση και Ανάπτυξη περιγράφουν τον τρόπο με τον οποίο θα υλοποιηθεί η στρατηγική του οργανισμού. Ο οργανισμός διευθύνει τις εσωτερικές του λειτουργίες και με την ανάπτυξη

των Ανθρώπων, των Πληροφοριών και του ίδιου του Οργανισμού είναι σε θέση να επιτύχει τους στόχους της στρατηγικής με τελικό αποτέλεσμα την κερδοφορία.

Συγκεκριμένα, οι **Εσωτερικές Λειτουργίες** του οργανισμού μπορούν να χωριστούν σε τέσσερις κατηγορίες:

- Λειτουργίες διαχείρισης εργασιών: οι βασικές, καθημερινές λειτουργίες παραγωγής προϊόντων ή υπηρεσιών και παράδοσης τους στους πελάτες.
- Λειτουργίες διαχείρισης πελατών: οι λειτουργίες διατήρησης και ανάπτυξης των σχέσεων με τους πελάτες.
- Λειτουργίες καινοτομίας: οι λειτουργίες παραγωγής νέων προϊόντων ή υπηρεσιών καθώς και διείσδυσης σε καινούργιες αγορές.
- Ρυθμιστικές και κοινωνικές λειτουργίες: οι λειτουργίες που σχετίζονται με την συνεργασία με τοπικές κοινωνίες και κυβερνήσεις για παραγωγή και πώληση.

Οι διευθυντές του οργανισμού πρέπει να αναγνωρίσουν τις λειτουργίες εκείνες που είναι οι πλέον σημαντικές και κρίσιμες για την στρατηγική. Η πραγματική αξία της διοίκησης ενός οργανισμού φαίνεται στην ικανότητα της αναγνώρισης και αριστοποίησης των λειτουργιών που είναι σημαντικές για την παραγωγή αξίας στους πελάτες. Οι λειτουργίες αυτές ανήκουν και στις τέσσερις παραπάνω κατηγορίες.

Τέλος, η προοπτική της **Μάθησης και Ανάπτυξης** περιγράφει τα άυλα κεφάλαια του οργανισμού που δεν είναι άλλα από τους Ανθρώπους, τις Πληροφορίες και ο τρόπος λειτουργίας του ίδιου του Οργανισμού (Κουλτούρα). Ο Στρατηγικός Χάρτης δείχνει τους συγκεκριμένους ανθρώπους, συστήματα πληροφοριών και κλίματος στον οργανισμό που είναι απαραίτητα για την στρατηγική του οργανισμού.

Στρατηγικός Σχεδιασμός (Στόχοι και Έργα)

Ο στρατηγικός χάρτης περιγράφει την λογική της στρατηγικής δείχνοντας τους στόχους των κρίσιμων λειτουργιών που δημιουργούν αξία καθώς και τα απαιτούμενα άυλα κεφάλαια του οργανισμού. Η Balanced Scorecard μεταφράζει τους στόχους του στρατηγικού χάρτη σε μετρικές και συγκεκριμένους στόχους. Ο στρατηγικός σχεδιασμός σχετίζεται με τον καθορισμό των συγκεκριμένων Στρατηγικών Στόχων (Objectives) και των Στρατηγικών Έργων (Initiatives) των προγραμμάτων, δηλαδή που πρέπει να αναπτυχθούν για την επίτευξη των στόχων. Για κάθε μετρική της Balanced Scorecard οι διευθυντές του οργανισμού πρέπει να αναγνωρίσουν τα στρατηγικά έργα που απαιτούνται για την επίτευξη του στρατηγικού στόχου.

Τα πλάνα που καθορίζουν τους πόρους που απαιτούνται για τα στρατηγικά έργα πρέπει να είναι ευθυγραμμισμένα με τις κρίσιμες λειτουργίες του οργανισμού και πρέπει να θεωρούνται σαν ένα ενιαίο σύνολο από επενδύσεις στον οργανισμό παρά σαν ξεχωριστά projects. Συνήθως οι στρατηγικοί στόχοι και τα στρατηγικά έργα σχεδιάζονται στον οργανισμό βασισμένα σε ένα πλάνο διάρκειας από 3 έως 5 έτη.

Στην συνέχεια παρουσιάζουμε τα επόμενα βήματα της διαδικασίας ελέγχου που σχετίζονται με τον τρόπο με τον οποίο υλοποιούνται οι στρατηγικοί στόχοι στον οργανισμό. Τα βήματα αυτά σχετίζονται με την Μοντελοποίηση και την εφαρμογή συγκεκριμένων Κανόνων.

Μοντελοποίηση του Οργανισμού

Στο βήμα αυτό δημιουργείται ένα μοντέλο του οργανισμού δομείται δηλαδή ο οργανισμός για να μπορέσουμε να διαχειριστούμε καλύτερα τον έλεγχο του. Προσπαθούμε να εντοπίσουμε τις οντότητες στις οποίες μπορούμε να δομήσουμε τον οργανισμό. Ύστερα από συζητήσεις με τους υπεύθυνους της ΒΙΟΣΕΡ και ανάλυση των δεδομένων, που συγκεντρώσαμε καταλήξαμε στο συμπέρασμα ότι ο οργανισμός μπορεί να δομηθεί με την βοήθεια των παρακάτω οντοτήτων:

- **Τύποι**: είναι οι γενικές οντότητες του οργανισμού όπως για παράδειγμα οι Πελάτες, Προμηθευτές, Είδη, Έξοδα, Έσοδα κλπ. Είναι απαραίτητο να δημιουργήσουμε τους Τύπους για τις οντότητες που υπάρχουν στον οργανισμό για να μπορούμε να τις διαχειριστούμε μέσω των εφαρμογών του οργανισμού.
- **Δομές**: είναι οι βασικότερες οντότητες μέσα στον οργανισμό οι οποίες αναπαριστούν την δόμηση του. Συγκεκριμένα, μια Δομή είναι μια ιεραρχική παρουσίαση ενός Τύπου (για παράδειγμα, έχουμε μια Δομή Πελατών, Εξόδων, Ειδών κλπ.). Ιεραρχική παρουσίαση σημαίνει ότι δημιουργούμε ένα δέντρο δομής με πολλά επίπεδα,τα οποία καταλήγουν στο τελευταίο επίπεδο που αποτελεί και το στοιχείο που θέλουμε να δούμε (για παράδειγμα ένας Πελάτης, ένα Είδος ή ένα συγκεκριμένο στοιχείο Εξόδων). Στόχος των Δομών είναι η αναπαράσταση και διαχείριση με την ίδια φιλοσοφία και λογική οποιασδήποτε οντότητας μέσα στον οργανισμό.

Με την βοήθεια των Δομών μπορούμε να δημιουργήσουμε ένα μοντέλο του οργανισμού δηλαδή να αναπαραστήσουμε οποιαδήποτε οντότητα του οργανισμού με την βοήθεια ενός δέντρου Δομής. Το βασικό πλεονέκτημα των Δομών είναι ότι μπορούμε να παραστήσουμε την ίδια πληροφορία (που είναι ο Τύπος) – για παράδειγμα στοιχεία Πελατών ή Εσόδων – με διαφορετικό τρόπο χρησιμοποιώντας κάθε φορά μια διαφορετική δομή. Στην συνέχεια για οποιαδήποτε εργασία θέλουμε να πραγματοποιήσουμε για ένα στοιχείο του οργανισμού για παράδειγμα καταχώρηση εξόδων, στοιχεία πωλήσεων σε πελάτες, αποθέματα ειδών κλπ. χρησιμοποιούμε μια αντίστοιχη Δομή και καταχωρούμε τα κατάλληλα δεδομένα.

Επίσης, εισαγάγουμε και την έννοια των Κέντρων Ευθύνης στον οργανισμό τα οποία δεν είναι τίποτα άλλο από τα διάφορα τμήματα του οργανισμού. Η έννοια της ευθύνης στα κέντρα αυτά αφορά τους στόχους του κάθε κέντρου (όσων αφορά πωλήσεις, έξοδα, έσοδα κλπ. ανάλογα με τον τύπο του κάθε κέντρου) οι οποίοι καταχωρούνται σε ένα σύστημα προϋπολογισμού. Η καταχώρηση των στόχων γίνεται στο παρακάτω βήμα της διαδικασίας.

Επιχειρησιακός Σχεδιασμός (Εφαρμογή Προϋπολογισμού)

Στο βήμα αυτό χρησιμοποιούμε ένα σύστημα Προϋπολογισμού για να καταχωρήσουμε τους στόχους για κάποια τμήματα του οργανισμού προκειμένου να μεταφέρουμε το μακροχρόνιο στρατηγικό πλάνο του οργανισμού στο ετήσιο πλάνο του Προϋπολογισμού (Βραχυπρόθεσμο πλάνο). Οι στρατηγικοί στόχοι που έχουν καθοριστεί για να επιτευχθεί το μακροπρόθεσμο στρατηγικό πλάνο του οργανισμού (από 3-5 έτη) μεταφέρονται με την βοήθεια της εφαρμογής του Προϋπολογισμού σε ένα ετήσιο βραχυπρόθεσμο πλάνο που εκφράζεται με τον ίδιο τον Προϋπολογισμό. Για κάθε Κέντρο Ευθύνης του οργανισμού ορίζονται οι ετήσιοι στόχοι και καταχωρούνται στο σύστημα του Προϋπολογισμού. Σκοπός του επόμενου βήματος είναι να ανακοινωθούν οι στόχοι αυτοί στους υπευθύνους των

Κέντρων Ευθύνης και αυτοί με την σειρά τους να ανακοινώσουν τους στόχους αυτούς στους υπόλοιπους ενδιαφερόμενους.

Απόδοση Κέντρων Ευθύνης

Σε αυτό το βήμα, μοιράζουμε υπευθυνότητες (νούμερα) στα διάφορα Κέντρα Ευθύνης του οργανισμού. Τα νούμερα (στόχοι) που έχουν καταχωρηθεί στο σύστημα του Προϋπολογισμού και μπορούν να θεωρηθούν σαν τους Κανόνες που ελέγχουν την υλοποίηση των στόχων μοιράζονται στους υπευθύνους των Κέντρων Ευθύνης του οργανισμού. Η πληροφορία μέσα από το σύστημα του Προϋπολογισμού θα μεταδοθεί στην συνέχεια σε ολόκληρο τον κέντρο ευθύνης και θα γίνει ενημέρωση των υπολοίπων μελών για τους στόχους του κέντρου στο οποίο ανήκουν. Η καταχώρηση των δεδομένων και η μετάδοση τους πραγματοποιείται με την βοήθεια των Δομών οι οποίες δημιουργούνται για τις αντίστοιχες περιπτώσεις. Μέσα από την δόμηση του οργανισμού έχουν καθοριστεί οι Δομές, που σχετίζονται με τους υπευθύνους και τα μέλη των Κέντρων Ευθύνης. Στην συνέχεια, τα νούμερα αυτά μεταφέρονται σε κάθε στοιχείο της Δομής και πραγματοποιείται η εκχώρηση της υπευθυνότητας σε κάθε Κέντρο Ευθύνης.

Δεν είναι απαραίτητο να γίνει η ενημέρωση αυτή μόνο μέσα από κάποια εφαρμογή αλλά μπορεί να γίνει μέσα από συναντήσεις και με την έκδοση των αντίστοιχων αναφορών. Αφού γίνει η πληροφόρηση κάθε ενδιαφερόμενου για τους στόχους του στην συνέχεια είναι απαραίτητο να γίνει καταχώρηση στο σύστημα των στόχων αυτών για κάθε Δομή και κάθε Κέντρο Ευθύνης. Ο λόγος που καθιστά απαραίτητη την καταχώρηση των στόχων στο σύστημα είναι για να γίνει στην συνέχεια ο έλεγχος και η αποτίμηση της απόδοσης κάθε Κέντρου Ευθύνης για να γίνουν οι απαραίτητες διορθωτικές κινήσεις.

Αποτίμηση Στρατηγικής – Έλεγχος Απόδοσης

Στο τελευταίο βήμα της διαδικασίας γίνεται η αποτίμηση της στρατηγικής που έχει ακολουθηθεί από τον οργανισμό και γίνεται έλεγχος της απόδοσης των Κέντρων Ευθύνης του οργανισμού. Η αποτίμηση και ο έλεγχος της επίδοσης των Κέντρων Ευθύνης γίνεται συνήθως ανα μικρά χρονικά διαστήματα, για παράδειγμα κάθε τρεις μήνες. Συγκρίνεται η πραγματική απόδοση των Κέντρων Ευθύνης με τους στόχους, που έχουν καθοριστεί τόσο στο βήμα της κατάρτισης του Προϋπολογισμού όσο και στο βήμα του στρατηγικού σχεδιασμού. Εάν η απόδοση των Κέντρων Ευθύνης είναι ικανοποιητική δεν γίνεται καμία διορθωτική κίνηση αλλά η διαδικασία συνεχίζεται πάλι με τον έλεγχο της απόδοσης των Κέντρων Ευθύνης προκειμένου να εντοπιστούν πιθανές αποκλίσεις από τους στόχους.

Εάν η απόδοση των Κέντρων Ευθύνης δεν είναι ικανοποιητική ξεκινάει η διαδικασία αλλαγών και αναθεωρήσεων προκειμένου να γίνουν τυχόν διορθωτικές κινήσεις. Η διαδικασία των αλλαγών γίνεται από κάτω προς τα πάνω. Αρχικά γίνονται κάποιοι έλεγχοι στην λειτουργία των Κέντρων Ευθύνης προκειμένου να εντοπιστούν πιθανά προβλήματα και να γίνουν κάποιες διορθωτικές κινήσεις στην λειτουργία κάποιων από τα κέντρα αυτά. Γίνεται πάλι έλεγχος της απόδοσης των Κέντρων Ευθύνης και στην περίπτωση που δεν είναι ικανοποιητική η απόδοση τους προχωράμε στο επόμενο επίπεδο αναθεωρήσεων αυτή τη φορά στο βήμα της κατάρτισης του Προϋπολογισμού. Το βήμα της αναθεώρησης του Προϋπολογισμού γίνεται και αυτό βραχυπρόθεσμα συνήθως μετά από μερικούς μήνες.

Στην αναθεώρηση του προϋπολογισμού εντοπίζονται πιθανά λάθη στα νούμερα που έχουν καταχωρηθεί ή κάποια λάθη στην καταχώρηση των βραχυπρόθεσμων στόχων για τα

Κέντρα Ευθύνης. Αφού γίνει η αναθεώρηση του Προϋπολογισμού συνεχίζονται τα επόμενα βήματα της διαδικασίας, δηλαδή το μοίρασμα της ευθύνης στους υπευθύνους των Κέντρων Ευθύνης και ο έλεγχος της απόδοσης τους. Εάν και πάλι δεν είναι ικανοποιητική η απόδοση των Κέντρων Ευθύνης, το επόμενο βήμα είναι η αναθεώρηση στο μοντέλο του οργανισμού.

Με τις πιθανές αναθεωρήσεις στο μοντέλο του οργανισμού επιχειρείται να εντοπιστούν λάθη στις δομές που έχουν δημιουργηθεί ή στα Κέντρα Ευθύνης του οργανισμού. Έτσι μπορεί να απαιτείται να αναθεωρηθεί κάποια συγκεκριμένη δομή ή να πρέπει να προστεθεί ή να αφαιρεθεί κάποιο Κέντρο Ευθύνης στον οργανισμό. Οι διορθωτικές κινήσεις που αναφέραμε παραπάνω αναφέρονται σε μια μικρή χρονική διάρκεια συνήθως από μερικούς μήνες μέχρι το τέλος του έτους.

Στην περίπτωση που τελικά η απόδοση των Κέντρων Ευθύνης δεν είναι ικανοποιητική ίσως να απαιτείται να γίνουν κάποιες αναθεωρήσεις στους Στρατηγικούς Στόχους και στα Στρατηγικά Έργα του οργανισμού. Εντοπίζονται πιθανοί στόχοι ή έργα τα οποία να είναι τελικά ανέφικτα και να πρέπει να αναθεωρηθούν. Επίσης καθορίζονται κάποια νέα έργα ή διορθώνονται κάποια υπάρχοντα έτσι ώστε η νέα διαδικασία ελέγχου να μπορέσει μετά από μερικούς μήνες λειτουργίας να επιφέρει ικανοποιητικά αποτελέσματα.

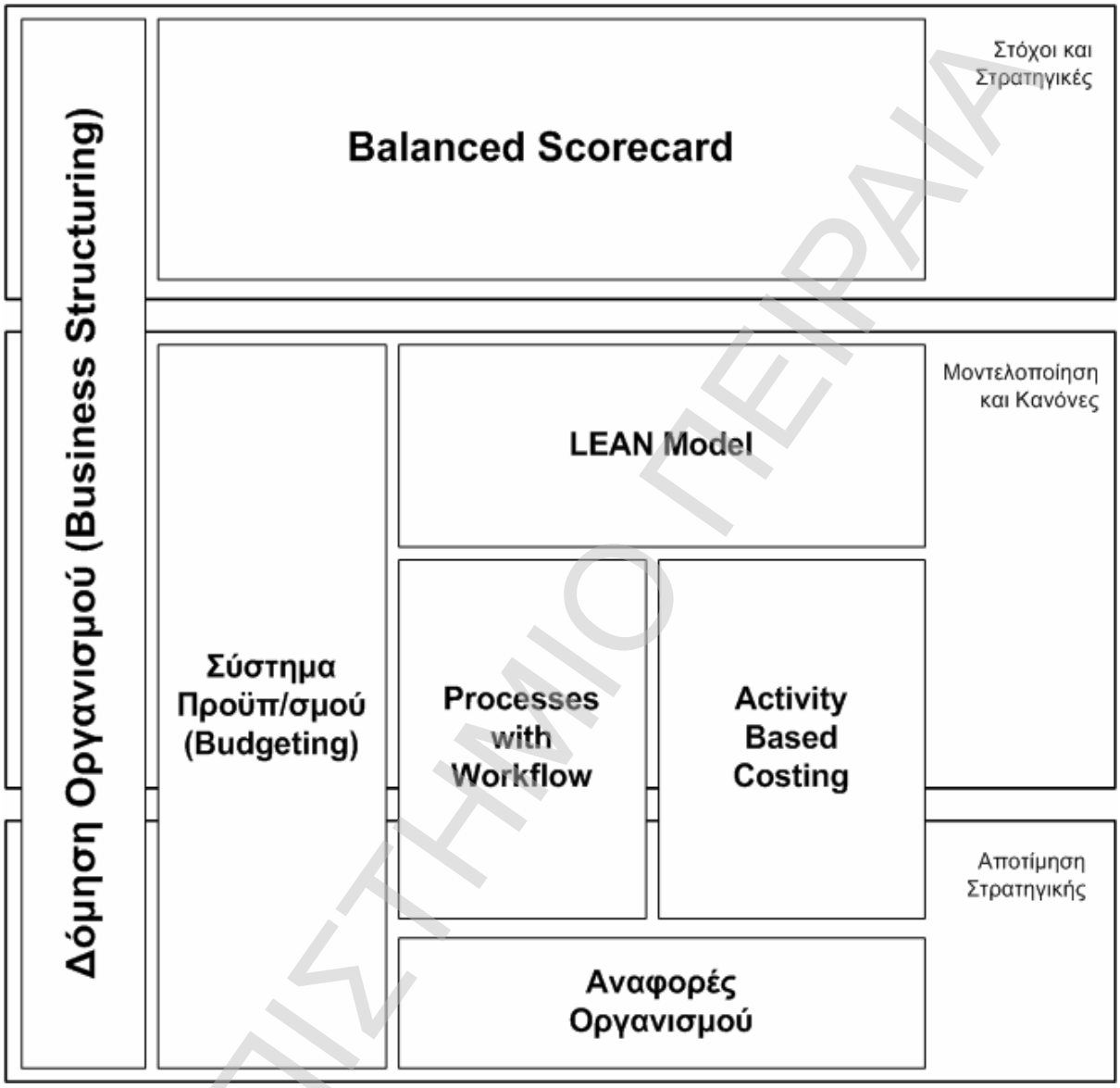
Αφού γίνουν οι απαραίτητες αλλαγές η διαδικασία επαναλαμβάνεται με την αποτίμηση και ελέγχεται πάλι η απόδοση των Κέντρων Ευθύνης. Στην περίπτωση που πάλι δεν είναι ικανοποιητική η απόδοση των Κέντρων Ευθύνης επαναλαμβάνονται τα παραπάνω βήματα των αναθεωρήσεων και των διορθωτικών κινήσεων. Γενικώς η διαδικασία που παρουσιάσαμε είναι μια συνεχής και επαναλαμβανόμενη διαδικασία ελέγχων και αποτίμησης της απόδοσης των Κέντρων Ευθύνης στον οργανισμό. Για την υποστήριξη της διαδικασίας στην ΒΙΟΣΕΡ πρέπει να αναπτυχθούν συγκεκριμένες εφαρμογές στον οργανισμό, οι οποίες θα λειτουργούν και θα υποστηρίζουν τον έλεγχο της διοίκησης στον οργανισμό. Αποτέλεσμα του συνόλου των εφαρμογών που απαιτούνται είναι ένα **Θεωρητικό Μοντέλο** που υποστηρίζει την διαδικασία ελέγχου.

Θεωρητικό Μοντέλο

Αφού παρουσιάσαμε την διαδικασία του ελέγχου διοίκησης του οργανισμού στην συνέχεια θα παρουσιάσουμε το Θεωρητικό Μοντέλο, που θα υποστηρίξει την διαδικασία αυτή. Το μοντέλο αυτό είναι ένα σύνολο εφαρμογών που πρέπει να αναπτυχθούν στον οργανισμό προκειμένου να υποστηρίξουν την διαδικασία του ελέγχου διοίκησης της ΒΙΟΣΕΡ.

Οι εφαρμογές, που απαιτούνται για την υποστήριξη της διαδικασία ελέγχου της ΒΙΟΣΕΡ παρουσιάζεται στο παρακάτω διάγραμμα.

Εφαρμογές Θεωρητικού Μοντέλου



Διάγραμμα 31: Θεωρητικό Μοντέλο Ελέγχου Διοίκησης (Προτεινόμενες Εφαρμογές)

Οι εφαρμογές, που ανήκουν στο παραπάνω μοντέλο είναι οι παρακάτω:

- **Balanced Scorecard**: το σύστημα της Balanced Scorecard ορίζεται στην κορυφή του μοντέλου. Αναλύονται μέσω του Στρατηγικού Χάρτη οι τέσσερις προοπτικές (Χρηματοοικονομικοί Όροι, Πελάτες, Εσωτερικές Λειτουργίες, Μάθηση και Ανάπτυξη) βάσει των οποίων καθορίζονται οι στρατηγικοί στόχοι και τα στρατηγικά έργα, που θα υποστηρίξουν την υλοποίηση της στρατηγικής του οργανισμού.
- **Δόμηση Οργανισμού (Business Structuring)**: οι Δομές, βάσει των οποίων απεικονίζεται ο οργανισμός. Επίσης η δόμηση μπορεί να περιλαμβάνει ένα απλό οργανόγραμμα της εταιρείας. Οι δόμηση του οργανισμού καλύπτει και τα τρία επίπεδα του μοντέλου.

- **Σύστημα Προϋπολογισμού (Budgeting)**: αποτελεί την κλασική εφαρμογή καταχώρησης και διαχείρισης του Προϋπολογισμού του οργανισμού. Καλύπτει το επίπεδο της Μοντελοποίησης και της Αποτίμησης της στρατηγικής.
- **LEAN Model**: το Λιτό μοντέλο είναι μια εφαρμογή η οποία έχει σαν στόχο την απλοποίηση των διαδικασιών σε ένα οργανισμό και την ελαχιστοποίηση του κόστους λειτουργίας τους. Το μοντέλο αυτό χρησιμοποιεί την εφαρμογή διαχείρισης διαδικασιών (Workflow) για την μέτρηση του χρόνου εκτέλεσης και την εφαρμογή Activity Based Costing (ABC) για την μέτρηση του κόστους εκτέλεσης των διαδικασιών.
- **Processes with Workflow**: η εφαρμογή διαχείρισης των διαδικασιών ασχολείται με την μέτρηση του χρόνου εκτέλεσης των διαδικασιών στον οργανισμό.
- **Activity Based Costing**: η εφαρμογή κοστολόγησης βάσει δραστηριοτήτων ασχολείται με την μέτρηση του κόστους εκτέλεσης των δραστηριοτήτων μέσα στον οργανισμό.
- **Αναφορές Οργανισμού**: είναι το σύστημα δημιουργίας και παρουσίασης των αναφορών σε κάθε ενδιαφερόμενο στον οργανισμό. Οι αναφορές χρησιμοποιούνται για την έλεγχο και την εκτίμηση της απόδοσης των Κέντρων Ευθύνης στον οργανισμό.

