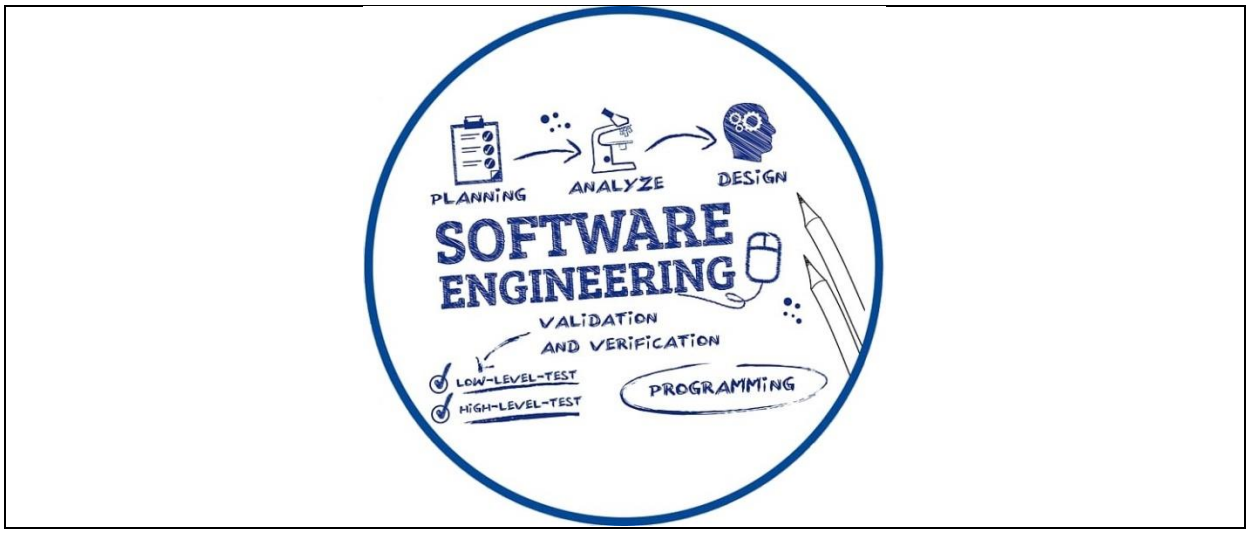
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΗΣ ΑΞΙΟΛΟΓΗΣΗΣ ΓΡΑΠΤΩΝ ΕΡΓΑΣΙΩΝ ΓΙΑ ΤΟ ΜΑΘΗΜΑ ΤΗΣ ΜΗΧΑΝΙΚΗΣ ΛΟΓΙΣΜΙΚΟΥ



Του φοιτητή
Καρυπίδη Μιχαήλ
Αρ. Μητρώου: 063021

Επιβλέπων
Ονοματεπώνυμο κ. Δεληγιάννης Ιγνάτιος
Βαθμίδα: Καθηγητής

Ημερομηνία 10/01/2021

Τίτλος Δ.Ε.: Ανάπτυξη εφαρμογής αξιολόγησης γραπτών εργασιών για το μάθημα Μηχανική Λογισμικού

Κωδικός Δ.Ε.: 19076

Ονοματεπώνυμο φοιτητή: Καρυπίδης Μιχαήλ

Ονοματεπώνυμο εισηγητή: Δεληγιάννης Ιγνάτιος

Ημερομηνία ανάληψης Δ.Ε: 24/11/2019

Ημερομηνία περάτωσης Δ.Ε. 10/01/2021

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

"The purpose of software engineering is to control complexity, not to create it".
Pamela Zave

Πρόλογος

Η ανάπτυξη λογισμικού κατά κύριο λόγο σημαίνει δημιουργικότητα. Απαιτείται ιδέα, έμπνευση, αφοσίωση και υλοποίηση. Όπως συμβαίνει με τη συγγραφή ενός βιβλίου, μιλάμε για κάτι το οποίο δεν υπάρχει και σταδιακά δημιουργείται μπροστά στα μάτια σου. Το λογισμικό λοιπόν είναι ισάξιο με οποιαδήποτε άλλη μορφή δημιουργίας που προέρχεται είτε από ατομική έμπνευση ή από κοινωνική ανάγκη, ασχέτως αν μιλάμε για την δημιουργία των Windows 1.0 ή της τηλεφωνικής ατζέντας που όλοι σχεδόν προγραμματίσαμε σε Basic.

Η επιλογή της πτυχιακής εργασίας μου έγινε για τους παραπάνω λόγους. Προτίμησα δηλαδή μια εργασία που αποτελείται από μια αρχική απαίτηση πελάτη (του ζητούμενου της πτυχιακής εργασίας στην προκειμένη περίπτωση) χωρίς περιορισμούς σε γλώσσα, πλατφόρμα ή συγκεκριμένη τεχνολογία. Άλλωστε, το μάθημα της Μηχανικής του Λογισμικού αφορά ακριβώς σε αυτήν την διαδικασία- την έρευνα και την επιλογή του βέλτιστου τρόπου ανάπτυξης του λογισμικού.

Τα οφέλη από αυτήν την ενασχόλησή μου είναι ποικίλα: εκτός από την εκμάθηση της VBA και την εμπειρία εκπόνησης μιας πτυχιακής εργασίας, προσπάθησα για πρώτη φορά ως επαγγελματίας να ανταποκριθώ στις απαιτήσεις του πελάτη.

Περίληψη

Σε αυτήν την εργασία περιγράφεται η υλοποίηση μιας εφαρμογής αξιολόγησης γραπτών εργασιών, η οποία προσαρμόζεται ως πρόσθετο στη πλατφόρμα του MS Word και για την ανάπτυξή της χρησιμοποιήθηκε η γλώσσα VBA. Η εφαρμογή θα χρησιμοποιηθεί για την αξιολόγηση εργασιών του μαθήματος «Μηχανική Λογισμικού».

Στο πρώτο μέρος της εργασίας θα γίνει αναφορά σε έννοιες που θα πρέπει να είναι κατανοητές όπως λογισμικό, μηχανική λογισμικού, επαναχρησιμοποιήσιμη τεχνολογία, UML κ.α., τα οποία αφορούν το μάθημα της Μηχανικής Λογισμικού.

Στο δεύτερο μέρος συμπεριλαμβάνονται περισσότερο τεχνικά θέματα, προσομοιάζοντας περισσότερο σε τεχνικό εγχειρίδιο, και περιλαμβάνει οδηγίες εγκατάστασης και χρήσης. Επιπλέον θα επεξηγηθούν συγκεκριμένα κομμάτια κώδικα σε VBA ή/και XML ώστε να γίνει κατανοητή η λογική του τρόπου ανάπτυξης και της λειτουργίας της εφαρμογής.

Development of an application for evaluating written assignments for the course of software engineering

Michail Karypidis

Abstract

This essay describes the implementation of a written assessment application, which is adapted as an add-on to the MS Word platform and uses the VBA language for its development. The application will be used to evaluate assignments of the course "Software Engineering". In the first part, the essay will be referred to concepts that should be understood such as software, software engineering, reusable technology, UML, etc., which relate to the course of Software Engineering.

The second part includes more technical issues, more like a technical manual, and includes installation and use instructions. In addition, specific lines of code in VBA and / or XML will be explained in order to be understood the logic of how the application is developed and operated.

Ευχαριστίες

Αισθάνομαι την ανάγκη να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Δεληγιάννη Ιγνάτιο για την εμπιστοσύνη που έδειξε στο πρόσωπό μου, καθώς και για την καθοδήγηση που απλόχερα μου παρείχε σε όλα τα στάδια της υλοποίησης της παρούσας εργασίας.

Θα ήταν παράλειψη να μην ευχαριστήσω την σύζυγο μου Άννα και την κόρη μου Χαρά, για την υπομονή που έδειξαν καθ' όλη την διάρκεια των σπουδών μου.

Τέλος, θα ήθελα να κάνω ιδιαίτερη αναφορά στα διδάγματα του μεγάλου Γερμανού φιλοσόφου Φρειδερίκου Νίτσε, που σαν φάρος φώτιζαν και συνεχίζουν να φωτίζουν τον δρόμο μου.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Συντομογραφίες

MS	Microsoft
VBA	Visual Basic for Applications
UML	Unified Modeling Language
ΜΛ	Μηχανική Λογισμικού
IEEE	Institute of Electrical and Electronics Engineers
SE	Software engineering
ACM	Association for Computing Machinery
IBM	International Business Machines Corporation
OMT	Object-Modeling Technic
MBASE	Model-Based Architecture and Software Engineering
ISO	International Organization for Standardization
SAQ	Software Quality Assurance
MTBF	Mean Time Between Failure
MTTR	Mean Time To Repair
MTTC	Mean Time to Change
OMG	Object Management Group
VBE	Visual Basic Editor

Κεφάλαιο 1ο: Μηχανική Λογισμικού

1.1 Εισαγωγή

Οι πρώτοι υπολογιστές εμφανίστηκαν τη δεκαετία του 1940 και ταυτόχρονα εμφανίστηκε η ανάγκη για ανάπτυξη λογισμικού. Τα πρώτα χρόνια, αυτή την ανάγκη εξυπηρετούσε η εταιρεία που κατασκεύαζε τους υπολογιστές (hardware), συχνά χωρίς επιπλέον χρέωση για τον πελάτη. Οι εταιρείες ανάπτυξης λογισμικού εμφανίστηκαν αργότερα (συνήθως αποτελούμενες από ένα άτομο), καθώς τα προγράμματα στα πρώτα χρόνια του προγραμματισμού ήταν πολύ μικρά σε σχέση με τα σημερινά, ειδικευμένα σε συγκεκριμένο τομέα εφαρμογής και αναπτύσσονταν συνήθως από ένα μόνο άτομο. Σήμερα τα προγράμματα είναι πολύ μεγαλύτερα και αναπτύσσονται από μεγάλες εταιρείες και ομάδες διασκορπισμένες σε όλον τον κόσμο. Παράλληλα η χρήση των υπολογιστών και του λογισμικού αυξήθηκε κατακόρυφα και εισχώρησε σε όλο και περισσότερο κρίσιμους για την ασφάλεια και την υγεία τομείς. Οποιαδήποτε απόκλιση στην ποιότητα, στην απόδοση και σε καθυστερήσεις που οφείλονται στην πολυπλοκότητα και στη μη επαρκή επικοινωνία ανάμεσα στους προγραμματιστές, θα μπορούσε να στοιχήσει μεγάλα χρηματικά ποσά ή ακόμη και ανθρώπινες ζωές. Ο κλάδος της Μηχανικής Λογισμικού (ΜΛ) εμφανίστηκε για να λύσει ακριβώς αυτά τα προβλήματα της πολυπλοκότητας στην ανάπτυξη λογισμικού και της οργάνωσης και επικοινωνίας των απομακρυσμένων προγραμματιστών μεταξύ τους. Η ΜΛ δηλαδή, μετά τη συστηματική και λεπτομερή μελέτη του σχεδιασμού, της ανάπτυξης και της συντήρησης του λογισμικού, προτείνει τον βέλτιστο τρόπο ανάπτυξης και παρακολούθησης του εκάστοτε έργου με στόχο την ικανοποίηση των αναγκών του πελάτη.

1.2 Λογισμικό

Σύμφωνα με την Misty E. Vermaat, ο υπολογιστής είναι μια ηλεκτρονική συσκευή η οποία λειτουργεί υπό τον έλεγχο των εντολών που είναι αποθηκευμένες στη μνήμη του και μπορεί να δεχτεί δεδομένα (είσοδος), να επεξεργαστεί αυτά τα δεδομένα σύμφωνα με καθορισμένους κανόνες, να παράγει πληροφορίες (έξοδος) και να αποθηκεύσει τις πληροφορίες για μελλοντική χρήση [1]. Οι αποθηκευμένες στη μνήμη εντολές που αναφέρονται παραπάνω είναι οργανωμένες σε προγράμματα, ενώ ένα ή περισσότερα προγράμματα ονομάζονται λογισμικό.

1.2.1 Τι ονομάζουμε λογισμικό

Το λογισμικό είναι ένας γενικός όρος για οργανωμένες συλλογές δεδομένων και οδηγιών του υπολογιστή, που χωρίζονται κυρίως σε δύο κύριες κατηγορίες: στο **λογισμικό συστήματος (System Software)** που παρέχει τις βασικές λειτουργίες του υπολογιστή και δεν αφορά την εκτέλεση συγκεκριμένων εργασιών και στο **λογισμικό εφαρμογών (Application Software)** που χρησιμοποιείται από τους χρήστες για την εκτέλεση συγκεκριμένων εργασιών.

1.2.2 Κατηγορίες Λογισμικού

Είδαμε παραπάνω ότι το λογισμικό χωρίζεται σε δυο κύριες κατηγορίες: στο λογισμικό συστήματος και στο λογισμικό εφαρμογών.

Το λογισμικό συστήματος είναι υπεύθυνο για τον έλεγχο, την ενσωμάτωση και τη διαχείριση των μεμονωμένων στοιχείων υλικού ενός συστήματος υπολογιστή, έτσι ώστε οποιοδήποτε άλλο λογισμικό και οι χρήστες του συστήματος να το αντιλαμβάνονται ως λειτουργική μονάδα. Έτσι, δεν χρειάζεται

να ασχολούνται με τις λεπτομέρειες χαμηλού επιπέδου, όπως η μεταφορά δεδομένων από τη μνήμη στο δίσκο ή την απόδοση κειμένου σε μια οθόνη. Γενικά, το λογισμικό συστήματος αποτελείται από ένα λειτουργικό σύστημα και ορισμένα βασικά βοηθητικά προγράμματα, όπως διαμορφωτές δίσκων, διαχειριστές αρχείων, διαχειριστές οθόνης, επεξεργαστές κειμένου, έλεγχο ταυτότητας χρήστη (σύνδεση) και εργαλεία διαχείρισης, και λογισμικό δικτύωσης και ελέγχου συσκευών.

Αντίθετα, το λογισμικό εφαρμογών χρησιμοποιείται για την εκπλήρωση συγκεκριμένων εργασιών. Μπορεί να αποτελείται από ένα μόνο πρόγραμμα, όπως για παράδειγμα ένα πρόγραμμα προβολής εικόνων, μια μικρή συλλογή προγραμμάτων (συχνά ονομάζεται πακέτο λογισμικού) που συνεργάζονται στενά για την ολοκλήρωση μιας εργασίας, όπως ένα υπολογιστικό φύλλο ή ένα σύστημα επεξεργασίας κειμένου ή μια μεγαλύτερη συλλογή (συχνά ονομάζεται σουίτα λογισμικού) σχετικών αλλά ανεξάρτητων προγραμμάτων και πακέτων που έχουν κοινή διεπαφή χρήστη ή κοινόχρηστη μορφή δεδομένων. Ένα τέτοιο παράδειγμα σουίτας λογισμικού είναι το Microsoft Office, το οποίο αποτελείται από ενσωματωμένο επεξεργαστή κειμένου, υπολογιστικό φύλλο, βάση δεδομένων κ.λπ. ή ένα σύστημα διαχείρισης βάσεων δεδομένων, το οποίο είναι μια συλλογή θεμελιωδών προγραμμάτων που μπορεί να παρέχουν κάποια υπηρεσία σε μια ποικιλία άλλων ανεξάρτητων εφαρμογών.

Σχήμα 1: Κατηγορίες λογισμικού

Οι βασικές δυο κατηγορίες διαιρούνται επίσης σε επιμέρους κατηγορίες (σχήμα 1). Το λογισμικό εφαρμογών συμπεριλαμβάνει το **λογισμικό γενικού σκοπού** και το **λογισμικό για επιχειρησιακές χρήσεις**. Αντίστοιχα, το λογισμικό συστήματος συμπεριλαμβάνει το **λογισμικό ανάπτυξης συστημάτων** και το **λογισμικό διαχείρισης συστημάτων**.

Το λογισμικό γενικού σκοπού/χρήσης αναφέρεται σε εφαρμογές υπολογιστών που δεν έχουν σχεδιαστεί για συγκεκριμένη επιχείρηση, βιομηχανία ή τμήμα. Αυτές οι εφαρμογές μπορούν επομένως, να αγοραστούν από το ράφι (off-the-shelf) και να εφαρμοστούν από πολλούς επαγγελματίες. Κάποια από τα πλεονεκτήματα της χρήσης λογισμικού γενικής χρήσης είναι:

- Είναι σχετικά φθηνό.
- Εύκολη πρόσβαση, καθώς είναι διαθέσιμο στα περισσότερα καταστήματα υπολογιστών.
- Έχει δοκιμαστεί διεξοδικά, οπότε δεν θα υπάρξουν σοβαρά προβλήματα ή σφάλματα.
- Υπάρχει μεγάλη υποστήριξη χρηστών, δηλαδή βιβλία, οδηγοί χρηστών, ηλεκτρονική βοήθεια και φόρουμ συζήτησης στο Διαδίκτυο.

Το λογισμικό για επιχειρήσεις είναι λογισμικό σχεδιασμένο για επαγγελματική χρήση και προσαρμοσμένο σε εταιρείες ή επιχειρήσεις, και όχι μεμονωμένα άτομα. Η επιλογή δεν είναι πάντα εύκολη και συχνά εξαρτάται από το μέγεθος, τις ιδιαίτερες ανάγκες της επιχείρησης και τις εργασίες που πρέπει να αυτοματοποιηθούν. Παραδείγματα λογισμικού επιχειρήσεων είναι τα προγράμματα επεξεργασίας κειμένου, λογισμικό τήρησης λογαριασμών, λογισμικό μισθοδοσίας και λογισμικό βάσης δεδομένων.

Στο λογισμικό ανάπτυξης συστημάτων ή λογισμικό προγραμματισμού συμπεριλαμβάνονται εργαλεία με τη μορφή προγραμμάτων ή εφαρμογών, τα οποία χρησιμοποιούν οι προγραμματιστές λογισμικού για τη δημιουργία, τον εντοπισμό σφαλμάτων, τη συντήρηση ή άλλες υπηρεσίες υποστήριξης άλλων προγραμμάτων και εφαρμογών. Ο όρος αναφέρεται συνήθως σε σχετικά απλά προγράμματα όπως μεταγλωττιστές, προγράμματα εντοπισμού σφαλμάτων και διερμηνείς. Πρακτικά, στην κατηγορία αυτή συμπεριλαμβάνονται όλες οι γλώσσες προγραμματισμού όπως η C, η C++, η Java, η Python κ.α.

Το λογισμικό διαχείρισης συστημάτων ελέγχει τους πόρους και της συσκευές του συστήματος. Για παράδειγμα, διαχειρίζεται τη μνήμη του συστήματος, όταν τρέχουν ταυτόχρονα δύο προγράμματα ή όταν χρειάζεται να γίνει μόνιμη μεταφορά δεδομένων από την προσωρινή μνήμη στον σκληρό δίσκο. Επίσης, διαχειρίζεται τις διακοπές (interrupts) και τις αστοχίες λογισμικού και υλικού εκτελώντας διαγνωστικούς ελέγχους. Τέλος, μια από τις πιο σημαντικές λειτουργίες του λογισμικού διαχείρισης συστημάτων είναι η επιλογή και ο έλεγχος των περιφερειακών συσκευών με την χρήση μικρών προγραμμάτων, τα οποία ονομάζονται οδηγοί (drivers).

1.2.3 Φύση του λογισμικού

Ο Philippe Kruchten, διερευνά σε άρθρο του [2] τα τέσσερα βασικά χαρακτηριστικά διαφοροποίησης του λογισμικού σε σχέση με τα φυσικά προϊόντα:

- **Απουσία θεμελιώδους θεωριών**: Το λογισμικό, σε αντίθεση με ένα φυσικό προϊόν, δεν υπάγεται σε κάποιο θεωρητικό πλαίσιο ή σε κάποιες βασικές φυσικές αρχές στις οποίες πρέπει να συμμορφώνεται, όπως για παράδειγμα οι φυσικοί νόμοι. Ωστόσο, το λογισμικό θα πρέπει να συμμορφώνεται με τις απαιτητές προδιαγραφές, στις διεπαφές με άλλα εσωτερικά μέρη λογισμικού και στις συνδέσεις με το περιβάλλον στο οποίο λειτουργεί.
- **Ευκολία αλλαγής**: Το λογισμικό είναι το στοιχείο που αλλάζει συχνότερα σε ένα σύστημα, ακριβώς επειδή είναι το πιο εύπλαστο στοιχείο (εύκολα τροποποιήσιμο). Ωστόσο, αυτό δεν σημαίνει ότι το λογισμικό είναι πάντα εύκολο να τροποποιηθεί. Η πολυπλοκότητα και η ανάγκη συμμόρφωσης μπορούν να κάνουν την τροποποίηση λογισμικού εξαιρετικά δύσκολη εργασία.
- **Ταχεία εξέλιξη των τεχνολογιών**: Οι τεχνικές ανάπτυξης του λογισμικού, καθώς και το ίδιο το περιβάλλον του λογισμικού, αλλάζουν ραγδαία και σε τέτοιο ρυθμό που δεν επιτρέπει την υλοποίηση λογισμικού σε μεγάλο βάθος χρόνου. Αυτό ασκεί μεγάλη πίεση στις εταιρείες,

ώστε να εκπαιδεύουν και να επανεκπαιδεύουν τους μηχανικούς λογισμικού που έχουν στη διάθεσή τους.

- **Πολύ χαμηλό κόστος κατασκευής** : Η «πραγματική» κατασκευή του λογισμικού δεν συνεπάγεται σχεδόν κανένα κόστος. Ακόμη και η διανομή: ένα CD-ROM, για παράδειγμα, κοστίζει λιγότερο από ένα δολάριο, και η παράδοση μέσω Διαδικτύου μόνο μερικά λεπτά. Επίσης, όπως έχει αναφερθεί παραπάνω, η ευκολία στην αλλαγή που προσφέρει το λογισμικό μας επιτρέπει, ακόμη και σε περίπτωση κακού σχεδιασμού, να το διορθώσουμε και να το κατασκευάσουμε ξανά, δυνατότητα που δεν είναι διαθέσιμη σε άλλα φυσικά προϊόντα.

Αλλά και ο Fred Brooks, στο διάσημο βιβλίο του «The Mythical Man-Month» [3], αναφέρεται στην μοναδικότητα του λογισμικού ως προϊόν σε σχέση με τα υπόλοιπα φυσικά προϊόντα. Συγκεκριμένα, παραθέτει τους ακόλουθους λόγους:

- Το λογισμικό δεν έχει φυσικές ιδιότητες.
- Το λογισμικό είναι προϊόν ομαδικής εργασίας και σκέψης.
- Η παραγωγικότητα των μηχανικών λογισμικού διαφέρει από την παραγωγικότητα των άλλων επιστημονικών κλάδων.
- Ο προϋπολογισμός και ο σχεδιασμός των έργων λογισμικού χαρακτηρίζεται από υψηλό βαθμό αβεβαιότητας, ο οποίος μπορεί εν μέρει να μετριαστεί από την εύρεση και εφαρμογή των βέλτιστων πρακτικών.
- Η διαχείριση των κινδύνων στα έργα λογισμικού είναι, κατά κύριο λόγο, προσανατολισμένη στις διαδικασίες.
- Το λογισμικό δεν είναι αυτόνομο, καθώς είναι πάντα μέρος ενός μεγαλύτερου συστήματος.
- Το λογισμικό είναι το στοιχείο που μεταβάλλεται πιο συχνά σε ένα σύστημα.

Πριν την εισαγωγή στη ΜΛ και μιας και στεκόμαστε στο βιβλίο του Brooks, είναι σκόπιμο να θυμηθούμε την διάσημη δυσοίωνη ρήση του Brooks (η οποία πλέον συμπεριλαμβάνεται στην αναθεωρημένη έκδοση του ανωτέρω βιβλίου του [3]), σύμφωνα με την οποία δεν υπάρχει καμία ασημένια σφαίρα που να σκοτώνει την αντιπαραγωγικότητα. Συγκεκριμένα, επιμένει ότι "δεν υπάρχει καμία εξέλιξη, ούτε στην τεχνολογία ούτε στην τεχνικές διαχείρισης, η οποία από μόνη της να μπορεί να υποσχεθεί ακόμη και μια τάξη μεγέθους [δεκαπλάσια] βελτίωση μέσα σε μια δεκαετία στην παραγωγικότητα, στην αξιοπιστία, στην απλότητα". Παρακάτω θα αναλυθούν τεχνικές με τις οποίες η ΜΛ προσπαθεί να απαντήσει σε αυτή την πεποίθηση.

1.3 Μηχανική Λογισμικού

Τα τελευταία τριάντα χρόνια, το μέγεθος και η πολυπλοκότητα του λογισμικού που παράγεται και χρησιμοποιείται από τη σύγχρονη κοινωνία μας, έχουν υποστεί μια ασύγκριτη αύξηση. Πλέον υπάρχει η ανάγκη καθορισμού συγκεκριμένων στόχων (ή λειτουργικών απαιτήσεων), πρόβλεψης των απαραίτητων πόρων (όπως για παράδειγμα η εκτίμηση του κόστους) για την επίτευξη αυτών των στόχων και την επιτυχή διαχείριση των προσδοκιών των πελατών.

Η υλοποίηση μιας εφαρμογής δεν αποτελείται πλέον απλώς από τη συγγραφή κώδικα, αλλά ακολουθεί επίσης συγκεκριμένες οδηγίες, όπως η γραπτή τεκμηρίωση (documentation) και οδηγίες μονάδων ελέγχου. Τα διαφορετικά κομμάτια που δημιουργούνται από διαφορετικά τμήματα ανάπτυξης θα πρέπει να ταιριάζουν μεταξύ τους, όπως θα πρέπει επίσης να είμαστε σε θέση να

εντοπίσουμε προβληματικές περιοχές χρησιμοποιώντας μετρήσεις και να βελτιώσουμε την ποιότητα σε αυτές τις περιοχές. Ο κώδικας θα πρέπει να ακολουθεί ορισμένα πρότυπα για να διευκολύνει τη συνεργασία ανάμεσα στις ομάδες.

Με άλλα λόγια, το έργο δεν τελειώνει με την επιτυχή ολοκλήρωση του κώδικα- αντιθέτως, για τα μεγάλα έργα η συντήρηση λογισμικού (όπως και κάθε άλλου είδους συντήρηση) μπορούν να κρατήσουν πολλούς ανθρώπους απασχολημένους για μεγάλο χρονικό διάστημα.

1.3.1 Ορισμοί της Μηχανικής Λογισμικού (ΜΛ)

Για την ΜΛ έχουν προταθεί πολλοί διαφορετικοί ορισμοί, στην προσπάθεια των επιστημόνων να συμπεριλάβουν σε αυτούς όλο και περισσότερα από τα χαρακτηριστικά της ΜΛ. Σε ακολουθία του ως άνω αναφερόμενου γεγονότος, οι ορισμοί της ΜΛ που ακολουθούν παρακάτω επισημαίνουν διαφορετικά χαρακτηριστικά ή προοπτικές της ΜΛ:

Ο ορισμός για τη ΜΛ που δίδεται από το Ινστιτούτο Ηλεκτρολόγων και Μηχανικών Ηλεκτρονικής (IEEE) είναι αρκετά συμπαγής και ορίζεται ως «η εφαρμογή συστηματικής, πειθαρχημένης, ποσοτικοποιημένης προσέγγισης στην ανάπτυξη, τη λειτουργία και τη συντήρηση του λογισμικού»[4].

Ο I. Sommerville στην όγδοη έκδοση του βιβλίου του σχετικά με την ΜΛ (Software engineering), επεκτείνει τον παραπάνω ορισμό και προτείνει «οι μηχανικοί λογισμικού θα πρέπει να υιοθετήσουν μια συστηματική και οργανωμένη προσέγγιση στη δουλειά τους και να χρησιμοποιήσουν τα κατάλληλα εργαλεία και τεχνικές ανάλογα με το εκάστοτε πρόβλημα, τους περιορισμούς ανάπτυξης και τους διαθέσιμους πόρους»[5].