Παραπάνω παρουσιάσαμε το πρώτο ζήτημα του προβλήματος που αντιμετωπίζει η ΒΙΟΣΕΡ που είναι η δημιουργία ενός θεωρητικού μοντέλου υλοποίησης μιας διαδικασίας ελέγχου διοίκησης του οργανισμού. Αποτέλεσμα της επίλυσης του πρώτου ζητήματος είναι το Θεωρητικό Μοντέλο. Το επόμενο ζήτημα του προβλήματος, που αντιμετωπίζει η ΒΙΟΣΕΡ είναι η παρουσίαση μιας διαδικασίας ανάπτυξης αντικειμενοστραφών εφαρμογών. Η διαδικασία ανάπτυξης θα πρέπει να χρησιμοποιεί την UML για κατασκευή του μοντέλου του συστήματος, την καταγραφή των απαιτήσεων και την ανάλυση του συστήματος.

Στην συνέχεια της παρουσίασης θα δείξουμε την διαδικασία χρήσης της UML. Θα παρουσιάσουμε τα βασικά βήματα της διαδικασίας καθώς επίσης θα δώσουμε ένα πραγματικό παράδειγμα χρήσης της διαδικασίας αυτής. Συγκεκριμένα θα παρουσιάσουμε την διαδικασία ανάπτυξης των εφαρμογών που ανήκουν στο Θεωρητικό Μοντέλο ελέγχου διοίκησης. Οι εφαρμογές που θα αναπτύξουμε με την αντικειμενοστραφή διαδικασία χρήσης της UML είναι:

- Σύστημα Προϋπολογισμού (Budgeting)
- Δόμηση Οργανισμού (Business Structuring)
- Αναφορές Οργανισμού

Να τονίσουμε στο σημείο αυτό ότι στόχος της παρακάτω παρουσίασης δεν είναι η ανάπτυξη ενός ολοκληρωμένου συστήματος που να υλοποιεί τις παρακάτω εφαρμογές αλλά η παρουσίαση χρήσης της αντικειμενοστραφούς διαδικασίας ανάπτυξης εφαρμογών. Το αποτέλεσμα της διαδικασίας που θα παρουσιαστεί δεν θα είναι ένα πλήρως λειτουργικό σύστημα αλλά δίνεται μεγαλύτερη βαρύτητα στα βήματα της διαδικασίας και στην παρουσίαση των διαγραμμάτων της UML που χρησιμοποιούνται.

Διαδικασία χρήσης της UML

Η UML παρέχει συμβολισμούς και κανόνες στους εμπλεκόμενους στην ανάπτυξη εφαρμογών προκειμένου να σχεδιάσουν διάφορα αντικειμενοστραφή μοντέλα. Παρόλα αυτά η UML δεν περιγράφει πώς πρέπει να γίνει η εργασία αυτή, ποια μεθοδολογία δηλαδή πρέπει να χρησιμοποιηθεί για την ανάπτυξη της εφαρμογής. Η UML έχει σχεδιαστεί για να χρησιμοποιηθεί από διάφορες μεθοδολογίες ανάπτυξης εφαρμογών είναι δηλαδή ένα γενικό πρότυπο ανεξάρτητο από συγκεκριμένες διαδικασίες.

Ωστόσο, για να χρησιμοποιηθεί με επιτυχία η UML πρέπει να γίνει χρήση μιας συγκεκριμένης διαδικασίας κυρίως σε περιπτώσεις που συμμετέχει στον σχεδιασμό και την ανάπτυξη της εφαρμογής μια μεγάλη ομάδα ανθρώπων για να υπάρχει ένας κοινός τρόπος επικοινωνίας και μια δυνατότητα μέτρησης της προόδου και βελτίωσης της εργασίας.

Μια διαδικασία περιγράφει τι πρέπει να γίνει, πώς θα γίνει, πότε πρέπει να γίνει και γιατί πρέπει να γίνει με ένα συγκεκριμένο τρόπο. Περιγράφει έναν αριθμό από δραστηριότητες που πρέπει να γίνουν με μια συγκεκριμένη σειρά για να επιτευχθεί ο συγκεκριμένος στόχος. Η ορθή ολοκλήρωση των επιμέρους δραστηριοτήτων της διαδικασίας δηλώνει ότι έχει επιτευχθεί ο στόχος της που στην προκειμένη περίπτωση είναι η ανάπτυξη της εφαρμογής.

Μια διαδικασία ανάπτυξης μιας αντικειμενοστραφούς εφαρμογής με την χρήση της UML πρέπει να έχει συγκεκριμένα χαρακτηριστικά. Η διαδικασία πρέπει να **καθοδηγείται από Use Cases** (use-case-driven), να είναι **επικεντρωμένη σε μια αρχιτεκτονική** (architecture-centric), να είναι **επαναληπτική** (iterative) και **αυξητική** (incremental). Οι όροι αυτοί αναλύονται στη συνέχεια.

Χαρακτηριστικά Διαδικασίας χρήσης της UML

- **Καθοδήγηση από Use Cases**

  Τα Use Cases χρησιμοποιούνται για να περιγράψουν τις λειτουργικές απαιτήσεις του συστήματος, τι πρέπει δηλαδή να κάνει το σύστημα. Τα Use Cases «οδηγούν» την εργασία της δημιουργίας των υπολοίπων διαγραμμάτων και εγγράφων. Τα Use Cases χρησιμοποιούνται σε όλες τις φάσεις της ανάπτυξης της εφαρμογής για την πιστοποίηση ότι οι απαιτήσεις που έχουν καταγραφεί υλοποιούνται στο σύστημα και για περαιτέρω έλεγχο. Το πρώτο χαρακτηριστικό που θα πρέπει να έχει η διαδικασία, που περιγράφουμε είναι ο έλεγχος σε κάθε βήμα της διαδικασίας ότι υλοποιούνται οι απαιτήσεις που έχουν καταγραφεί με την μορφή των Use Cases.

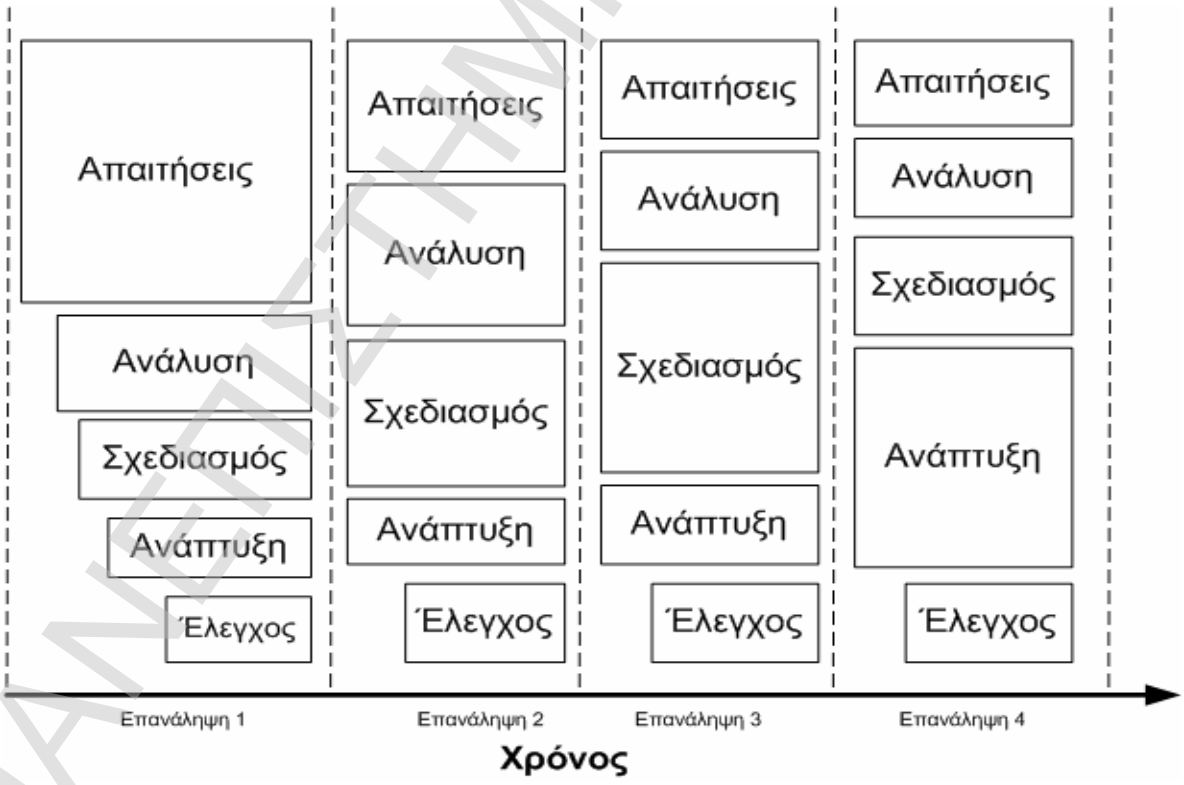
- **Επικέντρωση σε Αρχιτεκτονική**

  Μια διαδικασία που είναι επικεντρωμένη σε μια αρχιτεκτονική σημαίνει ότι από την αρχή της ανάπτυξης της εφαρμογής πρέπει να ληφθεί υπόψη των διαφόρων ατόμων μια σαφώς ορισμένη αρχιτεκτονική του συστήματος. Η UML παρέχει στον αναγνώστη των διαφόρων διαγραμμάτων διάφορες όψεις του συστήματος ανάλογα με τα διαγράμματα που χρησιμοποιούνται. Έτσι από την αρχή της διαδικασίας πρέπει να προσδιοριστεί η βασική αρχιτεκτονική του συστήματος και σε κάθε βήμα η αρχιτεκτονική αυτή να βελτιώνεται μέχρι την ολοκλήρωση της εφαρμογής. Η αρχιτεκτονική αυτή λειτουργεί σαν ένας χάρτης που παρουσιάζει τα διάφορα μέρη του συστήματος, τις συσχετίσεις τους και τους μηχανισμούς επικοινωνίας μεταξύ των μερών αυτών.

Ένα είδος αρχιτεκτονικής σύμφωνα με την οποία έχει σχεδιαστεί το σύστημα είναι αυτή της Αρχιτεκτονικής N-tier. Σύμφωνα με την αρχιτεκτονική αυτή η εφαρμογή αποτελείται από διάφορα επίπεδα κάθε ένα από τα οποία εκτελεί μια συγκεκριμένη λειτουργία και είναι ανεξάρτητο από τα υπόλοιπα επίπεδα. Οι διάφορες όψεις της UML όπως αυτές παρουσιάζονται με τα διάφορα διαγράμματα δίνουν στον αναγνώστη τον τρόπο με τον οποίο έχουν υλοποιηθεί λογικά και φυσικά τα διάφορα επίπεδα της Αρχιτεκτονικής N-tier.

- **Επαναληπτική**

Σε αντίθεση με τις παραδοσιακές, ακολουθιακές μεθοδολογίες ανάπτυξης όπως αυτή του μοντέλου του Καταρράκτη, όπου η έναρξη κάθε βήματος σήμαινε ολοκλήρωση του προηγούμενου βήματος οι διαδικασίες ανάπτυξης εφαρμογών που χρησιμοποιούν την UML είναι επαναληπτικές. Αυτό σημαίνει ότι κάθε βήμα της διαδικασίας δεν είναι ανεξάρτητο από τα προηγούμενα βήματα αλλά επιτρέπεται η επιστροφή σε προηγούμενο βήμα για βελτίωση κάποιων αδυναμιών, τροποποίηση κάποιων παρατηρήσεων και διόρθωση ή βελτίωση κάποιων διαγραμμάτων της UML, όπου βέβαια αυτό απαιτείται. Με τον τρόπο αυτό σε όλη την διαδικασία υπάρχει συνεχής επανατροφοδότηση (feedback) σε όλα τα στάδια της διαδικασίας με τελικό σκοπό την βελτίωση της ίδιας της διαδικασίας και φυσικά της ίδιας της εφαρμογής.



Διάγραμμα 32: Επαναληπτική Διαδικασία

Η επαναληπτική διαδικασία μπορεί να θεωρηθεί σαν ένα σύνολο από επαναλαμβανόμενα βήματα κάθε ένα από τα οποία εξαρτάται από το σημείο στο οποίο βρισκόμαστε και την πρόοδο που έχει επιτευχθεί. Για παράδειγμα, τις απαιτήσεις των χρηστών του συστήματος τις καταγράφουμε στα πρώτα βήματα της διαδικασίας και όσο

προχωράει η διαδικασία γίνονται όλο και λιγότερες αλλαγές σε αυτές. Θα είναι μεγάλο πρόβλημα εάν δεν έχει γίνει σωστή καταγραφή των απαιτήσεων στην αρχή της διαδικασίας και πρέπει να αλλάξουν όταν ολοκληρώνεται το σύστημα. Βέβαια μια καλή αρχιτεκτονική του συστήματος που έχει υιοθετηθεί δεν θα πρέπει να μας απαγορεύει να κάνουμε αλλαγές στις απαιτήσεις των χρηστών αλλά να γίνονται όποιες αλλαγές απαιτούνται με το λιγότερο δυνατό κόστος[11].

- **Αυξητική**

  Κάθε επανάληψη κατά την διάρκεια της ανάπτυξης της εφαρμογής πρέπει να παράγει ένα αποτέλεσμα που πρέπει να ελεγχθεί για να διαπιστωθεί η εξέλιξη του συστήματος. Μια διαδικασία είναι αυξητική όταν σε κάθε επανάληψη βελτιώνονται κάποιες αδυναμίες και παράγεται μια καινούργια έκδοση (version) του συστήματος. Βέβαια η έκδοση του συστήματος δεν σημαίνει ότι το προϊόν είναι έτοιμο για παράδοση αλλά σημαίνει ότι το κάθε βήμα έχει ολοκληρωθεί και έχει ελεγχθεί από τους υπευθύνους γεγονός που σημαίνει ότι πρέπει να υπάρχουν και οι κατάλληλοι μηχανισμοί ελέγχου στο τέλος του κάθε βήματος της επαναληπτικής διαδικασίας.

  Συνήθως όταν αναπτύσσεται ένα αντικειμενοστραφής σύστημα δίνεται βάρος αρχικά στην υλοποίηση των απαιτήσεων του συστήματος (τι πρέπει να κάνει το σύστημα) και στην υλοποίηση των κανόνων και της λογικής της επιχείρησης (τα οποία αποτελούν το Business layer στην αρχιτεκτονική N-Επιπέδων) χωρίς να δίνεται βάρος στο user interface της εφαρμογής (το Presentation layer της αρχιτεκτονικής). Αφού γίνουν οι απαραίτητοι έλεγχοι για την ορθότητα των αποτελεσμάτων η διαδικασία συνεχίζεται στην βελτίωση του συστήματος, προσθέτοντας και την ανάπτυξη των διαφόρων οθονών για την παρουσίαση των αποτελεσμάτων.

  Έτσι σε κάθε βήμα της διαδικασίας η εφαρμογή χτίζεται σιγά-σιγά, βελτιώνοντας και διορθώνοντας τα αντίστοιχα διαγράμματα της UML όπου αυτά χρησιμοποιούνται και προσθέτοντας καινούργιες λειτουργίες στο σύστημα.

Μοντέλο Διαδικασίας

Στην συνέχεια θα παρουσιαστεί το μοντέλο μιας αντικειμενοστραφούς διαδικασίας ανάπτυξης εφαρμογών Η διαδικασία, που παρουσιάζουμε περιλαμβάνει τα παρακάτω βήματα:

- Συλλογή Απαιτήσεων
- Ανάλυση
- Σχεδιασμός
- Ανάπτυξη
- Έλεγχος

[11] Η Αρχιτεκτονική N-tier έχει το μεγάλο πλεονέκτημα του διαχωρισμού των διαφόρων επιπέδων της εφαρμογής. Έτσι είναι δυνατή η αλλαγή σε κάποιο επίπεδο, για παράδειγμα στο Επίπεδο της Παρουσίασης χωρίς να απαιτούνται μεγάλες αλλαγές στα υπόλοιπα επίπεδα ή η αλλαγή του Επιπέδου της Βάσης Δεδομένων χωρίς να επηρεάζεται η λειτουργία της εφαρμογής.

Κάθε βήμα της διαδικασίας παρέχει και ένα συγκεκριμένο αποτέλεσμα, το οποίο περιλαμβάνει ένα αριθμό διαγραμμάτων UML. Στόχος του παραδείγματος της διαδικασίας είναι να παρουσιαστεί η χρήση της UML και συγκεκριμένα ποια διαγράμματα χρησιμοποιούνται σε κάθε βήμα της διαδικασίας. Η διαδικασία που περιγράφουμε είναι επαναληπτική και αυξητική πράγμα που σημαίνει ότι δεν απαγορεύεται η επιστροφή σε ένα προηγούμενο βήμα για να γίνουν οι απαραίτητες αλλαγές και διορθώσεις σε κάποια διάγραμμα της UML ή σε οτιδήποτε άλλο έχει καταγραφεί. Στη συνέχεια παρουσιάζονται τα βήματα της διαδικασίας χρήσης της UML.

Συλλογή Απαιτήσεων

Το βήμα αυτό είναι το πρώτο και το πιο σημαντικό σε ολόκληρη τη διαδικασία. Είναι πολύ σημαντικό να γνωρίζουμε τι θέλουν οι πελάτες από το σύστημα. Η συλλογή των απαιτήσεων είναι ένα είδος «συμφωνίας» μεταξύ των πελατών και των προμηθευτών του συστήματος. Οι πελάτες του συστήματος μπορεί να είναι οι πραγματικοί πελάτες μιας εφαρμογής που πληρώνουν για να την αποκτήσουν, οι χρήστες της εφαρμογής σε ένα οργανισμό ή οποιοσδήποτε άλλος θα χρησιμοποιήσει την εφαρμογή. Η συμφωνία αυτή σχετίζεται με την καταγραφή των απαιτήσεων των χρηστών, τι θέλουν δηλαδή να κάνει το σύστημα. Όταν δεν είναι δυνατή η καταγραφή λεπτομερειών σχετικά με τις λειτουργίες του συστήματος στο βήμα αυτό καταγράφεται απλώς μια ιδέα και μια απλή αναφορά για το τι θα προσφέρει το σύστημα.

Το αποτέλεσμα της συλλογής των απαιτήσεων είναι συνήθως ένα πρώτο μοντέλο της εφαρμογής, στο οποίο παρουσιάζονται:

- **Όραμα (Vision) του συστήματος**: με τον όρο όραμα του συστήματος εννοούμε μια απλή αναφορά για τις βασικές λειτουργίες του συστήματος. Η παρουσίαση του οράματος του συστήματος γίνεται εγγράφως και ύστερα από συζητήσεις και συνεντεύξεις με τους πελάτες του συστήματος.
- **Απαιτήσεις του συστήματος**: καταγράφονται οι απαιτήσεις του συστήματος, τι δηλαδή θέλουμε να κάνει το σύστημα. Οι απαιτήσεις του συστήματος αναπαρίστανται με **Use Case διαγράμματα**, τα οποία δείχνουν τις βασικές λειτουργίες του συστήματος, όπως τις βλέπουν οι εξωτερικοί χρήστες (actors).
- **Παρουσίαση Λειτουργιών**: η ανάλυση των Use Case διαγραμμάτων και η χρήση κάποιων use case σεναρίων οδηγεί στην καταγραφή των βασικών λειτουργιών της εφαρμογής. Η καταγραφή αυτή γίνεται με την χρήση **Activity διαγραμμάτων** που περιγράφουν την υλοποίηση των use cases.
- **Ανάλυση του περιβάλλοντος του πελάτη (domain model)**: μέσα από συνεντεύξεις και συζητήσεις αναλύεται το περιβάλλον (domain) του πελάτη με στόχο να καταγραφούν οι βασικές οντότητες (αντικείμενα) του συστήματος. Η καταγραφή των οντοτήτων γίνεται σε ένα γενικό **Class διάγραμμα** χωρίς μεγάλη λεπτομέρεια αναφορικά με τις συσχετίσεις μεταξύ των αντικειμένων αυτών.

Επαναλαμβάνουμε ότι στο βήμα αυτό δεν απαιτείται μεγάλη λεπτομέρεια για το **πως** θα υλοποιηθούν όλα όσα καταγράφονται στην συλλογή απαιτήσεων αλλά το σύστημα παρουσιάζεται **εξωτερικά** όπως το βλέπουν οι εξωτερικοί χρήστες χωρίς πολλές τεχνικές λεπτομέρειες.

Τέλος, εκτός από τα διαγράμματα της UML δημιουργείται ένα **απλό γλωσσάρι** των βασικών όρων που χρησιμοποιούνται στην εφαρμογή και στα διαγράμματα. Το γλωσσάρι είναι σε αρχική μορφή και κατά την διάρκεια της ανάπτυξης της εφαρμογής εμπλουτίζεται και συμπληρώνεται με καινούργιους όρους.

Ανάλυση

Το επόμενο βήμα της διαδικασίας είναι αυτό της ανάλυσης. Στο βήμα αυτό τα αποτελέσματα της καταγραφής των απαιτήσεων, που έγινε στο προηγούμενο βήμα αναλύονται και βελτιώνονται. Στόχος είναι αναλυθεί το περιβάλλον του προβλήματος έτσι ώστε να εντοπιστούν οι κύριες κλάσεις του «πραγματικού κόσμου». Οι κλάσεις αυτές καθώς και οι σχέσεις που υπάρχουν ανάμεσα τους θα χρησιμοποιηθούν στην εφαρμογή (εφόσον μιλάμε για αντικειμενοστραφή διαδικασία). Συγκεκριμένα, οι λειτουργίες που γίνονται σε αυτό το βήμα είναι οι παρακάτω:

- **Ανάλυση των Use Cases και των Αντικειμένων**: τα Use Case και τα αντικείμενα του προηγούμενου βήματος αναλύονται για την εύρεση των κλάσεων[12] της εφαρμογής και την πρόσθεση λεπτομερειών σε αυτές (χαρακτηριστικά και μέθοδοι στις κλάσεις, συσχετίσεις και κληρονομικότητα ανάμεσα στις κλάσεις κλπ.). Η καταγραφή αντικειμένων και των συσχετίσεων μεταξύ του γίνεται στα Class διαγράμματα, που δείχνουν την στατική επικοινωνία των κλάσεων
- **Εντοπισμός νέων κλάσεων**: εκτός από την βελτίωση των υπαρχόντων κλάσεων νέες κλάσεις ανακαλύπτονται και προστίθενται στο μοντέλο.
- **Καταγραφή της επικοινωνίας μεταξύ των κλάσεων**: καταγράφεται η επικοινωνία μεταξύ των αντικειμένων. Για ορισμένες σημαντικές λειτουργίες της εφαρμογής που πρέπει να παρουσιαστούν καταγράφεται η επικοινωνία των αντικειμένων με την χρήση των αντίστοιχων **Sequence** ή/και **Collaboration** διαγραμμάτων.
- **Ανάλυση των καταστάσεων βασικών αντικειμένων**: με την χρήση **Statechart** διαγραμμάτων αναλύεται και παρουσιάζεται η αλλαγή της κατάστασης των βασικών αντικειμένων της εφαρμογής.

Το αποτέλεσμα της ανάλυσης είναι ένα μοντέλο που περιγράφει το πρόβλημα το οποίο προσπαθεί να διαχειριστεί το σύστημα. Επίσης στο μοντέλο παρουσιάζονται οι βασικές κλάσεις του συστήματος καθώς και οι αντίστοιχες σχέσεις ανάμεσα τους.