Οι Ghezzi et al. με λακωνικό τρόπο ορίζουν την ΜΛ ως «το πεδίο της επιστήμης των υπολογιστών που ασχολείται με τη δημιουργία συστημάτων λογισμικού, τα οποία είναι τόσο μεγάλα ή τόσο περίπλοκα που για την υλοποίησή τους απαιτείται η σύσταση ομάδας ή ομάδων από μηχανικούς»[6].

Τέλος, ο W. Emmerich στο σχετικό βιβλίο του [6], μας ενημερώνει ότι «η μηχανική λογισμικού είναι ο κλάδος της εφαρμοσμένης μηχανικής συστημάτων, η οποία ασχολείται με την ανάπτυξη μεγάλων και σύνθετων λογισμικών συστημάτων». Στο ίδιο βιβλίο αναφέρει επίσης ότι η ΜΛ «ασχολείται επίσης με τις διαδικασίες, τις μεθόδους και τα εργαλεία που απαιτούνται για την ανάπτυξη συστημάτων λογισμικού που προκύπτουν από ένα οικονομικό και χρονικό πλαίσιο».

Σε μια προσπάθεια να συνοψίζουμε τους ανωτέρω ορισμούς, μπορούμε να πούμε ότι η ΜΛ είναι ο κλάδος της επιστήμης των υπολογιστών που εστιάζει στην ανάπτυξη προϊόντων λογισμικού όλων των σχημάτων, μεγεθών και πεδίων (θεωρείται απαραίτητη στα μεγαλύτερα και πολυπλοκότερα έργα, αυτό όμως δεν σημαίνει ότι και τα μικρότερα έργα δεν μπορούν να ωφεληθούν από τις πρακτικές της). Λειτουργεί μέσα από σύνολα αρχών, βέλτιστων πρακτικών και μεθόδων που έχουν δοκιμαστεί σε βάθος χρόνου. Καθώς το λογισμικό και η τεχνολογία συνεχώς αλλάζουν και συγχωνεύονται δημιουργώντας έτσι ένα ολοένα και πιο περίπλοκο περιβάλλον, η ΜΛ θα πρέπει επίσης να αλλάζει και να εξελίσσεται ώστε να προσαρμόζεται πάντα στις αλλαγές.

1.3.2 Ιστορική αναδρομή της ΜΛ

Ήδη από την ανάπτυξη των πρώτων ψηφιακών προγραμμάτων εμφανίστηκε η ανάγκη ύπαρξης μιας συστηματικής προσέγγισης σχεδιασμού για την επίτευξη καλύτερων προγραμμάτων. Χαρακτηριστικά να αναφέρουμε τη γνωστή ρήση του Maurice Wilkes, ο οποίος ήταν ένας από τους πρώτους που δημιουργούσε προγράμματα για ψηφιακό υπολογιστή, σχετικά με την «ακριβή στιγμή που

συνειδητοποίησα ότι ένα μεγάλο μέρος της υπόλοιπης ζωής μου θα ξοδέψω ψάχνοντας να βρω λάθη στα δικά μου προγράμματα»[8].

Ο όρος «Μηχανική Λογισμικού» χρησιμοποιήθηκε για πρώτη φορά σε ένα συνέδριο το 1968, όταν και συζητήθηκαν οι δυσκολίες και οι παγίδες του σχεδιασμού σύνθετων συστημάτων λογισμικού. Ξεκίνησε μια αναζήτηση λύσεων, η οποία επικεντρώθηκε κυρίως στην εύρεση καλύτερων μεθοδολογιών και εργαλείων. Οι γλώσσες προγραμματισμού αποτέλεσαν τα πιο ελπιδοφόρα εργαλεία σε αυτήν την αναζήτηση λύσεων, καθώς η ΜΛ συνδέεται στενά με την εμφάνιση και τη βελτίωσή τους. Επίσης σημαντικές ήταν οι προσπάθειες συστηματοποίησης, ακόμη και αυτοματοποίηση της τεκμηρίωσης και των δοκιμών του προγράμματος. Επίσης σημαντικές ήταν οι προσπάθειες συστηματοποίησης, ακόμη και αυτοματοποίηση της τεκμηρίωσης και των δοκιμών του προγράμματος. Πιο πρόσφατα, η ταχεία ανάπτυξη της υπολογιστικής ισχύος κατέστησε δυνατή την εφαρμογή του υπολογιστή σε ολοένα και πιο περίπλοκες εργασίες. Αυτή η τάση αύξησε δραματικά τις απαιτήσεις στους μηχανικούς λογισμικού. Τα προγράμματα και τα συστήματα έγιναν περίπλοκα και σχεδόν αδύνατο να κατανοηθούν πλήρως, και για το λόγο αυτόν υπάρχει η ανάγκη παράλληλης επικαιροποίησης των προσεγγίσεων που προτείνονται από την ΜΛ.

1.3.2.1 Προέλευση του όρου «Μηχανική Λογισμικού»

Για τους περισσότερους, ο όρος επινοήθηκε το 1968 από τον Friedrich Bauer, κατά τη διάρκεια του συνεδρίου του NATO. Άλλοι επισημαίνουν την επιστολή του Anthony Oettinger προς το ACM το 1966, όπου χρησιμοποίησε τον όρο «μηχανική λογισμικού» για να κάνει τη διάκριση μεταξύ της επιστήμης των υπολογιστών και της δημιουργίας συστημάτων λογισμικού[9]. Ακόμη νωρίτερα, στο τεύχος του Ιουνίου 1965 του περιοδικού «Computers and Automation», εμφανίστηκε μια διαβαθμισμένη διαφήμιση που αναζητούσε έναν «μηχανικό συστημάτων λογισμικού». Τέλος, για κάποιους, το άτομο που επινόησε τον όρο για πρώτη φορά ήταν η Margaret Hamilton. Έχοντας δουλέψει στο πρόγραμμα SAGE (Semi-automatic Ground Environment), έγινε η βασική προγραμματίστρια των διαστημικών προγραμμάτων Skylab και Apollo. Σύμφωνα με μια (αδημοσίευτη) προφορική ιστορία, ξεκίνησε να χρησιμοποιεί τον όρο «μηχανική λογισμικού» το 1963 ή το 1964 για να διακρίνει το έργο της από τη μηχανική υλικού που χρησιμοποιούσε το νέο διαστημικό πρόγραμμα των ΗΠΑ.

1.3.2.2 Πριν την δεκαετία του 1960

Η Ada Lovelace το 1843, δημοσίευσε ένα άρθρο το οποίο περιλαμβάνει ένα "σχέδιο" σχετικά με το πώς η Αναλυτική Μηχανή του Charles Babbage θα μπορούσε να υπολογίζει αριθμούς Μπερνούλι. Αυτό το "σχέδιο" θεωρείται από τους ιστορικούς το πρώτο πρόγραμμα υπολογιστή. Την ίδια στιγμή, ο George Boole πρότεινε έναν νέο τρόπο σκέψης στους μαθηματικούς και τους φιλόσοφους του κόσμου, όπως αυτός αναλύεται στο κλασικό πλέον βιβλίο του, «The Laws of Thought» [10]. Στο τέλος του 19ου αιώνα, είδαμε τους πρώτους ανθρώπινους υπολογιστές (κυρίως γυναίκες). Γύρω στις αρχές του νέου αιώνα, καθώς τα υπολογιστικά προβλήματα άρχισαν να αυξάνονται και καθώς τα μηχανικά βοηθήματα υπολογισμού έγιναν πιο αξιόπιστα και οικονομικά, η διαδικασία του υπολογισμού υποβλήθηκε σε περαιτέρω διαμόρφωση. Ήταν συνηθισμένη πια η ύπαρξη μεγάλων δωματίων γεμάτα γραφεία με ανθρώπινους υπολογιστές (πάλι, κυρίως γυναίκες), παραταγμένα σε σειρές. Το 1940, ο Wallace Eckert δημοσίευσε μια μέθοδο που χρησιμοποιούσε διάτρητες κάρτες [11], η οποία θεωρείται ως η πρώτη υπολογιστική μεθοδολογία ή γλώσσα προτύπων. Καθώς ο πόλεμος πλησίαζε την Ευρώπη, ο George Stibitz εφάρμοσε τις ιδέες του George Boole σχετικά με τη δυαδική λογική, για να κατασκευάσει τον πρώτο ψηφιακό αθροιστή από ηλεκτρομηχανικά ρελέ. Το

καλοκαίρι του 1944, ο John von Neumann (ο οποίος τότε εργαζόταν στο Manhattan Project) μαζί με τον Herman Goldstine και τον John Mauchly δημιουργήσανε τον ENIAC και ένα χρόνο αργότερα, παρουσιάστηκε ο EDVAC [12], ο οποίος ήταν δυαδικός υπολογιστής και όχι δεκαδικός σαν τον προκάτοχό του. Έτσι, γεννήθηκε ένας νέος τρόπος σκέψης: η έννοια ενός προγραμματιζόμενου, ηλεκτρονικού υπολογιστή με τις οδηγίες του αποθηκευμένες στη μνήμη. Μετά τον Δεύτερο Παγκόσμιο Πόλεμο, Ο Herman Goldstine μαζί με τον John von Neumann, επινόησαν μια σημειολογία που τελικά κατέληξε σε αυτό που σήμερα ονομάζουμε διάγραμμα ροής. Παράλληλα, οι Maurice Wilkes, David Wheeler και Stanley Gill, επινόησαν την έννοια της υπορουτίνας, αυξάνοντας έτσι περισσότερο τα επίπεδα αφαιρετικότητας του υπολογιστή. Τα τέλη της δεκαετίας του 1950, υπήρξαν μια ουσιαστική περίοδος για την επιστήμη των υπολογιστών καθώς όλο και περισσότεροι μεγάλοι υπολογιστές έγιναν διαθέσιμοι σε ερευνητικά ιδρύματα και πανεπιστήμια. Τότε ήταν που εμφανίστηκαν οι πρώτες γλώσσες προγραμματισμού. Η κύρια ιδέα ήταν να αντικατασταθούν οι ακολουθίες του ειδικού κώδικα οδηγιών με μαθηματικούς τύπους. Η πρώτη ευρέως γνωστή γλώσσα προγραμματισμού, η Fortran, εκδόθηκε από την IBM (1957), ακολουθούμενη σύντομα από την Algol (1958). Καθώς οι υπολογιστές χρησιμοποιήθηκαν τότε για μαθηματικούς υπολογισμούς και όχι για αποθήκευση δεδομένων και επικοινωνία, αυτές οι γλώσσες αφορούσαν κυρίως τα μαθηματικά.

1.3.2.3 Δεκαετίες 1960-1970

Τις συγκεκριμένες δεκαετίες μπήκαν οι βάσεις για την διαμόρφωση της ΜΛ όπως την γνωρίζουμε σήμερα. Το 1962 το Τμήμα Άμυνας των ΗΠΑ εξέδωσε την γλώσσα Cobol για επιχειρηματική χρήση. Το 1965 ο Dijkstra δημοσίευσε τις περίφημες σημειώσεις του για τον Δομημένο Προγραμματισμό [13] και εισήγαγε την έννοια της πειθαρχίας στον προγραμματισμό. Επίσης το 1965, ο Hoare δημοσίευσε ένα σημαντικό έγγραφο σχετικά με τη δομή δεδομένων [14]. Αυτές οι ιδέες είχαν μεγάλη επίδραση στις νέες γλώσσες προγραμματισμού, ιδίως στην Pascal που δημιούργησε ο Wirth [15]. Πλέον με την άφιξη της Pascal, ο δομημένος προγραμματισμός υποστηρίχθηκε από μια δομημένη γλώσσα προγραμματισμού. Επιπλέον, το 1966 ο Dijkstra έγραψε μια εργασία σχετικά με την αρμονική συνεργασία των διεργασιών [16], διατυπώνοντας μια πειθαρχία βασισμένη σε σηματοφόρους ως πρωταρχικά για συγχρονισμό τριών ταυτόχρονων διαδικασιών. Ο Hoare ακολούθησε το 1966 εισάγοντας τις Διαδοχικές Διαδικασίες Επικοινωνίας (CSP) [17], συνειδητοποιώντας ότι στο μέλλον οι προγραμματιστές θα έπρεπε να αντιμετωπίσουν την πρόκληση των παράλληλα εκτελούμενων διαδικασιών. Την ίδια εποχή οι Ole-Johan Dahl και Kristen Nygaard παρουσίασαν την έννοια της αντικειμενοστρέφειας, διαμέσου της Simula, μια αντικειμενοστρεφή και όχι αλγοριθμική γλώσσα προγραμματισμού. Ο Winston Royce πρότεινε τότε την ιδέα μιας επίσημης διαδικασίας ανάπτυξης λογισμικού, την διαδικασία καταρράκτη. Αν και σήμερα θεωρείται αρκετά παρωχημένη, η μεθοδολογία του ήταν για την εποχή της αρκετά προχωρημένη: εισήγαγε την έννοια της επαναληπτικής ανάπτυξης, τη σημασία του προτύπου και την αξία των αντικειμένων πέρα από τον ίδιο τον πηγαίο κώδικα. Σε συνδυασμό με τις ιδέες του David Parnas για την απόκρυψη πληροφοριών, τις ιδέες της Barbara Liskov για τους τύπους αφηρημένων δεδομένων και τις προσεγγίσεις του Peter Chen στη μοντελοποίηση οντοτήτων-σχέσεων, ο τομέας της ΜΛ απέκτησε πλέον ένα ζωντανό σύνολο ιδεών ώστε να βρει πεδίο εφαρμογής.

1.3.2.4 Δεκαετία 1980 έως σήμερα

Την δεκαετία του 1980 άρχισε να καθιερώνεται το μοντέλο της αντικειμενοστρέφειας. Εμφανίστηκαν νέες γλώσσες προγραμματισμού οι οποίες υποστήριζαν αυτό το μοντέλο, μεταξύ των οποίων η Smalltalk το 1980, η Object-Pascal το 1985, η C ++ επίσης το 1985 και η Oberon το 1988. Η μέθοδος

Booch [18] αναπτύχθηκε από τον Grady Booch το 1992 και πρόκειται για μια μέθοδο αντικειμενοστρεφούς ανάπτυξης λογισμικού. Αποτελείται από μια γλώσσα μοντελοποίησης αντικειμένων, μια επαναλαμβανόμενη διαδικασία αντικειμενοστραφής ανάπτυξης και ένα σύνολο προτεινόμενων πρακτικών. Ένα χρόνο πριν, το 1991, ο Jim Rumbaugh πρότεινε την τεχνική μοντελοποίησης αντικειμένων OMT (Object-Modeling Technic), η οποία αναπτύχθηκε ως προσέγγιση στην ανάπτυξη λογισμικού προτείνοντας τρεις βασικούς τύπους μοντέλων: μοντέλο αντικειμένου, δυναμικό μοντέλο και λειτουργικό μοντέλο. Την ίδια χρονιά, παρουσιάστηκε το Objectory, το οποίο είναι μια αντικειμενοστρεφής μεθοδολογία. Δημιουργήθηκε κυρίως από τον Ivar Jacobson, ο οποίος έχει συνεισφέρει σε μεγάλο βαθμό στην αντικειμενοστραφή μηχανική λογισμικού. Το πλαίσιο του Objectory είναι μια τεχνική σχεδιασμού που ονομάζεται σχεδίαση με δομικά στοιχεία. Με την τεχνική δομικών στοιχείων, ένα σύστημα θεωρείται ως σύστημα σύνδεσης μπλοκ με κάθε μπλοκ που αντιπροσωπεύει μια υπηρεσία συστήματος. Οι τρεις τελευταίοι . δηλαδή οι Booch, Rumbaugh και Jacobson παρουσίασαν στα μέσα της δεκαετίας του 1990 την Ενοποιημένη Γλώσσα Μοντελοποίησης (UML), η οποία θα αναλυθεί περαιτέρω σε επόμενο κεφάλαιο. Το 1995, ο Philippe Kruchten [19], πρότεινε το Μοντέλο Αρχιτεκτονικής Προβολής 4 + 1 (σχήμα 2). Το μοντέλο

Σχήμα 2: Μοντέλο Αρχιτεκτονικής Προβολής 4 + 1

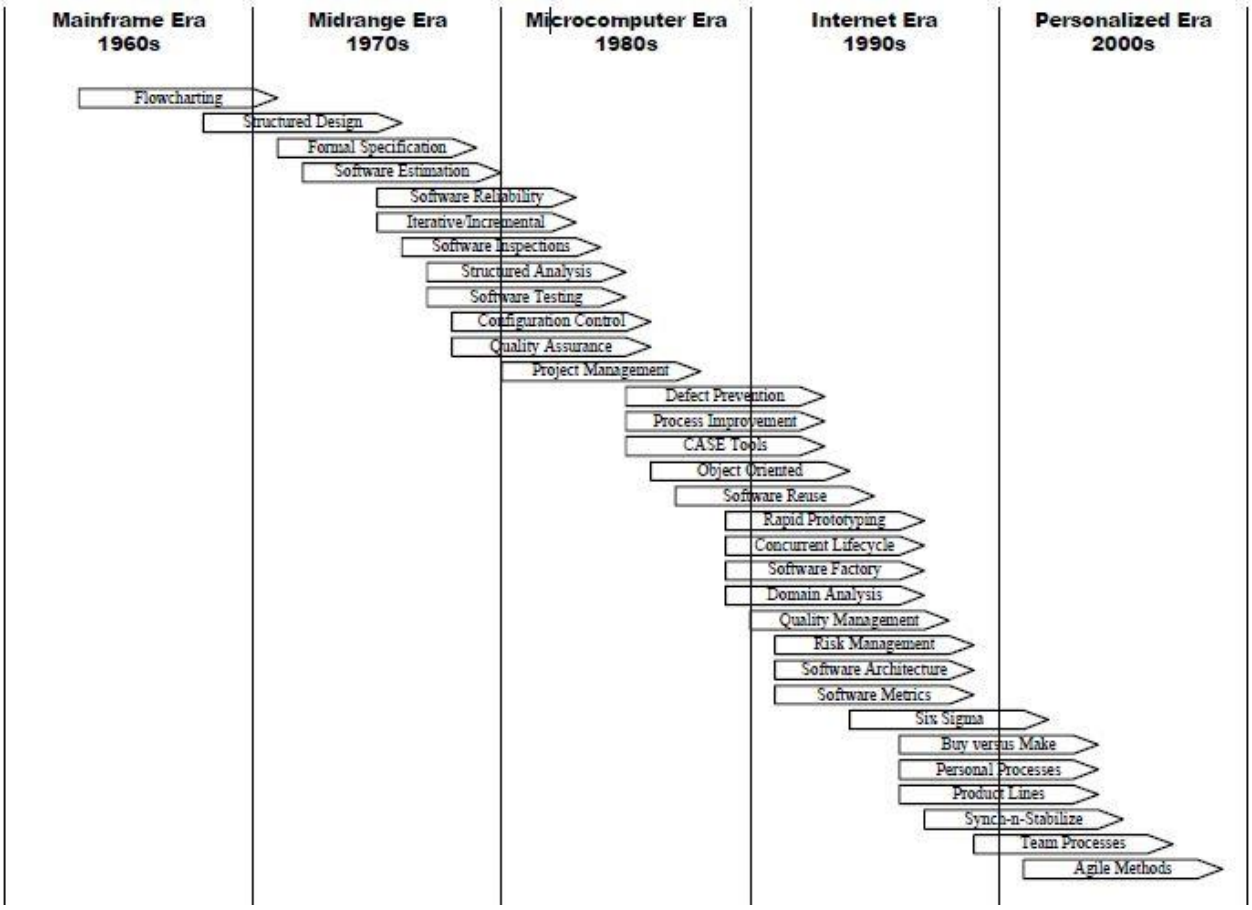
προβολών 4 + 1 είναι ένα πλαίσιο οργάνωσης πληροφοριών. Αποτελείται από την λογική προβολή, την προβολή διεργασίας, την προβολή ανάπτυξης και τη φυσική προβολή μιας εφαρμογής και πληροφορίες από την προοπτική του τελικού χρήστη. Ο Vic Basili παρουσίασε την προσέγγιση της Πειραματικής Μηχανικής Λογισμικούς (Experimental software engineering), η οποία επικεντρώνεται στη συλλογή αποδεικτικών στοιχείων, μέσω μετρήσεων και πειραμάτων που περιλαμβάνουν τα συστήματα λογισμικού (προϊόντα λογισμικού, διαδικασίες και πόροι). Ο Barry Boehm δημιούργησε το σπειροειδές μοντέλο ανάπτυξης λογισμικού, στο οποίο οι φάσεις ανάπτυξης επανεξετάζονται επανειλημμένα. Αυτή η επαναληπτική διαδικασία ανάπτυξης λογισμικού ενέπνευσε την προσέγγιση

MBASE και τον ακραίο προγραμματισμό. Οι Harlan Mills et al. [20] πρότειναν την Διαδικασία Μηχανικής Λογισμικού Cleanroom, η οποία είναι μια διαδικασία ανάπτυξης λογισμικού που αποσκοπεί στην παραγωγή λογισμικού με πιστοποιημένο επίπεδο αξιοπιστίας. Ο στόχος της διαδικασίας cleanroom είναι η πρόληψη των λαθών και όχι η εκ των υστέρων διόρθωση. Οι βασικές αρχές της διαδικασίας cleanroom είναι τρεις: ανάπτυξη λογισμικού βασισμένη σε τυπικές μεθόδους, αυξητική εφαρμογή κάτω από στατιστικό έλεγχο ποιότητας και στατιστικά ορθός έλεγχος.

Τέλος, υπήρξε η έλευση νέων γλωσσών προγραμματισμού, όπως για παράδειγμα, η JavaScript, η Python, η C #, η PHP και η Swift. Είχαμε την έλευση του Διαδικτύου, την εξέλιξή του σε de facto πλατφόρμα υπολογιστών και την μεταφορά των δεδομένων σε τεχνολογία cloud.

1.3.2.5 Χρονοδιάγραμμα

Αν και αρκετά μακροσκελές το κεφάλαιο με την ιστορική αναδρομή του όρου της ΜΛ, δεν αναφέρθηκαν όλα τα ορόσημα της ιστορίας της ΜΛ – άλλωστε η υπερανάλυση αυτή δεν θα ενέπιπτε στα πλαίσια αυτής της εργασίας. Ωστόσο, στο σχήμα 3 μπορούμε να δούμε το χρονοδιάγραμμα που ετοίμασε ο David F. Rico:



Σχήμα 3: Χρονοδιάγραμμα ΜΛ

Το χρονοδιάγραμμα χωρίζεται σε δεκαετίες και σύμφωνα με τον ίδιο τον Rico, «προσδιορίζονται τριάντα δύο μεγάλες κλάσεις μεθόδων λογισμικού που έχουν εμφανιστεί τα τελευταία 50 χρόνια. Υπάρχουν πολλές παραλλαγές κάθε μείζονος κλάσης μεθόδου λογισμικού, η οποία καθιστά τον αριθμό των μεθόδων λογισμικού στις εκατοντάδες»[21].

1.3.3 Η ανάγκη για την εφαρμογή της ΜΛ

Η ανάγκη για την εφαρμογή της ΜΛ προκύπτει λόγω του όλο και υψηλότερου ποσοστού των αλλαγών στις απαιτήσεις των χρηστών και του περιβάλλοντος στο οποίο λειτουργεί το λογισμικό. Η ΜΛ θεωρείται σημαντική επειδή στις μέρες μας χρησιμοποιείται ειδικευμένο λογισμικό σε σχεδόν κάθε κλάδο, σε κάθε επιχείρηση και για κάθε λειτουργία. Η ανάγκη για γρήγορη, αξιόπιστη και αποτελεσματική παράδοση των εφαρμογών συνεχώς αυξάνεται, καθώς το λογισμικό γίνεται όλο και μεγαλύτερο και πολυπλοκότερο. Σε γενικές γραμμές μπορούμε να πούμε ότι η ΜΛ εφαρμόζεται στην προσπάθεια επίτευξης των κάτωθι:

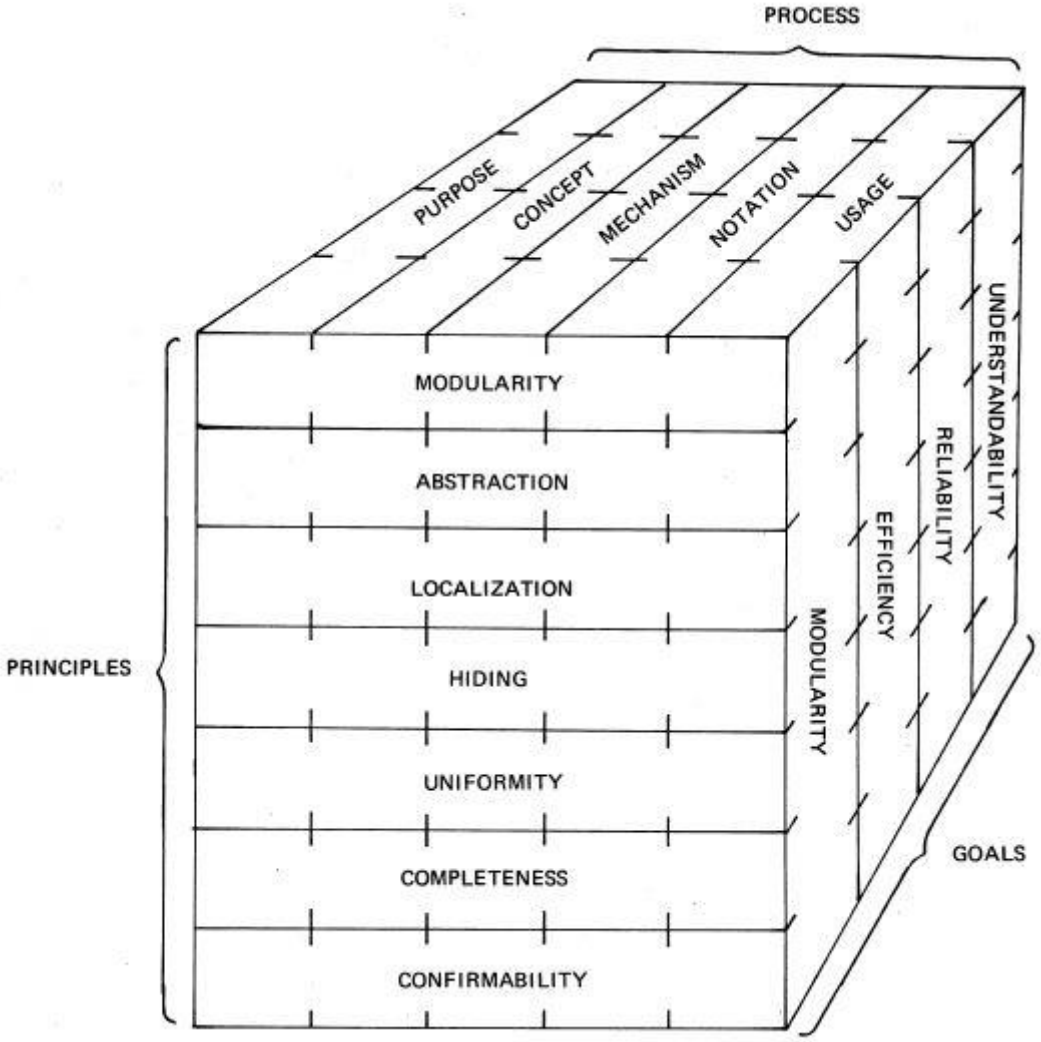
- **Μείωση της πολυπλοκότητας:** τα μεγάλα έργα λογισμικού είναι πάντα περίπλοκα και δύσκολο να αναπτυχθούν. Η ΜΛ έχει μια εξαιρετική λύση για τη μείωση της πολυπλοκότητας κάθε έργου: χωρίζει μεγάλα προβλήματα σε πολλά μικρά προβλήματα. Κατόπιν επιλύεται ένα προς ένα κάθε μικρό πρόβλημα. Όλα αυτά τα μικρά προβλήματα επιλύονται ανεξάρτητα μεταξύ τους από την ίδια ή διαφορετικές ομάδες ανάπτυξης.
- **Ελαχιστοποίηση του κόστους λογισμικού:** είναι γνωστό ότι, σε γενικές γραμμές, για την ανάπτυξη του λογισμικού απαιτούνται πολλές εργατοώρες και ότι οι μηχανικοί λογισμικού είναι υψηλόμισθοι επαγγελματίες. Απαιτείται δηλαδή μεγάλη ανθρώπινη δύναμη για την ανάπτυξη λογισμικού που αποτελείται από εκατομμύρια γραμμές κώδικα. Με την αρωγή ωστόσο της ΜΛ, οι προγραμματιστές σχεδιάζουν τα πάντα με τέτοιο τρόπο, ώστε να μειωθούν όλα αυτά που δεν θεωρούνται απαραίτητα. Σαν φυσικό επακόλουθο, το κόστος παραγωγής λογισμικού γίνεται μικρότερο σε σύγκριση με οποιοδήποτε λογισμικό που δεν χρησιμοποιεί την προσέγγιση που προτείνει η ΜΛ.
- **Μείωση του χρόνου:** οποιαδήποτε διαδικασία ανάπτυξης δεν ακολουθήσει το επιλεγμένο πλάνο, θα έχει σαν αποτέλεσμα τη χρονική καθυστέρηση του έργου. Στην περίπτωση ανάπτυξης μεγάλου σε μέγεθος λογισμικού, ίσως χρειαστούν πολλές δοκιμές στον κώδικα μέχρι η ομάδα ανάπτυξης να καταλήξει στον τελικό κώδικα λειτουργίας. Η παραπάνω διαδικασία είναι πολύ χρονοβόρα και, εάν δεν διαχειριστεί σωστά, μπορεί να κοστίσει πολύ χρόνο στην ολοκλήρωση του έργου. Στην αντίθετη περίπτωση, δηλαδή αν ακολουθηθεί πιστά το σχέδιο που έχει προτείνει η ΜΛ, τότε ο χρόνος ανάπτυξης θα μειωθεί κατά πολύ.
- **Επιτυχή διαχείριση μεγάλων έργων:** τα μεγάλα έργα δεν μπορούν να ολοκληρωθούν σε λίγες μόνο μέρες και απαιτούν υπομονή, ολοκληρωμένο σχεδιασμό και σωστή διαχείριση. Οι εταιρείες-πελάτες επενδύουν πόρους στα έργα λογισμικού και για να ολοκληρωθούν αυτά απαιτούνται από την ομάδα ανάπτυξης σωστός προγραμματισμός, κατεύθυνση, συνεχείς δοκιμές και συντήρηση μετά την ολοκλήρωση του έργου. Σε συνέχεια όλων των παραπάνω, είναι ξεκάθαρο ότι για τον σωστό χειρισμό των μεγάλων έργων λογισμικού, θα πρέπει να ακολουθηθεί η κατάλληλη προσέγγιση που θα προτείνει η ΜΛ.
- **Ανάπτυξη αξιόπιστου λογισμικού:** το λογισμικό θα πρέπει να είναι αξιόπιστο, αυτό σημαίνει ότι μετά την παράδοση στον πελάτη, θα πρέπει να λειτουργεί απρόσκοπτα τουλάχιστον για το δεδομένο χρονικό διάστημα που έχει προσυμφωνηθεί. Σε περίπτωση που διαπιστωθούν σφάλματα στο λογισμικό, θα πρέπει να γίνουν άμεσα οι κατάλληλες επεμβάσεις ώστε αυτά να επιλυθούν. Η ΜΛ επιμένει αρκετά στην εφαρμογή επανειλημμένων δοκιμών και στη συντήρηση σε τακτά χρονικά διαστήματα, ώστε να διασφαλίζεται έτσι η αξιοπιστία του λογισμικού.
- **Αποτελεσματικότητα:** η αποτελεσματικότητα ακολουθεί κατά κανόνα την πιστή εφαρμογή των κατάλληλων προτύπων. Τα πρότυπα λογισμικού, τα οποία συστήνονται από την ΜΛ,

αποτελούν το συστατικό στο οποίο εστιάζουν περισσότερο οι εταιρείες ανάπτυξης λογισμικού για την επίτευξη αποτελεσματικότερου λογισμικού.

- **Διαχείριση ποιότητας:** η επιλογή της καταλληλότερης διαδικασίας ανάπτυξης λογισμικού είναι αυτή που εντέλει επιφέρει σαν αποτέλεσμα το καλύτερο και ποιοτικό προϊόν.

1.3.4 Αρχές και Στόχοι της ΜΛ

Ήδη από το 1975, οι Douglas T. Ross et al. [22] αναφέρονται στις αρχές και τους στόχους της ΜΛ, προτείνοντας παράλληλα ένα σχήμα που περιέχει τα δομικά στοιχεία της Μηχανικής Λογισμικού (σχήμα 4):



Σχήμα 4: Τα δομικά στοιχεία της ΜΛ

Στο παραπάνω σχήμα φαίνονται τα δομικά στοιχεία που χρησιμοποιεί η ΜΛ και ο τρόπος που αυτά συνδέονται και αλληλεπιδρούν μεταξύ τους. Χωρίζονται σε τρεις κατηγορίες, τις αρχές (principles), τους στόχους (goals) και την διαδικασία (process). Δημιουργούνται επιμέρους κατάλληλα μπλοκ που αποτελούνται από μία από τις αρχές, το στόχο που θέλουμε να εκπληρώσουμε και την διαδικασία που έχει επιλεχθεί για την επίτευξη του στόχου.

Οι αρχές είναι επτά και χρησιμοποιούνται, μεμονωμένα και σε συνδυασμό, για τον προσδιορισμό και τον έλεγχο των σχέσεων που δημιουργούνται με τα υπόλοιπα δομικά στοιχεία. Χρησιμοποιούνται

ουσιαστικά ως κριτήρια απόφασης για να διασφαλιστεί ότι η προκύπτουσα σχέση επιτυγχάνει τους στόχους μας. Οι επτά αρχές είναι οι ακόλουθες:

- **Αρθρωτότητα (Modularity):** ο καλύτερος ορισμός για την αρθρωτότητα δίδεται από τον Myers, ο οποίος διευκρινίζει ότι «η αρθρωτότητα δεν επιτυγχάνεται απλά με την αυθαίρετη διαίρεση ενός μεγάλου προγράμματος σε μικρότερα μέρη ή ενότητες. Ο πρωταρχικός στόχος θα πρέπει να είναι η αποδόμηση του προγράμματος να γίνεται με τέτοιο τρόπο ώστε οι ενότητες να είναι ιδιαίτερα ανεξάρτητες η μία από την άλλη»[23]. Η αρχή της αρθρωτότητας λοιπόν, καθορίζει τον κατάλληλο τρόπο με τον οποίο δομείται ένα σύστημα λογισμικού.
- **Αφαιρετικότητα (Abstraction):** η αρχή της αφαιρετικότητας είναι η εξαγωγή των βασικών ιδιοτήτων, παραλείποντας παράλληλα τις λεπτομέρειες. Με αυτόν τον τρόπο επιτυγχάνεται μια μορφή ιεραρχικής αποδόμησης, κατά την οποία τα επίπεδα βρίσκονται στην αφαιρετική μορφή τους. Ουσιαστικά, το κάθε επίπεδο παρουσιάζει περιληπτικά αυτά που περιέχονται στα κατώτερα από αυτό επίπεδα, με τη λογική ότι οι λεπτομέρειες υπάγονται σε χαμηλότερα επίπεδα. Η αρχή της αφαιρετικότητας όταν συνδυαστεί με την αρχή της πληρότητας (completeness) διασφαλίζει ότι ένα δεδομένο επίπεδο κατά την αποδόμηση θα είναι κατανοητό ως μονάδα, χωρίς να απαιτούνται είτε λεπτομερείς γνώσεις από χαμηλότερα επίπεδα ή τον ρόλο που εκτελεί στο σύστημα (όπως αυτό φαίνεται από το επίπεδο που βρίσκεται πάνω από αυτό).
- **Τοπικοποίηση (Localization):** η αρχή της τοπικοποίησης αφορά την φυσική εγγύτητα. Τα συναφή στοιχεία θα πρέπει να βρίσκονται όλα συγκεντρωμένα σε ένα μέρος. Έτσι, οι άλλες αρχές μπορούν εύκολα να συσχετίσουν τα τοπικοποιημένα στοιχεία ώστε να εξυπηρετήσουν συγκεκριμένους σκοπούς.. Οι υπορουτίνες, οι πίνακες, οι λογικές και φυσικές εγγραφές είναι παραδείγματα της αρχής της τοπικοποίησης.
- **Απόκρυψη (Hiding):** συχνά συνδέεται με την ιδέα της «αναβολής δεσμευτικών αποφάσεων» στην επίλυση των προβλημάτων με χρήση της προσέγγισης «από πάνω προς τα κάτω», αλλά δεν είναι ακριβώς το ίδιο. Ο σκοπός της απόκρυψης είναι παρόμοιος με αυτόν της αφαιρετικότητας, όσον αφορά τουλάχιστον την απαίτηση του να παραμένουν ορατές μόνο εκείνες οι ιδιότητες της μονάδας που απαιτούνται για τη διασύνδεση με τις άλλες ενότητες. Ωστόσο, η απόκρυψη διαφέρει από την αφαιρετικότητα στο γεγονός ότι ο σκοπός της απόκρυψης είναι να κάνει απρόσιτες ορισμένες λεπτομέρειες, οι οποίες δεν θα πρέπει να επηρεάζουν άλλα μέρη ενός συστήματος. Ενώ δηλαδή η αφαιρετικότητα βοηθά στον εντοπισμό λεπτομερειών που πρέπει να κρύβονται, η απόκρυψη αφορά τον ορισμό και την επιβολή περιορισμών πρόσβασης που, χωρίς την αρχή της απόκρυψης, θα ήταν έμμεσες μόνο σε κάποιο σκοπό, έννοια, μηχανισμό, σημειογραφία ή περιγραφή χρήσης.
- **Ομοιομορφία (Uniformity):** η ομοιομορφία, η έλλειψη δηλαδή της ασυνέπειας και των περιττών διαφορών, είναι επίσης μια σημαντική αρχή. Όταν εφαρμόζεται σε σημειογραφικά θέματα, η ομοιομορφία αποδίδει αποτελέσματα χωρίς σύγχυση και δαπανηρές ασυνέπειες. Πρακτικά, η αρχή της ομοιομορφίας εξασφαλίζει τη συνέπεια σε ένα σύστημα λογισμικού.
- **Πληρότητα (Completeness):** η πληρότητα αποτελεί προφανέστατα μια σημαντική αρχή. Ο σκοπός αυτής της αρχής είναι να διασφαλιστεί ότι όλα τα βασικά στοιχεία μιας αφαίρεσης, για παράδειγμα, είναι ρητά και ότι δεν έχει παραλειφθεί κάποιο από αυτά, που ίσως να είναι απαραίτητο. Αυτό βέβαια, δεν σημαίνει ότι θα πρέπει να εμφανίζονται τα πάντα με κάθε λεπτομέρεια – απλώς το σύνολο των αφηρημένων εννοιών θα πρέπει να είναι σε θέση να καλύψει κάθε λεπτομέρεια.

- **Επιβεβαίωση (Confirmability):** η επιβεβαίωση είναι μια αρχή που κατευθύνει την προσοχή της στους μεθόδους, ώστε να διαπιστωθεί εάν οι δηλωθέντες στόχοι έχουν επιτευχθεί. Όταν εφαρμόζεται σε θέματα σχεδιασμού, η δυνατότητα επιβεβαίωσης αναφέρεται στη δομή ενός συστήματος, ούτως ώστε αυτή να ελέγχεται άμεσα. Θα πρέπει να υπάρχει η δυνατότητα διέγερσης κάποιου δομημένου συστήματος με ελεγχόμενο τρόπο, ώστε να μπορεί να αξιολογηθεί η ορθότητα της απόκρισης του.

Όσον αφορά τους στόχους της μηχανικής λογισμικού που προτείνονται από το σχήμα των δομικών στοιχείων της ΜΛ, τέσσερις είναι οι ιδιότητες που θεωρούνται αρκετά γενικές ώστε να γίνονται αποδεκτές ως στόχοι για ολόκληρη την πειθαρχία της μηχανικής λογισμικού. Αυτές είναι η δυνατότητα τροποποίησης, η αποδοτικότητα, η αξιοπιστία και η ευκολία στην κατανόηση:

- **Δυνατότητα τροποποίησης (Modifiability):** κάθε τροποποίηση συνεπάγεται κάποιες ελεγχόμενες αλλαγές, κατά τις οποίες ορισμένα μέρη ή σημεία θα πρέπει να παραμείνουν ίδια, ενώ άλλα θα πρέπει να τροποποιηθούν. Όλα αυτά θα πρέπει να γίνουν με τέτοιον τρόπο, ώστε να επιτυγχάνεται το επιθυμητό νέο αποτέλεσμα. Η δυσκολία της επιτυχούς τροποποίησης αυξάνεται καθώς υπάρχουν πάρα πολλοί λόγοι για τους οποίους μπορεί να χρειαστούν αλλαγές, όπως επίσης και πολλοί διαφορετικοί τρόποι προσέγγισης για την επίλυση.
- **Αποδοτικότητα (Efficiency):** η αποδοτικότητα μετράει πόση την σπατάλη χρόνου και εργασίας κατά τη διάρκεια ανάπτυξης μιας διαδικασίας ή ενός συστήματος. Μια διαδικασία υψηλής απόδοσης έχει λιγότερα χαμένα στοιχεία. Όταν εφαρμόζεται σε έργα πληροφορικής, η αποδοτικότητα μετρά πώς το επίπεδο στελέχωσης επηρεάζει πόση δουλειά μπορεί να γίνει. Το πρόβλημα είναι ότι ενώ σαν ιδέα είναι αρκετά απλή, είναι δύσκολη η εφαρμογή της διότι απαιτεί μια συστημοκεντρική άποψη κατά την ανάπτυξη του λογισμικού. Ως εκ τούτου, είναι δύσκολο να μετρηθεί άμεσα.
- **Αξιοπιστία (Reliability):** ο σκοπός της αξιοπιστίας είναι από τη μία η αποτροπή των αστοχιών κατά τη σύλληψη, το σχεδιασμό και την ανάπτυξη, και από την άλλη η δυνατότητα άμεσης ανάκαμψης μετά από αστοχία λειτουργίας ή απόδοσης Σε αντίθεση με την αποδοτικότητα, η οποία συχνά εφαρμόζεται πρόωρα, η αξιοπιστία εφαρμόζεται πολύ αργά ή καθόλου στις περισσότερες προσπάθειες ανάπτυξης λογισμικού. Η αξιοπιστία μπορεί να ενσωματωθεί μόνο κατά την έναρξη του έργου και δεν μπορεί να προστεθεί στο τέλος. Ως εκ τούτου, η αξιοπιστία έχει διεισδυτική και κρίσιμη επίδραση στις πρακτικές μηχανικής λογισμικού.
- **Ευκολία στην κατανόηση (Understandability):** ο τελευταίος στόχος που ασκεί ισχυρή επιρροή σε όλες τις πτυχές της ΜΛ είναι η κατανόηση. Θα πρέπει να σημειωθεί ότι ευκολία στην κατανόηση δεν σημαίνει κατανόηση σε επίπεδο αναγνωσιμότητας – αντιθέτως αποτελεί μια σημαντικότερη έννοια, η οποία συμπεριλαμβάνει ολόκληρη την εννοιολογική δομή του έργου. Ένα άλλο σημείο το οποίο θα πρέπει να διευκρινιστεί είναι ότι, σε οποιαδήποτε δεδομένη περίπτωση, ένα αποδεκτό επίπεδο κατανόησης είτε υπάρχει είτε όχι (δεν υπάρχει ενδιάμεσο στάδιο).

Η διαδικασία αποτελείται από μια ακολουθία πέντε βημάτων, η οποία προσομοιάζει τη διαδικασία της ανάπτυξης λογισμικού. Τα πέντε βήματα της διαδικασίας είναι τα εξής:

- **Σκοπός (Purpose):** καθορισμός των απαιτήσεων για ένα σύστημα.

- **Σχέδιο (Concept):** σχεδίαση της αρχιτεκτονικής ενός συστήματος λογισμικού, ούτως ώστε να ικανοποιηθούν οι απαιτήσεις και να προσδιοριστούν οι ενότητες που αποτελούν το σύστημα.
- **Μηχανισμός (Mechanism):** υλοποίηση του συστήματος λογισμικού (οι μηχανισμοί εδώ είναι προφανώς ο κώδικας, ο εντοπισμός σφαλμάτων, ο συντονισμός των δραστηριοτήτων κατά τη διαδικασία της ανάπτυξης).
- **Σημειογραφία (Notation):** καθορισμός της γλώσσας εντολών ή άλλων μέσων, τα οποία θα χρησιμοποιήσει ο χρήστης για να επικαλεστεί τις δυνατότητες του συστήματος λογισμικού.
- **Χρήση (Usage):** περιγραφή του τρόπου ελέγχου του συστήματος λογισμικού (αυτή η περιγραφή ενδέχεται να έχει τη μορφή εγχειριδίου χρήστη για το σύστημα).

1.3.5 Ηθικές αρχές της ΜΛ

Το Διοικητικό Συμβούλιο της IEEE Computer Society δημιούργησε μια διευθύνουσα επιτροπή τον Μάιο του 1993 για την αξιολόγηση, τον προγραμματισμό και τον συντονισμό δράσεων που σχετίζονται με την καθιέρωση του μηχανικού λογισμικού ως επάγγελμα. Την ίδια χρονιά, το συμβούλιο της ACM, συνέστησε τη σύσταση της Επιτροπής Μηχανικής Λογισμικού . Τον Ιανουάριο του 1994, και οι δύο εταιρείες συγκρότησαν μια κοινή συντονιστική επιτροπή για να καθοριστούν τα κατάλληλα πρότυπα για την επαγγελματική πρακτική του επαγγέλματος του μηχανικού λογισμικού, στις οποίες θα μπορούσαν να βασίζονται οι βιομηχανικές αποφάσεις, η επαγγελματική πιστοποίηση και τα εκπαιδευτικά προγράμματα. Έτσι, οι επαγγελματίες της ΜΛ θα δεσμεύονται να πραγματοποιούν την ανάλυση, τις προδιαγραφές, τον σχεδιασμό, την ανάπτυξη, τον έλεγχο και την συντήρηση του λογισμικού κατά τέτοιο τρόπο, ώστε να διατηρείται η καλή φήμη και να βελτιώνεται η εικόνα του επαγγέλματος του μηχανικού λογισμικού.

Οι Gotterbarn et al. σε άρθρο τους [24] μας υποδεικνύουν τις ακόλουθες οκτώ ηθικές αρχές της ΜΛ (σύμφωνα με τη δέσμευσή τους σχετικά με την υγεία, την ασφάλεια και την ευημερία του κοινού), που θα πρέπει να τηρούν οι μηχανικοί λογισμικού :

1. Δημόσιο συμφέρον: Οι μηχανικοί λογισμικού θα πρέπει να ενεργούν με συνέπεια και με γνώμονα το δημόσιο συμφέρον.

2. Πελάτης και εργοδότης: Οι μηχανικοί λογισμικού θα πρέπει να ενεργούν με τον καλύτερο δυνατό τρόπο για τα συμφέροντα του πελάτη και του εργοδότη τους, σύμφωνα πάντα με το δημόσιο συμφέρον.

3. Προϊόν: Οι μηχανικοί λογισμικού θα πρέπει να διασφαλίζουν ότι τα προϊόντα τους, καθώς και οι σχετικές τροποποιήσεις των προϊόντων τους, πληρούν τα υψηλότερα δυνατά επαγγελματικά πρότυπα.

4. Ακεραιότητα: Οι μηχανικοί λογισμικού θα πρέπει πάντα να διατηρούν την ακεραιότητα και την ανεξαρτησία τους, όσον αφορά την επαγγελματική τους κρίση.

5. Διαχείριση: Οι υπεύθυνοι του έργου καθώς και αυτοί που ηγούνται των ομάδων ανάπτυξης, θα πρέπει να ακολουθούν μια ηθική προσέγγιση στη διαχείριση της ανάπτυξης και συντήρησης του λογισμικού.

6. Εικόνα Επαγγέλματος: Οι μηχανικοί λογισμικού θα πρέπει πάντα να προωθούν την ακεραιότητα και την καλή φήμη του επαγγέλματος, ώστε να συνάδει με το δημόσιο συμφέρον.

7. Συναδελφικότητα: Οι μηχανικοί λογισμικού θα πρέπει να είναι δίκαιοι και να υποστηρίζουν τους συναδέλφους τους.

8. Προσωπική ανάπτυξη: Οι μηχανικοί λογισμικού θα πρέπει να συμμετέχουν στη διά βίου μάθηση σχετικά με την πρακτική του επαγγέλματός τους, όπως και να προωθούν μια ηθική προσέγγιση κατά την πρακτική του.

Οι παραπάνω ηθικές αρχές δεν θα πρέπει να αντιμετωπίζονται σαν την απόλυτη διαχωριστική γραμμή ανάμεσα στη αποδεκτή και στη μη αποδεκτή επαγγελματική συμπεριφορά για όλες τις πρακτικές καταστάσεις. Δεν αποτελούν δηλαδή έναν απλό ηθικό αλγόριθμο που δημιουργεί ηθικές αποφάσεις. Σε ορισμένες περιπτώσεις, κάποιες από τις παραπάνω αρχές μπορεί να είναι αντικρουόμενες μεταξύ τους ή με πρότυπα από άλλες πηγές. Αυτές οι καταστάσεις απαιτούν από τον μηχανικό λογισμικού να χρησιμοποιεί την ηθική του κρίση, να ενεργεί δηλαδή κατά τέτοιο τρόπο ώστε να είναι συνεπής με το πνεύμα του Κώδικα Δεοντολογίας και Επαγγελματικής Πρακτικής, λαμβάνοντας πάντα υπόψη τις περιστάσεις.

1.4 Ποιότητα Λογισμικού

Οι εταιρείες ανάπτυξης λογισμικού λειτουργούν μέσα σε μια εξαιρετικά ανταγωνιστική αγορά, μέσα στην οποία προσπαθούν αδιάκοπα να προσελκύσουν νέους και να διατηρήσουν τους παλιούς πελάτες. Ένα από τα βασικά κριτήρια επιτυχίας στο συγκεκριμένο κλάδο, είναι η διασφάλιση υψηλής ποιότητας προϊόντων λογισμικού. Η ποιότητα του λογισμικού είναι ο βαθμός συμμόρφωσής του με τις δηλωμένες λειτουργικές και μη λειτουργικές απαιτήσεις.

1.4.1 Ορισμοί

Το 1979 ο Philip Crosby όρισε την ποιότητα ως την «συμμόρφωση με τις απαιτήσεις των χρηστών»[25], ενώ οι Juran και Grynna το 1995 όρισαν την ποιότητα ως «την καταλληλότητα προς χρήση» [26]. Η IBM χρησιμοποίησε τη φράση «ποιότητα με γνώμονα την αγορά», η οποία βασίζεται στην επίτευξη της απόλυτης ικανοποίησης των πελατών. Οι Bevan και Azuma διευκρίνισαν ότι η ποιότητα στο λογισμικό είναι ένα υποκειμενικό μέγεθος. Δεν έχουν δηλαδή όλοι χρήστες ενός λογισμικού την ίδια άποψη αναφορικά με την ποιότητα του. Σε κάθε περίπτωση, οι χρήστες είναι σε θέση να εκτιμήσουν την διεπαφή χρήστη, την αποδοτικότητα και τη συχνότητα εμφάνισης σφαλμάτων, δηλαδή τα **εξωτερικά χαρακτηριστικά** του λογισμικού, ώστε να καθορίσουν άποψη για το λογισμικό. Πριν όμως το λογισμικό φτάσει στους χρήστες, αξιολογείται ως προς την ποιότητά του από τους κατασκευαστές του. Εξετάζουν και αξιολογούν τα **εσωτερικά χαρακτηριστικά** του λογισμικού, δηλαδή το πλήθος και το είδος των αστοχιών που συναντούν κατά τις δοκιμές, χρησιμοποιώντας ειδικές μετρήσεις.

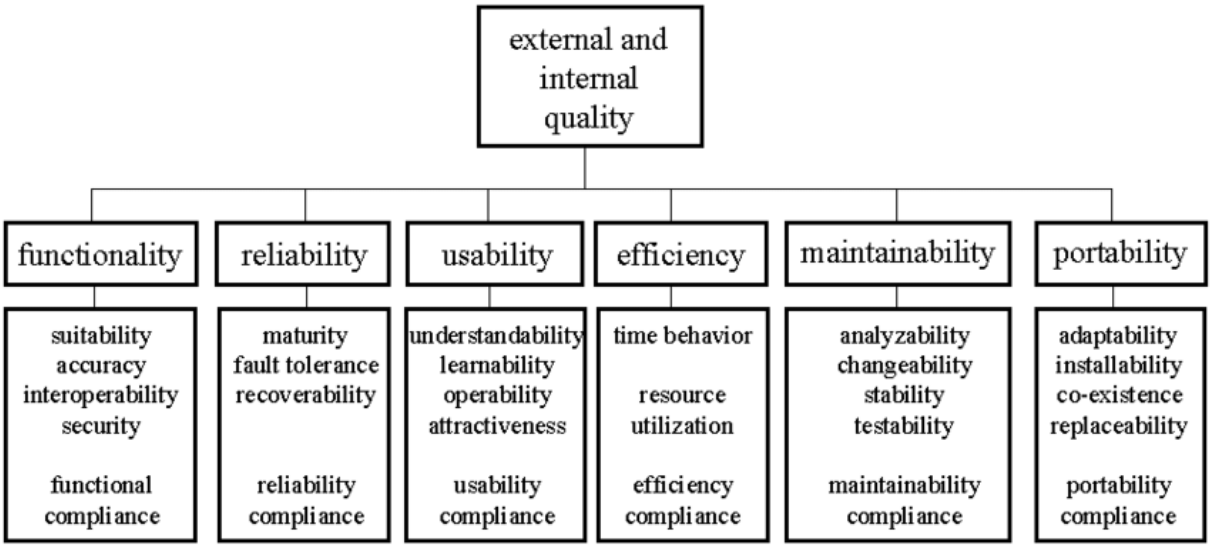
1.4.2 Γενικά χαρακτηριστικά και πρότυπα ποιότητας λογισμικού

Σε γενικές γραμμές μπορούμε να πούμε ότι κάποια από τα χαρακτηριστικά που φέρει ένα ποιοτικό λογισμικό είναι:

- Αξιοπιστία
- Διαθεσιμότητα
- Συντηρητικότητα
- Επεκτασιμότητα
- Αποδοτικότητα

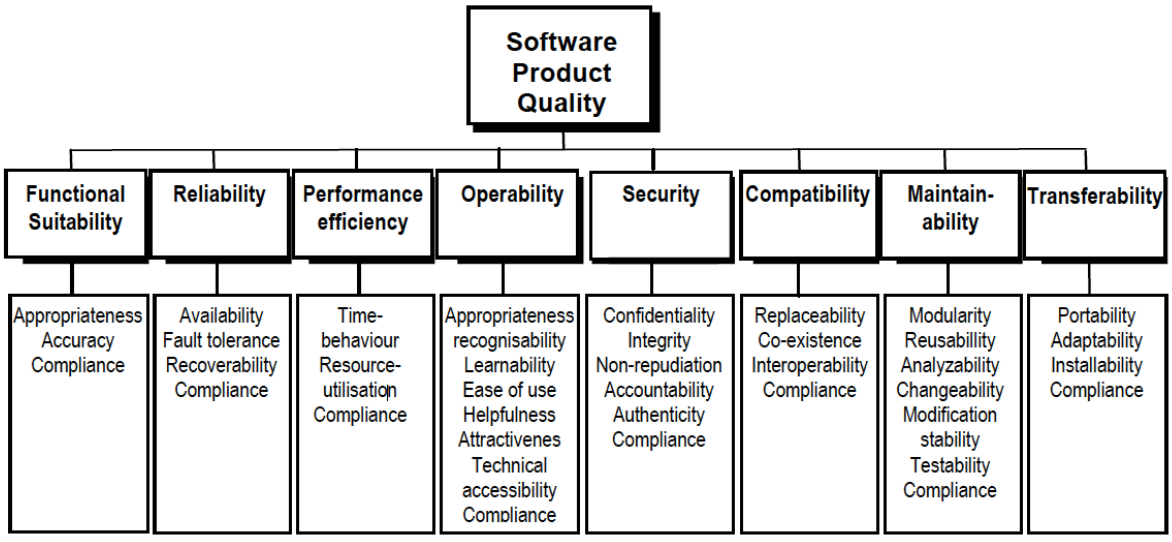
- Ευχρηστία
- Προσαρμοστικότητα

Συγκεκριμένα, το μοντέλο ποιότητας προϊόντος λογισμικού που προτείνεται από το πρότυπο ISO / IEC 9126-1 [27] ορίζει έξι χαρακτηριστικά ποιότητας: λειτουργικότητα, αξιοπιστία, χρηστικότητα, συντηρησιμότητα, φορητότητα και αποδοτικότητα (Σχήμα 5). Επιπλέον, εισάγει και την **ποιότητα στη χρήση** , η οποία ορίζεται από την αποτελεσματικότητα, την παραγωγικότητα, την ασφάλεια και την ικανοποίηση. Τα αναφερόμενα έξι ποιοτικά χαρακτηριστικά έχουν ορίσει υποχαρακτηριστικά, ώστε να καλύπτονται όλες οι ποιοτικές πτυχές που ενδιαφέρουν τα περισσότερα προϊόντα λογισμικού και να λειτουργεί έτσι ως λίστα ελέγχου για την εξασφάλιση της πλήρους κάλυψης της ποιότητας.