Σχεδιασμός

Στο τρίτο βήμα του σχεδιασμού οι εμπλεκόμενοι στην ανάπτυξη της εφαρμογής παίρνουν τα αποτελέσματα της ανάλυσης και προσπαθούν να σχεδιάσουν την εφαρμογή. Οι εμπλεκόμενοι έχοντας μπροστά τους το μοντέλο της ανάλυσης κάνουν τις απαραίτητες διορθώσεις και βελτιώσεις σε ότι έχει καταγραφεί. Σε πολλές περιπτώσεις ανάπτυξης εφαρμογών τα βήματα της Ανάλυσης και του Σχεδιασμού δεν είναι εντελώς αποκομμένα αλλά συνδυάζονται και θεωρούνται σαν ένα ενιαίο βήμα.

[12] Όπως έχουμε τονίσει, οι όροι Αντικείμενα και Κλάσεις είναι ταυτόσημοι. Τα Αντικείμενα είναι ένας γενικότερος όρος, ενώ η Κλάση ουσιαστικά αναφέρεται στην κλάση-οντότητα της αντικειμενοστραφούς εφαρμογής, στην οντότητα δηλαδή που θα δημιουργηθεί με την αντικειμενοστραφή γλώσσα προγραμματισμού που θα χρησιμοποιήσουμε.

Οι βασικές λειτουργίες που γίνονται στο βήμα του σχεδιασμού παρουσιάζονται στη συνέχεια:

- **Βελτίωση των κλάσεων**: οι κλάσεις της εφαρμογής βελτιώνονται με στόχο την λειτουργικότητα τους (επικοινωνία με τις οθόνες της εφαρμογής, επικοινωνία με άλλα συστήματα, επικοινωνία με την Βάση Δεδομένων).
- **Σχεδιασμός user interface**: σχεδιάζονται οι οθόνες παρουσίασης των δεδομένων, οι αναφορές και τα συστήματα επικοινωνίας με άλλα συστήματα. Οι οθόνες σχεδιάζονται με την βοήθεια των Use Case και των Sequence διαγραμμάτων, που έχουν δημιουργηθεί από τα προηγούμενα βήματα της διαδικασίας. Το αποτέλεσμα της σχεδίασης είναι κάποια **πρότυπα σχέδια των οθονών** που παρουσιάζονται στους πελάτες αλλά και τους προγραμματιστές της εφαρμογής για να συζητηθεί η μορφή τους έτσι ώστε να γίνουν απαραίτητες προτάσεις για πιθανές αλλαγές.
- **Σχεδιασμός βιβλιοθηκών**: σχεδιάζονται οι απαραίτητες **βιβλιοθήκες** κλάσεων και οθονών, που θα υποστηρίξουν την αρχιτεκτονική της εφαρμογής.
- **Σχεδιασμός αρχιτεκτονικής**: παραγωγή των σχεδίων εκείνων που θα βοηθήσουν σε τεχνικά θέματα, όπως ασφάλεια, επικοινωνίες και επικοινωνία με την Βάση Δεδομένων. Σχεδιάζεται ο τρόπος φυσικής επικοινωνίας των συστατικών μερών της εφαρμογής καθώς και τα θέματα ασφάλειας. Τα αποτελέσματα του σχεδιασμού αρχιτεκτονικής αναπαρίστανται με **Component** και **Deployment διαγράμματα**.

Ένα πολύ σημαντικό θέμα που δεν πρέπει να παραληφθεί είναι αυτό της **τεκμηρίωσης των αποτελεσμάτων** (διαγραμμάτων, οθονών κλπ.) από τους υπεύθυνους. Τα διαγράμματα που έχουν δημιουργηθεί θα αποτελέσουν την βάση για τα μετέπειτα βήματα της ανάπτυξης και του ελέγχου και η τεκμηρίωση είναι ένας σημαντικός παράγοντας για να υποστηριχθεί η εργασία των προγραμματιστών, που θα αναλάβουν την ανάπτυξη του κώδικα της εφαρμογής.

Ανάπτυξη

Το τέταρτο βήμα της ανάπτυξης περιλαμβάνει την συγγραφή του κώδικα της εφαρμογής. Εάν τα αποτελέσματα των προηγούμενων βημάτων είναι σωστά και λεπτομερή η συγγραφή του κώδικα δεν θα είναι δύσκολη διαδικασία. Οι εργασίες που γίνονται σε αυτό το βήμα είναι οι παρακάτω:

- **Συγγραφή κώδικα**: με την χρήση των Class, των Activity και των Component διαγραμμάτων οι προγραμματιστές αρχίζουν την συγγραφή του κώδικα της εφαρμογής.
- **Δημιουργία οθονών**: στο βήμα αυτό ολοκληρώνεται η δημιουργία των οθονών (user interface) της εφαρμογής. Επίσης ελέγχεται η επικοινωνία των οθονών με τις κλάσεις που έχουν δημιουργηθεί (ελέγχεται η επικοινωνία του Presentation επιπέδου με το Business επίπεδο όπως αυτά παρουσιάστηκαν στην αρχιτεκτονική N-επιπέδων).
- **Ολοκλήρωση τεκμηρίωσης**: ολοκληρώνεται η συγγραφή της τεκμηρίωσης των αποτελεσμάτων των βημάτων της διαδικασίας. Επίσης ολοκληρώνεται η δημιουργία του γλωσσάριου με όλους τους όρους, που έχουν χρησιμοποιηθεί κατά την διάρκεια

Έλεγχος

Σκοπός του πέμπτου και τελευταίου βήματος του ελέγχου είναι να ανακαλυφθούν τυχόν λάθη στον κώδικα. Γίνεται χρήση ορισμένων Use Case διαγραμμάτων που έχουν καταγράψει την λειτουργικότητα του συστήματος και βοηθούν στον έλεγχο ότι το σύστημα λειτουργεί σωστά. Επίσης γίνεται χρήση και των υπολοίπων διαγραμμάτων της UML τα οποία δείχνουν τις διάφορες όψεις της εφαρμογής.

Στόχος του τελευταίου βήματος είναι η εύρεση λαθών. Να τονίσουμε στο σημείο αυτό ότι οποιαδήποτε εύρεση λαθών δεν είναι αποτυχία αλλά επιτυχία αυτού του βήματος αυτού.

Στη συνέχεια παρουσιάζεται η διαδικασία χρήσης της UML, για την υλοποίηση της εφαρμογής, που υλοποιεί μέρος από το Θεωρητικό Μοντέλο Ελέγχου Διοίκησης (Management Control System). Η εφαρμογή που θα παρουσιαστεί είναι ένα σύστημα προϋπολογισμού. Επίσης, υλοποιείται το πρόβλημα της δόμησης του οργανισμού με την χρήση των Δομών καθώς επίσης υλοποιείται και το πρόβλημα της δημιουργίας αναφορών για τον έλεγχο και την αποτίμηση της λειτουργίας του συστήματος.

Μεθοδολογία Ανάπτυξης Εφαρμογής

Η μεθοδολογία που θα παρουσιάσουμε στην συνέχεια δείχνει την χρήση της UML στην διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών. Η μεθοδολογία χρήσης της UML περιλαμβάνει τα παρακάτω βήματα:

- Συλλογή απαιτήσεων
- Ανάλυση
- Σχεδιασμός
- Ανάπτυξη
- Έλεγχος

Σκοπός της παρουσίασης της μεθοδολογίας είναι να δείξουμε τον τρόπο με τον οποίο χρησιμοποιούμε τα διάφορα διαγράμματα της UML για τον σχεδιασμό, ανάπτυξη και τεκμηρίωση αντικειμενοστραφών εφαρμογών. Στην συνέχεια παρουσιάζονται τα βήματα της μεθοδολογίας.

Συλλογή Απαιτήσεων

Όραμα του συστήματος

Σκοπός της εφαρμογής είναι η παροχή ενός γενικού μηχανισμού υλοποίησης του προϋπολογισμού που θα χρησιμοποιηθεί στην εταιρεία ΒΙΟΣΕΡ με στόχο την υποστήριξη της εργασίας της διοίκησης. Θα παρουσιαστούν τα τμήματα της εφαρμογής, που γίνεται διαχείριση και επεξεργασία των βασικών οντοτήτων της εφαρμογής. Επίσης, μέσα από την εφαρμογή γίνεται προσπάθεια να παρουσιαστούν διάφορες πληροφορίες, όπως για παράδειγμα στοιχεία πωλήσεων, στοιχεία προϋπολογισμών, στοιχεία εξόδων και δαπανών, στοιχεία παραγωγής κ.λ.π. στους χρήστες της εφαρμογής ανάλογα με τον συγκεκριμένο ρόλο, στον οποίο ανήκουν οι χρήστες αυτοί. Οι πληροφορίες, που παρουσιάζονται μέσα από την εφαρμογή του προϋπολογισμού και την έκδοση των σχετικών αναφορών, βοηθούν του διοικητές του οργανισμού στο έργο τους και αποτελούν ένα μέσο για την υποστήριξη της διαδικασίας του ελέγχου της διοίκησης της ΒΙΟΣΕΡ.

Βασικός σκοπός της εφαρμογής είναι η διαχείριση του προϋπολογισμού, με την καταχώρηση των ετήσιων στόχων, που πρέπει να επιτύχει κάθε Κέντρο Ευθύνης του οργανισμού καθώς και ο έλεγχος και αποτίμηση της επίδοσης των Κέντρων Ευθύνης του οργανισμού, προκειμένου να υποστηριχθεί η επαναληπτική διαδικασία ελέγχου του οργανισμού, όπως παρουσιάστηκε στο πρώτο μέρος της επίλυσης του προβλήματος.

Επίσης, παρουσιάζεται η επίλυση του προβλήματος της δόμησης του οργανισμού, για να μπορούμε εύκολα να διαχειριστούμε τον έλεγχο της διοίκησης. Η δόμηση του οργανισμού θα πραγματοποιηθεί με την χρήση των Δομών που είναι οι βασικές οντότητες, που χρησιμοποιούνται σε όλες τις εφαρμογές της ΒΙΟΣΕΡ.

Οι βασικές οντότητες, που θα χρησιμοποιηθούν στην εφαρμογή είναι οι Παρουσιάσεις, τα Αντικείμενα και τα Αποτελέσματα και θα παρουσιαστούν στην συνέχεια. Οι οντότητες αυτές σχετίζονται με τον κάθε χρήστη που συνδέεται στην εφαρμογή βάσει συγκεκριμένων δικαιωμάτων που έχουν οριστεί για τους ρόλους στους οποίους ανήκει ο χρήστης και

αφορούν μια πληθώρα πληροφοριών που σχετίζονται με την συνολική λειτουργία της εταιρείας ΒΙΟΣΕΡ.

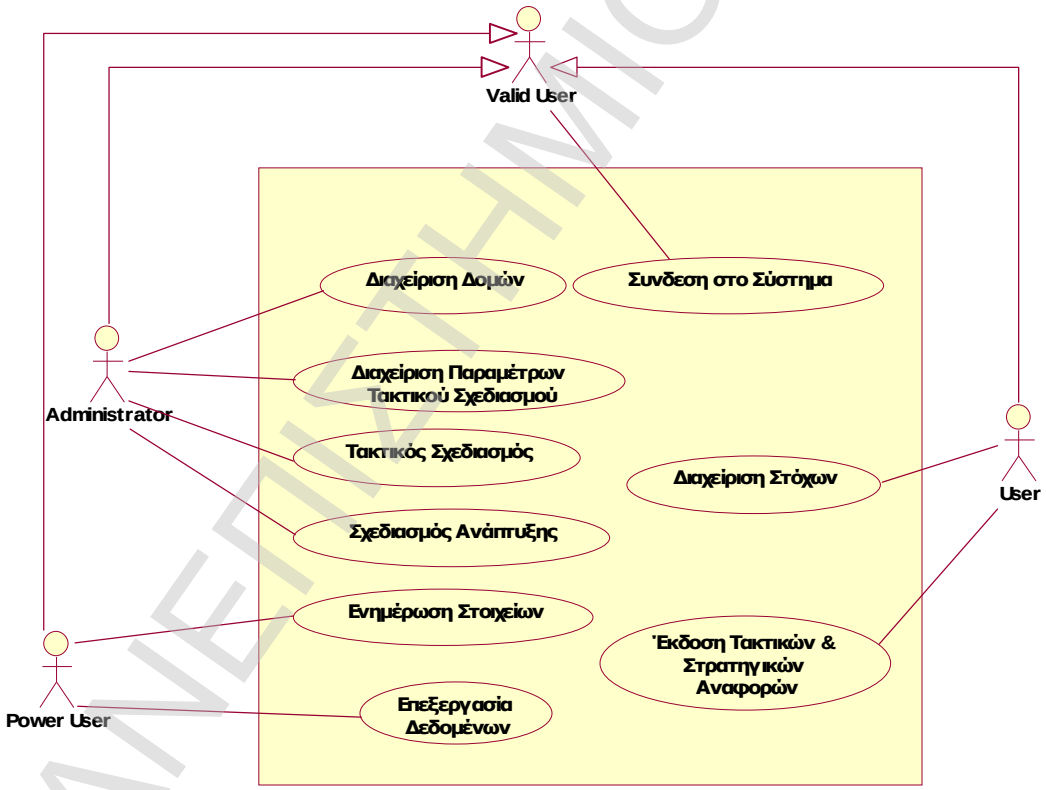
Απαιτήσεις συστήματος

Οι απαιτήσεις του συστήματος δηλαδή τι θέλουν οι χρήστες να κάνει το σύστημα πραγματοποιείται με την παρουσίαση των βασικών λειτουργιών του συστήματος όπως αυτές παρουσιάζονται στους εξωτερικούς χρήστες του συστήματος. Η παρουσίαση των βασικών λειτουργιών του συστήματος δεν δείχνει λεπτομέρειες για το πως υλοποιούνται οι λειτουργίες αυτές αλλά απλώς παρουσιάζονται αυτές όπως θα τις έβλεπε ένας εξωτερικός χρήστης του συστήματος.

Για την παρουσίαση των βασικών λειτουργιών του συστήματος θα χρησιμοποιηθούν Use Case διαγράμματα.

Χρήση Use Case διαγραμμάτων

Main Use Case Διάγραμμα



Οι actors, τα πρόσωπα δηλαδή που αλληλεπιδρούν με το σύστημα και βρίσκονται εξωτερικά από αυτό είναι:

- **Valid User**: ένας χρήστης που έχει συνδεθεί με επιτυχία στο σύστημα. Ο actor αυτός δεν εκτελεί κάποια συγκεκριμένη λειτουργία αλλά χρησιμοποιείται για να κληρονομήσουν την ιδιότητα του αυτή (σύνδεση στο σύστημα) οι υπόλοιποι actors.

- **Administrator**: ο διαχειριστής του συστήματος ο οποίος εκτελεί συγκεκριμένες λειτουργίες δημιουργίας διαφόρων οντοτήτων του συστήματος προκειμένου να χρησιμοποιηθούν από τους άλλους χρήστες.
- **Power User**: ένα ειδικός χρήστης του συστήματος ο οποίος εκτελεί συγκεκριμένες λειτουργίες ενημέρωσης στοιχείων με δεδομένα έτσι ώστε αυτά να χρησιμοποιηθούν από τους απλούς χρήστες του συστήματος.
- **User**: ο απλός χρήστης του συστήματος ο οποίος ανάλογα με τον ρόλο που έχει οριστεί για αυτόν χρησιμοποιεί το σύστημα και εκτελεί τις καθημερινές λειτουργίες του συστήματος.

Οι actors Administrator, Power User και User κληρονομούν την ιδιότητα του Valid User της σύνδεσης στο σύστημα δηλαδή για να εκτελέσουν τις λειτουργίες τους όλοι οι actors πρέπει πρώτα να έχουν συνδεθεί με επιτυχία στο σύστημα.

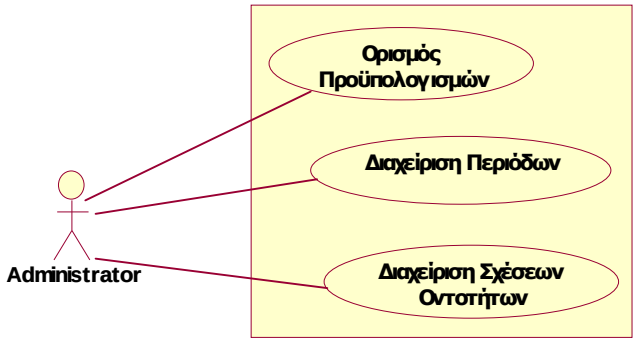
Οι βασικές λειτουργίες του συστήματος, τα βασικά use cases δηλαδή είναι[13]:

- Σύνδεση στο Σύστημα
- Διαχείριση Δομών
- Διαχείριση Παραμέτρων Τακτικού Σχεδιασμού
- Τακτικός Σχεδιασμός
- Διαχείριση Στόχων
- Σχεδιασμός Ανάπτυξης
- Ενημέρωση Στοιχείων
- Επεξεργασία Δεδομένων
- Έκδοση Τακτικών & Στρατηγικών Αναφορών

Από τα παραπάνω use cases υπάρχουν κάποια τα οποία τα θεωρούμε σημαντικά και πρέπει να δείξουμε περισσότερες λεπτομέρειες για να καταλάβουμε τον τρόπο λειτουργίας τους. Τα use cases που θεωρούμε σημαντικά είναι το «Διαχείριση Παραμέτρων Τακτικού Σχεδιασμού», το «Τακτικός Σχεδιασμός» και το «Διαχείριση Στόχων». Για κάθε ένα από τα use cases αυτά θα χρησιμοποιήσουμε επιπλέον Use Case διαγράμματα.

[13] Στην παρουσίαση των βασικών λειτουργιών του συστήματος αναφέρονται και ορισμένες οντότητες του συστήματος, όπως Δομές, Παρουσιάσεις, Περίοδοι, Αντικείμενα κλπ. Οι οντότητες αυτές, θα παρουσιαστούν με μεγαλύτερη λεπτομέρεια στην συνέχεια της παρουσίασης στο τμήμα της παρουσίασης των οντοτήτων της εφαρμογής.

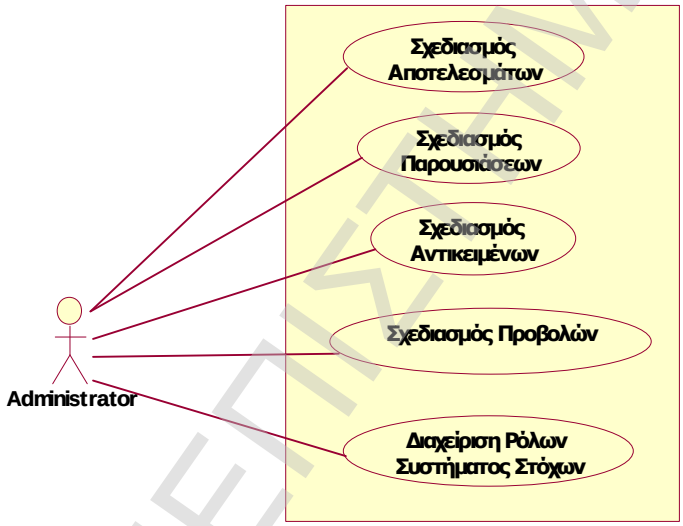
Use Case: Διαχείριση Παραμέτρων Τακτικού Σχεδιασμού



Οι βασικές λειτουργίες (use cases) που εκτελεί το use case «Διαχείριση Παραμέτρων Τακτικού Σχεδιασμού» είναι οι:

- Ορισμός Προϋπολογισμών
- Διαχείριση Περιόδων
- Διαχείριση Σχέσεων Οντοτήτων

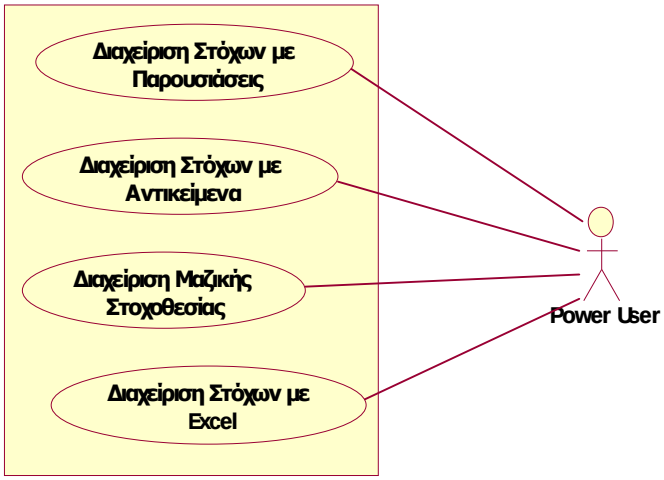
Use Case: Τακτικός Σχεδιασμός



Οι βασικές λειτουργίες (use cases) που εκτελεί το use case «Τακτικός Σχεδιασμός» είναι οι:

- Σχεδιασμός Αποτελεσμάτων
- Σχεδιασμός Παρουσιάσεων
- Σχεδιασμός Αντικειμένων
- Σχεδιασμός Προβολών
- Διαχείριση Ρόλων Συστήματος Στόχων

Use Case: Διαχείριση Στόχων



Οι βασικές λειτουργίες (use cases) που εκτελεί το use case «Διαχείριση Παραμέτρων Τακτικού Σχεδιασμού» είναι οι:

- Διαχείριση Στόχων με Παρουσιάσεις
- Διαχείριση Στόχων με Αντικείμενα
- Διαχείριση Μαζικής Στοχοθεσίας
- Διαχείριση Στόχων με Excel

Οι περισσότερες από τις παραπάνω λειτουργίες καθώς και οι διάφορες οντότητες που αναφέρονται σε αυτές (π.χ. Παρουσιάσεις, Αντικείμενα, Περίοδοι, Δομές κλπ.) θα παρουσιαστούν με μεγαλύτερη ανάλυση στην συνέχεια της παρουσίασης.

Παρουσίαση λειτουργιών

Αφού έχουμε παρουσιάσει τα Use Case διαγράμματα για να δείξουμε τις βασικές λειτουργίες του συστήματος στη συνέχεια θα προχωρήσουμε στην ανάλυση μερικών από τις λειτουργίες αυτές. Σκοπός είναι να παρουσιάσουμε τον τρόπο λειτουργίας των use cases αυτών. Επίσης στο βήμα αυτό δημιουργούμε κάποια use case σενάρια που περιγράφουν συγκεκριμένες λειτουργίες που χρήζουν μεγαλύτερης ανάλυσης.

Οι λειτουργίες που θα αναλύσουμε στην συνέχεια είναι οι παρακάτω:

- Διαχείριση Δομών
- Διαχείριση Περιόδων
- Ορισμός Προϋπολογισμών
- Σχεδιασμός Παρουσιάσεων
- Σχεδιασμός Αντικειμένων

Παρουσίαση use case σεναρίων

Στη συνέχεια της παρουσίασης θα δείξουμε κάποια use case σενάρια για ορισμένες από τις δασικές λειτουργίες του συστήματος. Ένα σενάριο είναι ένα σύνολο από διαδοχικά βήματα που γίνονται προκειμένου να εκτελεστεί η λειτουργία του συστήματος στην οποία αναφέρεται το σενάριο. Τα σενάρια βοηθούν τον αναγνώστη τους να καταλάβει τον τρόπο εκτέλεσης της κάθε λειτουργίας. Κάθε σενάριο στην συνέχεια αναπαρίσταται γραφικά με ένα Activity διάγραμμα.

Τα use case σενάρια των βασικότερων λειτουργιών του συστήματος παρουσιάζονται παρακάτω:

Use case σενάριο: Διαχείριση Δομών

1. Επιλογή από Κεντρική οθόνη – Άνοιγμα οθόνης επιλογής Τύπων
2. Ανάκτηση Τύπων
   - 2.1. Ανάκτηση διαθέσιμων Τύπων από Βάση Δεδομένων
   - 2.2. Εμφάνιση στην οθόνη
3. Επιλογή συγκεκριμένου Τύπου
   - 3.1. Ανάκτηση διαθέσιμων Δομών για τον επιλεγμένο Τύπου
   - 3.2. Εμφάνιση στην οθόνη (Λίστα Δομών)
4. Επιλογή συγκεκριμένης Δομής
   - 4.1. Άνοιγμα οθόνης διαχείρισης Δομών
   - 4.2. Ανάκτηση Περιγραφής Δομής
   - 4.3. Ανάκτηση Στοιχείων Δομής
   - 4.4. Εμφάνιση στην οθόνη (Δέντρο στοιχείων Δομής)
5. Επεξεργασία Δομής
   - 5.1. Προσθήκη Στοιχείου
   - 5.2. Διαγραφή Στοιχείου
   - 5.3. Αποκοπή – Επικόλληση Στοιχείου
   - 5.4. Μετακίνηση Στοιχείου στην Δομή (Πάνω-Κάτω)
   - 5.5. Προβολή ιδιοτήτων Στοιχείου Δομής
6. Αποθήκευση αλλαγών

Use case σενάριο: Διαχείριση Περιόδων

1. Άνοιγμα οθόνης διαχείρισης Περιόδων 2. Επιλογή Τύπου Περιόδου 2.1. Επιλογή Περιόδων Έτους 2.1.1. Δώσε Έτος 2.1.2. Επέλεξε από την λίστα των Μηνών ή 2.1.3. Επέλεξε από λίστα συγκεκριμένων Ορισμών Περιόδων **ή**
2.1. Επιλογή Κυλιόμενων Περιόδων 2.1.1. Δώσε Έτος Από / Μήνα Από 2.1.2. Δώσε Έτος Έως / Μήνα Έως 2.1.3. Δώσε αριθμό Περιόδων 2.1.4. Δώσε προβολή Ανά **ή**
2.1. Επιλογή Δομών Περιόδων 2.1.1. Ανάκτηση διαθέσιμων Δομών Περιόδων 2.1.2. Επιλογή Δομής Περιόδου από τη λίστα 2.1.3. Επέλεξε τα Στοιχεία της Δομής Περιόδου
3. Αποθήκευση επιλογών

Το παραπάνω σενάριο δείχνει τρεις διαφορετικές επιλογές που έχει ο χρήστης για να επιλέξει μια Περίοδο. Μπορεί να επιλέξει είτε Περιόδους έτους είτε Κυλιόμενες Περιόδους είτε μια Δομή Περιόδου. Η διαδικασία της επιλογής περιόδου θα γίνει περισσότερο σαφής στη συνέχεια που θα παρουσιαστεί το αντίστοιχο Activity διάγραμμα του συγκεκριμένου σεναρίου.

Use case σενάριο: Ορισμός Προϋπολογισμών

1. Άνοιγμα οθόνης ορισμού Προϋπολογισμών
2. Ανάκτηση Προϋπολογισμών από τη Βάση Δεδομένων
3. Ανάκτηση Τύπων για τις 7 διαστάσεις του Προϋπολογισμού
   3.1. Εμφάνιση Τύπων στην οθόνη
4. Επιλογή συγκεκριμένου Προϋπολογισμού
   4.1. Ανάκτηση διαστάσεων για τον επιλεγμένο Προϋπολογισμό
   4.2. Εμφάνιση διαστάσεων στην οθόνη
5. Επεξεργασία Προϋπολογισμού
   5.1. Για κάθε διάσταση που θέλουμε να ορίσουμε
      5.1.1. Επιλογή Τύπου για την διάσταση
      5.1.2. Καθαρισμός διάστασης που δεν θέλουμε
6. Αποθήκευση αλλαγών

Use case σενάριο: Σχεδιασμός Παρουσιάσεων

1. Άνοιγμα οθόνης σχεδιασμού Παρουσιάσεων
2. Ανάκτηση διαθέσιμων Τύπων
3. Ανάκτηση διαθέσιμων Προϋπολογισμών
4. Ανάκτηση Παρουσιάσεων από Βάση Δεδομένων
5. Επιλογή συγκεκριμένης Παρουσίασης
   5.1. Ανάκτηση στοιχείων επιλεγμένης Παρουσίασης (Δομές)
   5.2. Για κάθε στοιχείο της Παρουσίασης (Διάσταση)
      5.2.1. Ανάκτηση ονομασίας στοιχείου Παρουσίασης
   5.3. Εμφάνιση στοιχείων Παρουσίασης στην οθόνη
6. Επιλογή Προσθήκης Στοιχείου
   6.1. Επιλογή διαθέσιμου Προϋπολογισμού
   6.2. Επιλογή των διαστάσεων του Προϋπολογισμού (Τύποι)
   6.3. Για κάθε μια διάσταση (Τύπο)
      6.3.1. Ανάκτησε διαθέσιμες Δομές για τον Τύπο
      6.3.2. Επέλεξε Δομή για την διάσταση
   6.4. Αποδοχή αλλαγών
7. Αποθήκευση αλλαγών

Use case σενάριο: Σχεδιασμός Αντικειμένων

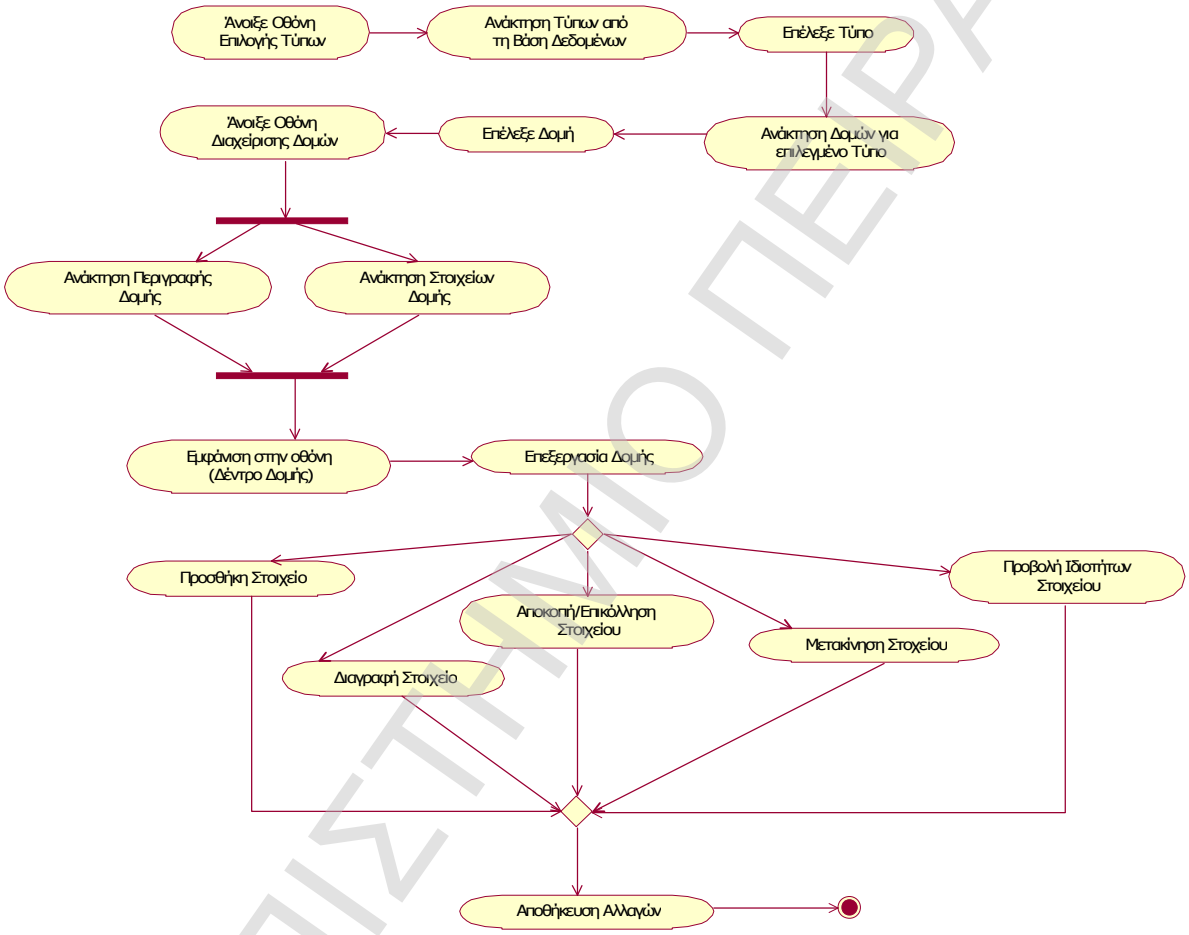
1. Άνοιγμα οθόνης σχεδιασμού Αντικειμένων
2. Ανάκτηση στοιχείων Ρόλου χρήστη
3. Ανάκτηση Αντικειμένων, βάσει Ρόλου χρήστη
4. Ανάκτηση Παρουσιάσεων, βάσει Ρόλου χρήστη
5. Επιλογή συγκεκριμένου Αντικειμένου
    5.1. Ανάκτηση σχετιζόμενης Παρουσίασης
    5.2. Ανάκτηση στοιχείων σχετιζόμενης Παρουσίασης (Δομές)
    5.3. Για κάθε στοιχείο της Παρουσίασης (Διάσταση)
        5.3.1. Ανάκτηση ονομασίας στοιχείου Παρουσίασης
        5.3.2. Ανάκτηση άλλων πληροφοριών
        5.3.3. Ανάκτηση στοιχείων Δομών
    5.4. Εμφάνιση στοιχείων Παρουσίασης στην οθόνη
    5.5. Για κάθε στοιχείο της Δομής
        5.5.1. Ανάκτηση Επιλογών χρήστη από τη Βάση Δεδομένων
        5.5.2. Τσεκάρισμα αντίστοιχου στοιχείου Δομής
6. Ανάκτηση πληροφοριών για Διαστάσεις
7. Επεξεργασία στοιχείων Παρουσίασης
8. Αποθήκευση αλλαγών

Στη συνέχεια για κάθε use case σενάριο που έχει δημιουργηθεί κατασκευάζεται ένα αντίστοιχο Activity διάγραμμα. Σκοπός κάθε Activity διαγράμματος είναι η οπτικοποίηση του use case σεναρίου στο οποίο αναφέρεται.

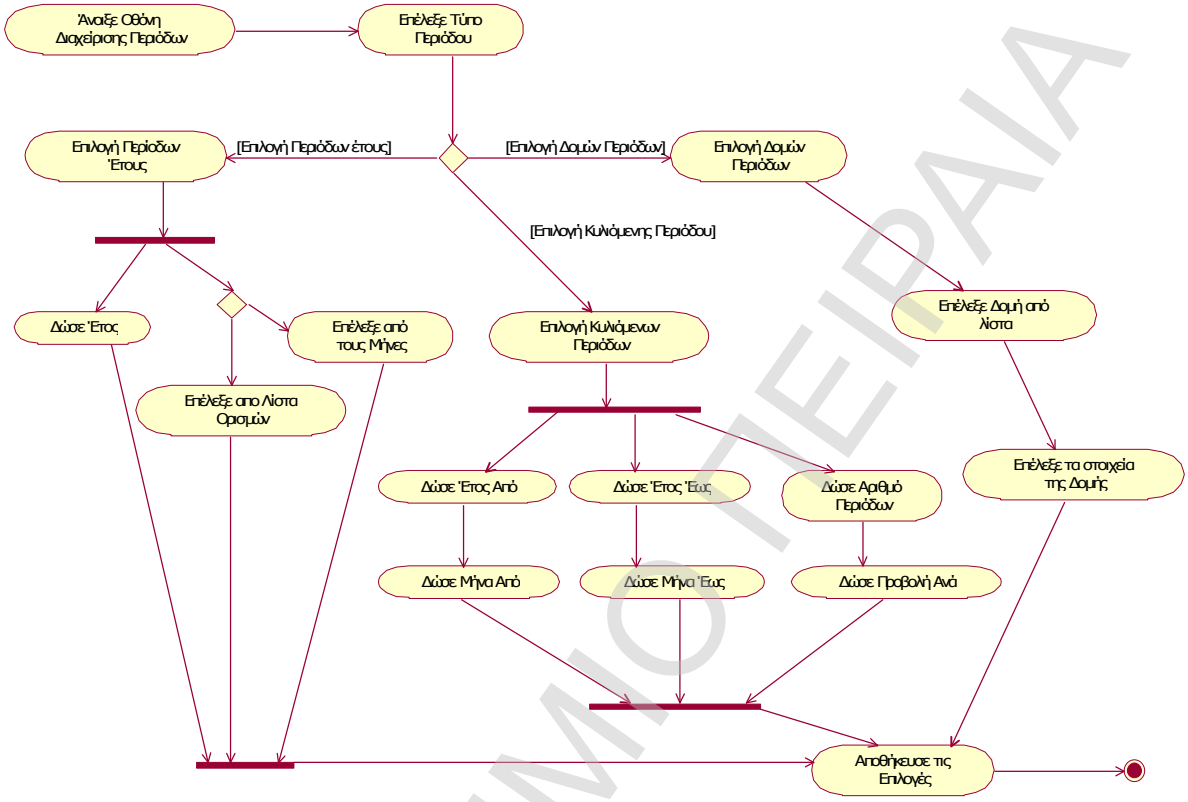
Παρουσίαση Activity Διαγραμμάτων

Στην συνέχεια παρουσιάζονται τα Activity διαγράμματα για τα use case σενάρια που έχουν δημιουργηθεί.

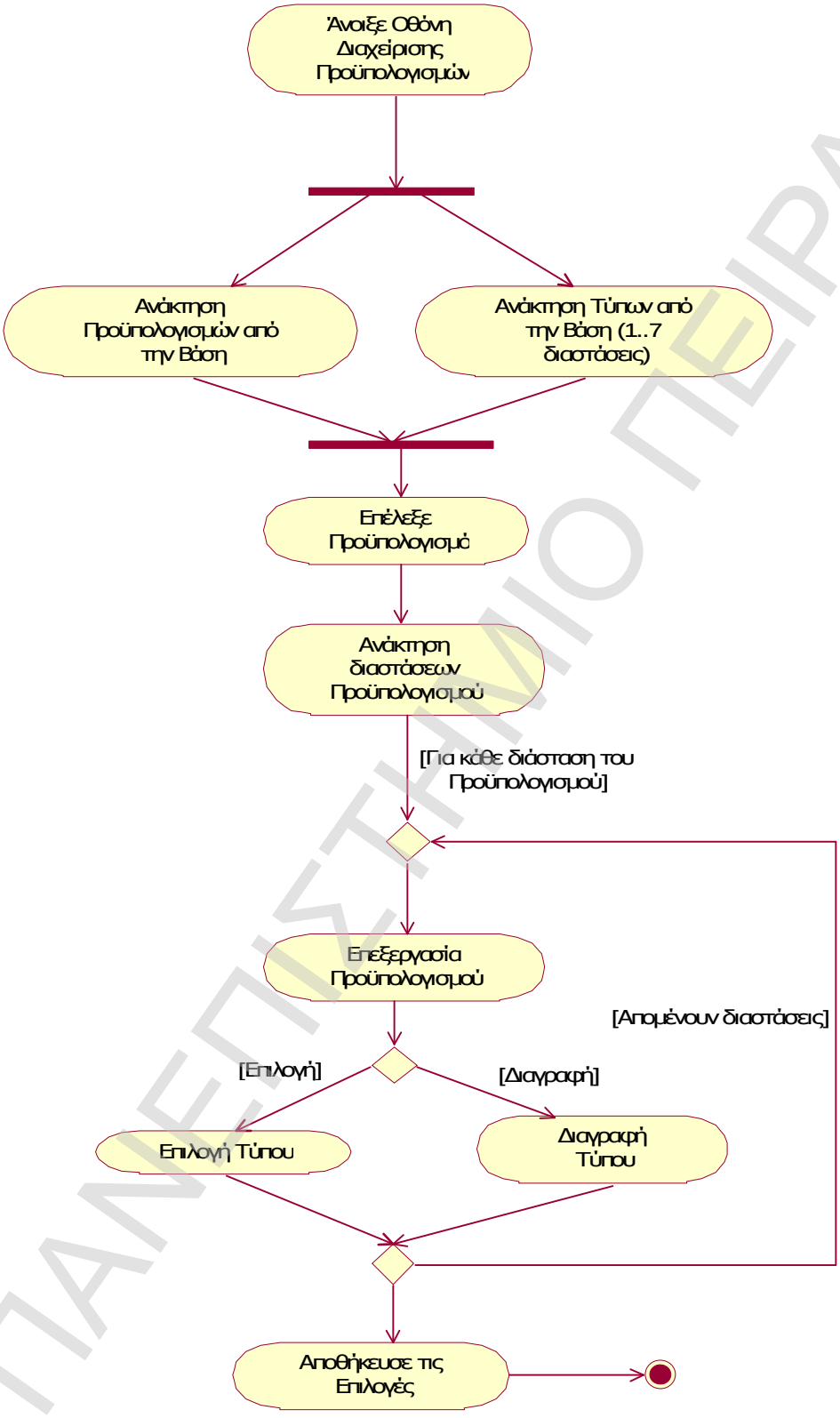
Activity διάγραμμα: Διαχείριση Δομών



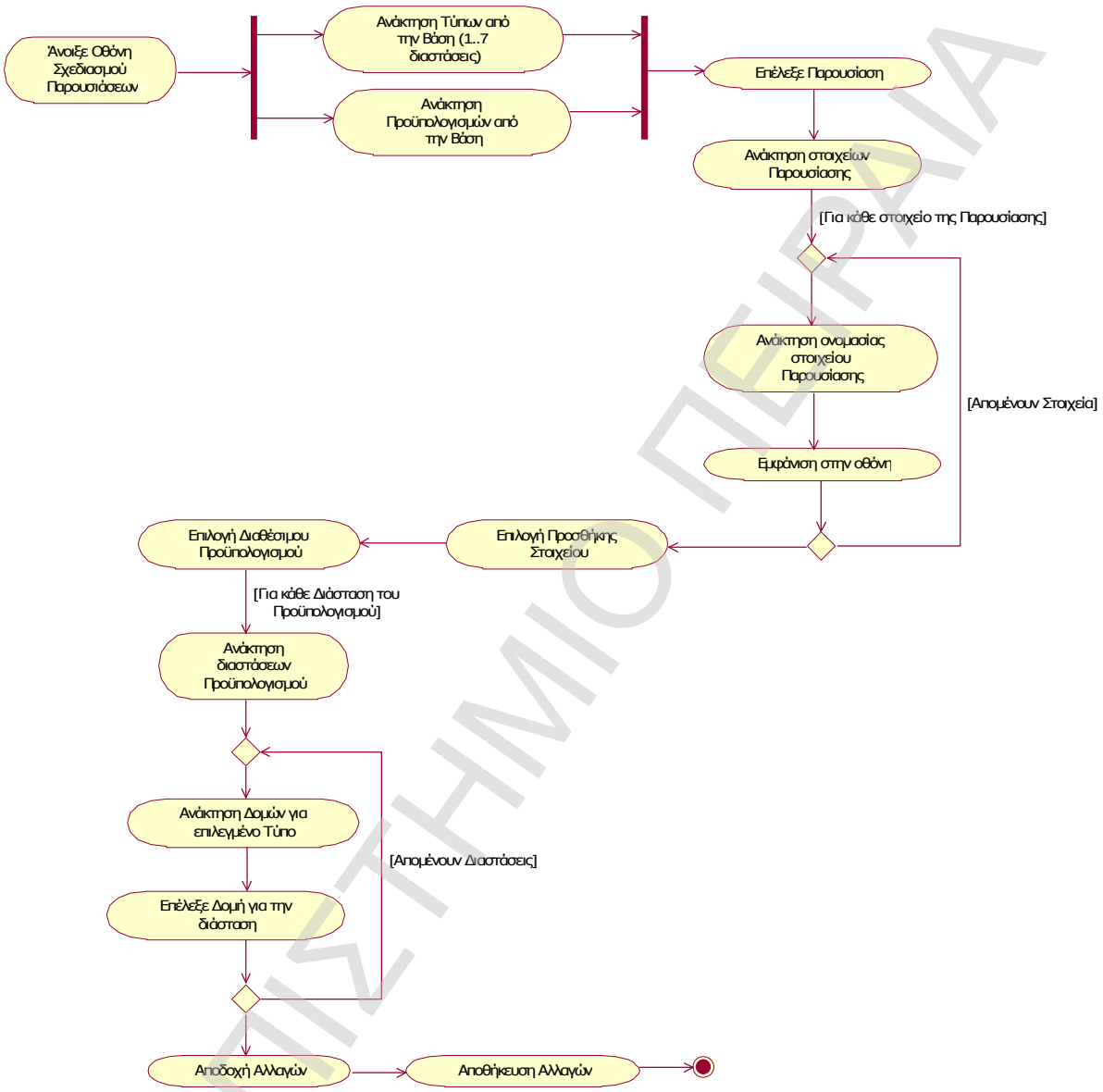
Activity διάγραμμα: Διαχείριση Περιόδων



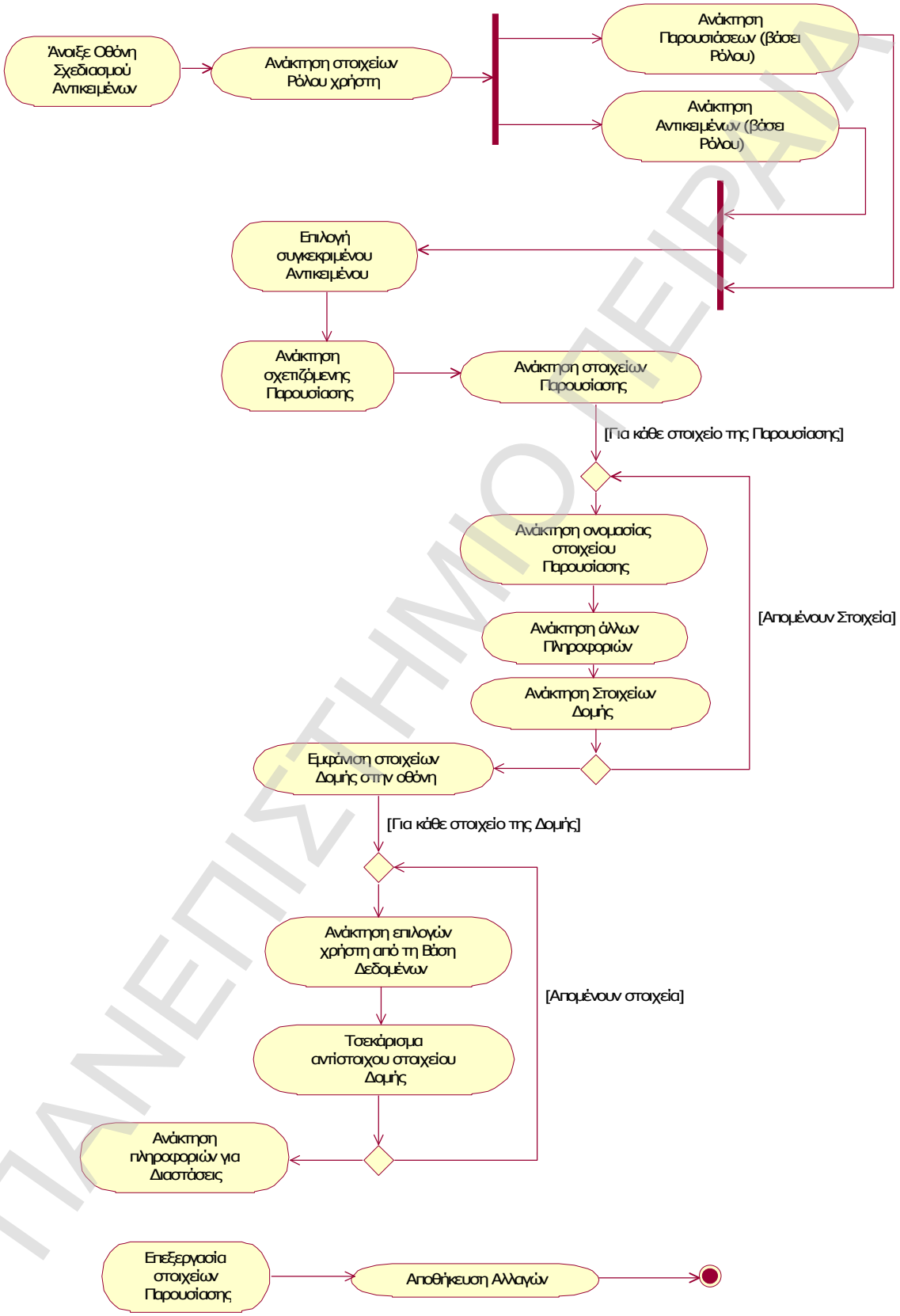
Activity διάγραμμα: Ορισμός Προϋπολογισμών