Σχήμα 5: Το πρότυπο ISO / IEC 9126-1

Ακολούθησε το πρότυπο ISO/IEC 25010:2011 το οποίο επέκτεινε τα χαρακτηριστικά ποιότητας και πλέον ανέρχονται σε οχτώ: Λειτουργική καταλληλότητα, αξιοπιστία, λειτουργικότητα, απόδοση, ασφάλεια, συμβατότητα, συντηρησιμότητα και φορητότητα (Σχήμα 6).



Σχήμα 6: Το πρότυπο ISO/IEC 25010:2011

Επιπλέον, το πρότυπο ISO/IEC 25010:2011 ορίζει ένα μοντέλο ποιότητας που αποτελείται από πέντε χαρακτηριστικά, όσον αφορά τη χρήση του λογισμικού: Αποτελεσματικότητα, αποδοτικότητα, ικανοποίηση, ασφάλεια και ευχρηστία.

1.4.3 Οπτικές θεωρήσεις αξιολόγησης της ποιότητας λογισμικού

Ο Garvin [28] εισήγαγε το 1984 πέντε οπτικές γωνίες μέσω των οποίων πρότεινε να γίνεται η αξιολόγηση του λογισμικού:

- **End user view (θεώρηση από την πλευρά του χρήστη):** Η ποιότητα ορίζεται ως ο βαθμός ετοιμότητας και καταλληλότητας του λογισμικού για χρήση. Περιλαμβάνει έλεγχο της αξιοπιστίας και της συχνότητας εμφάνισης σφαλμάτων του λογισμικού.
- **Manufacturing View (κατασκευαστική θεώρηση):** Η ποιότητα σχετίζεται με την ικανοποίηση των απαιτήσεων και των ιδιαιτεροτήτων που εκφράζονται από τον χρήστη. Συγκεκριμένα, εξετάζεται αν ο σχεδιασμός και η κατασκευή του λογισμικού έγινε με τον καλύτερο τρόπο, ώστε να περιοριστούν οι διορθώσεις σε αργότερο στάδιο.
- **Product View (θεώρηση από την πλευρά του προϊόντος):** Η ποιότητα εξωτερικεύεται από τα χαρακτηριστικά του προϊόντος. Βασίζεται στην υπόθεση ότι το τελικό προϊόν θα είναι αξιόλογο, μόνο αν τα εσωτερικά χαρακτηριστικά του λογισμικού είναι ποιοτικά.
- **Value based view (θεώρηση βάσει της αξίας):** Η ποιότητα θα πρέπει να συναρτάται άμεσα από την αξία που είναι διατεθειμένος να πληρώσει ο πελάτης για να αποκτήσει το προϊόν. Βέβαια, είναι αρκετά δύσκολη η εφαρμογή αυτής της θεώρησης στην πράξη καθώς είναι πολύ δύσκολο να βρεθεί η χρυσή τομή στη σχέση τιμή/απόδοση του προϊόντος.
- **Transcendental View (υπερβατική θεώρηση):** Η ποιότητα είναι κάτι που είναι δυνατόν να αναγνωριστεί, άλλα όχι να επιτευχθεί ή να οριστεί με πληρότητα.

Για να αξιολογηθεί σωστά η ποιότητα του λογισμικού, θα πρέπει να ελεγχθεί από διαφορετικές οπτικές γωνίες, λαμβάνοντας υπόψη τα χαρακτηριστικά ποιότητας. Ένα γενικό μοντέλο για την αξιολόγηση της ποιότητας παρέχει συνήθως μια συνολική εικόνα των ποιοτικών στοιχείων του λογισμικού. Στην πράξη όμως, τα χαρακτηριστικά ποιότητας εξαρτώνται από πολλούς παράγοντες, όπως για παράδειγμα από τον τύπο του λογισμικού ή το κοινό στο οποίο απευθύνεται.

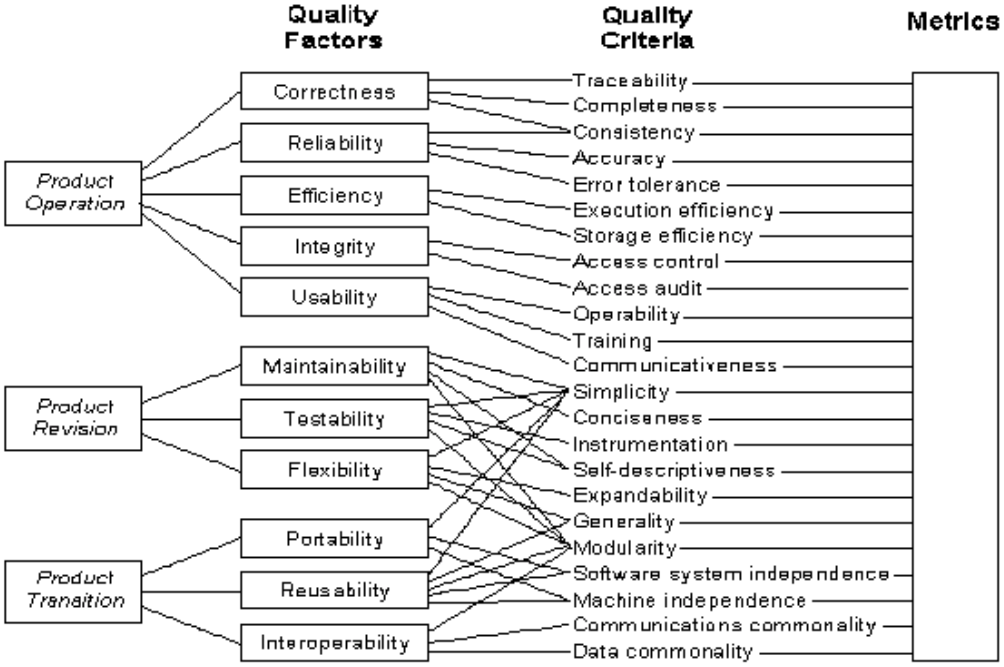
1.4.4 Μοντέλα διασφάλισης ποιότητας

Αρκετά μοντέλα ποιότητας έχουν καθοριστεί από διαφορετικά άτομα και οργανισμούς. Παρακάτω συνοψίζονται εν συντομία μερικά από τα πιο τυπικά και γνωστά μοντέλα ποιότητας.

1.4.4.1 Το μοντέλο McCall

Το μοντέλο McCall για την ποιότητα του λογισμικού [29] συνδυάζει ορισμένα κριτήρια σχετικά με τις λειτουργίες, τις αναθεωρήσεις και τις μεταβάσεις των προϊόντων. Αυτό το μοντέλο αναπτύχθηκε για την αξιολόγηση των σχέσεων μεταξύ των εξωτερικών παραγόντων και των κριτηρίων ποιότητας του προϊόντος. Το μοντέλο McCall χρησιμοποιείται στις Ηνωμένες Πολιτείες σε πολύ μεγάλα έργα στο στρατιωτικό, στο διαστημικό και στο δημόσιο τομέα. Τα ποιοτικά χαρακτηριστικά ταξινομούνται σε τρεις βασικούς τύπους: στους έντεκα παράγοντες που περιγράφουν την εξωτερική θεώρηση του λογισμικού (θεώρηση χρήστη), στα είκοσι τρία κριτήρια ποιότητας που περιγράφουν την εσωτερική θεώρηση του λογισμικού (θεώρηση προγραμματιστή) και σε ορισμένες μετρήσεις που ορίστηκαν και χρησιμοποιούνται ώστε να παρέχεται μια μέθοδος μέτρησης. Ο αριθμός των παραγόντων ορίστηκαν

σε μόλις έντεκα, ώστε η απλότητα να παραμείνει στο μοντέλο. Αυτοί λοιπόν οι παράγοντες είναι η ορθότητα, η αξιοπιστία, η αποτελεσματικότητα, η ακεραιότητα, η χρηστικότητα, η συντηρησιμότητα, η δυνατότητα εφαρμογής δοκιμών, η ευελιξία, η φορητότητα, η επαναχρησιμοποίηση και η διαλειτουργικότητα.

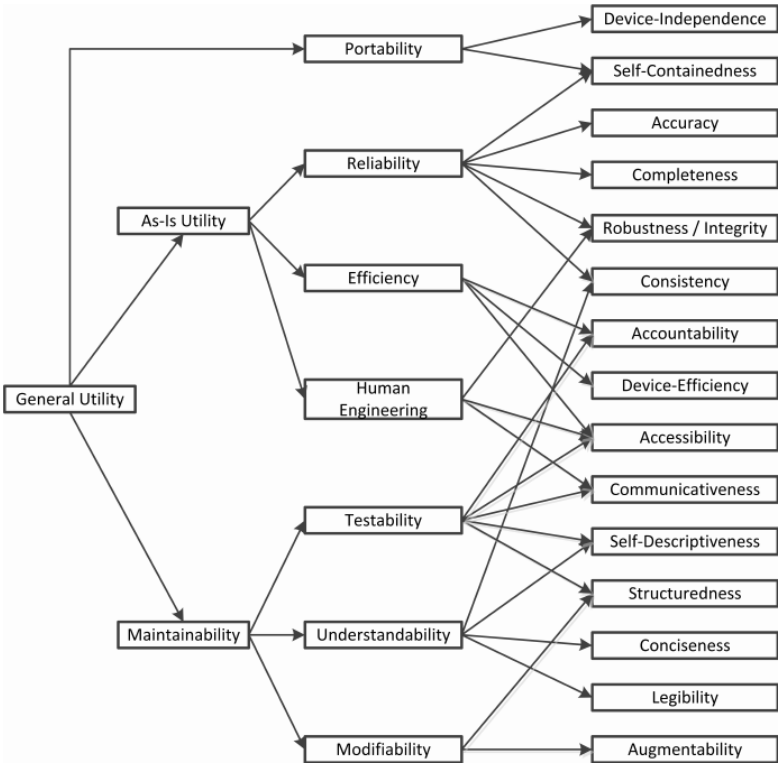


Σχήμα 7: Το μοντέλο McCall

Η κύρια συμβολή αυτού του μοντέλου είναι η σχέση μεταξύ των ποιοτικών χαρακτηριστικών και των μετρήσεων. Θα πρέπει να τονιστεί ωστόσο, ότι το μοντέλο δεν εξετάζει άμεσα τη λειτουργικότητα των προϊόντων λογισμικού.

1.4.4.2 Το μοντέλο Boehm

Στο μοντέλο του ο Boehm [30], όρισε ως πρωταρχικό χαρακτηριστικό της ποιότητας την «γενική χρησιμότητα». Ο στόχος αυτού του μοντέλου είναι να αντιμετωπίσει τις σύγχρονες αδυναμίες των μοντέλων που αξιολογούν αυτόματα και ποσοτικά την ποιότητα του λογισμικού. Το μοντέλο Boehm αντιπροσωπεύει μια ιεραρχική δομή χαρακτηριστικών, καθένα από τα οποία συμβάλλει στη συνολική ποιότητα. Τα χαρακτηριστικά υψηλού επιπέδου αντιπροσωπεύουν τις βασικές απαιτήσεις μιας πραγματικής χρήσης, στην οποία θα μπορούσε να γίνει αξιολόγηση της ποιότητας του λογισμικού. Περιλαμβάνει την τρέχουσα χρησιμότητα (as-is utility), τη συντηρησιμότητα και τη φορητότητα. Τα χαρακτηριστικά ενδιάμεσου επιπέδου περιέχουν επτά χαρακτηριστικά ποιότητας, τα οποία αντιπροσωπεύουν τις αναμενόμενες ιδιότητες από ένα σύστημα λογισμικού: Φορητότητα, αξιοπιστία, αποδοτικότητα, ευχρηστία, δυνατότητα εφαρμογής δοκιμών, εύκολη κατανόηση και ευελιξία (σχήμα 8). Στη χαμηλότερη δομή των επιπέδων της ιεραρχίας των χαρακτηριστικών, συναντάται η ιεραρχία μετρήσεων των πρωτόγονων χαρακτηριστικών. Σε αυτά τα πρωταρχικά χαρακτηριστικά περιλαμβάνονται μεταξύ άλλων η ακρίβεια, η συνέπεια, η ευρωστία, η προσβασιμότητα, η αποδοτικότητα, η ανεξαρτησία της συσκευής, η πληρότητα, η υπευθυνότητα, η επικοινωνία, η λεπτομερής κατανόηση, η αναγνωσιμότητα, η δομή, η δυνατότητα αύξησης, η συνοπτικότητα κ.α.

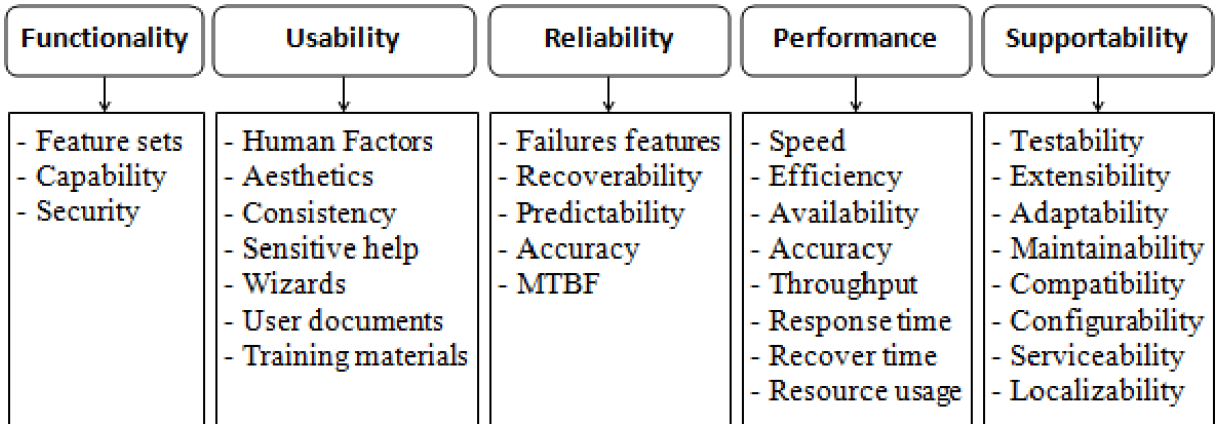


Σχήμα 8: Το μοντέλο Boehm

Το μοντέλο Boehm είναι παρόμοιο με το μοντέλο McCall, καθώς και τα δυο μοντέλα προτείνουν μια ιεραρχική δομή χαρακτηριστικών, καθένα από τα οποία συμβάλλει στη συνολική ποιότητα. Το μοντέλο του Boehm ωστόσο, αν και περιλαμβάνει τις ανάγκες των χρηστών όπως συμβαίνει και στο μοντέλο McCall, προσθέτει επιπλέον τα χαρακτηριστικά απόδοσης υλικού τα οποία δεν συναντώνται στο μοντέλο McCall. Από την άλλη μεριά όμως, το μοντέλο του Boehm αποτελεί απλά ένα διάγραμμα, χωρίς καμία πρόταση για τη μέτρηση των ποιοτικών χαρακτηριστικών.

1.4.4.3 Το μοντέλο FURPS

Το μοντέλο FURPS (Σχήμα 9) παρουσιάστηκε αρχικά από τον Robert Grady στη Hewlett Packard (αργότερα επεκτάθηκε από την Rational Software, η οποία πλέον ανήκει στην IBM και ονομάζεται IBM Rational Software, στο μοντέλο FURPS+). Αυτό το μοντέλο ταξινομεί τα χαρακτηριστικά σε δύο διαφορετικών ειδών απαιτήσεων, δηλαδή τις **Λειτουργικές Απαιτήσεις (F)**, οι οποίες ορίζονται από τις αναμενόμενες εισόδους και εξόδους, και τις **Μη Λειτουργικές Απαιτήσεις (URPS)**. Στο ακρωνύμιο URPS των μη λειτουργικών απαιτήσεων, το U επισημαίνει τη Χρησιμότητα (Usability) και περιλαμβάνει τους ανθρώπινους παράγοντες, την αισθητική, την τεκμηρίωση του χρήστη και το υλικό εκπαίδευσης. Το R σημαίνει Αξιοπιστία (Reliability) και περιλαμβάνει τη συχνότητα εμφάνισης των λαθών, τη σοβαρότητα των αστοχιών και τον χρόνο ανάκαμψης του λογισμικού από κάποια βλάβη. Το P συνδέεται με την Απόδοση (Performance) και περιλαμβάνει τις λειτουργικές απαιτήσεις, ενώ το S συνδέεται με την Υποστηρικτικότητα (Supportability) και περιλαμβάνει την υποστήριξη του λογισμικού και τον απαραίτητο σχεδιασμό και την υλοποίηση τόσο της εφαρμογής, όσο και της διεπαφής.

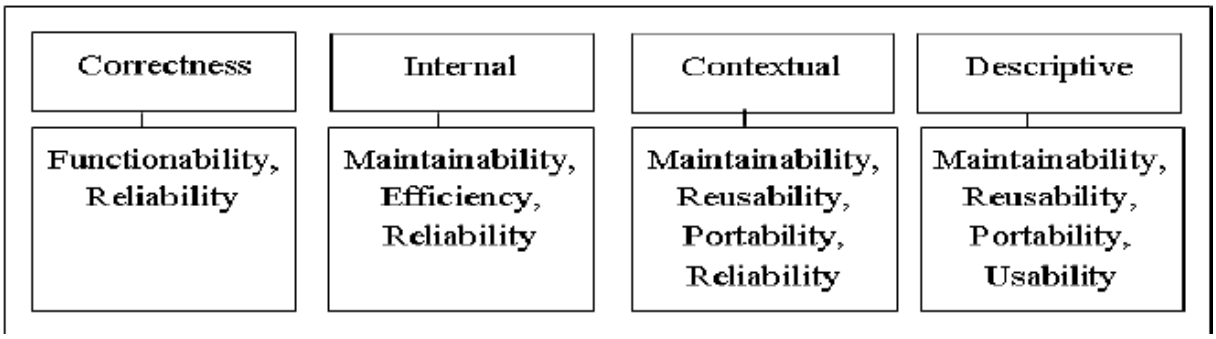


Σχήμα 9: Το μοντέλο FURPS

Όπως έχει ήδη αναφερθεί, το μοντέλο FURPS επεκτάθηκε από την IBM Rational Software στο μοντέλο FURPS +. Αυτό το μοντέλο λαμβάνει υπόψη μόνο τις απαιτήσεις χρήστη και αγνοεί τον παράγοντα του προγραμματιστή. Επιπλέον, δεν λαμβάνει υπόψη ορισμένα από τα χαρακτηριστικά του προϊόντος λογισμικού, όπως την φορητότητα και την συντηρησιμότητα.

1.4.4.4 Το μοντέλο Dromey

Το 1995, ο Dromey πρότεινε ένα πλαίσιο εργασίας για την αξιολόγηση των φάσεων του προσδιορισμού, του σχεδιασμού και της υλοποίησης των απαιτήσεων [31]. Το πλαίσιο αποτελείται ουσιαστικά από τρία μοντέλα: Το μοντέλο ποιότητας απαιτήσεων, το μοντέλο ποιότητας σχεδίασης και το μοντέλο ποιότητας υλοποίησης. Οι ιδιότητες προϊόντος υψηλού επιπέδου για το μοντέλο ποιότητας εφαρμογής περιλαμβάνουν: (i) Την ορθότητα, η οποία αξιολογεί εάν παραβιάζονται ορισμένες βασικές αρχές, με την λειτουργικότητα και την αξιοπιστία ως χαρακτηριστικά ποιότητας λογισμικού. (ii) Εσωτερικά μέτρα για το πόσο καλά έχει αναπτυχθεί ένα στοιχείο σύμφωνα με την προβλεπόμενη χρήση του, με την συντήρηση, την αποδοτικότητα και την αξιοπιστία ως τα χαρακτηριστικά ποιότητας λογισμικού (iii) Η συνάφεια αντιμετωπίζει τις εξωτερικές επιρροές στη χρήση ενός στοιχείου, με χαρακτηριστικά τη δυνατότητα συντήρησης, την επαναχρησιμοποίηση, την φορητότητα και την αξιοπιστία. (iv) Μετρήσεις της περιγραφικότητας ενός στοιχείου, με χαρακτηριστικά τη συντηρησιμότητα, την επαναχρησιμοποίηση, τη φορητότητα και τη χρηστικότητα.

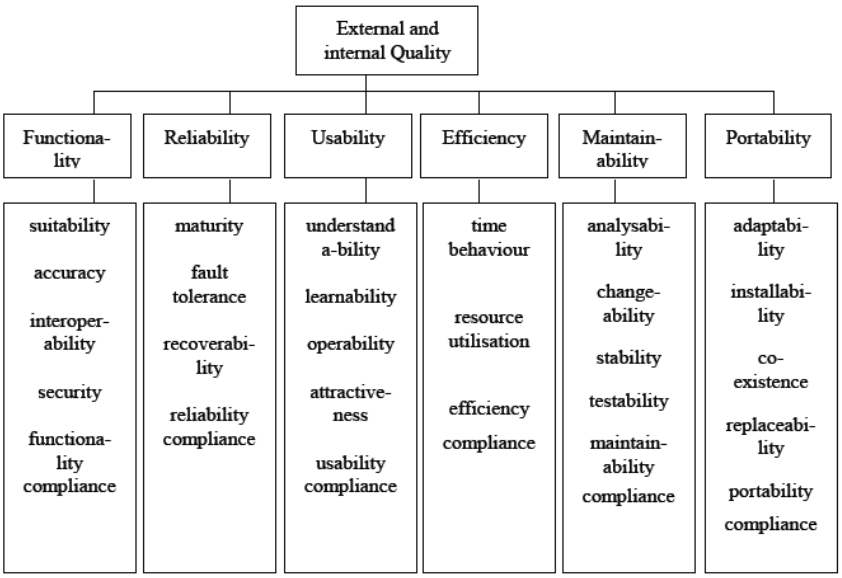


Σχήμα 10: Το μοντέλο Dromey

Το μοντέλο Dromey επιχειρεί να αυξήσει την κατανόηση της σχέσης μεταξύ των χαρακτηριστικών και των υπό-χαρακτηριστικών της ποιότητας. Επιπλέον, επιδιώκει να εντοπίσει τις ιδιότητες του λογισμικού που επηρεάζουν τα χαρακτηριστικά της ποιότητας.

1.4.4.5 Το πρότυπο ISO IEC 9126

Λόγω της εμφάνισης όλο και περισσότερο προτάσεων για νέα μοντέλα ποιότητας λογισμικού, επήλθε σύγχυση στο χώρο και προέκυψε έτσι η ανάγκη ενός νέου προτύπου. Επομένως, ο Παγκόσμιος Οργανισμός Τυποποίησης ISO / IEC JTC1 ξεκίνησε την ανάπτυξη ενός προτύπου για να επιτευχθεί η απαιτούμενη συναίνεση από όλους και να ενθαρρυνθεί η τυποποίηση παγκοσμίως. Το ISO 9126 αποτελεί μέρος του προτύπου ISO 9000, το οποίο είναι ίσως το πιο σημαντικό πρότυπο για τη διασφάλιση ποιότητας. Οι πρώτες σκέψεις ξεκίνησαν ήδη από το 1978, όμως η ανάπτυξη του ISO / IEC 9126 ξεκίνησε το 1985. Σε αυτό το μοντέλο, το σύνολο των χαρακτηριστικών ποιότητας ενός προϊόντος λογισμικού ταξινομείται σε μια ιεραρχική δομή δέντρου. Το υψηλότερο επίπεδο αυτής της δομής αποτελείται από τα ποιοτικά χαρακτηριστικά και το χαμηλότερο επίπεδο αποτελείται από τα κριτήρια ποιότητας του λογισμικού. Το μοντέλο καθορίζεται από τα εξής έξι χαρακτηριστικά: Λειτουργικότητα, αξιοπιστία, ευχρηστία, αποδοτικότητα, δυνατότητα συντήρησης και φορητότητα, τα οποία και χωρίζονται περαιτέρω σε είκοσι ένα δευτερεύοντα χαρακτηριστικά (Σχήμα 11). Αυτά τα δευτερεύοντα χαρακτηριστικά αποτελούν το αποτέλεσμα των εσωτερικών ιδιοτήτων του λογισμικού και εκδηλώνονται εξωτερικά όταν το λογισμικό χρησιμοποιείται ως μέρος ενός συστήματος.



Σχήμα 11: Το πρότυπο ISO IEC 9126

Τα καθορισμένα χαρακτηριστικά ισχύουν για κάθε είδος λογισμικού, συμπεριλαμβανομένων των προγραμμάτων υπολογιστών αλλά και των δεδομένων που περιέχονται προγραμματισμένα στο υλικό του συστήματος, και παρέχει συνεπή ορολογία για τον καθορισμό της ποιότητας των προϊόντων λογισμικού.

1.4.5 Μετρήσεις και μετρικές ποιότητας λογισμικού

Στη Μηχανική Λογισμικού, οι μετρήσεις πραγματοποιούνται με βάση ορισμένες μετρικές λογισμικού, οι οποίες και θεωρούνται ως η μέθοδος για τη μέτρηση των διαφόρων χαρακτηριστικών ενός λογισμικού. Όσον αφορά την ποιότητα, η διασφάλιση ποιότητας λογισμικού (SAQ) διασφαλίζει την ποιότητα και εφαρμόζεται αδιαλείπτως καθ' όλη τη διαδικασία της ανάπτυξης του λογισμικού. Όπως οι γενικές μετρήσεις, έτσι και η ποιότητα λογισμικού μετράται με βάση ορισμένες μετρικές ποιότητας λογισμικού. Ο όρος "μετρικές ποιότητας λογισμικού" απεικονίζει την εικόνα της μέτρησης των ιδιοτήτων του λογισμικού, καταγράφοντας τον αριθμό ελαττωμάτων ή των κενών ασφαλείας που πιθανόν να υπάρχουν στο λογισμικό. Ωστόσο, η μέτρηση της ποιότητας δεν περιορίζεται στην καταμέτρηση των ελαττωμάτων ή των τρωτών σημείων, αλλά καλύπτει και άλλες πτυχές των ιδιοτήτων, όπως τη συντήρηση, την αξιοπιστία, την ακεραιότητα, τη χρηστικότητα, την ικανοποίηση πελατών κ.λπ.

1.4.5.1 Ορισμοί μετρήσεων

Ο Stevens ορίζει την μέτρηση ως την «εκχώρηση αριθμών σε αντικείμενα ή συμβάντα σύμφωνα με κάποιον κανόνα. Ο κανόνας ανάθεσης μπορεί να είναι οποιοσδήποτε συνεπής κανόνας. Ο μόνος κανόνας που δεν επιτρέπεται είναι η τυχαία εκχώρηση, γιατί η τυχαιότητα προσομοιάζει σαν αποτέλεσμα στην μη ύπαρξη κανόνα» [32]. Ο Finkelstein υποστηρίζει ότι «η μέτρηση είναι η διαδικασία εμπειρικής, αντικειμενικής, ανάθεσης αριθμών σε ιδιότητες αντικειμένων ή γεγονότων του πραγματικού κόσμου με τέτοιο τρόπο ώστε να τα περιγράφει» [33]. Τέλος, οι Fenton και Pfleeger ενώνουν τους ορισμούς των προηγουμένων αναφέροντας ότι «μέτρηση είναι η διαδικασία με την οποία οι αριθμοί ή τα σύμβολα αποδίδονται σε χαρακτηριστικά οντοτήτων στον πραγματικό κόσμο με τέτοιο τρόπο ώστε να τα χαρακτηρίζουν σύμφωνα με σαφώς καθορισμένους κανόνες» [34].

1.4.5.2 Μετρικές ποιότητας λογισμικού

Σε γενικές γραμμές, οι μετρικές ποιότητας του λογισμικού είναι στην ουσία μια υποκατηγορία των μετρήσεων ποιότητας λογισμικού, η οποία δίνει έμφαση κυρίως στα ποιοτικά στοιχεία του προϊόντος και της διαδικασίας του έργου. Η μέτρηση του λογισμικού αποτελεί μια ευρύτερη έννοια που ενσωματώνει σε αυτήν τις μετρικές ποιότητας λογισμικού και αποτελείται κυρίως από τρεις τύπους μετρήσεων:

- **Μετρικές προϊόντων (Product Metrics):** Περιλαμβάνει το μέγεθος, τον σχεδιασμό, την πολυπλοκότητα, την απόδοση και άλλες παραμέτρους που σχετίζονται με την ποιότητα του προϊόντος.
- **Μετρικές διαδικασίας (Process Metrics):** Περιλαμβάνει παραμέτρους όπως η χρονική διάρκεια εντοπισμού και αφαίρεσης ελαττωμάτων, ο χρόνος απόκρισης για επίλυση προβλημάτων κ.λπ.
- **Μετρικές έργου (Project Metrics):** Μπορεί να περιλαμβάνει αριθμό των ομάδων εργασίας, τους εμπλεκόμενους προγραμματιστές, το κόστος και τη διάρκεια του έργου κ.λπ.

Ορισμένες μετρικές ανήκουν σε περισσότερες από μία κατηγορίες. Για παράδειγμα, οι μετρικές ποιότητας κατά τη διαδικασία ενός έργου ανήκουν και στις μετρικές διεργασίας, αλλά και στις μετρικές έργου. Οι μετρικές ποιότητας λογισμικού είναι ένα υποσύνολο μετρικών λογισμικού που εστιάζουν στις πτυχές ποιότητας του προϊόντος, της διαδικασίας και του έργου. Αυτές συνδέονται στενότερα με τις μετρικές διαδικασίας και προϊόντων παρά με τις μετρικές έργων.

Σε κάθε περίπτωση, υπάρχει ένας μεγάλος αριθμός διαθέσιμων μετρικών βάσει των οποίων μετράται η ποιότητα του λογισμικού. Αλλά μεταξύ αυτών υπάρχουν κάποιες μετρικές που έχουν αποδειχτεί σημαντικότερες στη μέτρηση της ποιότητας του λογισμικού. Αυτές είναι:

- **Ποιότητα του κώδικα:** Οι μετρικές ποιότητας του κώδικα μετρούν την ποιότητα του κώδικα που χρησιμοποιείται για την ανάπτυξη έργου λογισμικού. Η διατήρηση της ποιότητας του κώδικα σε καλά επίπεδα, χωρίς δηλαδή να περιέχονται σε αυτόν σφάλματα και να είναι σημασιολογικά σωστός, αποτελεί μια πολύ σημαντική παράμετρο για την καλή ανάπτυξη έργου λογισμικού. Επίσης, στην ποιότητα του κώδικα μετρώνται τόσο οι ποσοτικές μετρήσεις (όπως για παράδειγμα ο αριθμός των γραμμών, η πολυπλοκότητα, οι συναρτήσεις, ο ρυθμός εμφάνισης σφαλμάτων κ.λπ.), όσο και οι ποιοτικές μετρήσεις (όπως η αναγνωσιμότητα, η σαφήνεια κώδικα, η αποδοτικότητα, η συντήρηση κ.λπ.).
- **Αξιοπιστία:** Οι μετρικές αξιοπιστίας μετρούν την αντίδραση του λογισμικού σε διαφορετικές συνθήκες. Αν δηλαδή το λογισμικό είναι σε θέση να παρέχει ακριβή εξυπηρέτηση τη σωστή στιγμή ή όχι. Η αξιοπιστία μπορεί να ελεγχθεί χρησιμοποιώντας τον μέσο χρόνο μεταξύ αποτυχίας (MTBF) και τον μέσο χρόνο επισκευής (MTTR).
- **Απόδοση:** Οι μετρικές απόδοσης χρησιμοποιούνται για τη μέτρηση της απόδοσης ενός λογισμικού. Κάθε λογισμικό έχει αναπτυχθεί για συγκεκριμένους σκοπούς. Έτσι οι μετρικές απόδοσης, αφού μετρήσουν την απόδοση του λογισμικού, καθορίζουν εάν το λογισμικό πληροί τις απαιτήσεις του χρήστη ή όχι, αναλύοντας τον χρόνο και τους πόρους που χρησιμοποιεί για την παροχή της υπηρεσίας.
- **Ευχρηστία:** Οι μετρικές ευχρηστίας ελέγχουν εάν το πρόγραμμα είναι φιλικό προς το χρήστη ή όχι. Έτσι λοιπόν, είναι σημαντικό σε κάθε λογισμικό να μπορεί να διαπιστωθεί εάν ο τελικός χρήστης είναι ευχαριστημένος ή όχι χρησιμοποιώντας αυτό το λογισμικό.
- **Ορθότητα:** Η ορθότητα είναι μια από τις σημαντικές μετρικές ποιότητας λογισμικού, καθώς ελέγχει εάν το σύστημα ή το λογισμικό λειτουργούν απρόσκοπτα, ικανοποιώντας έτσι τις απαιτήσεις του χρήστη. Η ορθότητα αποδίδει το βαθμό εξυπηρέτησης που παρέχει μία λειτουργία, ώστε να ικανοποιηθεί ο σκοπός για τον οποίο δημιουργήθηκε.
- **Συντηρησιμότητα:** Κάθε προϊόν λογισμικού απαιτεί συντήρηση και αναβάθμιση. Η συντήρηση όμως είναι μια δαπανηρή και χρονοβόρα διαδικασία και ως εκ τούτου, θα πρέπει στο προϊόν λογισμικού να παρέχεται εύκολη συντήρηση. Οι μετρικές συντήρησης περιλαμβάνουν τον χρόνο που απαιτείται για την προσαρμογή σε νέες λειτουργίες, τη Μέση Ώρα Για Αλλαγή (MTTC), την απόδοση σε τροποποιημένα περιβάλλοντα κ.λπ.
- **Ακεραιότητα:** Η ακεραιότητα του λογισμικού είναι σημαντική καθώς αφορά το πόσο εύκολη είναι η ενσωμάτωσή του σε άλλα λογισμικά, αυξάνοντας έτσι τις δυνατότητες και τη συνολική χρηστικότητα του λογισμικού. Επίσης αποτρέπεται η ενσωμάτωση του λογισμικού σε μη εξουσιοδοτημένο λογισμικό, μειώνοντας έτσι τις πιθανότητες επιθέσεων στον κυβερνοχώρο.
- **Ασφάλεια:** Οι μετρικές ασφαλείας μετράνε το κατά πόσο ασφαλές είναι το λογισμικό. Στην εποχή της τρομοκρατίας στον κυβερνοχώρο, η ασφάλεια αποτελεί το πιο ουσιαστικό κομμάτι κάθε λογισμικού. Η ασφάλεια διασφαλίζει ότι δεν θα υπάρξουν μη εξουσιοδοτημένες αλλαγές και κανένας φόβος για επιθέσεις στον κυβερνοχώρο, όταν το προϊόν λογισμικού θα χρησιμοποιηθεί από τον τελικό χρήστη.

1.4.5.3 Χαρακτηριστικά των αποτελεσματικών μετρικών ποιότητας λογισμικού

Σε γενικές γραμμές, για να είναι αποτελεσματικές οι μετρικές ποιότητας θα πρέπει αρχικά να αφορούν μία συγκεκριμένη μέτρηση ενός συγκεκριμένου χαρακτηριστικού ή ενός χαρακτηριστικού μεγαλύτερης σημασίας. Μία εξειδικευμένη μετρική είναι κατά πάσα πιθανότητα αποτελεσματικότερη σε συγκεκριμένη μέτρηση από μία άλλη. Από την άλλη όμως, είναι καλό για μια μετρική ποιότητας να είναι περιεκτική, ώστε να είναι σε θέση να χρησιμοποιηθεί σε μεγάλη ποικιλία σεναρίων. Θα πρέπει επίσης να σημειωθεί ότι, κατά κανόνα, δεν θα πρέπει να λαμβάνονται υπόψη χαρακτηριστικά τα οποία έχουν ήδη μετρηθεί από κάποια άλλη μέτρηση. Μια μετρική ποιότητας θα πρέπει να είναι αξιόπιστη να λειτουργεί παρόμοια σε όλες τις συνθήκες και, τέλος, να είναι εύκολη στην κατανόηση και στη χρήση.

1.4.5.4 Η σημασία των μετρικών ποιότητας λογισμικού

Με τη χρήση των μετρικών ποιότητας λογισμικού επιτυγχάνεται η υψηλότερη ποιότητα του προϊόντος, αντλούνται μετρήσεις για την αξιολόγηση, την τροποποίηση και τη βελτίωση της διαδικασίας ανάπτυξης σε βάθος χρόνου. Σαν αποτέλεσμα ακολουθεί μείωση των εξόδων, μείωση του κόστους και του χρόνου ανάπτυξης και η προώθηση ευέλικτων (Agile) περιβαλλόντων ανάπτυξης λογισμικού. Μέσω των μετρικών ποιότητας μπορούν να οριστούν και να κατηγοριοποιηθούν όλα τα στοιχεία, ούτως ώστε να καταστούν περισσότερο κατανοητές οι διαδικασίες. Επίσης, η χρήση των μετρικών επιτρέπει την αξιολόγηση καθεμίας από αυτές τις διαδικασίες και να προσδίδει τις δεδομένες απαιτήσεις και προδιαγραφές, να προβλέπονται και να προγραμματίζονται μελλοντικές κινήσεις σχετικά με τις απαιτήσεις λογισμικού των επιχειρήσεων και την επίτευξη της συνολικής ποιότητας της διαδικασίας, του προϊόντος και εντέλει, ολόκληρου του έργου.

1.5 Ευέλικτες Μεθοδολογίες (Agile)

Οι περισσότερες μεθοδολογίες ανάπτυξης λογισμικού μπορούν να χωριστούν σε δύο κατηγορίες: **Παραδοσιακές** και **Ευέλικτες Μεθοδολογίες**. Οι παραδοσιακές μεθοδολογίες οδηγούνται από εκτενή σχεδιασμό και λεπτομερή τεκμηρίωση. Ένα παράδειγμα παραδοσιακής μεθοδολογίας είναι το μοντέλο καταρράκτη (waterfall), όπου κάθε φάση ακολουθεί την προηγούμενη. Για την ομάδα ανάπτυξης του λογισμικού , αυτό έχει σαν αποτέλεσμα να αντιμετωπίζει σημαντικές δυσκολίες σε οποιαδήποτε τροποποίηση ζητηθεί εκ των υστέρων. Σε αντίθεση, οι ευέλικτες μεθοδολογίες αποτελούνται από μικρούς επαναληπτικούς και επαυξητικούς κύκλους ανάπτυξης, με συχνή παράδοση λειτουργήσιμου λογισμικού και καθορίζεται από τη στενή συνεργασία της ομάδας ανάπτυξης με τους πελάτες.

1.5.1 Μανιφέστο ευέλικτων μεθοδολογιών (Agile Manifest)

Τη δεκαετία του 1990, η βιομηχανία της ανάπτυξης λογισμικού συνειδητοποίησε ότι αδυνατούσε να κινηθεί αρκετά γρήγορα ώστε να ικανοποιήσει τις απαιτήσεις των εφαρμογών και τις απαιτήσεις των πελατών. Το κυριότερο πρόβλημα προερχόταν από την εφαρμογή των παραδοσιακών μοντέλων ανάπτυξης, τα οποία ακολουθούσαν χρονολογική προσέγγιση και η ανάπτυξη συνέβαινε διαδοχικά, ενώ το τελικό προϊόν δεν μπορούσε να αποκαλυφθεί στους πελάτες μέχρι το τελικό βήμα. Αυτό άφηνε λίγα περιθώρια για ευελιξία όσον αφορά τις προόδους αξιολογήσεων και τις αλλαγές. Έτσι, μέχρι την ολοκλήρωση της εφαρμογής, ήταν πολύ πιθανό ότι οι απαιτήσεις και τα συστήματα των αρχικών στόχων του έργου να έχουν αλλάξει. Αυτό επέφερε μεγάλη σπατάλη προσπάθειας και χρημάτων, ενώ

πολλά έργα ακυρώθηκαν χωρίς να ολοκληρωθούν. Έτσι, οι επαγγελματίες της κοινότητας λογισμικού θεώρησαν ότι ήταν ώρα για μια νέα, ανανεωμένη προσέγγιση.

Συγκεκριμένα, τον Φεβρουάριο του 2001 στη Γιούτα των Ηνωμένων Πολιτειών, δεκαεπτά επιστήμονες του χώρου διατύπωσαν τις βασικές αρχές της ευέλικτης μεθοδολογίας που, ως σύνολο, αποτελούν το λεγόμενο «Agile Manifesto». Πρακτικά, πρόκειται για την δήλωση των αξιών και των αρχών που εκφράζουν την ευέλικτη μεθοδολογία. Αποτελείται από τέσσερις θεμελιώδεις αξίες και δώδεκα βασικές αρχές, και στοχεύει να βοηθήσει στην δημιουργία καλύτερων τρόπων ανάπτυξης λογισμικού παρέχοντας μια σαφή και μετρήσιμη δομή που προωθεί την επαναληπτική ανάπτυξη, τη συνεργασία της ομάδας και την αναγνώριση αλλαγών. Έτσι, ο όρος «Ευέλικτες Μεθοδολογίες» αναφέρεται σε οποιαδήποτε μεθοδολογία που ευθυγραμμίζεται με τις έννοιες που καθορίζει το ως άνω αναφερόμενο μανιφέστο.

1.5.1.1 Οι τέσσερις αξίες του ευέλικτου μανιφέστου

Όπως έχει ήδη αναφερθεί, το ευέλικτο μανιφέστο αποτελείται από τέσσερις θεμελιώδεις αξίες και δώδεκα υποστηρικτικές αρχές. Μπορεί η κάθε ευέλικτη μεθοδολογία να εφαρμόζει αυτές τις τέσσερις αξίες με διαφορετικούς τρόπους, όλες τους όμως βασίζονται σε αυτές τις αξίες για να καθοδηγήσουν την ανάπτυξη και την παράδοση υψηλής ποιότητας λογισμικού εργασίας.

Οι τέσσερις θεμελιώδεις αξίες του ευέλικτου μανιφέστου είναι οι εξής:

- **Τα άτομα και οι αλληλεπιδράσεις βρίσκονται πάνω από τις διαδικασίες και τα εργαλεία:** Κατά το παρελθόν, πολλές ομάδες λογισμικού είχαν την τάση να εστιάζουν στην εξασφάλιση των καλύτερων εργαλείων και διαδικασιών για την κατασκευή του λογισμικού τους. Το ευέλικτο μανιφέστο αξιολογεί ότι, αν και τα εργαλεία και οι διαδικασίες είναι σημαντικά, το ανθρώπινο δυναμικό πίσω από τις διαδικασίες είναι σημαντικότερο. Αυτό βέβαια δεν σημαίνει ότι οι ευέλικτες μεθοδολογίες αποθαρρύνουν τις τυποποιημένες διαδικασίες ή τα εξελιγμένα εργαλεία. Ωστόσο, αυτά είναι άχρηστα εάν η ομάδα ανάπτυξης δεν μπορεί να συνεργαστεί σωστά. Ακόμη και σε περίπτωση που δεν θα είναι δυνατό να διατεθούν σε μια ικανή ομάδα τα απαραίτητα εργαλεία και διαδικασίες, είναι πιθανό ότι με την ενθάρρυνση και τα κατάλληλα κίνητρα θα βρεθεί κάποιος τρόπος να ολοκληρωθεί το έργο.
- **Το λογισμικό που λειτουργεί αξιολογείται πάνω από την εκτενή τεκμηρίωση:** Ιστορικά, οι εταιρείες ανάπτυξης λογισμικού έχουν αφιερώσει πάρα πολύ χρόνο για την τεκμηρίωση του προϊόντος κατά την τελική παράδοση. Η τεκμηρίωση περιλαμβάνει τεχνικές προδιαγραφές, τεχνικές απαιτήσεις, τεχνικό ενημερωτικό δελτίο, έγγραφα σχεδιασμού διεπαφής, σχέδια δοκιμών, σχέδια τεκμηρίωσης και οι εγκρίσεις που απαιτούνται για καθένα από αυτά. Ο κατάλογος ήταν εκτενής και αποτελούσε συχνά την αιτία των μεγάλων καθυστερήσεων κατά την ανάπτυξη. Οι ευέλικτες μεθοδολογίες δεν στοχεύουν στην εξάλειψη της τεκμηρίωσης, αλλά της βελτιστοποίησής της σε μια μορφή που δίνει στον προγραμματιστή αυτό που χρειάζεται για να κάνει τη δουλειά του, χωρίς όμως να τον εγκλωβίσουν σε λεπτομέρειες. Οι ευέλικτες απαιτήσεις εγγράφων ως ιστορίες χρηστών, επαρκούν για να ξεκινήσει ο προγραμματιστής λογισμικού το έργο δημιουργίας μιας νέας λειτουργίας. Το ευέλικτο μανιφέστο εκτιμά την τεκμηρίωση, αλλά εκτιμά περισσότερο το λογισμικό που λειτουργεί.
- **Η συνεργασία με τον πελάτη αξιολογείται πάνω από τις συμβατικές διαπραγματεύσεις:** Η περίοδος της διαπραγμάτευσης είναι η περίοδος κατά την οποία ο πελάτης και ο εκπρόσωπος της εταιρείας ανάπτυξης λογισμικού επεξεργάζονται τις λεπτομέρειες της

ανάπτυξης του προϊόντος, θέτοντας παράλληλα τα σημεία στην πορεία όπου οι λεπτομέρειες θα μπορούν να επαναδιαπραγματευθούν. Ωστόσο, η συνεργασία με τον πελάτη αποτελεί μια εντελώς διαφορετική έννοια. Με τα μοντέλα ανάπτυξης, όπως για παράδειγμα το μοντέλο καταρράκτης, οι πελάτες διαπραγματεύονται τις απαιτήσεις για το προϊόν, συχνά με μεγάλη λεπτομέρεια, πριν από την έναρξη της εργασίας. Αυτό πρακτικά σημαίνει ότι ο πελάτης συμμετέχει στη διαδικασία ανάπτυξης πριν αυτή ξεκινήσει και μετά από την ολοκλήρωσή της, αλλά όχι κατά τη διάρκεια της διαδικασίας. Αντίθετα, το ευέλικτο μανιφέστο περιγράφει έναν πελάτη ο οποίος συμμετέχει και συνεργάζεται καθ 'όλη τη διαδικασία ανάπτυξης, καλύπτοντας έτσι έγκαιρα και με μεγαλύτερη ευκολία τις ανάγκες του. Αυτό επιτυγχάνεται με συναντήσεις με τον πελάτη ανά τακτά διαστήματα για περιοδικές επιδείξεις εκδόσεων του προϊόντος. Επίσης το ευέλικτο μανιφέστο ενθαρρύνει την παρουσία ενός από τους τελικούς χρήστες ως καθημερινό μέλος της ομάδας ώστε να παρακολουθεί όλες τις συναντήσεις, διασφαλίζοντας έτσι ότι το προϊόν ανταποκρίνεται στις επιχειρηματικές ανάγκες του πελάτη.

- **Η ανταπόκριση στην αλλαγή αξιολογείται πάνω από την τήρηση ενός προδιαγεγραμμένου σχεδίου:** Ένα σημαντικό όφελος της ευέλικτης μεθοδολογίας είναι ότι ενθαρρύνει τη συχνή αναθεώρηση και επανεξέταση των τρεχόντων σχεδίων βάσει των νέων πληροφοριών που η ομάδα συλλέγει και αναλύει συνεχώς. Ο χάρτης της πορείας του προϊόντος λοιπόν, δεν αποτελεί πλέον ένα στατικό έγγραφο, αλλά μια δυναμική στρατηγική. Οι εταιρείες ανάπτυξης λογισμικού, με την υιοθέτηση ευέλικτων μεθοδολογιών, θα είναι σε θέση να παρουσιάσουν στους πελάτες δυναμικούς χάρτες πορείας του προϊόντος, με διαφανή τρόπο που θα αντικατοπτρίζει και την πιθανότητα αλλαγής με βάση τις νέες γνώσεις. Δηλαδή, η ευέλικτη μεθοδολογία επιτρέπει σε μια ομάδα ανάπτυξης να προσαρμόζει τις προτεραιότητες και τα σχέδιά της όποτε αυτό έχει νόημα. Αυτές οι ομάδες δεν παραμένουν προσκολλημένες σε ένα ξεπερασμένο σχέδιο, μόνο και μόνο επειδή έχουν δεσμευτεί να το ακολουθήσουν.

Τέλος, επισημαίνεται ότι οι παραπάνω αξίες αντικατοπτρίζουν μια προσέγγιση και όχι ένα σύνολο αυστηρών κανόνων που θα πρέπει να ακολουθηθούν απαρέγκλιτα . Αυτές οι αξίες είναι ανοιχτές προς ερμηνεία και δεν είναι σταθερές οδηγίες.

1.5.1.2 Οι δώδεκα αρχές που διέπουν το ευέλικτο μανιφέστο

Οι δώδεκα αρχές που διέπουν το ευέλικτο μανιφέστο αποτελούν τις κατευθυντήριες αρχές για τις μεθοδολογίες που θεωρούνται ευέλικτες. Περιγράφουν μια προσέγγιση στην οποία η αλλαγή είναι ευπρόσδεκτη και ο πελάτης βρίσκεται στο επίκεντρο της εργασίας. Αποδεικνύουν επίσης την πρόθεση του κινήματος να ευθυγραμμίσει την ανάπτυξη με τις επιχειρηματικές ανάγκες. Οι δώδεκα αρχές που διέπουν το ευέλικτο μανιφέστο είναι οι εξής:

- Πρώτη μας προτεραιότητα είναι η ικανοποίηση του πελάτη μέσω της έγκαιρης και συνεχούς παράδοσης χρήσιμου λογισμικού.
- Οι αλλαγές στις απαιτήσεις είναι ευπρόσδεκτες, ακόμα και σε προχωρημένα στάδια της ανάπτυξης. Οι ευέλικτες διαδικασίες δαμάζουν τις αλλαγές με στόχο την ενίσχυση του ανταγωνιστικού πλεονεκτήματος του πελάτη.
- Παραδίδουμε συχνά λογισμικό που λειτουργεί, σε διαστήματα μερικών εβδομάδων ή μηνών, με προτίμηση στη συντομότερη χρονική κλίμακα.

- Οι προγραμματιστές και οι ειδικοί της αγοράς πρέπει να συνεργάζονται καθημερινά καθ' όλη τη διάρκεια του έργου.
- Θεμελιώνουμε τα έργα γύρω από άτομα με πάθος και ενδιαφέρον. Διαμορφώνουμε το κατάλληλο περιβάλλον, τους παρέχουμε την αναγκαία υποστήριξη, και εμπιστευόμαστε την ικανότητά τους να φέρουν σε πέρας την αποστολή τους.
- Η πιο αποδοτική και αποτελεσματική μέθοδος για τη μετάδοση πληροφορίας προς και εντός της ομάδας ανάπτυξης λογισμικού είναι η συνομιλία πρόσωπο με πρόσωπο.
- Το λογισμικό που λειτουργεί είναι το κύριο μέτρο προόδου.
- Οι ευέλικτες διαδικασίες προάγουν την αειφόρο ανάπτυξη. Οι χορηγοί, η ομάδα ανάπτυξης λογισμικού και οι χρήστες θα πρέπει να είναι σε θέση να διατηρούν ένα σταθερό ρυθμό επ' αόριστον.
- Η διαρκής έμφαση στην τεχνική αρτιότητα και στην εύρυθμη σχεδίαση ενισχύουν την ευελιξία.
- Η απλότητα -- η τέχνη της μεγιστοποίησης του όγκου της δουλειάς που δεν χρειάζεται να γίνει -- είναι ουσιώδης.
- Οι καλύτερες αρχιτεκτονικές, απαιτήσεις και σχέδια προκύπτουν από ομάδες πού οργανώνονται μόνες τους.
- Σε τακτά χρονικά διαστήματα, η ομάδα συλλογίζεται για το πώς θα γίνει πιο αποτελεσματική, ρυθμίζοντας και προσαρμόζοντας τη συμπεριφορά της αναλόγως.

Τα έργα τα οποία αναπτύσσονται με ευέλικτη μεθοδολογία είναι πελατοκεντρικά και ενθαρρύνουν την καθοδήγηση και τη συμμετοχή των πελατών. Ως αποτέλεσμα, η ευελιξία έχει εξελιχθεί σε μια καθολική προσέγγιση για ολόκληρη τη βιομηχανία ανάπτυξης λογισμικού- καθώς και μια βιομηχανία από μόνη της.