Activity διάγραμμα: Σχεδιασμός Παρουσιάσεων



Activity διάγραμμα: Σχεδιασμός Αντικειμένων



Ανάλυση περιβάλλοντος χρήστη (domain model)

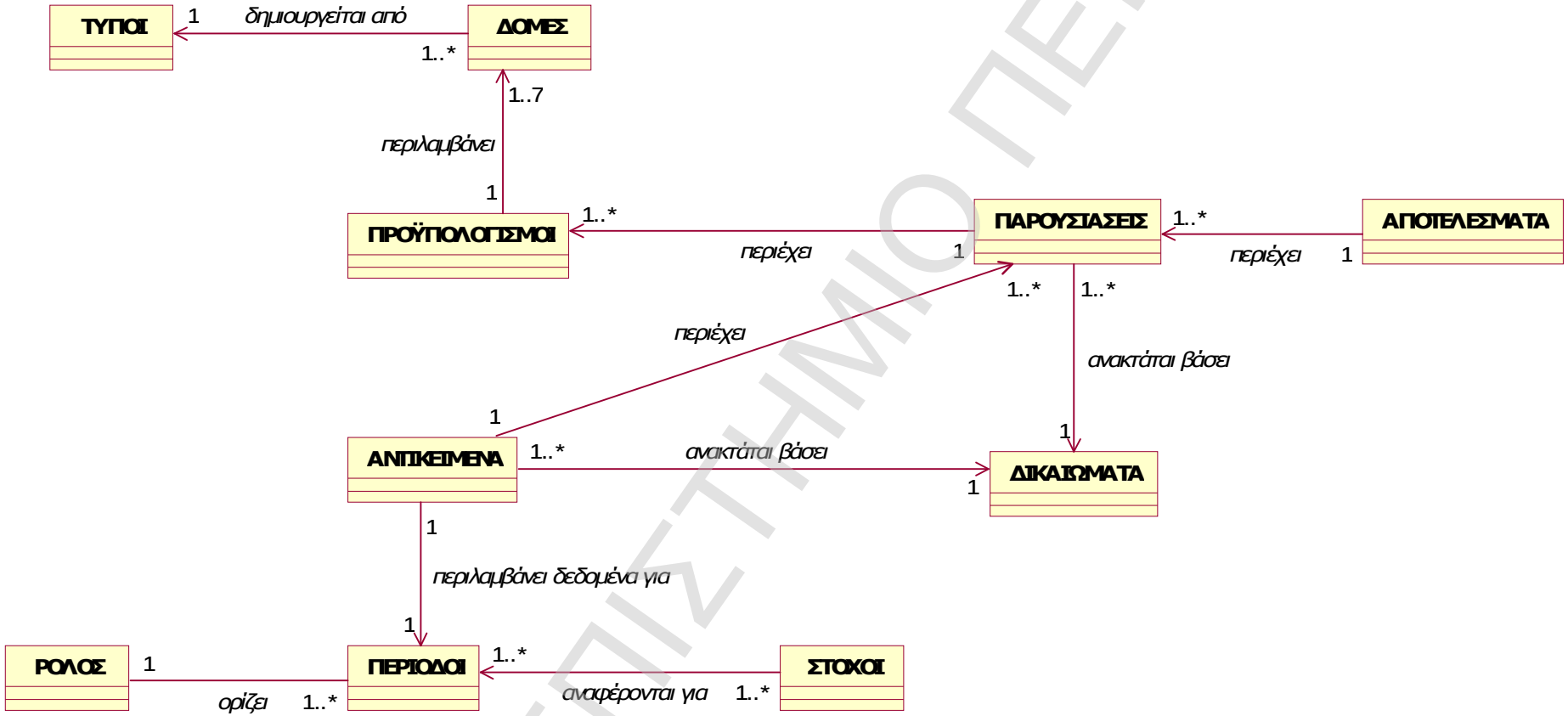
Το επόμενο βήμα σχετίζεται με την ανάλυση του περιβάλλοντος χρήστη του συστήματος (domain model). Το domain model παρουσιάζει τις βασικές οντότητες που θα χρησιμοποιηθούν στην εφαρμογή και γίνεται προσπάθεια καταγραφής τους για περαιτέρω ανάλυση τους. Μέσα από την ανάλυση των Use Case διαγραμμάτων και των use case σεναρίων και των αντίστοιχων Activity διαγραμμάτων έχουμε εντοπίσει κάποια αντικείμενα (οντότητες), που είναι πολύ πιθανό να συμμετέχουν στο σύστημα. Τα αντικείμενα αυτά είναι τα παρακάτω:

- Τύποι
- Δομές
- Περίοδοι
- Ρόλοι
- Δικαιώματα
- Προϋπολογισμοί
- Παρουσιάσεις
- Αντικείμενα
- Αποτελέσματα

Χρήση Class Διαγράμματος

Η ανάλυση της κάθε οντότητα θα παρουσιαστεί στις επόμενες σελίδες. Η καταγραφή των βασικών οντοτήτων του συστήματος γίνεται σε ένα Class διάγραμμα το οποίο παρουσιάζεται στη συνέχεια.

Διάγραμμα Domain Model



Παρουσίαση αντικειμένων του περιβάλλοντος της εφαρμογής

Τα βασικά αντικείμενα που θα χρησιμοποιηθούν στην εφαρμογή παρουσιάζονται στον επόμενο πίνακα:

Ονομασία	**Περιγραφή**
Τύποι	Τύπος είναι μια γενική οντότητα του συστήματος, όπως Πελάτες, Είδη, Έξοδα. Ένας Τύπος δεν εκτελεί κάποιες συγκεκριμένες λειτουργίες αλλά χρησιμοποιείται σαν βάση για την δημιουργία άλλων οντοτήτων όπως οι Δομές.
Δομές	Δομή είναι ένα αντικείμενο που παρουσιάζει την ιεραρχική δομή διάφορων Τύπων Για παράδειγμα έχουμε Δομές Πελατών, Ειδών, Εξόδων κλπ. Για κάθε Τύπο μπορούμε να έχουμε περισσότερες από μία Δομές προκειμένου να παραστήσουμε την ίδια πληροφορία με διαφορετικό τρόπο. Με μία Δομή μπορούμε να αναπαραστήσουμε και να διαχειριστούμε οποιαδήποτε οντότητα μέσα στην εφαρμογή. Η διαχείριση των Δομών γίνεται με ένα συγκεκριμένο τρόπο (ιεραρχική απεικόνιση) χωρίς να μας ενδιαφέρει τι δεδομένα περιέχει μέσα της η Δομή. Επίσης μπορούμε να συνδυάσουμε δυο ή περισσότερες Δομές (διαστάσεις) προκειμένου να δείξουμε διάφορες πληροφορίες, για παράδειγμα Έσοδα ανά Πελάτη, Έσοδα ανά Είδος, Πωλήσεις ανά Πελάτη και ανά Είδος κλπ.
Περίοδοι	Περίοδος είναι η διάσταση του χρόνου βάσει του οποίου θέλουμε να γίνει η παρουσίαση των δεδομένων. Για παράδειγμα έχουμε μηνιαία παρουσίαση (12 μήνες του έτους ή οποιοδήποτε άλλο σύνολο μηνών), έχουμε κυλιόμενες περιόδους (από κάποιο Μήνα/Ετος έως κάποιο Μήνα/Ετος) και τέλος έχουμε μια Δομή Περιόδου (που στην ουσία είναι μια ιεραρχική παρουσίαση του χρόνου που έχει δημιουργηθεί από τον χρήστη, π.χ. προβολή ανά ΕΤΟΣ-ΤΕΤΡΑΜΗΝΑ-ΜΗΝΑΣ-ΕΒΔΟΜΑΔΑ)
Ρόλοι	Η οντότητα του Ρόλου διαχειρίζεται τον ρόλο που έχει κάθε χρήστης που συνδέεται στην εφαρμογή. Βάσει του ρόλου ορίζεται τι έχει δικαίωμα να βλέπει ο χρήστης. Ένας ρόλος μπορεί να είναι μια συγκεκριμένη οντότητα-ρόλος, π.χ. ΔΙΕΥΘΥΝΤΗΣ, ΥΠΕΥΘΥΝΟΣ, ΧΡΗΣΤΗΣ κλπ. Επίσης μπορεί να είναι και ένα τμήμα του οργανογράμματος του οργανισμού δηλαδή ένα τμήμα της ιεραρχικής οργάνωσης της εταιρείας, π.χ. ΤΜΗΜΑ ΠΩΛΗΣΕΩΝ, ΛΟΓΙΣΤΗΡΙΟ κλπ. που περιέχει μέσα του διάφορες οντότητες-ρόλους.
Δικαιώματα	Η οντότητα των Δικαιωμάτων διαχειρίζεται τα στοιχεία που έχει δικαίωμα να βλέπει ο χρήστης βάσει του Ρόλου στον οποίο ανήκει. Τα στοιχεία, που έχει δικαίωμα να βλέπει ο χρήστης είναι οι Παρουσιάσεις και τα Αντικείμενα.
Προϋπολογισμοί	Προϋπολογισμός είναι ένας «καμβάς» πάνω στον οποίο δημιουργούμε από 1 έως 7 διαστάσεις που περιλαμβάνουν Τύπους. Ένας Προϋπολογισμός δεν εκτελεί κάποια συγκεκριμένη λειτουργία αλλά χρησιμοποιείται για να κατασκευαστούν πάνω σε αυτόν οι Παρουσιάσεις.
Παρουσιάσεις	Παρουσίαση είναι η οντότητα, που καθορίζει τι στοιχεία θα περιέχει η κάθε διάσταση. Συγκεκριμένα η Παρουσίαση κατασκευάζεται πάνω σε ένα Προϋπολογισμό ο οποίος καθορίζει τον αριθμό των διαστάσεων (από 1 έως 7) καθώς και τον Τύπο που θα έχει η κάθε διάσταση. Μια Παρουσίαση μπορεί να αποτελείται από ένα ή περισσότερους Προϋπολογισμούς ανεξαρτήτως διαστάσεων. Στη συνέχεια, για κάθε διάσταση της Παρουσίασης,επιλέγουμε μια συγκεκριμένη Δομή που να ανήκει στον Τύπο της κάθε διάστασης. Με την Παρουσίαση έχουμε την δυνατότητα να ομαδοποιήσουμε διαφορετικά στοιχεία (π.χ. Δομές Πελατών, Εξόδων, Ειδών κλπ.) κάτω από ένα κοινό όνομα.

Αντικείμενα	Αντικείμενο είναι η οντότητα που αποθηκεύει τις επιλογές του χρήστη για μια Παρουσίαση. Συγκεκριμένα, ένας χρήστης ανάλογα με τον Ρόλο στον οποίο ανήκει έχει Δικαίωμα να δει ορισμένες Παρουσιάσεις. Στην συνέχεια, για κάθε Παρουσίαση μπορεί να επιλέξει ορισμένα στοιχεία των Δομών της κάθε διάστασης της Παρουσίασης και να τα αποθηκεύσει. Έτσι, όποτε ο χρήστης θέλει να δει δεδομένα για τα στοιχεία που τον ενδιαφέρουν επιλέγει να δει το συγκεκριμένο Αντικείμενο.
Αποτελέσματα	Αποτέλεσμα είναι η οντότητα που διαχειρίζεται την ομαδοποίηση διαφόρων Παρουσιάσεων και την παρουσίαση του αποτελέσματος. Για παράδειγμα, εάν έχουμε πολλές Παρουσιάσεις που σχετίζονται με έξοδα και έσοδα το Αποτέλεσμα ομαδοποιεί τις Παρουσιάσεις αυτές, κάνει προσθέσεις και αφαιρέσεις (ανάλογα με τον Τύπο τους) και εμφανίζει το τελικό Αποτέλεσμα (άθροισμα) στον χρήστη

Πίνακας 5: Αντικείμενα του Περιβάλλοντος της Εφαρμογής

Ανάλυση

Μετά από το βήμα της Συλλογής Απαιτήσεων προχωράμε στο δεύτερο βήμα της Ανάλυσης. Στο σημείο αυτό αφού έχουμε καταγράψει τις απαιτήσεις των χρηστών, δηλαδή τι θέλουν να κάνει το σύστημα που σχεδιάζουμε και ποιες είναι οι βασικές οντότητες, που θα χρησιμοποιηθούν στην εφαρμογή πρέπει να προχωρήσουμε στην ανάλυση των αντικειμένων. Το βήμα αυτό είναι η «καρδιά» της διαδικασίας γιατί αναλύουμε τα αντικείμενα τα οποία αποτελούν την δομή της εφαρμογής (αφού άλλωστε μιλάμε για αντικειμενοστραφή εφαρμογή).

Στο βήμα αυτό αναλύουμε τα αντικείμενα, προσπαθούμε να βρούμε καινούργια αντικείμενα και καταγράφουμε τον τρόπο, με τον οποίο τα αντικείμενα αυτά επικοινωνούν μεταξύ τους.

Ανάλυση Use Cases και Αντικειμένων

Μέσα από την προηγούμενη ανάλυση των Use Case διαγραμμάτων, των use case σεναρίων και των αντίστοιχων Activity διαγραμμάτων έχουμε εντοπίσει τα βασικά αντικείμενα που θα χρησιμοποιηθούν στην εφαρμογή. Αυτά τα αντικείμενα θέλουμε να τα μετατρέψουμε σε κλάσεις, δηλαδή σε εκείνες τις οντότητες του αντικειμενοστραφούς προγραμματισμού που θα χρησιμοποιηθούν στην ανάπτυξη της εφαρμογής. Επομένως για κάθε οντότητα θα δημιουργήσουμε την αντίστοιχη κλάση.

Οι κλάσεις που θα χρησιμοποιήσουμε θα έχουν το πρόθεμα *cls* και το όνομα της αντίστοιχης οντότητας στα αγγλικά. Έτσι για τις ανάγκες της εφαρμογής του που σχεδιάζουμε θα δημιουργήσουμε τις παρακάτω κλάσεις:

- Τύποι – *clsTypes*
- Δομές – *clsDomes*
- Περίοδοι – *clsPeriod*
- Δικαιώματα – *clsRights*
- Προϋπολογισμοί – *clsBdgDef*
- Παρουσιάσεις – *clsBdgPar*
- Αντικείμενα – *clsBdgObjects*
- Αποτελέσματα – *clsBdgSyn*

ΣΗΜΕΙΩΣΗ: Για το αντικείμενο Ρόλοι δεν είναι απαραίτητο να δημιουργήσουμε μια ξεχωριστή κλάση αλλά μπορούμε να φτιάξουμε μια μεταβλητή τύπου Structure, που θα περιέχει τα πεδία του αντικειμένου Ρόλοι. Έτσι, για να αναζητούμε δεδομένα για κάποιο συγκεκριμένο Ρόλο θα χρησιμοποιούμε την συγκεκριμένη μεταβλητή.

Στις κλάσεις που θα δημιουργήσουμε πρέπει να προσθέσουμε τα κατάλληλα χαρακτηριστικά (μεταβλητές) και τις κατάλληλες μεθόδους (συναρτήσεις). Τα χαρακτηριστικά και οι μέθοδοι για κάθε κλάση ανακαλύπτονται μέσα από την λεπτομερή ανάλυση της λειτουργίας της κάθε κλάσης ξεχωριστά.

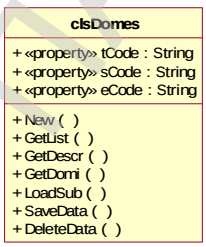
Για παράδειγμα, για την κλάση που διαχειρίζεται τις Δομές, την ***clsDomes*** πρέπει να χρησιμοποιήσουμε τα παρακάτω Χαρακτηριστικά, που σχετίζονται με την κλάση αυτή:

Χαρακτηριστικό	**Χρήση**
tCode	Κωδικός Τύπου Δομής
sCode	Κωδικός Δομής
eCode	Κωδικός στοιχείου Δομής

Επίσης για την διαχείριση, ανάκτηση δεδομένων για μια Δομή και την παρουσίαση τους στην εφαρμογή χρειαζόμαστε τις παρακάτω Μεθόδους:

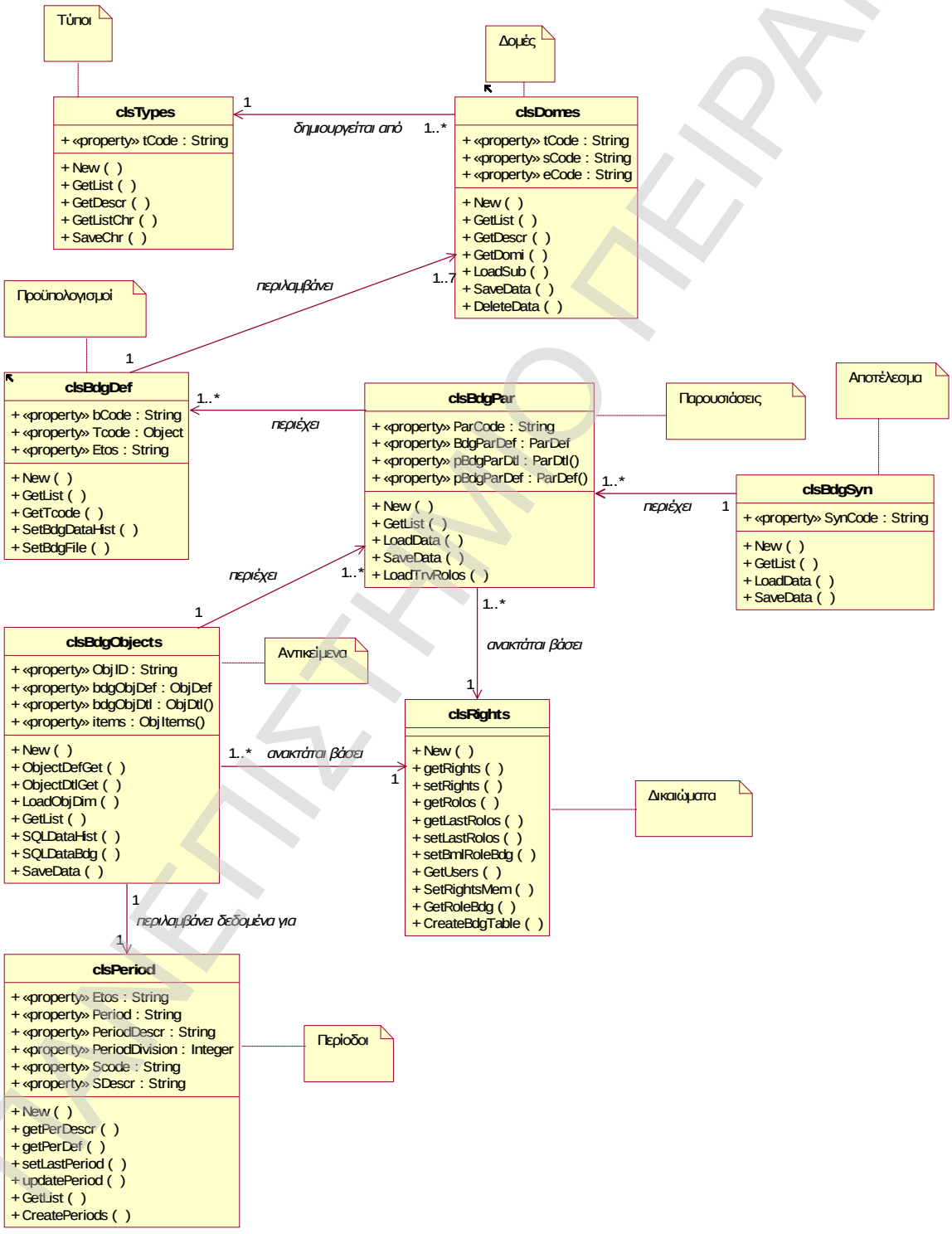
Μέθοδος	**Χρήση**
New	Constructor της κλάσης για την δημιουργία της
GetList	Ανάκτηση διαθέσιμων Δομών
GetDescr	Ανάκτηση περιγραφής της Δομής
GetDomi	Ανάκτηση των στοιχείων της Δομής
SaveData	Αποθήκευση δεδομένων
DeleteData	Διαγραφή δεδομένων

Για την γραφική αναπαράσταση της κλάσης Δομές θα χρησιμοποιήσουμε ένα ορθογώνιο που θα περιέχει τα Χαρακτηριστικά και τις Μεθόδους, για παράδειγμα:



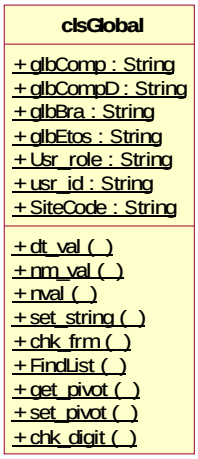
Την ανάλυση που κάναμε για την κλάση Δομές θα την επαναλάβουμε για όλες τις κλάσεις του συστήματος. Τις κλάσεις αυτές θα τις παραστήσουμε σε ένα Class διάγραμμα:

Χρήση Class Διαγράμματος

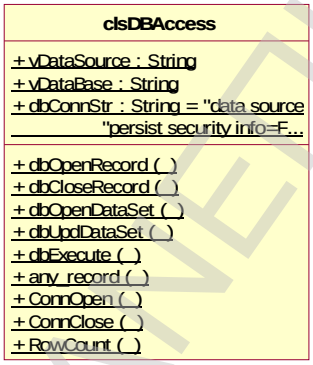


Εντοπισμός νέων Αντικειμένων

Το επόμενο βήμα της διαδικασίας είναι να εντοπίσουμε τυχών καινούργια αντικείμενα που θα βοηθήσουν στην ανάπτυξη της εφαρμογής. Πράγματι παρατηρούμε ότι πρέπει να χρησιμοποιήσουμε μια κλάση η οποία θα περιλαμβάνει τις καθολικές (global) μεταβλητές, οι οποίες θα είναι «ορατές» σε ολόκληρη την εφαρμογή. Η κλάση αυτή περιλαμβάνει και συναρτήσεις οι οποίες μπορούν να χρησιμοποιηθούν από οποιαδήποτε σημείο της εφαρμογής. Η καινούργια κλάση ονομάζεται *clsGlobal* και παρουσιάζεται στην συνέχεια:



Σχετικά με την επικοινωνία με την Βάση Δεδομένων είναι πολύ χρήσιμο να χρησιμοποιήσουμε μία κλάση η οποία θα περιλαμβάνει τις μεταβλητές και τις συναρτήσεις που είναι απαραίτητες για την επικοινωνία με την Βάση Δεδομένων. Έτσι, μπορούμε να χρησιμοποιήσουμε απλώς τις συναρτήσεις επικοινωνίας με την Βάση χωρίς να ξέρουμε το μηχανισμό με τον οποίο επιτυγχάνεται η επικοινωνία αυτή. Η κλάση επικοινωνίας με την Βάση Δεδομένων ονομάζεται *clsDBAccess*:



Τα καινούργια αντικείμενα τα προσθέτουμε στο Class διάγραμμα, που έχουμε ήδη δημιουργήσει για τα υπόλοιπα αντικείμενα της εφαρμογής, έτσι ώστε να έχουμε μια γενική άποψη για τα αντικείμενα της εφαρμογής και τον τρόπο με τον οποίο συσχετίζονται μεταξύ τους. Να τονίσουμε στο σημείο αυτό, ότι τα Class διαγράμματα αναπαριστούν την στατική συσχέτιση μεταξύ των αντικειμένων της εφαρμογής. Προκειμένου να δείξουμε την δυναμική επικοινωνία μεταξύ των αντικειμένων χρησιμοποιούμε τα Sequence (ή τα Collaboration) διαγράμματα.