1.5.1.3 Φάσεις του κύκλου ανάπτυξης της ευέλικτης μεθοδολογίας

Οι φάσεις του κύκλου ανάπτυξης σε μια ευέλικτη μεθοδολογία δεν λαμβάνουν χώρα απαραίτητα διαδοχικά. Είναι και αυτές ευέλικτες και πάντα εξελισσόμενες, με πολλές από αυτές να συμβαίνουν παράλληλα:

- **Σχεδιασμός:** Μόλις μια ιδέα κριθεί ως βιώσιμη, η ομάδα του έργου συγκεντρώνεται για να εντοπίσει τα βασικά χαρακτηριστικά, να δώσει βαθμό προτεραιότητας σε κάθε χαρακτηριστικό και να τα εισάγει στην επαναληπτική προσέγγιση.
- **Ανάλυση απαιτήσεων:** Ο πελάτης και η ομάδα ανάπτυξης συναντώνται για να προσδιορίσουν τις επιχειρηματικές απαιτήσεις του έργου, οι οποίες θα πρέπει είναι μετρήσιμες, σχετικές και λεπτομερείς.
- **Σχεδιασμός:** Ο σχεδιασμός πραγματοποιείται βάσει των απαιτήσεων που προσδιορίστηκαν από τον πελάτη και η ομάδα περνάει στο στάδιο της αναζήτησης μεθόδων, ώστε να αποφασιστεί η τελική μορφή του προϊόντος, η στρατηγική της ανάπτυξης του έργου και ο καθορισμός των επαναληπτικών δοκιμών που θα ακολουθηθούν.
- **Εφαρμογή, συγγραφή κώδικα και ανάπτυξη:** Σε αυτό το στάδιο πραγματοποιείται η συγγραφή του κώδικα, η ανάπτυξη των λειτουργιών του έργου και ο προγραμματισμός των επαναληπτικών κύκλων της ανάπτυξης.
- **Δοκιμή:** Έλεγχος του κώδικα με βάση τις απαιτήσεις, ούτως ώστε να υπάρχει η βεβαιότητα ότι το προϊόν ικανοποιεί πραγματικά τις ανάγκες του πελάτη. Αυτή η φάση περιλαμβάνει δοκιμές μονάδων, δοκιμές ολοκλήρωσης, δοκιμές συστήματος και δοκιμές αποδοχής.

- **Παράδοση:** Παράδοση του προϊόντος στον πελάτη. Μόλις οι τελικοί χρήστες αρχίσουν να χρησιμοποιούν το προϊόν, ενδέχεται να αντιμετωπίσουν νέα προβλήματα που θα πρέπει να αντιμετωπίσει η ομάδα του έργου σε μελλοντικές επανεξετάσεις.

1.5.1.4 Πλεονεκτήματα των ευέλικτων μεθοδολογιών

Οι ευέλικτες μεθοδολογίες εξελίχθηκαν ως διαφορετικές προσεγγίσεις ανάπτυξης λογισμικού κατά τη δεκαετία του 1990 και αποτελούν την απάντηση στην άκαμπτη, γραμμική μεθοδολογία των σειριακών μεθοδολογιών που επικρατούσαν μέχρι τότε. Επικεντρώνονται στην ευελιξία, τη συνεχή βελτίωση και την ταχύτητα. Παρακάτω ακολουθούν τα κυριότερα πλεονεκτήματα των ευέλικτων μεθοδολογιών:

- **Οι αλλαγές ενθαρρύνονται:** Με την εφαρμογή βραχύτερων κύκλων σχεδιασμού, υπάρχει πάντα η ευκαιρία για βελτίωση και επαναπροσδιορισμό των προτεραιοτήτων του έργου, δίδοντας έτσι την δυνατότητα ομαλότερης προσαρμογής των αλλαγών καθ' όλη τη διάρκεια της ανάπτυξης.
- **Ο τελικός στόχος μπορεί να είναι άγνωστος:** Οι ευέλικτες μεθοδολογίες είναι ιδιαίτερα επωφελείς για έργα των οποίων ο τελικός στόχος δεν είναι σαφώς καθορισμένος. Καθώς το έργο προχωρά, οι στόχοι αναδεικνύονται και ανάλογα προσαρμόζεται και η ομάδα ανάπτυξης.
- **Ταχύτερη, υψηλής ποιότητας παράδοση:** Η κατανομή του έργου σε επαναλήψεις επιτρέπει στην ομάδα να επικεντρωθεί στην ανάπτυξη, τη δοκιμή και τη συνεργασία υψηλής ποιότητας. Η διεξαγωγή δοκιμών σε κάθε επανάληψη σημαίνει ότι τα σφάλματα εντοπίζονται και επιλύονται πιο γρήγορα.
- **Ισχυρή ομαδική αλληλεπίδραση:** Οι ευέλικτες μεθοδολογίες ενθαρρύνουν την συχνή επικοινωνία και την αλληλεπίδραση πρόσωπο-με-πρόσωπο όλων των εμπλεκομένων στο έργο.
- **Συμμετοχή των πελατών στην ανάπτυξη:** Οι πελάτες έχουν πολλές ευκαιρίες να ελέγξουν το προϊόν κατά τη φάση της ανάπτυξης , καθώς τους παραδίδονται διαδοχικές εκδόσεις του έργου. Επομένως, μοιράζονται έγκαιρα τις απόψεις και τις προτάσεις τους με την ομάδα ανάπτυξης, συνεισφέροντας έτσι στο τελικό προϊόν.
- **Συνεχής βελτίωση:** Ενθαρρύνονται η ανάδραση και τα σχόλια των τελικών χρηστών από τα μέλη της ομάδας ανάπτυξης καθ 'όλη τη διάρκεια του έργου, ούτως ώστε τα συμπεράσματα που αντλήθηκαν να εισαχθούν σε μελλοντικές εκδόσεις.

1.5.1.5 Μειονεκτήματα των ευέλικτων μεθοδολογιών

Ενώ το πρόσημο στις agile μεθοδολογίες είναι κατά κανόνα θετικό, αυτό δεν σημαίνει ότι είναι απίθανο να προκύψουν και κάποια προβλήματα. Για παράδειγμα, μπορεί να είναι δύσκολος ο καθορισμός μιας σταθερής ημερομηνίας παράδοσης, ή μπορεί να παραμεληθεί η τεκμηρίωση, ενώ το τελικό προϊόν μπορεί να είναι πολύ διαφορετικό από το αρχικά προβλεπόμενο. Παρακάτω παρατίθενται κάποια από τα μειονεκτήματα που μπορεί να προκύψουν από την χρήση κάποιας ευέλικτης μεθοδολογίας:

- **Ο προγραμματισμός μπορεί να είναι λιγότερο διακριτός:** Επειδή οι διαχειριστές των ομάδων έργου συχνά αναδιανέμουν καθήκοντα μεταξύ των μελών της ομάδας, είναι πιθανό ορισμένες λειτουργίες που έχουν προγραμματιστεί για παράδοση να μην ολοκληρωθούν εγκαίρως. Επίσης, η υπερβολική χρήση των συναντήσεων σπριντ μπορεί εν τέλει να επιβαρύνει το συνολικό χρονοδιάγραμμα του έργου, αντί να προσφέρει κάτι ουσιώδες.

- **Τα μέλη της ομάδας θα πρέπει να είναι έμπειρα:** Οι ομάδες στις ευέλικτες μεθοδολογίες αποτελούνται συνήθως από λίγα μέλη, οπότε αυτά τα μέλη που συμμετέχουν θα πρέπει να έχουν υψηλή εξειδίκευση σε διάφορους τομείς και να κατανοούν τη τον τρόπο που λειτουργεί η ευέλικτη μεθοδολογία.
- **Οι προγραμματιστές θα πρέπει να δεσμεύσουν τον χρόνο τους:** Απαιτείται ενεργή συμμετοχή και συνεργασία καθ 'όλη τη διαδικασία μιας ευέλικτης μεθοδολογίας, η οποία είναι πιο χρονοβόρα από μια παραδοσιακή προσέγγιση.
- **Η τεκμηρίωση μπορεί να αγνοηθεί:** Οι ευέλικτες μεθοδολογίες συνιστούν τις συχνές παραδόσεις εκδόσεων του έργου, σε σχέση με την ολοκληρωμένη τεκμηρίωση. Ενώ η τεκμηρίωση από μόνη της δεν εξασφαλίζει την επιτυχία, οι ομάδες θα πρέπει να βρουν τη σωστή ισορροπία μεταξύ του ποσοστού τεκμηρίωσης και της συζήτησης.

1.5.1.6 Ευρέως χρησιμοποιούμενες Agile μεθοδολογίες

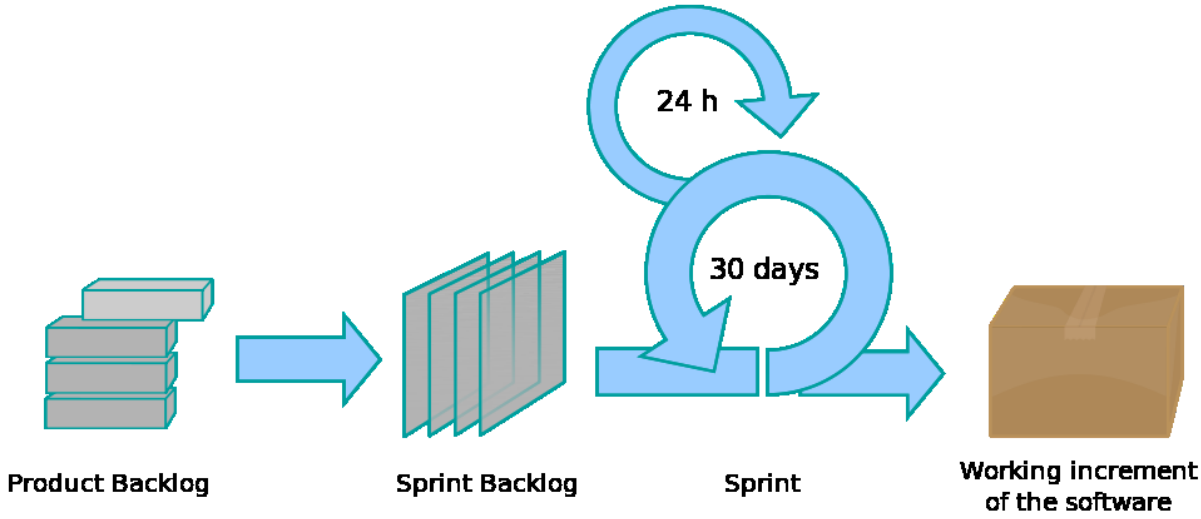
Οι μεθοδολογίες Agile που περιγράφονται παρακάτω μοιράζονται μεγάλο μέρος της ίδιας γενικής φιλοσοφίας, καθώς και πολλά από τα ίδια χαρακτηριστικά και πρακτικές. Ωστόσο, από την άποψη της εφαρμογής, το καθένα έχει το δικό του μοναδικό συνδυασμό πρακτικών, ορολογίας και τακτικής. Κάποιες από τις πιο ευρέως χρησιμοποιούμενες ευέλικτες μεθοδολογίες είναι οι κάτωθι:

- Scrum
- Lean Software Development
- Kanban
- Ακραίος Προγραμματισμός - Extreme Programming (XP)
- Crystal
- Dynamic Systems Development Method (DSDM)
- Feature Driven Development (FDD)

1.6 Μεθοδολογία Scrum

Το Scrum είναι ένα πλαίσιο για την ανάπτυξη και την ολοκλήρωση σύνθετων έργων (Σχήμα 12). Η διαχείριση των έργων αποτελεί το επίκεντρο της προσοχής του Scrum, καθώς αυτή είχε χαρακτηριστεί ως ένας από τους πιο κρίσιμους παράγοντες που οδηγούσαν σε αποτυχία τα έργα λογισμικού. Ως ευέλικτη μέθοδος, το Scrum συμμορφώνεται με τις αξίες και τις αρχές του Agile που υποδεικνύονται από το ευέλικτο μανιφέστο, το οποίο έχει ήδη αναλυθεί σε προηγούμενη παράγραφο. Οι Rising et al. το περιγράφουν ως «μια διαδικασία για τη σταδιακή δημιουργία λογισμικού σε σύνθετα περιβάλλοντα» [35]. Η ανάπτυξη βασίζεται σε επαναλήψεις που ονομάζονται σπριντ (Sprint) και συνήθως διαρκούν από μια έως τρεις εβδομάδες, ενώ το κάθε σπριντ καθορίζεται από μια σειρά οδηγιών που ονομάζεται backlog. Οι Glaiel et al. [36] συνεισφέρουν στην κατανόηση της έννοιας του backlog, εξηγώντας ότι το backlog αποτελεί βασικά ένα σύνολο εργασιών που επιλέχθηκαν για εκτέλεση κατά τη διάρκεια της επαναληπτικής διαδικασίας του σπριντ και σε αυτό περιλαμβάνονται όλα τα χαρακτηριστικά που το προϊόν θα πρέπει να έχει. Ο Cho [37] εξηγεί εν συντομία όλη την διαδικασία των επαναληπτικών διαδικασιών σπριντ: Η διαδικασία περιλαμβάνει πολλά στάδια- αρχικά υπάρχουν οι συναντήσεις για τον προγραμματισμό των σπριντ, όπου οι στόχοι και οι εργασίες που θα εκτελεστούν κατά τη διάρκεια κάθε σπριντ παρουσιάζονται και αποφασίζονται. Στη συνέχεια λαμβάνουν χώρα οι καθημερινές συναντήσεις Scrum, όπου τα μέλη της ομάδας μπορούν να

παρουσιάσουν την πρόοδο της δουλειά τους, τα προβλήματά τους και τη γνώμη τους για την πορεία του έργου.



Σχήμα 12: Το πλαίσιο Scrum

Τέλος, η ομάδα ανάπτυξης συναντά τον πελάτη και αναλύουν τα αποτελέσματα και τα δεδομένα που προέκυψαν από το σπριντ. Η παραδοτέα έκδοση του προϊόντος ελέγχεται από τον πελάτη και λαμβάνεται η απόφαση: είτε ο πελάτης επικυρώνει την έκδοση, ή οι λειτουργίες της έκδοσης θα πρέπει να επεξεργαστούν εκ νέου σε ένα επερχόμενο σπριντ.

1.6.1 Οι τρεις πυλώνες της πλαισίου Scrum

Το πλαίσιο Scrum βασίζεται στην εμπειρική θεωρία ελέγχου της διαδικασίας ή στον εμπειρισμό. Ο εμπειρισμός ισχυρίζεται ότι η γνώση προέρχεται από την εμπειρία και τη λήψη αποφάσεων με βάση αυτό που είναι γνωστό. Έτσι, το πλαίσιο Scrum χρησιμοποιεί μια επαναληπτική, σταδιακή προσέγγιση για τη βελτιστοποίηση του τρόπου πρόβλεψης και τον έλεγχο του κινδύνου.

Τρεις πυλώνες υποστηρίζουν κάθε εφαρμογή του εμπειρικού ελέγχου της διαδικασίας: η διαφάνεια (Transparency), η επιθεώρηση (Inspection) και η προσαρμογή (Adaptation).

1.6.1.1 Διαφάνεια

Η διαφάνεια ορίζει ότι τα γεγονότα θα πρέπει να παρουσιάζονται όπως πραγματικά είναι. Όλα τα εμπλεκόμενα άτομα - ο πελάτης, ο διευθύνων σύμβουλος, οι μεμονωμένοι συνεισφέροντες - θα πρέπει επίσης να λειτουργούν με διαφάνεια στις καθημερινές τους συναλλαγές με τους άλλους. Θα πρέπει να εμπιστεύεται ο ένας τον άλλον και έχουν το θάρρος να ενημερώνουν τους υπόλοιπους τόσο για τα καλά όσο και για τα άσχημα νέα. Όλοι θα πρέπει να συνεργάζονται συλλογικά για τον κοινό οργανωτικό στόχο και κανείς δεν θα πρέπει να έχει κρυφή ατζέντα. Η διαφάνεια αποτελεί ένα από τα βασικά πλεονεκτήματα της μεθοδολογίας Scrum, καθώς αποκαλύπτει τα σημεία προόδου της εργασίας και της ομάδας. Αυτό σημαίνει ότι όταν η ομάδα επιτυγχάνει τον στόχο της, οι υπεύθυνοι για αυτήν την επιτυχία μπορούν να αναγνωριστούν και, έτσι, να εκτιμηθούν οι προσπάθειες που έχουν καταβάλει.

Επιπλέον, η διαφάνεια απαιτεί αυτές οι πτυχές να καθορίζονται από ένα κοινό πρότυπο, ώστε οι παρατηρητές να μοιράζονται μια κοινή κατανόηση του τι βλέπουν. Για παράδειγμα, θα πρέπει να υπάρχει ένας κοινός κώδικας επικοινωνίας τον οποίο θα μοιράζονται όλοι στην ομάδα ή μια κοινή κατανόηση σχετικά με το πότε ένα προϊόν θα θεωρείται έτοιμο.

1.6.1.2 Επιθεώρηση

Η επιθεώρηση σε αυτό το πλαίσιο δεν υλοποιείται από κάποιον επιθεωρητή ή ελεγκτή, αλλά πραγματοποιείται με την συμμετοχή όλης της ομάδας Scrum. Ο έλεγχος μπορεί να γίνει πάνω στο προϊόν, στις διαδικασίες ή στις πρακτικές, με στόχο τη διαρκή βελτίωση. Για παράδειγμα, η ομάδα παρουσιάζει ανοιχτά και με διαφάνεια την έκδοση του προϊόντος στο τέλος του κάθε Σπριντ στον πελάτη, προκειμένου να αποκομίσει τη γνώμη και τα σχόλιά του. Εάν ο πελάτης αλλάξει τις απαιτήσεις του κατά τη διάρκεια αυτού του ελέγχου, η ομάδα δεν θα πρέπει να διαμαρτυρηθεί αλλά να προσαρμοστεί στα νέα δεδομένα, χρησιμοποιώντας μάλιστα τις νέες απαιτήσεις ως ευκαιρία για βελτίωση της συνεργασίας με τον πελάτη. Αν και οι χρήστες του Scrum θα πρέπει συχνά να ελέγχουν τις αναφορές του Scrum (Scrum Artifacts), η επιθεώρησή τους δεν πρέπει να είναι τόσο συχνή ώστε η επιθεώρηση να εμποδίζει την εργασία. Σε κάθε περίπτωση, οι επιθεωρήσεις είναι πιο επωφελείς όταν πραγματοποιούνται επιμελώς από εξειδικευμένους επιθεωρητές στο σημείο εργασίας. Η επιθεώρηση στη μεθοδολογία Scrum μπορεί να πραγματοποιηθεί με δραστηριότητες όπως:

- Χρήση ενός κοινού πίνακα Scrum και άλλων πληροφοριών, ώστε να αποσαφηνιστεί η τρέχουσα κατάσταση του έργου σε όλους τους εμπλεκόμενους.
- Συλλογή σχολίων από τον πελάτη και τους άλλους ενδιαφερόμενους καθ' όλη τη διάρκεια του σταδίου της ανάπτυξης.
- Δημιουργία των αναφορών του προϊόντος με σειρά προτεραιότητας και διεξαγωγή σχεδιασμού των διαδικασιών έκδοσης.
- Επιθεώρηση και έγκριση της παραδοτέας έκδοσης από τον πελάτη.
- Συμμετοχή του πελάτη στη διαδικασία επίδειξης και επικύρωσης των επαναληπτικών διαδικασιών σπριντ.

1.6.1.3 Προσαρμογή

Είναι πλέον κατανοητό ότι στις ευέλικτες μεθοδολογίες οι αλλαγές είναι πάντα ευπρόσδεκτες με στόχο τη διαρκή βελτίωση του προϊόντος. Όταν ένας επιθεωρητής διαπιστώσει ότι μία ή περισσότερες πτυχές μιας διαδικασίας αποκλίνουν από τους ορισμένους στόχους και το προκύπτον προϊόν δεν φαίνεται να είναι αποδεκτό, τότε η διαδικασία ή το υλικό που υποβάλλεται σε επεξεργασία θα πρέπει να αναπροσαρμοστεί. Αυτή η αναπροσαρμογή θα πρέπει να πραγματοποιηθεί το συντομότερο δυνατό ώστε για να ελαχιστοποιηθεί η περαιτέρω απόκλιση. Η έννοια της αναπροσαρμογής όπως αυτή αναφέρεται παραπάνω σημαίνει είτε την αλλαγή της μη λειτουργικής διαδικασίας, ή την μετατροπή μιας διαδικασίας ούτως ώστε να λειτουργεί αποδοτικότερα. Αυτό σημαίνει συνεχής εκτέλεση μικρών πειραμάτων, ώστε να διατηρείται ό, τι λειτουργεί και να αλλάζει ότι φαίνεται ανεπαρκές.

1.6.1.4 Συμπεράσματα

Στο πλαίσιο Scrum, οι αποφάσεις λαμβάνονται με βάση την παρατήρηση και τον πειραματισμό και όχι τον λεπτομερή εκ των προτέρων σχεδιασμό. Ο έλεγχος της εμπειρικής διαδικασίας βασίζεται στους τρεις πυλώνες: της διαφάνειας, της επιθεώρησης και της προσαρμογής. Αυτό σημαίνει ότι βάσει των αποτελεσμάτων ενός έργου θα πρέπει:

- Να είναι ορατές οι αρμοδιότητες στις σημαντικές πτυχές της διαδικασίας ώστε να αναγνωρίζεται η εργασία εκείνων που είναι υπεύθυνοι για το αποτέλεσμα και να λαμβάνουν τα εύσημα.
- Να διεξάγονται έγκαιροι έλεγχοι προόδου στους στόχους ενός Σπριντ για τον εντοπισμό των ανεπιθύμητων αποκλίσεων.
- Η αναπροσαρμογή μιας διαδικασίας να γίνεται το συντομότερο δυνατό ώστε να ελαχιστοποιηθούν τυχόν περαιτέρω αποκλίσεις.

Συμπερασματικά, ο εμπειρισμός και οι τρεις πυλώνες δεν είναι απλά σημαντικοί για το πλαίσιο Scrum, αλλά ουσιαστικά αποτελούν το θεμέλιο της. Αποτελεί τον τρόπο επίτευξης συνεχούς βελτίωσης τόσο στα προϊόντα, όσο και στις διαδικασίες που χρησιμοποιούνται στην ομάδα.

1.6.2 Οι ομάδες Scrum

Μέσα στο πλαίσιο του Scrum, όλη η εργασία που παραδίδεται στον πελάτη υλοποιείται από ειδικές ομάδες ανθρώπων που ονομάζονται ομάδες Scrum. Η ομάδα Scrum είναι μια συλλογή από άτομα που συνεργάζονται για την παράδοση των απαιτούμενων και των δεσμευμένων προσαυξήσεων των προϊόντων. Το πλαίσιο Scrum ενθαρρύνει ένα υψηλό επίπεδο επικοινωνίας μεταξύ των μελών της ομάδας, έτσι ώστε η ομάδα να μπορεί:

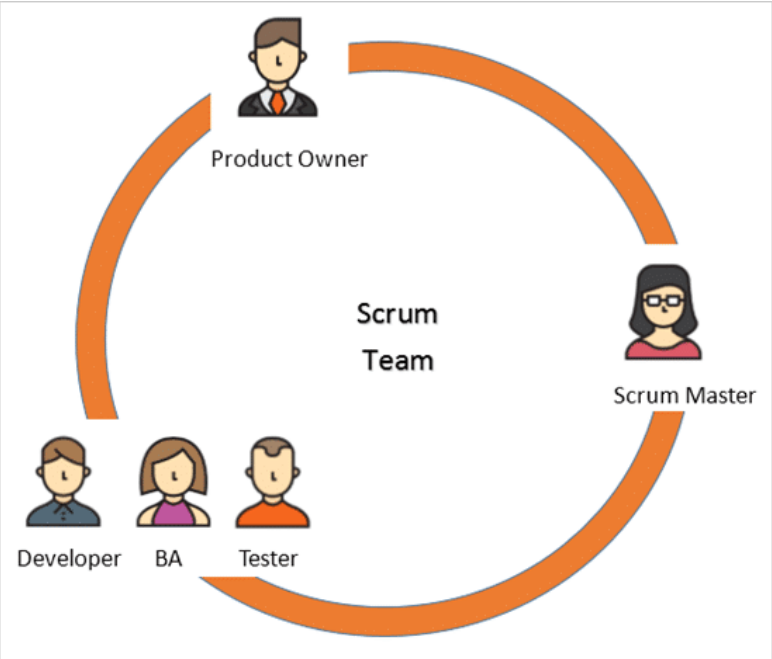
- Να ακολουθεί έναν κοινό στόχο.
- Να συμμορφώνεται με τους ίδιους κανόνες και κανονισμούς.
- Να δείχνει σεβασμό ο ένας στον άλλο.

1.6.2.1 Υποχρεώσεις των ομάδων Scrum

Η ομάδα Scrum και κάθε μέλος της ομάδας αναλαμβάνουν ορισμένες ευθύνες, οι οποίες θα πρέπει να εκπληρώνονται. Στην αρχή θα πρέπει να αναλύονται οι απαιτήσεις του έργου, να δημιουργούνται εργασίες, να εκτιμώνται και να διανέμονται. Με άλλα λόγια, αυτό σημαίνει τη δημιουργία της αναφοράς ανεκπλήρωτου προϊόντος (Backlog). Επίσης, έχουν την υποχρέωση να εκτελούν τη σύντομη ημερήσια συνάντηση Σπριντ. Θα πρέπει να διασφαλίζουν ότι στο τέλος του κάθε Σπριντ, η ομάδα θα είναι σε θέση να παραδώσει λειτουργική έκδοση του προϊόντος. Τέλος, έχουν την ευθύνη της καταγραφής της κατάστασης και των υπολοίπων προσπαθειών πάνω στα καθήκοντά της ομάδας, ώστε να επιτρέπεται η δημιουργία ενός καθοδικού διαγράμματος σπριντ.

1.6.3 Ρόλοι των ομάδων Scrum

Μια ομάδα Scrum αποτελείται από τον ιδιοκτήτη του προϊόντος (Product owner), την ομάδα ανάπτυξης (Development team) και τον Scrum Master (Σχήμα 13). Οι ομάδες Scrum είναι αυτό-οργανωμένες και διαλειτουργικές. Οι αυτό-οργανωμένες ομάδες επιλέγουν τον καλύτερο τρόπο για να ολοκληρώσουν τη δουλειά τους, αντί να κατευθύνονται από άλλους εκτός της ομάδας. Οι διαλειτουργικές ομάδες διαθέτουν όλες τις ικανότητες που απαιτούνται για την ολοκλήρωση της εργασίας, χωρίς να εξαρτώνται από άλλους που δεν αποτελούν μέρος της ομάδας. Το ομαδικό μοντέλο στο Scrum έχει σχεδιαστεί για να βελτιστοποιεί την ευελιξία, τη δημιουργικότητα και την παραγωγικότητα. Οι ομάδες Scrum παρέχουν επαναληπτικές εκδόσεις του προϊόντος και, σταδιακά, μεγιστοποιούν τις ευκαιρίες για ανατροφοδότηση. Οι αυξητικές παραδόσεις του τελικού προϊόντος διασφαλίζουν ότι θα υφίσταται πάντα διαθέσιμη μια δυνητικά χρήσιμη έκδοση του προϊόντος.



Σχήμα 13: Η σύσταση μιας ομάδας Scrum

1.6.3.1 Ιδιοκτήτης του προϊόντος (Product owner)

Ο ιδιοκτήτης του προϊόντος είναι υπεύθυνος για τη μεγιστοποίηση της αξίας του προϊόντος και της εργασίας που προκύπτει από την Ομάδα Ανάπτυξης. Ο τρόπος με τον οποίο αυτό επιτυγχάνεται, ποικίλλει ευρέως μεταξύ των οργανισμών, των ομάδων Scrum και των ατόμων. Ο ιδιοκτήτης του προϊόντος είναι το μοναδικό άτομο που είναι υπεύθυνο για την διαχείριση του ανεκτέλεστου προϊόντος (Backlog) . Αυτή η διαχείριση περιλαμβάνει:

- Ξεκάθαρη έκφραση και περιγραφή των στοιχείων του ανεκτέλεστου προϊόντος.
- Παραγγελία των αντικειμένων που περιλαμβάνονται στη λίστα του ανεκτέλεστου προϊόντος και τα οποία απαιτούνται για την καλύτερη επίτευξη των στόχων και των αποστολών της ομάδας.
- Βελτιστοποίηση της αξίας της εργασίας που εκτελεί η Ομάδα Ανάπτυξης.
- Διασφάλιση ότι τα στοιχεία που περιλαμβάνονται στο ανεκτέλεστο προϊόν είναι σε όλους ορατά, διαφανή και σαφή και υποδεικνύουν το τι θα δουλέψει η ομάδα Scrum στη συνέχεια.
- Διασφάλιση ότι η Ομάδα Ανάπτυξης κατανοεί τα στοιχεία του ανεκτέλεστου προϊόντος στο επίπεδο που απαιτείται.