Καταγραφή επικοινωνίας μεταξύ των Αντικειμένων

Αφού έχουμε εντοπίσει τα αντικείμενα, το επόμενο βήμα της διαδικασίας σχετίζεται με την καταγραφή του τρόπου επικοινωνίας μεταξύ των αντικειμένων της εφαρμογής. Στα προηγούμενα βήματα της διαδικασίας έχουμε δείξει τον στατικό τρόπο συσχέτισης μεταξύ των αντικειμένων. Η δυναμική επικοινωνία των αντικειμένων σχετίζεται με την ανταλλαγή μηνυμάτων μεταξύ των αντικειμένων στην διάρκεια του χρόνου.

Ανταλλαγή μηνυμάτων μεταξύ των αντικειμένων σημαίνει την κλήση μεθόδων μεταξύ των κλάσεων και παρίσταται με τα Sequence διαγράμματα. Εναλλακτικά, εάν δεν θέλουμε να παραστήσουμε την ανταλλαγή μηνυμάτων στην διάρκεια του χρόνου αλλά σε αύξουσα σειρά μπορούμε να χρησιμοποιήσουμε Collaboration διαγράμματα. Για χάριν ευκολίας της παρουσίασης στην συνέχεια θα χρησιμοποιήσουμε μόνο Sequence διαγράμματα.

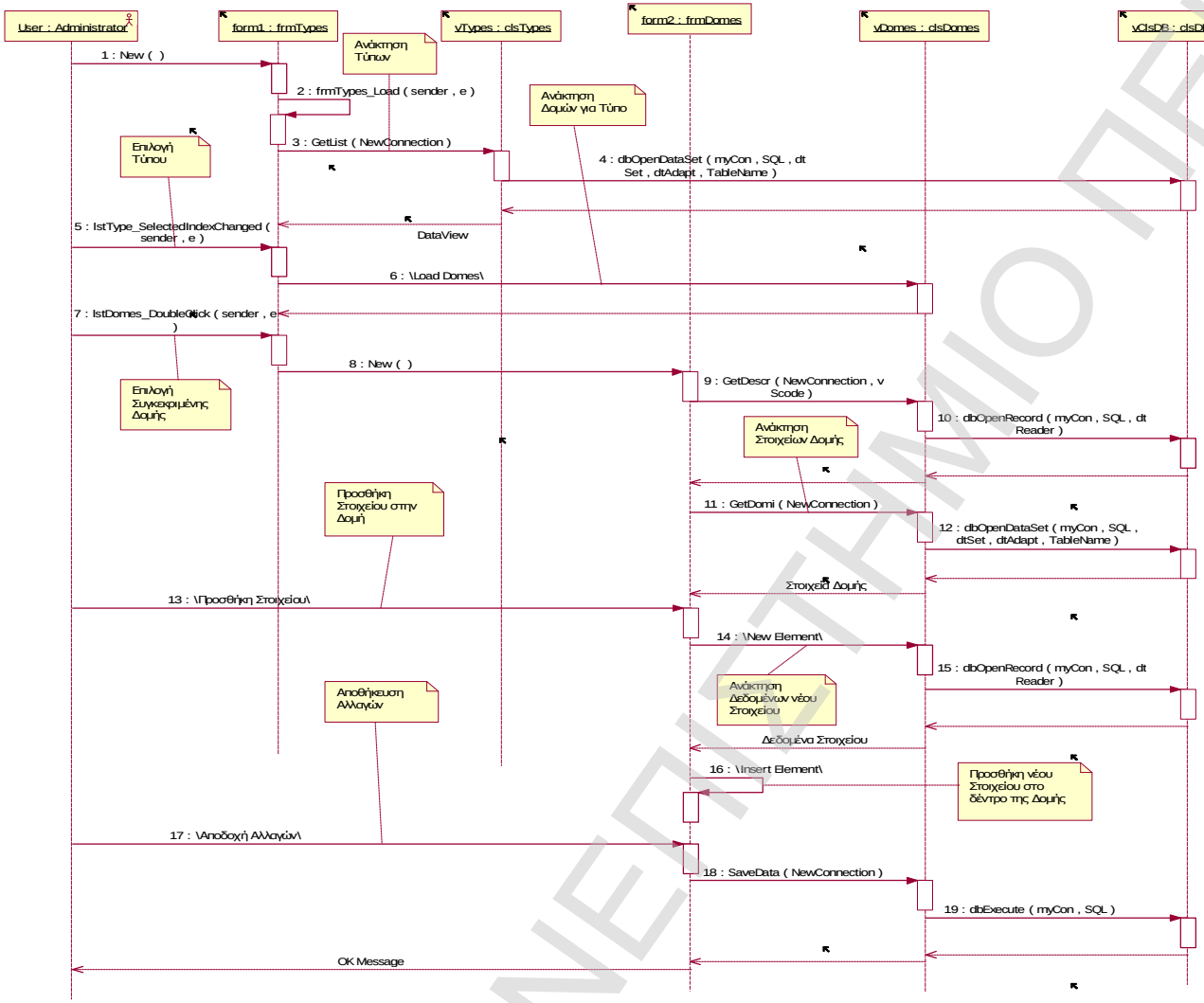
Η κατασκευή των Sequence διαγραμμάτων επιτυγχάνεται με την ανάλυση των Use Case και των Class διαγραμμάτων καθώς και των use case σεναρίων που έχουν δημιουργηθεί στα προηγούμενα βήματα της διαδικασίας. Στην συνέχεια παρουσιάζουμε τα Sequence διαγράμματα για τις λειτουργίες «Διαχείριση Δομών» και «Διαχείριση Προϋπολογισμών».

Χρήση Sequence Διαγραμμάτων

Sequence διάγραμμα: Διαχείριση Δομών

ΠΕΡΙΓΡΑΦΗ:	Ο Administrator συνδέεται στην οθόνη διαχείρισης Τύπων (*frmTypes*). Κατά το άνοιγμα της οθόνης διαχείρισης φορτώνονται οι διαθέσιμοι Τύποι από την Βάση Δεδομένων. Ο Administrator επιλέγει ένα συγκεκριμένο Τύπο φορτώνονται οι διαθέσιμες Δομές για τον επιλεγμένο Τύπο (λίστα Δομών). Ο Administrator επιλέγει μια συγκεκριμένη Δομή και ανοίγει η οθόνη διαχείρισης Δομών (*frmDomes*) . Γίνεται ανάκτηση της περιγραφής της Δομής καθώς και τα στοιχεία που αποτελούν την Δομή (δέντρο Δομής). Ο Administrator έχει την επιλογή να αλλάξει προσθέσει νέο στοιχείο στην Δομή. Φορτώνονται από την Βάση οι πληροφορίες του νέου στοιχείου και προστίθεται το νέο στοιχεί στο δέντρο της Δομής. Ο Administrator επιλέγει Αποδοχή. Οι πληροφορίες για την Δομή αποθηκεύονται στην Βάση Δεδομένων και ο Administrator ενημερώνεται για την επιτυχία της ενημέρωσης.

Sequence διάγραμμα «Διαχείριση Δομών»

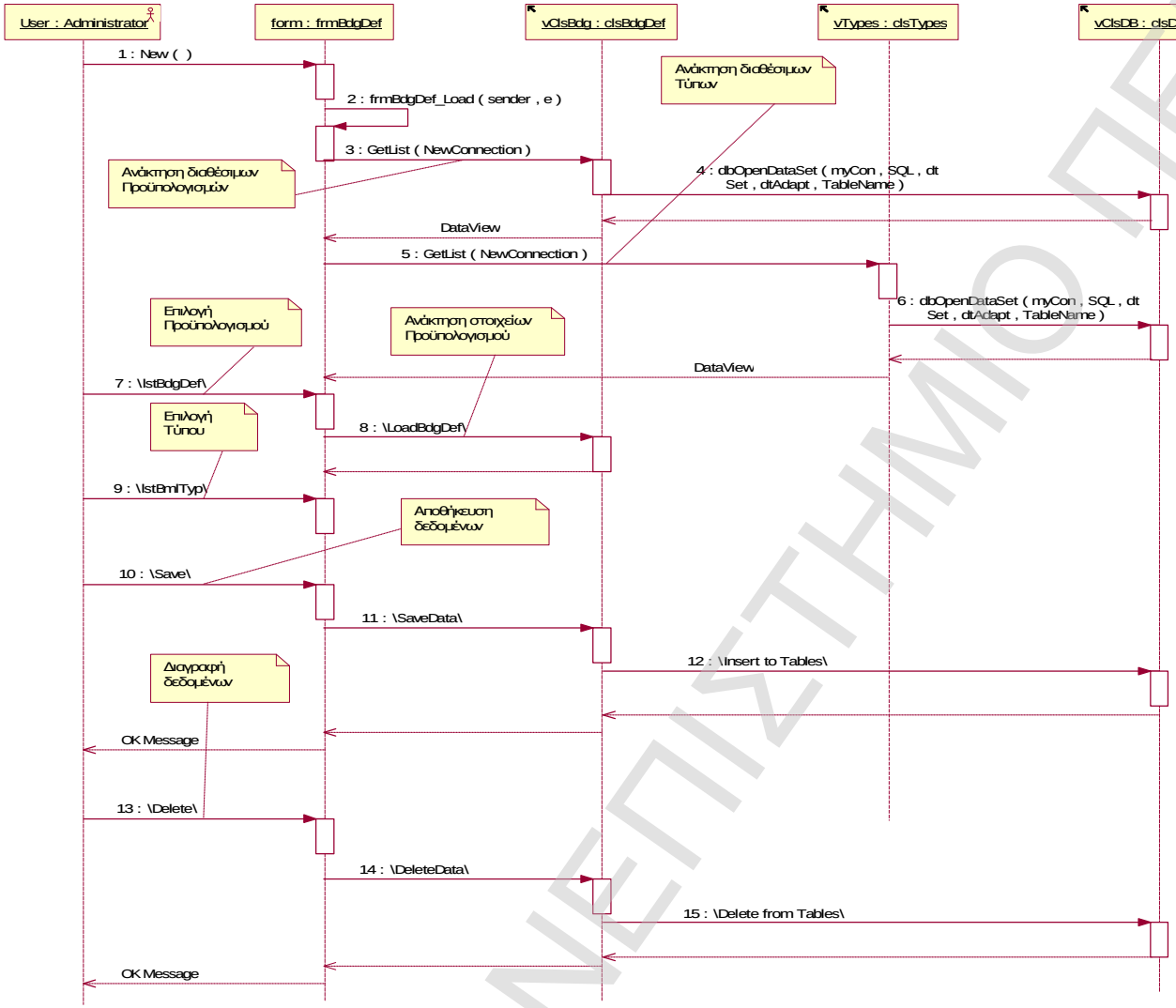


Sequence διάγραμμα: Διαχείριση Προϋπολογισμών

ΠΕΡΙΓΡΑΦΗ:	Ο Administrator συνδέεται στην οθόνη διαχείρισης Προϋπολογισμών (*frmBdgDef*). Κατά το άνοιγμα της οθόνης διαχείρισης φορτώνονται οι διαθέσιμοι Προϋπολογισμοί και οι Τύποι από την Βάση Δεδομένων. Ο Administrator επιλέγει ένα συγκεκριμένο Προϋπολογισμό και φορτώνονται οι διαστάσεις του Προϋπολογισμού, που επέλεξε (από 1 έως 7 διαστάσεις). Ο Administrator έχει την επιλογή να αλλάξει τους Τύπος για τις διαστάσεις του Προϋπολογισμού ή να προσθέσει μια νέα διάσταση και να αποθηκεύσει τις αλλαγές που έκανε. Επίσης υπάρχει η επιλογή διαγραφής του επιλεγμένου Προϋπολογισμού. Ο Administrator ενημερώνεται για την επιτυχία της αποθήκευσης ή της διαγραφής.

Το αντίστοιχο Sequence διάγραμμα της λειτουργίας «Διαχείριση Προϋπολογισμών» παρουσιάζεται στη συνέχεια.

Sequence διάγραμμα «Διαχείριση Προϋπολογισμών»



Σχεδιασμός

Μετά από την ανάλυση των αντικειμένων της εφαρμογής ακολουθεί το τρίτο βήμα του Σχεδιασμού. Στο βήμα αυτό σχεδιάζουμε τα διάφορα αντικείμενα έτσι ώστε στην συνέχεια οι προγραμματιστές της εφαρμογής να μπορούν εύκολα να τα υλοποιήσουν σε μια γλώσσα προγραμματισμού. Στο βήμα αυτό επίσης ασχολούμαστε με διάφορα τεχνικά θέματα όπως ο σχεδιασμός των βιβλιοθηκών κώδικα και οθονών που θα χρησιμοποιηθούν στην εφαρμογή καθώς επίσης και με τον σχεδιασμό της αρχιτεκτονικής βάσει της οποίας θα υλοποιηθεί η εφαρμογή.

Οι βασικές λειτουργίες του βήματος του Σχεδιασμού της εφαρμογής παρουσιάζονται στη συνέχεια.

Βελτίωση των Αντικειμένων

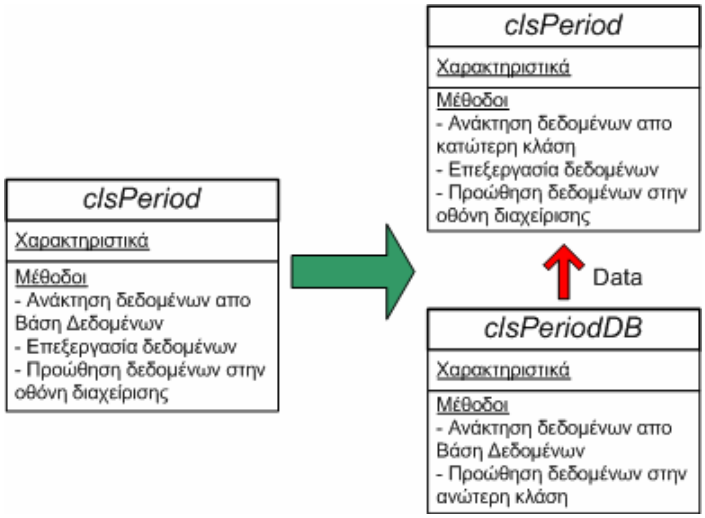
Η αρχιτεκτονική εφαρμογών πάνω στην οποία θα στηριχθεί η εφαρμογή που αναπτύσσουμε είναι η N-Tier Αρχιτεκτονική, που περιλαμβάνει τα παρακάτω επίπεδα:

- Presentation Layer
- Business Logic Layer
- Data Access Layer
- Data Layer

Κάθε επίπεδο θέλουμε να είναι λογικά απομονωμένο το ένα από το άλλο έτσι το κάθε ένα από αυτά να εκτελεί κάποιο συγκεκριμένη λειτουργία. Για παράδειγμα, το επίπεδο Business Logic θέλουμε να ασχολείται με την επεξεργασία των δεδομένων και την εφαρμογή των κανόνων και της λογικής του οργανισμού. Το επίπεδο Data Access θέλουμε να ασχολείται μόνο με την επικοινωνία με την Βάση Δεδομένων και τα υπόλοιπα αρχεία που περιέχουν δεδομένα (επίπεδο Data). Τέλος το επίπεδο Presentation θέλουμε να ασχολείται με την παρουσίαση των δεδομένων χωρίς κάποια περαιτέρω επεξεργασία τους.

Προκειμένου να επιτύχουμε τον διαχωρισμό λειτουργιών μεταξύ της ανάκτησης των δεδομένων από την Βάση και της επεξεργασία τους θεωρούμε ότι κάθε κλάση της εφαρμογής πρέπει να χωριστεί σε τουλάχιστον δύο οντότητες. Η πρώτη κλάση θα ασχολείται με την επεξεργασία των δεδομένων και την προώθηση τους στο επίπεδο της παρουσίασης και η δεύτερη κλάση θα ασχολείται με την ανάκτηση των δεδομένων από την Βάση και την προώθηση τους στην πρώτη κλάση.

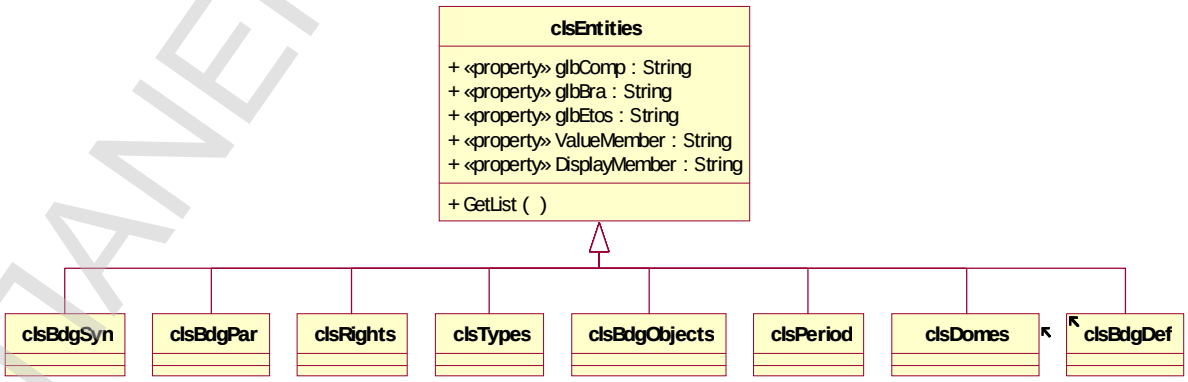
Κάθε κλάση που έχουμε χρησιμοποιήσει έως τώρα και αναφέρονται στις βασικές οντότητες της εφαρμογής θα την χωρίσουμε σε δύο νέες κλάσεις. Η πρώτη κλάση θα διατηρεί το όνομα που έχει έως τώρα δηλαδή με το πρόθεμα *cls* και το όνομα της οντότητας. Η δεύτερη κλάση, που θα ασχολείται με την επικοινωνία με την Βάση Δεδομένων, θα διατηρεί το όνομα της πρώτης κλάσης (*cls+Ονομα_Οντότητας*) και θα προσθέσουμε το επίθεμα *DB* για να δείχνει ότι επικοινωνεί με την Βάση Δεδομένων.



Διάγραμμα 33: Διαχωρισμός Κλάσεων

Παρατηρούμε ότι οι βασικές κλάσεις της εφαρμογής (Δομές, Προϋπολογισμοί, Αντικείμενα κλπ.) έχουν κάποια κοινά χαρακτηριστικά όπως κάποιες μεταβλητές για την εμφάνιση των αποτελεσμάτων στην οθόνη (συγκεκριμένα οι μεταβλητές *ValueMember* και *DisplayMember* οι οποίες χρησιμοποιούνται για την σύνδεση των δεδομένων που ανακτώνται από την Βάση Δεδομένων προς τα διάφορα combo boxes των οθονών της εφαρμογής) και κάποιες μεταβλητές που σχετίζονται με το Έτος, την Εταιρεία και το Υποκατάστημα που αναφέρονται τα δεδομένα. Επίσης υπάρχουν και κάποιες κοινές μέθοδοι όπως η *GetList* για την ανάκτηση των δεδομένων και την προώθηση τους στην οθόνη παρουσίασης.

Για να μην επαναλαμβάνεται ο ίδιος κώδικας σε κάθε κλάση της εφαρμογής μπορούμε να εκμεταλλευτούμε το πλεονέκτημα που μας παρέχει ο αντικειμενοστραφής προγραμματισμός, το οποίο δεν είναι άλλο από την Κληρονομικότητα. Έτσι ορίζουμε μια γενική Κλάση-Πατέρας που περιέχει όλα τα κοινά χαρακτηριστικά των οντοτήτων έστω την *clsEntities* η οποία προσφέρει τα περιεχόμενα της σε όλες τις υπόλοιπες βασικές κλάσεις της εφαρμογής. Στη συνέχεια η κάθε κλάση ξεχωριστά υλοποιεί τις μεθόδους της ξεχωριστά ανάλογα πάντα με τον τύπο της κλάσης.

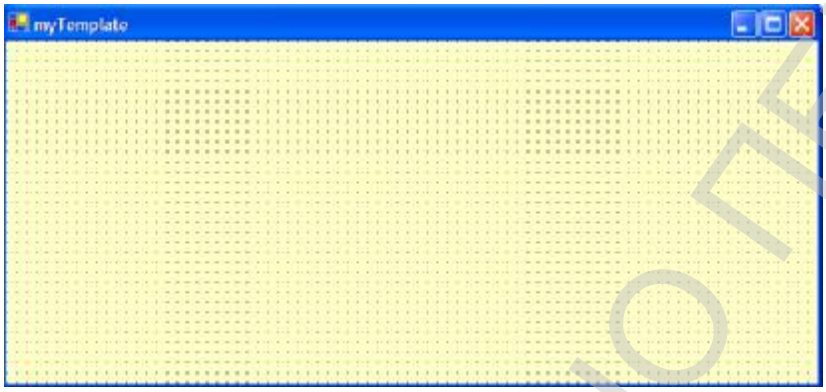


Διάγραμμα 34: Κληρονομικότητα στις βασικές κλάσεις

Σχεδιασμός User Interface

Τα πλεονεκτήματα της Κληρονομικότητας μπορούμε να τα εκμεταλλευτούμε και στην σχεδίαση των οθονών της εφαρμογής. Έτσι μπορούμε να δημιουργήσουμε ορισμένα πρότυπα οθονών με συγκεκριμένα χαρακτηριστικά (π.χ. χρωματισμός οθόνης, menu, διάφορα controls κλπ.) και στη συνέχεια τα πρότυπα αυτά να κληρονομούνται στις υπόλοιπες οθόνες της εφαρμογής.

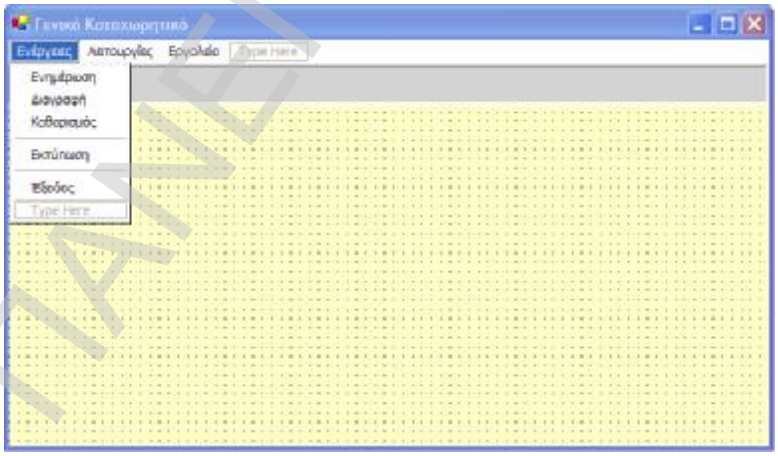
Δημιουργούμε αρχικά μια οθόνη την *myTemplate* η οποία θα διαχειρίζεται τον χρωματισμό της οθόνης.