Όλες οι παραπάνω εργασίες μπορούν να υλοποιηθούν είτε από τον ιδιοκτήτη του προϊόντος ή ακόμη και από ολόκληρη την Ομάδα Ανάπτυξης. Ωστόσο, ο ιδιοκτήτης του προϊόντος παραμένει υπόλογος για αυτές τις εργασίες. Θα πρέπει να σημειωθεί ότι ο ιδιοκτήτης του προϊόντος είναι μόνο ένα άτομο, όχι επιτροπή, αν και μπορεί να αντιπροσωπεύει τις επιθυμίες μιας επιτροπής. Ωστόσο, όσοι θέλουν να αλλάξουν την προτεραιότητα ενός ανεκτέλεστου προϊόντος, θα πρέπει να απευθύνονται αποκλειστικά στον ιδιοκτήτη προϊόντος. Για να καταφέρει να πετύχει ο ιδιοκτήτης του προϊόντος τον σκοπό του, θα πρέπει ολόκληρος ο οργανισμός να σέβεται τις αποφάσεις του. Κανείς δεν επιτρέπεται να πει στην ομάδα ανάπτυξης να ακολουθήσει ένα διαφορετικό σύνολο απαιτήσεων από αυτό που έχει επιβάλλει ο ιδιοκτήτης και η ομάδα ανάπτυξης δεν επιτρέπεται να ενεργεί σύμφωνα με όσα λέει κάποιος άλλος.

1.6.3.2 Scrum master

Οι Scrum masters είναι οι διαμεσολαβητές της μεθοδολογίας Scrum και ως τέτοιοι, λειτουργούν ως συντονιστές για την υπόλοιπη ομάδα. Ένας καλός Scrum master είναι αφοσιωμένος στα θεμέλια και τις αξίες του Scrum, αλλά ταυτόχρονα παραμένει ευέλικτος και ανοιχτός σε ευκαιρίες που θα μπορούσαν να βελτιώσουν τη ροή εργασίας της ομάδας. Αν και ο τίτλος του Scrum Master ακούγεται ισχυρός, αυτό δεν σημαίνει ότι είναι ο επικεφαλής του έργου και δεν θεωρείται υπεύθυνος για τα αποτελέσματα του έργου- αυτή η ευθύνη βαραίνει την ομάδα στο σύνολό της. Παρ' όλα αυτά παραμένει ένας πολύ δυναμικός ρόλος και φέρει την ευθύνη να:

- Βοηθήσει την ομάδα να επιτύχει συναίνεση για το τι μπορεί να επιτευχθεί σε μια συγκεκριμένη χρονική περίοδο, δηλαδή σε ένα σπριντ.
- Βοηθήσει την ομάδα να επιτύχει συναίνεση κατά τη διάρκεια του καθημερινού Scrum.
- Βοηθήσει την ομάδα να παραμείνει συγκεντρωμένη και να ακολουθήσει τους συμφωνημένους κανόνες που ισχύουν για τα καθημερινά Scrums.
- Καταργήσει τα εμπόδια που απειλούν την πρόοδο της ομάδας.
- Προστατεύει την ομάδας από εξωτερικούς περισπασμούς.
- Διασφαλίζει ότι τα στοιχεία του ανεκτέλεστου προϊόντος ορίζονται σαφώς και αποτελεσματικά.

Ο κύριος ρόλος του Scrum master είναι αυτός του διαμεσολαβητή. Εξασφαλίζει ότι ακολουθούνται οι βέλτιστες πρακτικές και ότι τα έργα της ομάδας προχωρούν, ενώ ενθαρρύνει τη διαφάνεια, τον έλεγχο και την προσαρμογή. Οι κοινές δεξιότητες που θα πρέπει να κατέχει ένας Scrum master περιλαμβάνουν:

- Την ικανότητα να διευκολύνει την επικοινωνία μεταξύ των μελών της ομάδας και να προωθεί την αίσθηση της κοινότητας.
- Την ικανότητα να βοηθά τα μέλη της ομάδας να προσαρμοστούν σε νέες καταστάσεις μέσω της αρωγής και της καθοδήγησης.
- Την υπευθυνότητα να κοινοποιεί την πρόοδο και τις ανάγκες της ομάδας σε εξωτερικές ομάδες.
- Την ηρεμία και την ενσυναίσθηση για τον χειρισμό μεταβαλλόμενων διαπροσωπικών δυναμικών, συμπεριφορικών προτύπων και επίλυσης συγκρούσεων.

Εκτός από την ομάδα ανάπτυξης, τα άτομα που ανήκουν στους υπόλοιπους ρόλους ενός Scrum, όπως για παράδειγμα ο διαχειριστής του έργου και ο ιδιοκτήτης του προϊόντος, θα πρέπει και αυτά να συνεργάζονται στενά με τον Scrum master για την επίτευξη ενός σαφώς καθορισμένου κοινού στόχου.

1.6.3.3 Ομάδα Ανάπτυξης

Οι ομάδες ανάπτυξης είναι καλά δομημένες ομάδες, οι οποίες είναι εξουσιοδοτημένες από τον οργανισμό να διαχειρίζονται αποτελεσματικά την εργασία τους. Αυτό έχει ως αποτέλεσμα μια μοναδική συνεργασία που βελτιστοποιεί τη συνολική αποτελεσματικότητα της ομάδας ανάπτυξης. Σύμφωνα με τους Cohen et al. [38], το ιδανικό μέγεθος μιας ομάδας είναι από επτά έως δέκα άτομα, χωρίς να συμπεριλαμβάνεται ο Scrum master και ο ιδιοκτήτης του προϊόντος. Με οποιοδήποτε μικρότερο μέγεθος από αυτό, η ομάδα δεν θα μπορέσει να αποδώσει ικανοποιητικά στο κάθε σπριντ. Αντίθετα, με οποιοδήποτε μεγαλύτερο μέγεθος ομάδας, η επικοινωνία θα γινόταν περίπλοκη και δυσκίνητη. Υπάρχουν μερικά εμφανή χαρακτηριστικά μιας ομάδας ανάπτυξης. Τα σημαντικότερα από αυτά είναι:

- Είναι αυτό-οργανωμένες ομάδες. Κανένας, ούτε ακόμη και ο Scrum master, δεν θα πρέπει να κατευθύνει την ομάδα ανάπτυξης σχετικά με τον τρόπο που θα μετατραπεί το ανεκτέλεστο προϊόν σε αυξητικές εκδόσεις του προϊόντος.
- Οι ομάδες ανάπτυξης είναι γενικά διαλειτουργικές. Αποτελούνται από μέλη με ποικίλες δεξιότητες. Σε επίπεδο ομάδας, αυτές οι συνδυασμένες δεξιότητες είναι απαραίτητες για τη δημιουργία ενός προϊόντος.
- Τα μέλη της ομάδας ανάπτυξης δεν έχουν ατομικούς τίτλους. Κάθε μέλος αναγνωρίζεται μόνο ως μέρος της ομάδας ανάπτυξης, ανεξάρτητα από την εργασία που εκτελείται από το άτομο.
- Η μεθοδολογία Scrum δεν αναγνωρίζει υπό-ομάδες μέσα στην ομάδα ανάπτυξης, αν και μπορεί να αποτελείται από διάφορους τομείς όπως ο τομέας δοκιμών, οι επιχειρηματικές αναλύσεις, οι λειτουργίες ή οι αρχιτεκτονικές.
- Η ομάδα ανάπτυξης φέρει στο σύνολό της την ευθύνη ενός έργου, όχι μεμονωμένα κάποια μέλη της.

Όσον αφορά τις υποχρεώσεις και τις ευθύνες που αναλαμβάνουν οι ομάδες ανάπτυξης, μεταξύ άλλων περιλαμβάνονται και τα ακόλουθα:

- **Η εκτέλεση των επαναλήψεων σπριντ:** Κατά τη διάρκεια της διεξαγωγής ενός σπριντ, τα μέλη της ομάδας ανάπτυξης εκτελούν τα καθήκοντα του σχεδιασμού, της ανάπτυξης, της ενσωμάτωσης και της δοκιμής των εκκρεμοτήτων των προϊόντων, με σκοπό τη δημιουργία της αυξητικής έκδοσης που θα αποσταλεί για έλεγχο στον πελάτη. Μέσω της αυτό-οργάνωσης, αποφασίζουν από κοινού τον τρόπο σχεδιασμού, διαχείρισης και εκτέλεσης της εργασίας. Η ομάδα ανάπτυξης αφιερώνει αρκετό χρόνο στην εκτέλεση σπριντ.
- **Επιθεώρηση και προσαρμογή:** Όλα τα μέλη της ομάδας ανάπτυξης θα πρέπει ανελλιπώς να συμμετέχουν στο καθημερινό Scrum. Κατά τη διάρκεια αυτής της διαδικασίας, τα μέλη της ομάδας επιθεωρούν συλλογικά την πρόοδό τους προς το στόχο του σπριντ και προσαρμόζουν ανάλογα το σχέδιο για την εργασία της τρέχουσας ημέρας. Εάν κάποια από τα μέλη της ομάδας δεν συμμετάσχουν στην καθημερινή διαδικασία, η ομάδα μπορεί να χάσει ζωτικά κομμάτια της μεγαλύτερης εικόνας του έργου, και έτσι ο στόχος του σπριντ θα οδηγηθεί σε αποτυχία.
- **Καλλωπισμός του ανεκτέλεστου προϊόντος backlog:** Σε κάθε σπριντ, η ομάδα ανάπτυξης θα πρέπει να αφιερώσει αρκετό χρόνο για την προετοιμασία του επόμενου σπριντ. Ένα μεγάλο μέρος αυτής της εργασίας θα πρέπει να επικεντρωθεί στον καλλωπισμό του ανεκτέλεστου προϊόντος. Αυτός ο καλλωπισμός περιλαμβάνει τη δημιουργία , τον εξευγενισμό, την εκτίμηση και την προτεραιότητα των στοιχείων του ανεκτέλεστου προϊόντος. Σε κάθε σπριντ, η ομάδα ανάπτυξης θα πρέπει να διαθέσει έως και 10% της συνολικής ικανότητάς της, ώστε να βοηθήσει τον ιδιοκτήτη του προϊόντος να παρουσιάσει μια ελκυστική αυξητική έκδοση του προϊόντος.
- **Σχεδιασμός του σπριντ:** Η ομάδα ανάπτυξης, στην αρχή του κάθε σπριντ, συμμετέχει στον προγραμματισμό του επόμενου σπριντ. Δηλαδή, σε συνεργασία με τον ιδιοκτήτη του προϊόντος, η ομάδα ανάπτυξης σχεδιάζει τον τρόπο επίτευξης του στόχου για το επόμενο σπριντ. Μόλις καθοριστεί ο στόχος, η ομάδα καθορίζει ένα υποσύνολο υψηλής προτεραιότητας εκκρεμοτήτων του προϊόντος που θα πρέπει να ολοκληρωθούν ώστε να επιτευχθεί ο στόχος. Ο χρόνος που απαιτείται για τον σχεδιασμό του σπριντ είναι άμεσα ανάλογος με το μέγεθος του σπριντ. Η διάρκεια προγραμματισμού για σπριντ δύο εβδομάδων είναι περίπου μισή ημέρα. Ένα σπριντ τεσσάρων εβδομάδων μπορεί να χρειαστεί έως και μια ολόκληρη μέρα για τον προγραμματισμό του. Αυτός ο σχεδιασμός ακολουθεί επίσης ένα

επαναληπτικό σχέδιο. Σε αντίθεση με τις παραδοσιακές μεθόδους σχεδιασμού έργων, η ομάδα δεν επικεντρώνεται σε ένα περίπλοκο και αβέβαιο σχέδιο στην αρχή της αναπτυξιακής της προσπάθειας. Αντίθετα, ακολουθεί μια σειρά μικρών βημάτων και σιγουρότερα και λεπτομερέστερα σχέδια στην αρχή κάθε σπριντ.

- **Επιθεώρηση και προσαρμογή του προϊόντος και της διαδικασίας:** Προς το τέλος του κάθε σπριντ, η ομάδα ανάπτυξης περιλαμβάνει τις δραστηριότητες επιθεώρησης και προσαρμογής-επισκόπηση του σπριντ και αναδρομή του σπριντ. Στην επισκόπηση σπριντ, συμμετέχουν επίσης η ομάδα ανάπτυξης, ο ιδιοκτήτης προϊόντος, ο Scrum master, οι ενδιαφερόμενοι, οι χορηγοί, οι πελάτες και τα ενδιαφερόμενα μέλη άλλων ομάδων. Εξετάζουν τα μόλις ολοκληρωμένα χαρακτηριστικά του τρέχοντος σπριντ και συζητούν για το πώς θα προχωρήσουν με αποτελεσματικό τρόπο. Η εκ των υστέρων δραστηριότητα του σπριντ (αναδρομή του σπριντ) συμβαίνει όταν η ομάδα του Scrum επιθεωρεί και προσαρμόζει τη διαδικασία και τις τεχνικές πρακτικές του Scrum , ώστε να βελτιώσει τον τρόπο που χρησιμοποιείται η μεθοδολογία Scrum και να προσφέρει στο προϊόν την καλύτερη επιχειρηματική αξία.

Τέλος, μια ομάδα ανάπτυξης θα πρέπει να έχει συγκεκριμένους ρόλους για τα μέλη της. Δεν θα πρέπει συγχέονται οι λειτουργικοί ρόλοι των μελών, όπως οι προγραμματιστές, οι υπεύθυνοι δοκιμών, οι αναλυτές επιχειρήσεων κ.λπ. Εκτός από αυτό, τα μέλη της ομάδας θα πρέπει σε προσωπικό επίπεδο να έχουν στόχους και κίνητρα, να είναι οργανωτικοί και να τους αρέσει να δουλεύουν ομαδικά.

1.6.4 Γεγονότα του Scrum (Scrum events)

Το πλαίσιο των διαδικασιών ενός Scrum υλοποιείται μέσω μιας ακολουθίας γεγονότων και των αντίστοιχων παραγόμενων αντικειμένων/εξόδων (Artifacts). Οι δραστηριότητες του Scrum αποτελούν γεγονότα μέσα σε ένα χρονικό πλαίσιο. Αυτό σημαίνει ότι σε ένα έργο, κάθε γεγονός Scrum έχει μια προκαθορισμένη μέγιστη διάρκεια και προβάλει με διαφάνεια την πρόοδο του έργου σε όλους όσους συμμετέχουν σε αυτό. Όλα τα συμβάντα Scrum, συμπεριλαμβανομένης της βελτίωσης των ανεκπλήρωτων προϊόντων, πραγματοποιούνται κατά τη διάρκεια ενός σπριντ. Επομένως, σε κάθε σπριντ, θα πρέπει να αναπτύσσεται και ένα αυξητικό προϊόν εργασίας, ενώ η διάρκειά του συνήθως κυμαίνεται από δύο εβδομάδες έως ένα μήνα και αυτή η διάρκεια παραμένει σταθερή για όλα τα σπριντ του έργου. Θα πρέπει επίσης να σημειωθεί ότι: α) δεν μπορούμε να έχουμε διαφορετικές χρονικές περιόδους για τα διαφορετικά σπριντ σε ένα έργο και β) ένα νέο σπριντ ξεκινά αμέσως μετά την ολοκλήρωση του προηγούμενου. Τέλος, ένα σπριντ θα πρέπει να ακυρωθεί εάν ο στόχος του καταστεί παρωχημένος. Αυτό μπορεί να συμβεί εάν ο οργανισμός αλλάξει κατεύθυνση ή εάν αλλάξουν οι συνθήκες αγοράς ή τεχνολογίας. Σε γενικές γραμμές, ένα σπριντ μπορεί να ακυρωθεί μόνο από τον κάτοχο του προϊόντος, αν και κάποιες φορές τα υπόλοιπα μέλη μπορούν να επηρεάσουν την απόφαση. Η μεθοδολογία Scrum ορίζει τέσσερα γεγονότα (κάποιες φορές ονομάζονται και τελετές), τα οποία λαμβάνουν χώρα σε κάθε σπριντ: Σχεδιασμός του σπριντ (Sprint Planning), Ημερήσιο Scrum (Daily Scrum), ανάλυση του σπριντ (Sprint Review) και αναδρομική ανάλυση του σπριντ (Sprint Retrospective).

1.6.4.1 Σχεδιασμός του σπριντ (Sprint Planning)

Οι εργασίες που θα εκτελεστούν στο σπριντ προγραμματίζονται στη συνάντηση σχεδιασμού του σπριντ. Η συνάντηση σχεδιασμού του σπριντ έχει διάρκεια το πολύ τέσσερις ώρες για ένα σπριντ που θα διαρκέσει δύο εβδομάδες και οκτώ ώρες για ένα σπριντ που θα διαρκέσει ένα μήνα. Είναι ευθύνη του Scrum Master να διασφαλίσει ότι η συνάντηση θα πραγματοποιηθεί απρόσκοπτα και ότι όλοι οι

απαιτούμενοι παρευρισκόμενοι είναι παρόντες και κατανοούν τον σκοπό της προγραμματισμένης συνάντησης. Επιπλέον, ο Scrum Master συντονίζει τη συζήτηση ώστε να είναι σε θέση να παρακολουθεί την πορεία της και, την προκαθορισμένη ώρα, να επιβάλει τον τερματισμό της. Ο σχεδιασμός ενός σπριντ επικεντρώνεται στις ακόλουθες δύο ερωτήσεις:

- Τι πρέπει να περιλαμβάνει η αυξητική έκδοση του προϊόντος ώστε να μπορεί να παραδοθεί;
- Πώς θα επιτευχθούν οι απαιτούμενες εργασίες για την επίτευξη των στόχων του σπριντ;

Από την άλλη, στις εισερχόμενες πληροφορίες, οι οποίες και αποτελούν την βάση της συζήτησης σε μία συνάντηση σχεδιασμού σπριντ, περιλαμβάνονται τα κάτωθι:

- Η λίστα με τις εκκρεμότητες του προϊόντος.
- Η τελευταία αυξητική έκδοση του προϊόντος.
- Πρόβλεψη για την απόδοση της ομάδας κατά τη διάρκεια του σπριντ.
- Εκτίμηση της προηγούμενης απόδοσης της ομάδας.

Η ομάδα Scrum συζητά τη λειτουργικότητα που μπορεί να αναπτυχθεί κατά τη διάρκεια του σπριντ. Ο ιδιοκτήτης του προϊόντος παρέχει διευκρινίσεις σχετικά με τα στοιχεία του προϊόντος τα οποία δεν έχουν εκπληρωθεί μέχρι τη δεδομένη στιγμή. Η ομάδα επιλέγει από τη λίστα των ανεκπλήρωτων εργασιών τα στοιχεία με τα οποία θα ασχοληθεί στο σπριντ, δημιουργώντας έτσι έναν μετρήσιμο τρόπο αξιολόγησης της απόδοσής της. Η ομάδα μπορεί επίσης να προσκαλέσει τρίτους (οι οποίοι δεν ανήκουν στην ομάδα Scrum) να παρευρεθούν στη συνάντηση ώστε να λάβουν εξειδικευμένες τεχνικές συμβουλές ή βοήθεια στην εκτίμηση.

1.6.4.2 Ημερήσιο Scrum (Daily Scrum)

Η ομάδα ανάπτυξης συνεδριάζει για δεκαπέντε λεπτά (ή και λιγότερο) κάθε μέρα της διεξαγωγής του σπριντ, ώστε για να ελέγξει την πρόοδό του. Περιγράφουν ο ένας στον άλλον την πορεία της δουλειάς τους, ζητούν βοήθεια όταν χρειάζεται και εξετάζουν εάν είναι ακόμα σε καλό δρόμο για την επίτευξη του στόχου του σπριντ. Πρακτικά, για την ομάδα ανάπτυξης, πρόκειται για μια ευκαιρία να επιθεωρεί την πορεία του έργου σε καθημερινή βάση και να προσαρμόζει ανάλογα το προϊόν και τις διαδικασίες. Αν και το ημερήσιο Scrum εκλαμβάνεται συχνά ως συμβάν παρακολούθησης της κατάστασης, ωστόσο στην πραγματικότητα είναι ένα συμβάν προγραμματισμού. Η ημερήσια συνάντηση Scrum πραγματοποιείται την ίδια ώρα και στον ίδιο χώρο κάθε μέρα για τη μείωση της πολυπλοκότητας. Κατά τη διάρκεια της συνάντησης, κάθε μέλος της ομάδας εξηγεί:

- Τι έκανε χθες που βοήθησε την ομάδα να επιτύχει τον στόχο του σπριντ;
- Τι θα κάνει σήμερα για να βοηθήσει την ομάδα να επιτύχει τον στόχο του σπριντ;
- Βλέπει τυχόν εμπόδια που εμποδίζουν τον ίδιο ή την ομάδα να επιτύχει τον στόχο του σπριντ;

Η συμβολή των μελών στη συνάντηση θα πρέπει να είναι η ανάλυση της πορείας της ομάδας προς την επίτευξη των στόχων και το αποτέλεσμα θα πρέπει να είναι ένα νέο ή αναθεωρημένο σχέδιο που θα βελτιστοποιεί τις προσπάθειες της ομάδας. Αν και ο Scrum Master είναι αυτός που συντονίζει την ημερήσια συνάντηση Scrum και διασφαλίζει ότι επιτυγχάνονται οι στόχοι της συνάντησης, η συνάντηση εξακολουθεί να αποτελεί ευθύνη όλης της ομάδας. Τέλος, εάν κριθεί απαραίτητο, η ομάδα μπορεί να συναντηθεί αμέσως μετά την ημερήσια συνάντηση Scrum, για λεπτομερείς συζητήσεις ή για να προγραμματίσει εκ νέου το υπόλοιπο έργο του σπριντ.

Τα οφέλη που προκύπτουν από τις καθημερινές συναντήσεις Scrum είναι:

- Βελτίωση της επικοινωνίας εντός της ομάδας.

- Προσδιορισμός των εμπόδιων και άμεση αντιμετώπισή τους, ώστε να ελαχιστοποιηθούν οι επιπτώσεις στο σπριντ.
- Επισημαίνονται και προωθούνται οι γρήγορες λήψεις αποφάσεων.
- Βελτίωση του επίπεδου γνώσεων της ομάδας.

1.6.4.3 Ανάλυση του σπριντ (Sprint Review)

Η ανάλυση πραγματοποιείται στο τέλος του κάθε σπριντ. Κατά τη διάρκεια της ανάλυσης του σπριντ και σε συνεργασία με τον πελάτη, παρουσιάζεται και εξετάζεται η αυξητική έκδοση του προϊόντος που προέκυψε. Σε αυτή τη συνάντηση, η ομάδα Scrum και οι ενδιαφερόμενοι συνεργάζονται για να εκτιμήσουν το αποτέλεσμα του σπριντ και να προχωρήσουν στα επόμενα βήματα που απαιτούνται ώστε να βελτιστοποιηθεί η αξία του προϊόντος. Πρακτικά, ο στόχος της ανάλυσης ενός σπριντ είναι η συλλογή ανατροφοδότησης και η από κοινού απόφαση για των επόμενων βημάτων. Η ανάλυση ενός σπριντ έχει χρονική διάρκεια περίπου δύο ώρες όταν αφορά σπριντ διάρκειας δύο εβδομάδων και τέσσερις ώρες όταν αφορά σπριντ διάρκειας ενός μηνός.

Η ανάλυση του σπριντ περιλαμβάνει τις ακόλουθες φάσεις:

- Στους συμμετέχοντες περιλαμβάνονται η ομάδα Scrum και τα βασικά ενδιαφερόμενα μέρη, όπως ο ιδιοκτήτης του προϊόντος.
- Ο ιδιοκτήτης του προϊόντος εξηγεί ποια στοιχεία από τη λίστα των εκκρεμοτήτων έχουν ολοκληρωθεί κατά τη διάρκεια του σπριντ και ποιά όχι.
- Η ομάδα συζητά τι πήγε καλά κατά τη διάρκεια του σπριντ, ποια προβλήματα αντιμετώπισε και πώς λύθηκαν αυτά τα προβλήματα.
- Η ομάδα επιδεικνύει το έργο που έχει ολοκληρώσει και απαντά σε ερωτήσεις, εάν υπάρχουν, σχετικά με την αυξητική αυτή έκδοση.
- Στη συνέχεια, ολόκληρη η ομάδα συζητά για τα επόμενα βήματα. Στην ουσία, η ανάλυση του σπριντ παρέχει πολύτιμη αρωγή στη σχεδίαση του επόμενου σπριντ.
- Ακολούθως, η ομάδα Scrum εξετάζει το χρονοδιάγραμμα, τον προϋπολογισμό, τις πιθανές δυνατότητες και την αγορά για την επόμενη αναμενόμενη έκδοση του προϊόντος.
- Το αποτέλεσμα της ανάλυσης του σπριντ αποτελεί ένα ενημερωμένο αρχείο ανεκπλήρωτου προϊόντος, το οποίο καθορίζει τις πιθανές εκκρεμότητες για το επόμενο σπριντ.

1.6.4.4 Αναδρομική ανάλυση του σπριντ (Sprint Retrospective)

Στην αναδρομή του σπριντ, η ομάδα επικεντρώνεται στη διαδικασία και συζητά τι πήγε σωστά και ποιοι από τους επιμέρους τομείς μπορεί να χρήζουν βελτίωσης. Κάνουν απτά σχέδια για το πώς θα βελτιώσουν τη δική τους διαδικασία, τα εργαλεία και τις σχέσεις. Η αναδρομή του σπριντ πραγματοποιείται μετά την ανάλυση του σπριντ και πριν από τον επόμενο σχεδιασμό. Συνήθως πρόκειται για συνάντηση μίας ώρας όταν αφορά σπριντ διάρκειας δύο εβδομάδων και συνάντηση τριών ωρών όταν αφορά σπριντ διάρκειας ενός μηνός.

Ο σκοπός της αναδρομικής ανάλυσης του σπριντ είναι:

- Να συνδυαστούν τα διδάγματα από το τελευταίο σπριντ, όσον αφορά τους ανθρώπους, τις σχέσεις, τη διαδικασία και τα εργαλεία.
- Να προσδιοριστούν τα κύρια στοιχεία που πήγαν καλά και οι πιθανές βελτιώσεις.
- Η δημιουργία σχεδίου για την εφαρμογή των βελτιώσεων, με σκοπό την αύξηση της ποιότητας του προϊόντος.

- Πρακτικά, η αναδρομική ανάλυση αποτελεί μια ευκαιρία για την ομάδα να διεισδύσει και να βελτιώσει το πλαίσιο της διαδικασίας Scrum, ώστε να καταστήσει το επόμενο σπριντ πιο αποτελεσματικό.

1.6.5 Αντικείμενα Scrum (Scrum Artifacts)

Τα αντικείμενα Scrum είναι οι πληροφορίες που χρησιμοποιούν η ομάδα και οι ενδιαφερόμενοι για να αναλύσουν λεπτομερώς το προϊόν που αναπτύσσεται, τις ενέργειες για την παραγωγή του και τις ενέργειες που πραγματοποιήθηκαν κατά τη διάρκεια του έργου. Αυτά τα αντικείμενα παρέχουν μεταδεδομένα που δίνουν πληροφορίες για την απόδοση ενός σπριντ. Είναι απαραίτητα εργαλεία για κάθε ομάδα, δεδομένου ότι επιτρέπουν την υλοποίηση των βασικών χαρακτηριστικών της μεθοδολογίας Scrum, όπως η διαφάνεια, η επιθεώρηση και η προσαρμογή. Τα αντικείμενα Scrum αντιπροσωπεύουν την εργασία ή την αξία και έχουν σχεδιαστεί για να μεγιστοποιούν τη διαφάνεια των βασικών πληροφοριών. Η μεθοδολογία Scrum ορίζει τρία βασικά αντικείμενα: Εκκρεμότητες προϊόντος (Product Backlog), εκκρεμότητες σπριντ (Sprint Backlog) και αυξητική έκδοση του προϊόντος (Product Increment).