Διάγραμμα 35: Παράδειγμα οθόνης myTemplate

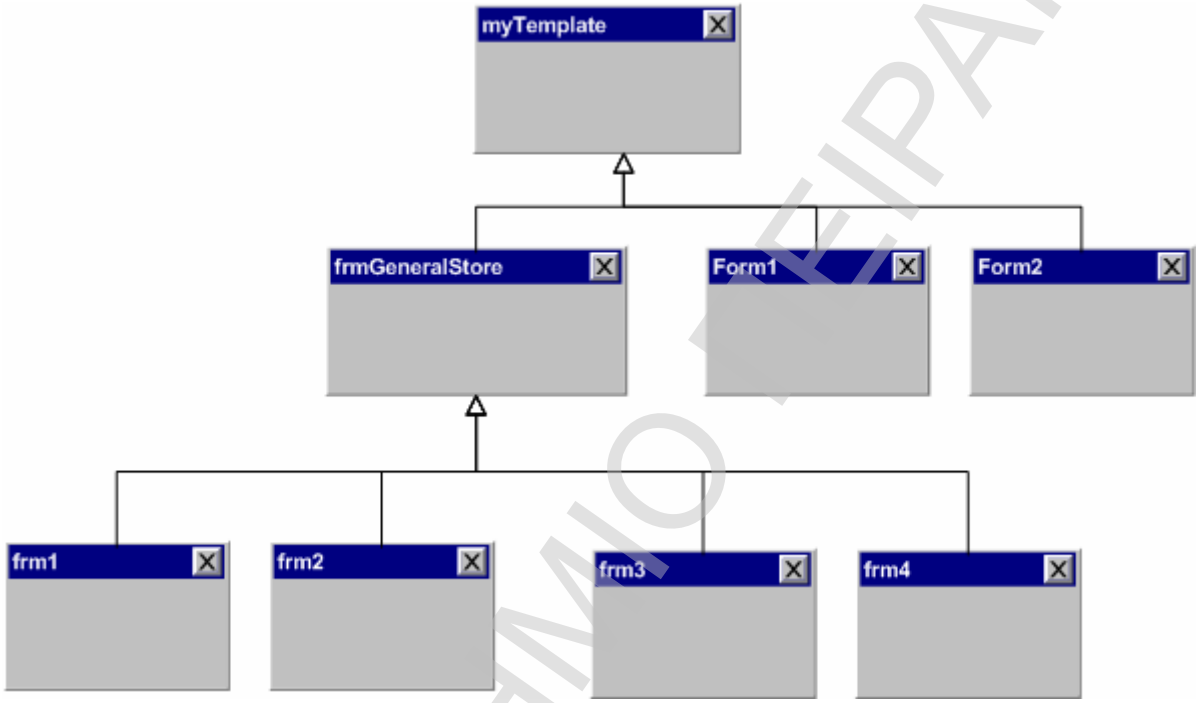
Την παραπάνω οθόνη την κληρονομούν όλες οι οθόνες της εφαρμογής προκειμένου να έχουν όλες τον ίδιο χρωματισμό. Μια από τις από τις οθόνες που θα κληρονομεί την myTemplate είναι η οθόνη *frmGeneralStore*, η οποία θα είναι μια οθόνη-πατέρας για όλες τις οθόνες που θα χρησιμοποιούνται για καταχώριση δεδομένων. Η οθόνη *frmGeneralStore* περιέχει το menu της οθόνης καθώς και τα διάφορα κουμπιά που εμφανίζονται σε κάθε οθόνη και εκτελούν συγκεκριμένες λειτουργίες.

Επίσης ο οθόνη αυτή θα διαχειρίζεται τα διάφορα events που συμβαίνουν κατά το άνοιγμα και το κλείσιμο της οθόνης. Ένα παράδειγμα της οθόνης αυτής, παρουσιάζεται παρακάτω:



Διάγραμμα 36: Παράδειγμα οθόνης frmGeneralStore

Δημιουργούμε μια ιεραρχία στις οθόνες της εφαρμογής γεγονός που απλοποιεί την διαχείριση τους. Για παράδειγμα, εάν θέλουμε να κάνουμε μια αλλαγή στον χρωματισμό των οθονών αλλάζουμε το χρώμα στην αρχική οθόνη-πατέρας ***myTemplate*** και όλες οι αλλαγές κληρονομούνται στις οθόνες-παιδιά. Το παρακάτω διάγραμμα δείχνει την σχέση κληρονομικότητας μεταξύ των οθονών της εφαρμογής μας.



Διάγραμμα 37: Κληρονομικότητα στις οθόνες της εφαρμογής

Σχεδιασμός Βιβλιοθηκών Κλάσεων και Οθονών

Το βήμα αυτό περιλαμβάνει την δημιουργία των απαραίτητων βιβλιοθηκών κλάσεων και οθονών για να μπορούμε εύκολα να τις διαχειριστούμε και να έχουμε την δυνατότητα επαναχρησιμοποίησης τους και σε άλλες εφαρμογές. Θέλουμε να χωρίσουμε τις κλάσεις της εφαρμογής μας σε διαφορετικές βιβλιοθήκες κλάσεων έτσι ώστε να μπορούν να υποστηρίξουν την αρχιτεκτονική N-tier της εφαρμογής μας, να χρησιμοποιούνται από διάφορες εφαρμογές και να μπορούν εύκολα να αλλάξουν χωρίς να επηρεάζουν τα υπόλοιπα τμήματα της εφαρμογής μας.

Επιπλέον θέλουμε να χωρίσουμε και τις οθόνες της εφαρμογής μας σε ξεχωριστές βιβλιοθήκες κλάσεων έτσι ώστε να μπορούν να χρησιμοποιούνται από διάφορες εφαρμογές που έχουν αναπτυχθεί στον οργανισμό.

Μέσα από την ανάλυση των απαιτήσεων τόσο της εφαρμογής όσο και των αναγκών του οργανισμού για υπάρχουσες και για νέες εφαρμογές προτείνουμε τις παρακάτω βιβλιοθήκες κλάσεων και οθονών:

Ονομασία	Περιγραφή Βιβλιοθήκης
myBudget	Αποτελεί την windows[14] εφαρμογή που αναπτύσσουμε. Περιλαμβάνει τις κλάσεις και τις οθόνες, που χρησιμοποιούνται αποκλειστικά από την εφαρμογή αυτή. Είναι το εκτελέσιμο αρχείο (.exe) που τρέχουν οι χρήστες.
myLib	Περιλαμβάνει τις οθόνες που χρησιμοποιούνται από την εφαρμογή myBydget καθώς και από οποιαδήποτε άλλη windows εφαρμογή του οργανισμού. Δεν χρησιμοποιείται από web εφαρμογές. Είναι ένα .dll αρχείο.
myBnDLib	Περιλαμβάνει τις κλάσεις που χρησιμοποιούνται από την εφαρμογή myBudget καθώς και από οποιαδήποτε άλλη εφαρμογή του οργανισμού (windows ή web εφαρμογή). Είναι ένα .dll αρχείο.
myBnDBudget	Περιλαμβάνει τις κλάσεις, που χρησιμοποιούνται μόνο από την εφαρμογή myBudget καθώς και από κάποια web εφαρμογή, που χρειάζεται κάποιες κλάσεις της myBudget. Είναι ένα .dll αρχείο.

Πίνακας 6: Βιβλιοθήκες Κλάσεων και Οθονών

Σχεδιασμός Αρχιτεκτονικής Εφαρμογής

Η αρχιτεκτονική της εφαρμογής που αναπτύσσουμε αποτελείται από τα παρακάτω επίπεδα:

- Presentation Layer – Επίπεδο παρουσίασης δεδομένων
- Business Logic Layer – Επίπεδο διαχείρισης δεδομένων
- Data Access Layer – Επίπεδο ανάκτησης δεδομένων
- Data Layer – Βάση Δεδομένων και άλλα αρχεία δεδομένων

Προκειμένου να υποστηριχτεί η αρχιτεκτονική που προτείνουμε για την ανάπτυξη της εφαρμογής οι κλάσεις και οι οθόνες της εφαρμογής πρέπει να είναι λογικά (τουλάχιστον) και σε μερικές περιπτώσεις φυσικά χωρισμένες σε διάφορα επίπεδα που αναπαριστούν τα ξεχωριστά επίπεδα της εφαρμογής. Για τον λόγο αυτό έχουμε δημιουργήσει τις βιβλιοθήκες των κλάσεων και τον οθονών. Οι βιβλιοθήκες αυτές είναι φυσικά απομονωμένες από τις υπόλοιπες δηλαδή είναι αποθηκευμένες σε διαφορετικό χώρο στον δίσκο. Κάθε μια όμως βιβλιοθήκη όπως θα δούμε παρακάτω δεν ανήκει αναγκαστικά σε ένα μόνο επίπεδο της αρχιτεκτονικής αλλά κάποιες βιβλιοθήκες μπορούν λογικά να ανήκουν σε δυο επίπεδα. Λέγοντας «λογικά» σε αντιδιαστολή με το «φυσικά» εννοούμε τον φυσικό διαχωρισμό των επιπέδων σε διαφορετικά σημεία αποθήκευσης στον δίσκο.

Στόχος μας είναι να έχουμε όσο το δυνατόν περισσότερες απομονωμένες συλλογές κλάσεων έτσι ώστε η αλλαγή σε ένα επίπεδο να μην επηρεάζει τα υπόλοιπα επίπεδα καθώς επίσης να υπάρχει η δυνατότητα χρησιμοποίησης των ίδιων κλάσεων σε περισσότερες εφαρμογές εκτός από αυτήν που αναπτύσσουμε.

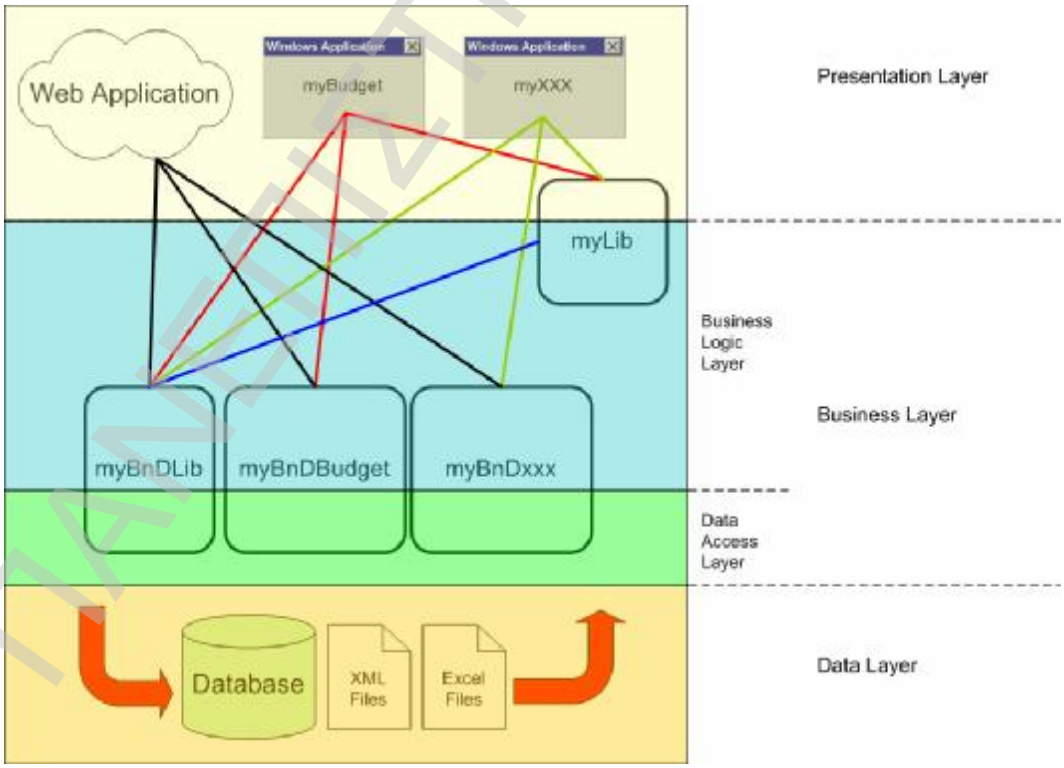
Ξεκινώντας από το τελευταίο επίπεδο και προχωρώντας προς τα πάνω έχουμε τα παρακάτω επίπεδα:

[14] Windows εφαρμογή είναι η εφαρμογή, που χρησιμοποιεί το παραθυρικό περιβάλλον των Microsoft Windows. Εναλλακτικά υπάρχουν οι web εφαρμογές, που λειτουργούν με την χρήση ενός Internet browser όπως ο Microsoft Internet Explorer.

Επίπεδο	Βιβλιοθήκες	Παρατηρήσεις
Data Layer	Βάση Δεδομένων, XML και Excel αρχεία	Στο επίπεδα αυτό περιλαμβάνεται η Βάση Δεδομένων καθώς επίσης και κάποια XML αρχεία και αρχεία Excel. Το επίπεδο αυτό είναι τελείως απομονωμένο από τα υπόλοιπα επίπεδα έτσι ώστε οποιαδήποτε αλλαγή στον τρόπο αποθήκευσης των δεδομένων να μην επηρεάζει την λειτουργία των εφαρμογών.
Data Access Layer	*myBnDLib* *myBnDBudget*	Στο επίπεδο αυτό περιλαμβάνονται οι κλάσεις *DB* των βιβλιοθηκών *myBnDLib* και *myBnDBudget* οι οποίες χρησιμοποιούνται για ανάκτηση και αποστολή δεδομένων επίπεδο Data.
Business Logic Layer	*myLib* *myBnDLib* *myBnDBudget*	Στο επίπεδο περιλαμβάνονται οι κλάσεις των βιβλιοθηκών *myLib*, *myBnDLib* και *myBnDBudget*. Οι κλάσεις αυτές χρησιμοποιούνται για την επεξεργασία των δεδομένων και την προώθηση τους στο επίπεδο Presentation.
Presentation Layer	*myLib* *myBudget*	Στο επίπεδο αυτό περιλαμβάνονται οι οθόνες της εφαρμογής μας και της βιβλιοθήκης *myLib* που χρησιμοποιούνται για την παρουσίαση των δεδομένων στους χρήστες της εφαρμογής.

Πίνακας 7: Σχεδιασμός Αρχιτεκτονικής Εφαρμογής

Στο παρακάτω διάγραμμα παρουσιάζεται η αρχιτεκτονική που προτείνουμε για όλες τις εφαρμογές (windows και web) της ΒΙΟΣΕΡ.

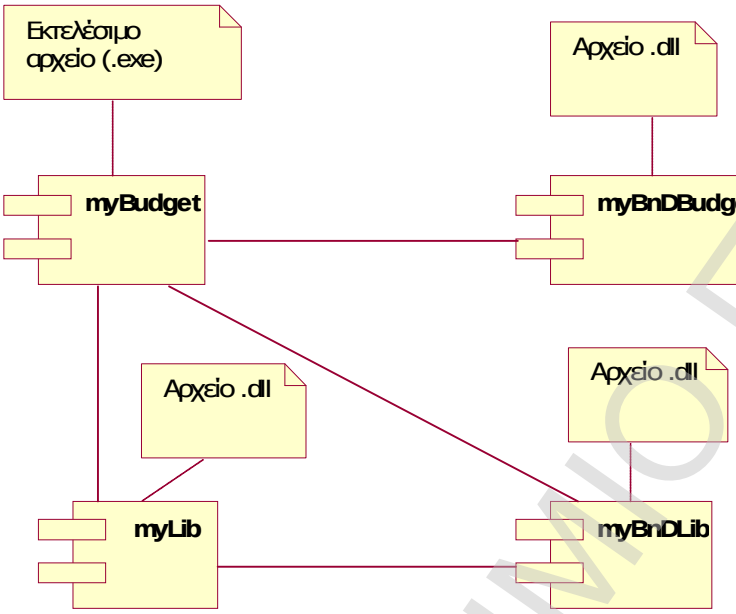


Διάγραμμα 38: Αρχιτεκτονική Εφαρμογών της ΒΙΟΣΕΡ

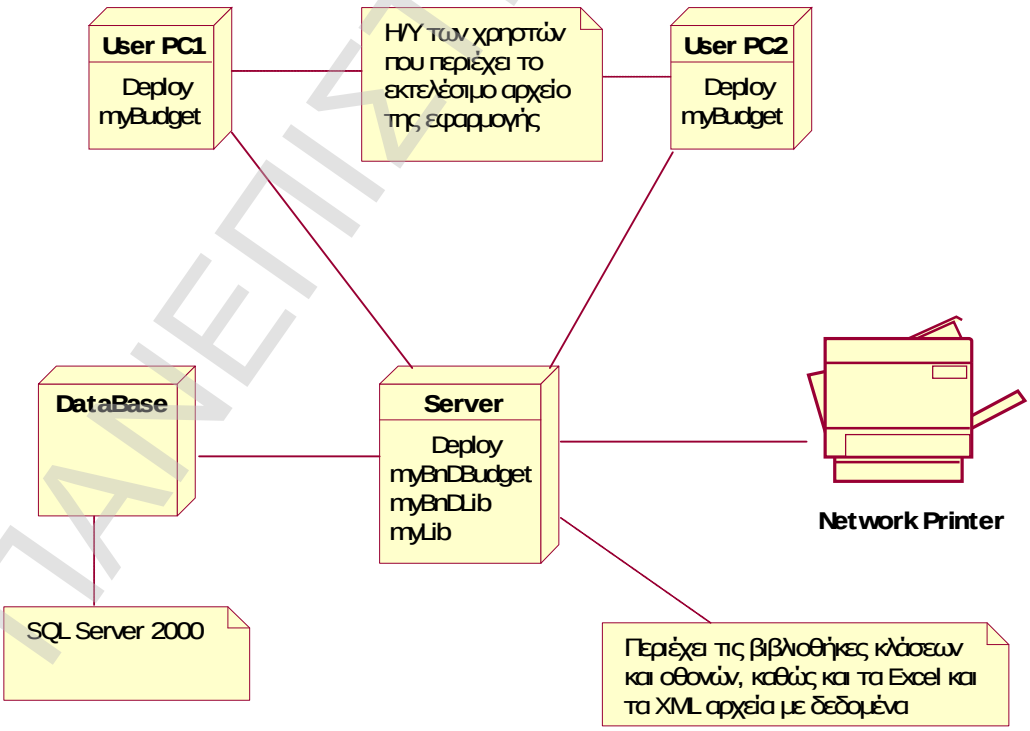
Χρήση Component και Deployment Διαγραμμάτων

Στην συνέχεια παρουσιάζουμε τα Deployment και τα Component διαγράμματα που αναπαριστούν την αρχιτεκτονική της εφαρμογής.

Component διάγραμμα εφαρμογής



Deployment διάγραμμα εφαρμογής



Τεκμηρίωση Διαγραμμάτων και Οθονών

Σε αυτό το βήμα της διαδικασίας ολοκληρώνεται η τεκμηρίωση των διαγραμμάτων της διαδικασίας. Διορθώνονται τυχόν λάθη στα διαγράμματα και τοποθετούνται σχόλια προκειμένου να δοθούν για χρήση από τους προγραμματιστές της εφαρμογής. Τα διαγράμματα θα χρησιμοποιηθούν στο επόμενο βήμα της ανάπτυξης κώδικα και χρειάζεται η τεκμηρίωση τους για να βλέπουν οι ενδιαφερόμενοι τα αποτελέσματα της ανάλυσης και του σχεδιασμού της εφαρμογής.

Γίνεται τεκμηρίωση των οθονών της εφαρμογής. Συγκεκριμένα οι οθόνες-πρότυπα που έχουν σχεδιαστεί παρουσιάζονται στους προγραμματιστές της εφαρμογής προκειμένου αυτοί να χρησιμοποιήσουν τα πρότυπα αυτά για την σχεδίαση των οθονών της εφαρμογής. Τέλος, βελτιώνεται και ολοκληρώνεται η συμπλήρωση του γλωσσάριου των όρων που χρησιμοποιούνται στην εφαρμογή.

Το γλωσσάριο των όρων της διαδικασίας παρουσιάζεται στην συνέχεια.

Γλωσσάρι Όρων Διαδικασίας

Όρος	Περιγραφή
Αναφορές:	Οι εκτυπώσεις της εφαρμογής, που παρουσιάζουν τα διάφορα αποτελέσματα της επεξεργασίας των δεδομένων. Είναι απαραίτητο στοιχείο κάθε εφαρμογής γιατί είναι ένα χρήσιμο εργαλείο ελέγχου και εκτίμησης της απόδοσης των Κέντρων Ευθύνης του οργανισμού.
Αντικείμενο:	Η οντότητα που αποθηκεύει τις επιλογές του χρήστη για μια Παρουσίαση. Ένας χρήστης ανάλογα με τον Ρόλο στον οποίο ανήκει έχει Δικαίωμα να δει ορισμένες Παρουσιάσεις. Στην συνέχεια για κάθε Παρουσίαση μπορεί να επιλέξει ορισμένα στοιχεία των Δομών της κάθε διάστασης της Παρουσίασης και να τα αποθηκεύσει. Έτσι όποτε ο χρήστης θέλει να δει δεδομένα για τα στοιχεία που τον ενδιαφέρουν επιλέγει να δει το συγκεκριμένο Αντικείμενο.
Αποτέλεσμα:	Η οντότητα που διαχειρίζεται την ομαδοποίηση διαφόρων Παρουσιάσεων και την παρουσίαση του αποτελέσματος. Για παράδειγμα, εάν έχουμε πολλές Παρουσιάσεις που σχετίζονται με έξοδα και έσοδα το Αποτέλεσμα ομαδοποιεί τις Παρουσιάσεις αυτές, κάνει προσθέσεις και αφαιρέσεις (ανάλογα με τον Τύπο τους) και εμφανίζει το τελικό Αποτέλεσμα (άθροισμα) στον χρήστη.
Αρχιτεκτονική N-tier:	Η αρχιτεκτονική που σχετίζεται με τον διαχωρισμό των εφαρμογών ενός οργανισμού σε ξεχωριστά επίπεδα. Κάθε επίπεδο είναι σαφώς ορισμένο και εκφράζει μια γενική λειτουργία της εφαρμογής. Ο διαχωρισμός των επιπέδων είναι φυσικός είτε λογικός.
Βιβλιοθήκες:	Η ομαδοποίηση κλάσεων και οθονών για να μπορούμε εύκολα να τις διαχειριστούμε και να έχουμε την δυνατότητα επαναχρησιμοποίησης τους και σε άλλες εφαρμογές. Οι κλάσεις και οι οθόνες μπορούν να χρησιμοποιούνται από διάφορες εφαρμογές και να μπορούν εύκολα να αλλάξουν χωρίς να επηρεάζουν τα υπόλοιπα τμήματα της εφαρμογής.
Διαχειριστής (Administrator):	Ο ειδικός χρήστης ο οποίος έχει συγκεκριμένες αρμοδιότητες στην εφαρμογή. Συγκεκριμένα ο διαχειριστής ασχολείται με τον σχεδιασμό και την διαχείριση των βασικών οντοτήτων της εφαρμογής, όπως τους Προϋπολογισμούς, τις Περιόδους, τα Αντικείμενα και τις Παρουσιάσεις.

Δικαιώματα:	Τα στοιχεία που έχει δικαίωμα να βλέπει ο χρήστης βάσει του Ρόλου στον οποίο ανήκει. Τα στοιχεία που έχει δικαίωμα να βλέπει ο χρήστης είναι οι Παρουσιάσεις και τα Αντικείμενα.
Δομή:	Το αντικείμενο που παρουσιάζει την ιεραρχική δομή διάφορων Τύπων (Δομές Πελατών, Ειδών, Εξόδων κλπ.) Για κάθε Τύπο μπορούμε να έχουμε περισσότερες από μία Δομές προκειμένου να παραστήσουμε την ίδια πληροφορία με διαφορετικό τρόπο. Με μία Δομή μπορούμε να αναπαραστήσουμε και να διαχειριστούμε οποιαδήποτε οντότητα μέσα στην εφαρμογή. Επίσης μπορούμε να συνδυάσουμε δυο ή περισσότερες Δομές για να δείξουμε διάφορες πληροφορίες (π.χ. Έσοδα ανά Πελάτη, Έσοδα ανά Είδος, Πωλήσεις ανά Πελάτη και ανά Είδος κλπ.)
Μέθοδοι (Κλάσης):	Οι συναρτήσεις που ανήκουν σε μια Κλάση (Αντικείμενο). Οι μέθοδοι μπορεί να είναι τοπικές (ορατές μόνο μέσα στην κλάση) ή καθολικές (ορατές από όλες τις κλάσεις της εφαρμογής)
Μηνύματα:	Οι μέθοδοι των κλάσεων οι οποίες χρησιμοποιούνται για την επικοινωνία μεταξύ των κλάσεων. Δείχνουν τον τρόπο επικοινωνίας μεταξύ των διάφορων αντικειμένων σε μια αντικειμενοστραφή εφαρμογή.
Παρουσίαση:	Η οντότητα που καθορίζει τι στοιχεία θα περιέχει η κάθε διάσταση. Κατασκευάζεται πάνω σε ένα Προϋπολογισμό ο οποίος καθορίζει τον αριθμό των διαστάσεων (από 1 έως 7) καθώς και τον Τύπο που θα έχει η κάθε διάσταση. Για κάθε διάσταση της Παρουσίασης επιλέγουμε μια συγκεκριμένη Δομή που να ανήκει στον Τύπο της κάθε διάστασης. Με την Παρουσίαση έχουμε την δυνατότητα να ομαδοποιήσουμε διαφορετικά στοιχεία (Πελάτες, Έξοδα, Είδη κλπ.) κάτω από ένα κοινό όνομα.
Περίοδος:	Η διάσταση του χρόνου βάσει του οποίου θέλουμε να γίνει η παρουσίαση των δεδομένων. Για παράδειγμα έχουμε μηνιαία παρουσίαση (12 μήνες του έτους), κυλιόμενες περιόδους (από κάποιο Μήνα/Ετος έως κάποιο Μήνα/Ετος) ή μια Δομή Περιόδου (ιεραρχική παρουσίαση του χρόνου που έχει δημιουργηθεί από τον χρήστη, π.χ. προβολή ανά ΕΤΟΣ-ΤΕΤΡΑΜΗΝΑ-ΜΗΝΑΣ).
Προϋπολογισμοί:	Ένας «καμβάς» πάνω στον οποίο δημιουργούμε από 1 έως 7 διαστάσεις που περιλαμβάνουν Τύπους. Ένας Προϋπολογισμός δεν εκτελεί κάποια συγκεκριμένη λειτουργία αλλά χρησιμοποιείται για να κατασκευαστούν πάνω σε αυτόν οι Παρουσιάσεις που χρησιμοποιούνται στις εφαρμογές.
Προϋπολογισμός (Budgeting):	Ένα χρήσιμο εργαλείο για βραχυπρόθεσμο προγραμματισμό και έλεγχο σε ένα οργανισμό. Ένας λειτουργικός προϋπολογισμός (operational budget) συνήθως αφορά στο τρέχον ημερολογιακό έτος και παρουσιάζει των προγραμματισμό των εσόδων και των εξόδων για το έτος αυτό.
Ρόλος:	Ο ρόλος που έχει κάθε χρήστης που συνδέεται στην εφαρμογή. Βάσει του ρόλου ορίζεται τι έχει δικαίωμα να βλέπει ο χρήστης. Ένας ρόλος μπορεί να είναι μια συγκεκριμένη οντότητα (ΔΙΕΥΘΥΝΤΗΣ, ΥΠΕΥΘΥΝΟΣ, ΧΡΗΣΤΗΣ) ή κάποιο τμήμα του οργανογράμματος του οργανισμού (ΤΜΗΜΑ ΠΩΛΗΣΕΩΝ, ΛΟΓΙΣΤΗΡΙΟ που περιέχει μέσα του διάφορους ρόλους).
Στόχος:	Η οντότητα που διαχειρίζεται τους στόχους του προϋπολογισμού που καταχωρούν οι χρήστες στο σύστημα. Καταχωρούνται για συγκεκριμένες Περιόδους και για να τους δει ο χρήστης πρέπει να έχει

	τα αντίστοιχα Δικαιώματα.
Συνδεδεμένος Χρήστης (Valid User) :	Ο χρήστης ο οποίος έχει συνδεθεί με επιτυχία στην εφαρμογή. Επιτυχία σύνδεσης στην εφαρμογή σημαίνει ότι ο χρήστης έχει δώσει σωστό κωδικό (password) και το αναγνωριστικό του (username) έχει δικαιώματα πρόσβασης στην εφαρμογή.
Τύποι:	Μια γενική οντότητα του συστήματος όπως Πελάτες, Είδη, Έξοδα. Ένας Τύπος δεν εκτελεί κάποιες συγκεκριμένες λειτουργίες αλλά χρησιμοποιείται σαν βάση για την δημιουργία άλλων οντοτήτων όπως οι Δομές.
Χαρακτηριστικά (Κλάσης):	Είναι οι μεταβλητές (variables), οι ιδιότητες (Properties) και οι μεταβλητές Structure που ανήκουν σε μια Κλάση.
Χρήστης:	Ο συνδεδεμένος χρήστης ο οποίος δουλεύει την εφαρμογή. Ανάλογα με τον Ρόλο που ανήκει ο κάθε χρήστης έχει και ανάλογα Δικαιώματα προβολής και επεξεργασίας δεδομένων στην εφαρμογή.

Πίνακας 8: Γλωσσάρι Όρων Διαδικασίας

Ανάπτυξη και Έλεγχος

Τα δυο τελευταία βήματα της αντικειμενοστραφούς διαδικασίας που παρουσιάζουμε είναι αυτά της Ανάπτυξης και του Ελέγχου. Στο βήμα της Ανάπτυξης οι προγραμματιστές της εφαρμογής προχωράνε στην συγγραφή κώδικα καθώς και στην δημιουργία όλων των οθονών που θα χρησιμοποιηθούν στην εφαρμογή.

Τέλος στο τελευταίο βήμα του Ελέγχου με την βοήθεια διαφόρων διαγραμμάτων της UML τα οποία παρουσιάζουν τις διάφορες όψεις της εφαρμογής προσπαθούμε να ελέγξουμε εάν οι λειτουργίες της εφαρμογής είναι σύμφωνες με τις απαιτήσεις των χρηστών. Επίσης γίνεται έλεγχος του κώδικα της εφαρμογής για τυχόν λάθη στις απαιτήσεις του συστήματος.

Κεφάλαιο 8. Συμπεράσματα – Προτάσεις

Συμπεράσματα

Στην παρούσα εργασία προσπαθήσαμε να αναφέρουμε κάποια από τα θέματα που απασχολούν τις επιχειρήσεις που αναπτύσσουν εφαρμογές λογισμικού και συγκεκριμένα την Βιομηχανία Παραγωγής Ορών ΒΙΟΣΕΡ Α.Ε. Τα θέματα αυτά σχετίζονται με την μετάβαση από τον συνήθη χρήση του δομημένου προγραμματισμού στην λογική της χρήσης αντικειμένων (ανάπτυξη εφαρμογών με αντικειμενοστραφή προγραμματισμό), τον σχεδιασμό της αρχιτεκτονικής των εφαρμογών, της βάσης δηλαδή πάνω στην οποία δομείται η εφαρμογή (n-Tier Αρχιτεκτονική) καθώς και την παρουσίαση μιας διαδικασίας ανάπτυξης αντικειμενοστραφών εφαρμογών με την χρήση της UML (ένα κοινό πρότυπο ανάλυσης, σχεδιασμού και τεκμηρίωσης εφαρμογών) η οποία θα μπορεί να χρησιμοποιηθεί στην ανάπτυξη κάθε αντικειμενοστραφούς εφαρμογής της εταιρείας ΒΙΟΣΕΡ.

Πέρα από το πρόβλημα της ανάπτυξης αντικειμενοστραφών εφαρμογών ένα άλλο ζήτημα που απασχολεί την ΒΙΟΣΕΡ είναι το πρόβλημα του ελέγχου της λειτουργίας και της επίτευξης των στόχων του οργανισμού. Παρουσιάστηκε μια διαδικασία Ελέγχου Διοίκησης, η οποία βασίζεται πάνω στην θεωρία των Συστημάτων Ελέγχου Διοίκησης (Management Control Systems) και στην Balanced Scorecard (τεχνική αποτίμησης απόδοσης του οργανισμού αλλά και στρατηγικό σύστημα διοίκησης).

Στο Κεφάλαιο 1 παρουσιάσαμε ορισμένα εισαγωγικά θέματα για την εταιρεία ΒΙΟΣΕΡ. Παρουσιάσαμε την οργανωτική δομή της, τα τμήματα της, τις λειτουργίες και τα είδη που εμπορεύεται η εταιρεία.

Στο Κεφάλαιο 2 παρουσιάσαμε τα θέματα αναφορικά με τον Αντικειμενοστραφή προγραμματισμό. Δείξαμε την ευκολία ανάπτυξης και συντήρησης εφαρμογών με την χρήση του αντικειμενοστραφούς προγραμματισμού καθώς και τα πλεονεκτήματα που προσφέρονται από την αναπαράσταση των όρων του πραγματικού κόσμου σε όρους εφαρμογών υπολογιστών με την χρήση των αντικειμένων.

Στο Κεφάλαιο 3 δείξαμε τα στοιχεία σχετικά με την N-tier Αρχιτεκτονική εφαρμογών. Παρουσιάσαμε τους προηγούμενους τύπους αρχιτεκτονικής και την μετάβαση στην N-tier Αρχιτεκτονική. Ιδιαίτερη έμφαση δόθηκε στην κατανόηση και διαχωρισμό των λειτουργιών μιας εφαρμογής, η οποία είναι δομημένη πάνω στην N-tier Αρχιτεκτονική καθώς και στα πλεονεκτήματα που προσφέρει ο συνδυασμός της N-tier αρχιτεκτονικής (σχεδιασμός ξεχωριστών επιπέδων εφαρμογής) με τον αντικειμενοστραφή προγραμματισμός (σχεδιασμός αντικειμένων εφαρμογής). Παράλληλα δείξαμε την ανάγκη χρήσης ενός μοντέλου, το οποίο θα αναπαραστήσει την εφαρμογή, η οποία αναπτύσσεται καθώς και την ανάγκη χρήσης ενός προτύπου επικοινωνίας και οπτικής αναπαράστασης του μοντέλου της εφαρμογής. Ένα τέτοιο πρότυπο είναι η Unified Modeling Language (UML).

Στο Κεφάλαιο 4 κάναμε μια παρουσίαση της UML. Συγκεκριμένα παρουσιάσαμε τα βασικά διαγράμματα της UML καθώς επίσης και τον τρόπο με τον οποίο χρησιμοποιούνται τα διαγράμματα αυτά για τον τελικό αποτέλεσμα που είναι η ανάλυση, σχεδιασμός και ανάπτυξη των εφαρμογών

Στο Κεφάλαιο 5 παρουσιάσαμε το πρόβλημα της ΒΙΟΣΕΡ που είναι αφενός μεν η ανάπτυξη μιας διαδικασίας ελέγχου της διοίκησης της εταιρείας, αφετέρου η παρουσίαση μιας διαδικασίας ανάπτυξης αντικειμενοστραφών εφαρμογών με την χρήση της UML.

Για να υποστηρίξουμε το πρόβλημα σχετικά με τον έλεγχο διοίκησης στο Κεφάλαιο 6 παρουσιάσαμε την θεωρία σχετικά με τα Συστήματα Ελέγχου Διοίκησης. Παρουσιάσαμε τα θέματα σχετικά με τα χαρακτηριστικά και το περιβάλλον του Ελέγχου Διοίκησης καθώς και μια διαδικασία χρήσης του. Παρουσιάσαμε, επίσης την Balanced Scorecard και τον τρόπο σύμφωνα με τον οποίο μπορεί να λειτουργήσει όχι μόνο σαν μια μέθοδος αποτίμησης της απόδοσης του οργανισμού καθώς και σαν ένα στρατηγικό εργαλείο διοίκησης και ελέγχου του οργανισμού.

Στο Κεφάλαιο 7 παρουσιάσαμε την επίλυση του προβλήματος που αντιμετωπίζει η εταιρεία ΒΙΟΣΕΡ. Στο πρώτο μέρος του κεφαλαίο παρουσιάσαμε μια Διαδικασία Ελέγχου Διοίκησης του οργανισμού. Η διαδικασία βασίζεται πάνω στην Balanced Scorecard και στους Στρατηγικούς Χάρτες για να δείξει τους στρατηγικούς στόχους του οργανισμού και είναι μια επαναληπτική διαδικασία ελέγχου, η οποία παρακολουθεί και αποτιμά την απόδοση της λειτουργίας του οργανισμού και λαμβάνει τα απαραίτητα διορθωτικά μέτρα. Παραγόμενο της διαδικασίας είναι ένα Θεωρητικό Μοντέλο, που περιέχει ένα σύνολο εφαρμογών που πρέπει να αναπτύξει ο οργανισμός για να υποστηρίξει την διαδικασία ελέγχου. Το δεύτερο μέρος παρουσιάζει μια διαδικασία ανάπτυξης αντικειμενοστραφών εφαρμογών με την χρήση της UML. Πέρα από την παρουσίαση των βημάτων της διαδικασίας παρουσιάσαμε την εφαρμογή της διαδικασίας σε ένα πραγματικό πρόβλημα της ΒΙΟΣΕΡ, που είναι η ανάπτυξη μιας αντικειμενοστραφούς Εφαρμογής Προϋπολογισμού.

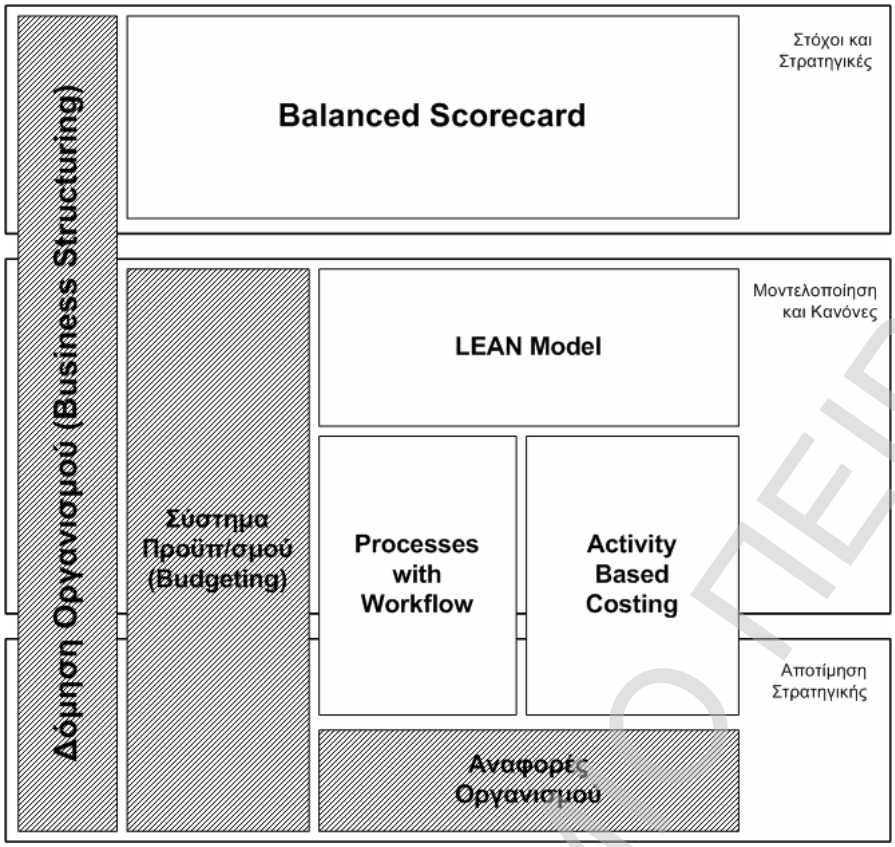
Συνοπτικά μπορούμε να πούμε, ότι ένας οργανισμός θα μπορέσει να ωφεληθεί από την χρήση αντικειμενοστραφών τεχνικών για την ανάπτυξη των εφαρμογών του και από την χρήση της N-tier Αρχιτεκτονικής για να στηρίξει πάνω τις εφαρμογές αυτές. Παράλληλα, αναδεικνύεται το πρόβλημα του Ελέγχου Διοίκησης σε ένα οργανισμό και από την διαδικασία ελέγχου διοίκησης, που προτείνουμε, αναδεικνύεται η ανάγκη χρήσης της Balanced Scorecard και των Στρατηγικών Χαρτών, που βοηθάνε στον εντοπισμό των κρίσιμων επιχειρηματικών λειτουργιών και των άυλων κεφαλαίων του οργανισμού (άνθρωποι, συστήματα κλπ.) που θα προσφέρουν αξία στους πελάτες και θα φέρουν τα αναμενόμενα χρηματοοικονομικά αποτελέσματα.

Προτάσεις

Στο σημείο αυτό μπορούμε να αναφέρουμε ορισμένες κατευθύνσεις που παρουσιάζουν ενδιαφέρον για περαιτέρω έρευνα, αναφορικά με διάφορα θέματα που παρουσιάστηκαν στην εργασία αυτή.

Αναφορικά με το ζήτημα της σχεδίασης του Θεωρητικού Μοντέλου ελέγχου διοίκησης για την εταιρεία ΒΙΟΣΕΡ, έχουμε υλοποιήσει τις ακόλουθες εφαρμογές:

- **Σύστημα Προϋπολογισμού (Budgeting)**: η εφαρμογή καταχώρησης και διαχείρισης του Προϋπολογισμού του οργανισμού.
- **Δόμηση Οργανισμού (Business Structuring)**: δόμηση του οργανισμού με την χρήση Δομών. Έχουν δημιουργηθεί οι Δομές, οι οποίες είναι απαραίτητες για την αναπαράσταση του οργανισμού. Επίσης έχουν καθοριστεί τα Κέντρα Ευθύνης του οργανισμού, που είναι τα διάφορα τμήματα του οργανισμού.
- **Αναφορές Οργανισμού**: μέσα από την εφαρμογή του Προϋπολογισμού παράγονται οι απαραίτητες αναφορές, οι οποίες χρησιμοποιούνται για την έλεγχο και την εκτίμηση της απόδοσης των Κέντρων Ευθύνης στον οργανισμό.



Διάγραμμα 39: Υλοποίηση Εφαρμογών

Όπως βλέπουμε από το παραπάνω διάγραμμα, οι εφαρμογές που πρέπει να υλοποιηθούν για να αντιμετωπιστεί το ζήτημα του Ελέγχου Διοίκησης του οργανισμού είναι οι παρακάτω:

- **Balanced Scorecard**: πρέπει να αναπαρασταθεί ο οργανισμός βάσει των τεσσάρων προοπτικών της Balanced Scorecard (Χρηματοοικονομικοί Όροι, Πελάτες, Εσωτερικές Λειτουργίες, Μάθηση και Ανάπτυξη). Στην συνέχεια βάσει των προοπτικών αυτών θα καθοριστούν οι στρατηγικοί στόχοι και τα στρατηγικά έργα, που θα υποστηρίξουν την υλοποίηση της στρατηγικής του οργανισμού.
- **LEAN Model**: η εφαρμογή η οποία έχει σαν στόχο την απλοποίηση των διαδικασιών σε ένα οργανισμό και την ελαχιστοποίηση του κόστους λειτουργίας τους. Η εφαρμογή αυτή βασίζεται στην εφαρμογή διαχείρισης διαδικασιών (Workflow) για την μέτρηση του χρόνου εκτέλεσης και την εφαρμογή Activity Based Costing (ABC) για την μέτρηση του κόστους εκτέλεσης των διαδικασιών.
- **Processes with Workflow**: η εφαρμογή διαχείρισης των διαδικασιών ασχολείται με την μέτρηση του χρόνου εκτέλεσης των διαδικασιών. Με την εφαρμογή αυτή θα μπορέσουμε να διαχειριστούμε την ροή της εκτέλεσης των διαδικασιών στον οργανισμό και να παρακολουθούμε τον χρόνο εκτέλεσης τους.
- **Activity Based Costing**: η εφαρμογή κοστολόγησης βάσει δραστηριοτήτων ασχολείται με την μέτρηση του κόστους εκτέλεσης των δραστηριοτήτων. Με την εφαρμογή αυτή θα έχουμε την δυνατότητα να εκτιμήσουμε το κόστος εκτέλεσης όλων των διαδικασιών και να έχουμε ένα πολύτιμο εργαλείο ελέγχου των διαδικασιών στον οργανισμό.

Σχετικά με το ζήτημα της χρήσης μιας αντικειμενοστραφούς διαδικασίας ανάπτυξης εφαρμογών, όπως αυτή παρουσιάστηκε παραπάνω, προτείνουμε την βελτίωση της διαδικασίας αναφορικά με δύο ζητήματα:

- Δυνατότητα Ανάπτυξης της εφαρμογής (συγγραφή κώδικα) με την χρήση των UML διαγραμμάτων, που έχουν κατασκευαστεί στα προηγούμενα βήματα της διαδικασίας. Πρέπει να ελεγχθεί ο τρόπος με τον οποίο δίνεται η δυνατότητα παραγωγής κώδικας μέσα από τα διαγράμματα της UML.
- Δυνατότητα αυτοματοποιημένου Ελέγχου της εφαρμογής με την χρήση των UML διαγραμμάτων, που έχουν κατασκευαστεί. Πρέπει να ελεγχθεί ο τρόπος και οι τεχνικές που χρησιμοποιούνται για να δημιουργεί μια διαδικασία ελέγχου της σωστής λειτουργίας της εφαρμογής.

Βιβλιογραφία

Ξένη Βιβλιογραφία

[1] Anthony Robert N., Govindarajan Vijay (2004) "*Management Control Systems*", 11th Edition, McGraw Hill

[2] Boggs Wendy, Boggs Michael (2002) "*Mastering UML with Rational Rose 2002*", Sybex

[3] Bortniker Matthew, Blexrud Christofer (2000) "*Professional Windows DNA: Building Distributed Web Applications with VB, COM+, MSMQ, SOAP, and ASP*", 1st edition, Chapter 1: Why Do We Need DNA?, Wrox Press

[4] Cernosek G.,Naiburg E. (Ιούνιος 2004) "*The Value of Modeling*", IBM, Rational Software, A technical discussion of software modeling

[5] Chandler A. (1962) "*Strategy and Structure: Chapters in the History of the American Industrial Enterprise*", MA: MIT Press, p13

[6] Eriksson H.,Penker M.,Lyons B.,Fado D. (OMG) (2004) "*UML 2 Toolkit*", Wiley Publishing, Inc

[7] Fowler Martin, (2004) "*UML Distilled. A brief guide to the standard Object Modeling Language*", 3rd Edition, Addison-Wesley

[8] Johnson G, Scholes K, (1999) "*Exploring Corporate Strategy: Text and Cases*", 5th Edition, London, Prentice Hall Europe, pp10

[9] Kaplan Robert S., Norton David P. (1996) "*The Balanced Scorecard. Translating Strategy into Action*", Harvard Business School Press

[10] Kaplan Robert S., Norton David P. (2004) "*Strategy Maps. Converting intangible assets into tangible outcomes*", Harvard Business School Press

[11] Langfield-Smith Kim (Φεβρουάριος 1997) "*Management control systems and strategy: A critical review*", Accounting, Organizations and Society, Volume 22, Issue 2, Pages 207-232

[12] Pender Tom (2002) "*UML Weekend Crash Course*", Wiley Publishing, Inc.

[13] Pender Tom (2003) "*UML Bible*", Wiley

[14] Quatrani Terry (1999) "*Visual Modeling with Rational Rose and UML*", Addison Wesley

[15] Quatrani Terry (Ιούνιος 2003) "*Introduction to the Unified Modeling Language*", IBM, Rational Software, A technical discussion of UML

[16] Schmuller Joseph (2004) "*Teach Yourself UML in 24 Hours*", SAMS

[17] Siman Si Alhir (Ιούλιος 1998) "*The True Value of the Unified Modeling Language (UML)*", DistributedComputing.com

[18] Siman Si Alhir (Απρίλιος 2000) "*Understanding Use Case Modeling*"

[19] Simons Robert. (1995) "*Levers of Control: How Managers use Innovative Control Systems to Drive Strategic Renewal*", Harvard Business School Press

[20] Object Management Group (OMG) (Μάρτιος 2003) "*Unified Modeling Language Specification*" έκδοση 1.5

Ελληνική Βιβλιογραφία

[21] Καζαντζής Χ. "*Σημειώσεις Διοικητικής Λογιστικής*" (σημειώσεις μαθήματος "Διοικητική Λογιστική", 7ο εξάμηνο, τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων, Πανεπιστήμιο Πειραιώς)

[22] Παπαδάκης Β. (2002) "*Στρατηγική των Επιχειρήσεων: Ελληνική και Διεθνής εμπειρία*", Εκδόσεις Μπένου

Internet Sites

[23] 15 Seconds - Application Architecture: An N-Tier Approach - Part 1 (http://www.15seconds.com/issue/011023.htm)

[24] MSDN - N-Tier Application Development with Microsoft.NET (http://www.microsoft.com/belux/nl/msdn/community/columns/hyatt/ntier1.mspx)

[25] Object Management Group (www.omg.org)

[26] Object Oriented Programming (OOP) (http://www.oop.esmartkid.com/)

[27] QuasarSoft - White Papers about Architecture (http://www.quasarsoft.com)

[28] TheFreeDictionary (http://encyclopedia.thefreedictionary.com/multitier+architecture)

[29] Wikipedia, the free encyclopedia - Object-oriented programming (http://en.wikipedia.org/wiki/Object-oriented)

[30] Ορισμός Στρατηγικής (http://home.att.net/~nickols/strategy_definition.htm)