1.6.5.1 Εκκρεμότητες προϊόντος (Product Backlog)

Οι εκκρεμότητες του προϊόντος είναι μια λίστα που περιλαμβάνει νέες δυνατότητες, βελτιώσεις, επιδιορθώσεις σφαλμάτων, εργασίες ή απαιτήσεις εργασίας που απαιτούνται για τη δημιουργία ενός προϊόντος. Διαμορφώνεται από διάφορες πηγές πληροφοριών, όπως η υποστήριξη πελατών, η ανάλυση των ανταγωνιστών, οι απαιτήσεις της αγοράς και η γενική επιχειρηματική ανάλυση. Οι εκκρεμότητες του προϊόντος είναι ένα συνεχώς μεταβαλλόμενο αντικείμενο, καθώς ενημερώνεται με την έλευση νέων πληροφοριών. Αποτελεί μια λίστα που διατηρείται μεταξύ των ομάδων και επιμελείται ο ιδιοκτήτης του προϊόντος. Περιέχει επίσης τις εργασίες που κάποτε βρίσκονταν σε ενεργό σπριντ, αλλά υποτιμήθηκαν και μετακινήθηκαν πίσω στο αρχείο με τις εκκρεμότητες.

1.6.5.2 Εκκρεμότητες σπριντ (Sprint Backlog)

Οι εκκρεμότητες του σπριντ είναι ένα σύνολο ανεκτέλεστων εργασιών, οι οποίες έχουν προωθηθεί για ανάπτυξη κατά την επόμενη έκδοση του προϊόντος. Καθορίζονται από τις ομάδες ανάπτυξης για να σχεδιάσουν και να αναλύσουν λεπτομερώς τις εργασίες που απαιτούνται για τη δημιουργία της παραδοτέας αυξητικής έκδοσης του προϊόντος. Το αντικείμενο των εκκρεμοτήτων του σπριντ δημιουργείται με την επιλογή μιας εργασίας από τις διαθέσιμες που υπάρχουν στη λίστα με τις εκκρεμότητες του προϊόντος. Στη συνέχεια αυτή η εργασία χωρίζεται σε μικρότερα, ενεργά στοιχεία του σπριντ. Επίσης, το αντικείμενο των εκκρεμοτήτων του σπριντ ενημερώνεται κατά τη φάση του σχεδιασμού του σπριντ. Οι μικρότερες εργασίες του σπριντ ανατίθενται στις σχετικές ομάδες, όπως οι ομάδες σχεδιασμού και ανάπτυξης. Εάν μια ομάδα δεν έχει την δυνατότητα να παραδώσει όλες τις εργασίες που περιλαμβάνονται στο σπριντ, αυτές θα μπούνε σε κατάσταση αναμονής ώστε να καταχωρηθούν σε ένα μελλοντικό σπριντ.

1.6.5.3 Αυξητική έκδοση του προϊόντος (Product Increment)

Η αυξητική έκδοση του προϊόντος είναι το άθροισμα όλων των εκκρεμοτήτων του προϊόντος που ολοκληρώθηκαν κατά τη διάρκεια ενός σπριντ, σε συνδυασμό με το άθροισμα όλων των εκκρεμοτήτων του προϊόντος που εκδόθηκαν σε όλα τα προηγούμενα σπριντ. Στο τέλος ενός σπριντ, η παραδοτέα έκδοση θα πρέπει να είναι ένα προϊόν που λειτουργεί, πράγμα που σημαίνει ότι θα πρέπει να βρίσκεται σε κατάσταση χρήσης. Από εκεί και πέρα, ο ιδιοκτήτης του προϊόντος θα αποφασίσει αν

η παραδοτέα έκδοση θα αποτελέσει την τελική μορφή του προϊόντος ή όχι. Σε κάθε περίπτωση όμως, καμιά εργασία δεν μπορεί να θεωρηθεί μέρος της αύξησης της παραδοτέας έκδοσης, αν δεν βρίσκεται σε λειτουργική κατάσταση.

1.6.6 Επιλογική σημείωση πάνω στο πλαίσιο Scrum

Έχοντας ήδη αναλύσει τα δομικά στοιχεία και τη σημασία τους, θεωρείται αναπόφευκτη η χρήση των ρόλων, των γεγονότων, των αντικείμενων και των κανόνων στην εφαρμογή του πλαισίου Scrum. Εάν εφαρμοστούν ορισμένα μόνο μέρη της μεθοδολογίας, το αποτέλεσμα δεν θα είναι Scrum. Το πλαίσιο Scrum θα πρέπει να εφαρμοστεί στο σύνολό της για να λειτουργήσει σωστά και να ευθυγραμμίζεται με άλλες τεχνικές, μεθοδολογίες και πρακτικές.

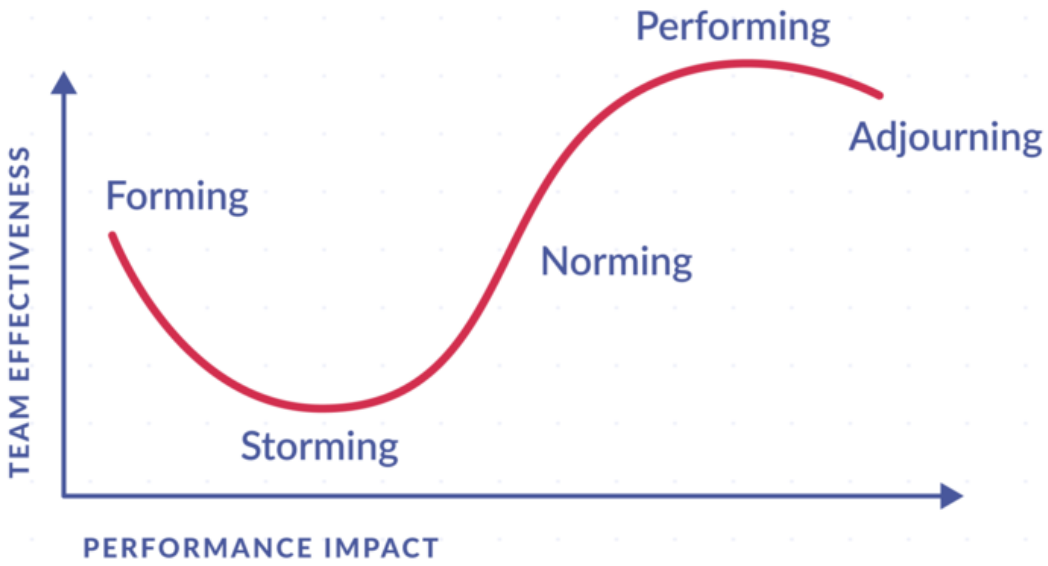
1.7 Το μοντέλο Tuckman

Κατά τη δημιουργία μιας νέας ομάδας Scrum, πρέπει πάντα να θεωρείται ως δεδομένο ότι καμία νέα ομάδα δεν θα είναι σε θέση να προσφέρει τα μέγιστα από την αρχή. Μετά τη δημιουργία της ομάδας, αυτή θα πρέπει να περάσει από συγκεκριμένες φάσεις όπως αυτές περιγράφονται από το Μοντέλο Tuckman [39]: Σχηματισμός (Forming), σύγκρουση (Storming), κανονικοποίηση (Norming), απόδοση (Performing). Αργότερα, πρόσθεσε ένα πέμπτο στάδιο, τη διακοπή (Adjourning- επίσης γνωστό και ως "πένθος") για να σηματοδοτήσει το τέλος του ταξιδιού μιας ομάδας (Σχήμα 14).

1.7.1 Σχηματισμός (Forming)

Στην στάδιο του σχηματισμού, όταν δηλαδή σχηματίζεται μια νέα ομάδα, μπορεί να μην είναι απολύτως ξεκάθαρα στα μέλη ο σκοπός της ομάδας, το κατά πόσο ταιριάζουν και αν θα συνεργαστούν καλά μεταξύ τους. Μπορεί να είναι ανήσυχοι, περίεργοι ή και ενθουσιασμένοι σε αυτήν τη φάση. Ωστόσο, η ευθύνη βαρύνει τον αρχηγό της ομάδας για να τους συντονίσει. Αυτό το στάδιο ίσως χρειάζεται περισσότερο χρόνο, καθώς οι άνθρωποι γνωρίζουν τους νέους συναδέλφους τους και τους τρόπους εργασίας τους. Αυτό το στάδιο χαρακτηρίζεται από:

- Υψηλή εξάρτηση από τον ηγέτη για οδηγίες και καθοδήγηση.
- Μικρή συμφωνία πάνω στους στόχους της ομάδας, εκτός από αυτούς που δίδονται από τον αρχηγό.
- Οι ατομικοί ρόλοι και οι ευθύνες είναι ακόμη ασαφείς.
- Ο αρχηγός πρέπει να είναι έτοιμος να απαντήσει σε πολλές ερωτήσεις σχετικά με τον σκοπό, τους στόχους και τις εξωτερικές σχέσεις της ομάδας. Στη παρούσα φάση, οι διαδικασίες συχνά αγνοούνται.
- Τα μέλη δοκιμάζουν την ανοχή του συστήματος και του ηγέτη.
- Ο αρχηγός της ομάδας κατευθύνει.



Σχήμα 14: Το μοντέλο Tuckman

1.7.2 Σύγκρουση (Storming)

Στο στάδιο της σύγκρουσης, τα μέλη αρχίζουν να αμφισβητούν τα καθορισμένα όρια. Υπάρχει η πιθανότητα να προκύψουν συγκρούσεις ή προστριβές μεταξύ των μελών της ομάδας, λόγω ασυμφωνίας των χαρακτήρων ή λόγω διαφωνίας πάνω στον τρόπο εργασίας. Σε αυτό το στάδιο τα μέλη της ομάδας μπορεί να αμφισβητήσουν την εξουσία ή τον τρόπο διαχείρισης, ή ακόμα και την αποστολή της ομάδας. Εάν η ομάδα δεν τεθεί υπό έλεγχο, θα μετατραπεί σε πεδίο εντάσεων και συγκρούσεων. Επίσης, αν δεν έχουν ξεκαθαριστεί ακόμη οι ρόλοι και οι ευθύνες, τα άτομα μπορεί να αρχίσουν να αισθάνονται αδικημένοι από τον φόρτο εργασίας τους ή να απογοητευτούν από την έλλειψη προόδου. Το στάδιο της σύγκρουσης χαρακτηρίζεται από:

- Έλλειψη συμφωνίας όσον αφορά τη λήψη αποφάσεων στην ομάδα. Τα μέλη της ομάδας προσπαθούν να προσδιορίσουν τον εαυτό τους και τη θέση τους σε σχέση με άλλα μέλη της ομάδας και τον αρχηγό, οι οποίοι ενδέχεται να λάβουν προκλήσεις από τα μέλη της ομάδας.
- Η σαφήνεια του σκοπού της ομάδας αυξάνεται, αλλά εξακολουθούν να υπάρχουν πολλές αβεβαιότητες.
- Σχηματίζονται κλίκες και φατρίες. Αυτό μπορεί να οδηγήσει σε αγώνες εξουσίας. Η ομάδα θα πρέπει να επικεντρωθεί στους στόχους της και να αποφύγει την απόσπαση της προσοχής από σχέσεις και συναισθηματικά ζητήματα.
- Ενδέχεται να απαιτούνται συμβιβασμοί για να καταστεί δυνατή η πρόοδος.
- Ο αρχηγός της ομάδας καθοδηγεί.

1.7.3 Κανονικοποίηση (Norming)

Σταδιακά, η ομάδα προχωράει στο στάδιο της κανονικοποίησης. Τα μέλη αρχίζουν να επιλύουν τις διαφορές τους, να εκτιμούν τα δυνατά σημεία των άλλων μελών και να σέβονται την εξουσία του

αρχηγού. Γνωρίζοντας πλέον καλύτερα ο ένας τον άλλον, τα μέλη της ομάδας αισθάνονται πιο άνετα ώστε να ζητούν βοήθεια και να προσφέρουν εποικοδομητικά σχόλια. Μοιράζονται μια ισχυρότερη δέσμευση για την υλοποίηση των στόχων της ομάδας και σημειώνουν ικανοποιητική πρόοδο. Το στάδιο της κανονικοποίησης χαρακτηρίζεται από:

- Η συμφωνία και η συναίνεση διαμορφώνονται πλέον σε μεγάλο βαθμό μεταξύ της ομάδας, τα μέλη της οποίας ανταποκρίνονται θετικά στην αρωγή του αρχηγού.
- Οι ρόλοι και οι ευθύνες είναι σαφείς και αποδεκτές.
- Οι μεγάλες αποφάσεις λαμβάνονται με ομαδική συμφωνία. Μικρότερες αποφάσεις μπορούν να ανατεθούν σε μεμονωμένα άτομα ή μικρές ομάδες εντός της ομάδας.
- Η αφοσίωση και η ενότητα έχουν ισχυροποιηθεί. Η ομάδα μπορεί να συμμετέχει σε διασκεδαστικές και κοινωνικές δραστηριότητες.
- Η ομάδα αναπτύσσει τις διαδικασίες και τον τρόπο εργασίας της μέσω συζήτησης.
- Υπάρχει γενικός σεβασμός για τον αρχηγό και οι ευθύνες της ηγεσίας μοιράζονται πλέον σε όλη την ομάδα.
- Ο αρχηγός βοηθά και διευκολύνει.

1.7.4 Απόδοση (Performing)

Στη φάση της απόδοσης, η ομάδα αποκτά ροή και αποδίδει στο μέγιστο των δυνατοτήτων της. Με σκληρή δουλειά και δομημένες διαδικασίες, η ομάδα βαδίζει με αποτελεσματικότητα προς την επίτευξη των στόχων της. Οι διαφορές μεταξύ των μελών εκτιμώνται και χρησιμοποιούνται για να ενισχύσουν την απόδοση της ομάδας. Η φάση της απόδοσης χαρακτηρίζεται από:

- Αυξημένη στρατηγική ευαισθητοποίηση της ομάδας. Είναι πλέον σαφές γιατί η ομάδα κάνει αυτό που κάνει.
- Το κοινό όραμα της ομάδας. Η ομάδα είναι πλέον ανεξάρτητη και δεν χρειάζεται παρέμβαση ή συμμετοχή από τον ηγέτη.
- Εστίαση στην επίτευξη στόχων και λήψη των περισσοτέρων αποφάσεων βάσει κριτηρίων που έχουν συμφωνηθεί με τον αρχηγό. Η ομάδα έχει υψηλό βαθμό αυτονομίας.
- Διαφωνίες. Ωστόσο, τώρα επιλύονται θετικά μέσα στην ομάδα και γίνονται οι απαραίτητες αλλαγές στις διαδικασίες και τη δομή.
- Η ομάδα εργάζεται για την επίτευξη του στόχου και την αντιμετώπιση θεμάτων σχέσεων, στυλ και διεργασιών τα οποία προκύπτουν στην πορεία.
- Τα μέλη της ομάδας φροντίζουν το ένα το άλλο.
- Η ομάδα απαιτεί την ανάθεση καθηκόντων και έργων από τον αρχηγό.
- Η ομάδα δεν χρειάζεται να λάβει οδηγίες ή βοήθεια. Τα μέλη της ομάδας μπορεί να ζητήσουν βοήθεια από τον αρχηγό με προσωπική και διαπροσωπική ανάπτυξη.
- Ο αρχηγός αναθέτει και επιβλέπει.

1.7.5 Διακοπή (Adjourning)

Τέλος, η φάση της διακοπής αποτελεί περισσότερο συμπλήρωμα του αρχικού μοντέλου των τεσσάρων σταδίων και όχι επέκτασή του - βλέπει την ομάδα από μια προοπτική πέρα από το σκοπό των τεσσάρων πρώτων σταδίων. Δηλαδή η συγκεκριμένη φάση είναι σχετική μόνο με τα άτομα της

ομάδας και την ευημερία τους, αλλά όχι με το κύριο καθήκον της διαχείρισης και ανάπτυξης μιας ομάδας. Σε κάθε περίπτωση, πολλές ομάδες φτάνουν σε αυτό το στάδιο φυσιολογικά. Για παράδειγμα, τα έργα τελειώνουν και οι μόνιμες ομάδες διαλύονται. Τα άτομα που τους αρέσει η ρουτίνα ή που έχουν αναπτύξει στενές εργασιακές σχέσεις με τους συναδέλφους, μπορεί να δυσκολευτούν αυτή τη στιγμή.

1.8 Ιστορίες χρηστών (User Stories) και εκτίμηση προσπάθειας (Planning Poker)

Οι ευέλικτες ομάδες σε όλο τον κόσμο χρησιμοποιούν πλέον την τεχνική της εκτίμησης προσπάθειας για την ανάπτυξη του προϊόντος τους. Αυτή η τεχνική (ονομάζεται επίσης και Scrum poker), βοηθά τις ομάδες να υπολογίσουν τον χρόνο και την προσπάθεια που απαιτείται για την ολοκλήρωση κάθε εκκρεμότητας του ανεκτέλεστου προϊόντος. Το όνομα αυτής της τεχνικής είναι εμπνευσμένο από το πόκερ επειδή οι συμμετέχοντες χρησιμοποιούν φυσικές κάρτες, οι οποίες μοιάζουν με κάρτες παιχνιδιού, και υπολογίζουν τον αριθμό των πόντων ιστορίας για κάθε εκκρεμότητα ή εργασία. Ο στόχος αυτής της διαδικασίας είναι να βοηθήσει τις εταιρείες ανάπτυξης λογισμικού να εκτιμήσουν με μεγαλύτερη ακρίβεια τα χρονοδιαγράμματα ανάπτυξης, να δημιουργήσουν συναίνεση μεταξύ των μελών της διαλειτουργικής ομάδας και να σχεδιάσουν πιο στρατηγικά την εργασία της ομάδας.

1.8.1 Απαιτήσεις

Οι Tsui και Karam ορίζουν τις απαιτήσεις ως «δηλώσεις που περιγράφουν το πως πρέπει να είναι το σύστημα λογισμικού αλλά όχι το πώς πρέπει να κατασκευαστεί» [40]. Αυτές οι δηλώσεις είναι επομένως πολύ σημαντικές, καθώς περιγράφουν τις απαιτήσεις του χρήστη από το σύστημα. Η συλλογή των απαιτήσεων με αποτελεσματικό τρόπο είναι ύψιστης σημασίας για τη διαδικασία ανάπτυξης λογισμικού, καθώς οι ελλιπείς προδιαγραφές απαιτήσεων είναι ένας από τους κορυφαίους λόγους αποτυχίας του έργου. Σε μια παραδοσιακή, μη ευέλικτη διαδικασία ο κύκλος θα προσανατολιζόταν κάπως έτσι:

- Καταγραφή όλων των απαιτήσεων.
- Ανάλυση των απαιτήσεων.
- Σχεδιασμός της λύσης.
- Κωδικοποίηση της λύσης.
- Έλεγχος.

Έχει ήδη αναφερθεί ότι σε τέτοια παραδοσιακά μοντέλα, οι πελάτες και οι χρήστες εμπλέκονται μόνο στην καταγραφή των απαιτήσεων. Στη συνέχεια εξαφανίζονται, έως ότου έρθει η ώρα να παραλάβουν το λογισμικό. Σύμφωνα με τον Cohn [41], αυτό δεν έχει ικανοποιητικά αποτελέσματα, καθώς οι πελάτες και οι χρήστες θα πρέπει να έχουν συμμετοχή σε ολόκληρο το έργο. Επομένως, δεν μπορούν να εξαφανιστούν στη μέση του έργου. Σε κάθε περίπτωση, έχει ήδη αναφερθεί σε προηγούμενες ενότητες ότι αυτά τα προβλήματα έχουν λυθεί πλέον με την έλευση των ευέλικτων μεθοδολογιών και πλαισίων. Με ποιο τρόπο όμως συλλέγονται οι απαιτήσεις; Τα τελευταία χρόνια έχει επικρατήσει μια τεχνική, η οποία χρησιμοποιεί τις ιστορίες χρηστών (user stories) για να συγκεντρώσει τις απαιτήσεις της εφαρμογής.

1.8.2 Ιστορίες Χρηστών (User Stories)

Οι ιστορίες χρηστών είναι σύντομες, απλές περιγραφές ενός χαρακτηριστικού που παρουσιάζεται από την προοπτική του ατόμου που επιθυμεί τη νέα δυνατότητα, δηλαδή του χρήστη ή του πελάτη. Αυτή η

επιθυμητή δυνατότητα γράφεται σε μια κάρτα και εκφράζεται σε καθημερινή γλώσσα, ακολουθώντας συνήθως ένα απλό πρότυπο (Σχήμα 15):

Σχήμα 15: Πρότυπο Ιστορίας Χρήστη

Οι ιστορίες χρηστών αποτελούνται από τρεις φάσεις: καταγραφή σε κάρτες (Card), συζήτηση (Conversation) και επιβεβαίωση (Confirmation). Κατά την πρώτη φάση, υπάρχει μια γραπτή περιγραφή της ιστορίας σε μια κάρτα που χρησιμοποιείται κατά τον προγραμματισμό των επαναλήψεων. Κατά τη δεύτερη φάση, δεδομένου ότι δεν περιλαμβάνονται όλες οι πληροφορίες στις κάρτες, συμπληρώνονται σταδιακά επιπλέον λεπτομέρειες με βάση τη συζήτηση. Αυτές οι συζητήσεις πραγματοποιούνται πολλές φορές κατά τη διάρκεια του έργου, όπως για παράδειγμα κατά την εκτίμηση ιστοριών και τον προγραμματισμό επανάληψης. Τέλος, κατά την τρίτη φάση, είναι πολύ σημαντικό η ομάδα και ο πελάτης να συμφωνήσουν ώστε να υπάρξει επιβεβαίωση ως προς τον χρόνο παράδοσης.

Οι κάρτες δεν περιλαμβάνουν όλες τις λεπτομέρειες σχετικά με τις ιστορίες, αλλά αποτελούν μόνο μια σύντομη περιγραφή ορισμένων λειτουργιών. Στην ουσία, το σημαντικό δεν είναι η ίδια η κάρτα, αλλά η συζήτηση που πραγματοποιείται μέσω αυτής μεταξύ του πελάτη και της ομάδας ανάπτυξης. Νέες ιστορίες μπορούν να γραφτούν ανά πάσα στιγμή καθ' όλη τη διάρκεια του έργου. Μπορεί να είναι λεπτομέρειες που εκφράζονται ως πρόσθετες ιστορίες, μια ιστορία η οποία είναι πολύ μεγάλη για να χωρέσει σε μία μόνο επανάληψη ή απλά η προσθήκη νέων χαρακτηριστικών. Οι ιστορίες χρηστών είναι γραμμένες σε φυσική γλώσσα και επομένως είναι εύκολα κατανοητές, τόσο για τους πελάτες όσο και για τους προγραμματιστές. Δεδομένου ότι δεν υπάρχουν όλες οι πληροφορίες στην κάρτα ιστορίας, η προφορική επικοινωνία είναι πολύ σημαντική και επομένως ενθαρρύνεται.

Μερικές φορές είναι απαραίτητο να χωριστεί μια ιστορία χρήστη σε δύο ή περισσότερες ιστορίες. Η ιστορία μπορεί να είναι πολύ μεγάλη για εφαρμογή κατά τη διάρκεια ενός σπριντ. Μπορεί επίσης να είναι απαραίτητο να χωριστεί μια ιστορία εάν απαιτείται μια πιο ακριβής εκτίμηση. Συνεπώς, στα μεγάλα έργα, μπορεί να υπάρχουν εκατοντάδες ή χιλιάδες ιστορίες χρηστών και αυτό συνεπάγεται μεγάλη δυσκολία στην οργάνωσή τους.

1.8.2.1 Πλεονεκτήματα της χρήσης των ιστοριών χρηστών

Οι ιστορίες χρηστών μειώνουν τον χρόνο που αφιερώνεται στη σύνταξη της εξαντλητικής τεκμηρίωσης, δίνοντας περισσότερο έμφαση στις πελατοκεντρικές συνομιλίες. Κατά συνέπεια, οι ιστορίες χρηστών επιτρέπουν στις ομάδες να παρέχουν ποιοτικό λογισμικό πιο γρήγορα, κάτι που

σαφώς προτιμούν οι πελάτες. Υπάρχουν αρκετά οφέλη που προκύπτουν από την υιοθέτηση της προσέγγισης ιστοριών χρηστών στην ευέλικτη ανάπτυξη, όπως:

- Οι ιστορίες εστιάζουν στον χρήστη. Μπορεί η λίστα των εκκρεμών εργασιών να είναι αυτή που διατηρεί την ομάδα επικεντρωμένη στις εργασίες που πρέπει να ελεγχθούν, αλλά μια συλλογή ιστοριών διατηρεί την ομάδα επικεντρωμένη στην επίλυση προβλημάτων που αφορούν πραγματικούς χρήστες.
- Η χρήση των ιστοριών ενθαρρύνουν τη συνεργασία. Με τον τελικό στόχο καθορισμένο, η ομάδα συνεργάζεται ώστε να επιτύχει τον στόχο και να αποφασίσει το πώς θα εξυπηρετήσει καλύτερα τον χρήστη.
- Οι ιστορίες οδηγούν σε δημιουργικές λύσεις. Οι ιστορίες ενθαρρύνουν την ομάδα να σκεφτεί κριτικά και δημιουργικά, ώστε να βρεθούν οι καλύτερες λύσεις.
- Οι ιστορίες αυξάνουν τη δυναμική της ομάδας. Με κάθε ιστορία που η ομάδα ανάπτυξης καταπιάνεται, απολαμβάνει μικρές προκλήσεις και μια μικρή νίκη. Έτσι, σταδιακά η ομάδα αποκτά αυξημένη δυναμική και καλύτερη ψυχολογία για την συνέχεια του έργου.

1.8.2.2 Ακρωνύμιο INVEST (Independent, Negotiable, Valuable, Estimable, Small, Testable)

Το ακρωνύμιο INVEST αποτελεί έναν πρακτικό τρόπο να μας θυμίζει ένα ευρέως αποδεκτό σύνολο κριτηρίων, ή μια λίστα ελέγχου, για την αξιολόγηση της ποιότητας μιας ιστορίας χρήστη. Εάν η ιστορία δεν πληροί ένα από αυτά τα κριτήρια, η ομάδα ίσως θα πρέπει να την επαναδιατυπώσει ή ακόμη και να εξετάσει το ενδεχόμενο επανεγγραφής (που συχνά μεταφράζεται σε φυσική καταστροφή της παλιάς κάρτας ιστορίας και σύνταξη μιας νέας).

Με βάση λοιπόν το ακρωνύμιο INVEST, μια ιστορία είναι καλή όταν είναι:

- **Ανεξάρτητη (Independent) :** Θα πρέπει να είναι αυτόνομη, δηλαδή να μην εξαρτάται από κάποια άλλη ιστορία.
- **Διαπραγματεύσιμη (Negotiable) :** Αρχικά πρέπει να αποτυπώνεται μόνο η ουσία των αναγκών του χρήστη, αφήνοντας έτσι χώρο για συνομιλία. Η ιστορία χρήστη δεν πρέπει να γράφεται σαν συμβόλαιο.
- **Πολύτιμο (Valuable) :** Επιστρέφει την αξία στον τελικό χρήστη.
- **Εκτιμήσιμη (Estimable) :** Οι ιστορίες χρηστών θα πρέπει να μπορούν να εκτιμηθούν, ώστε να μπορούν να δοθούν προτεραιότητες και να ενταχθούν σε σπριντ.
- **Μικρή (Small) :** Η ιστορία χρήστη θα πρέπει να είναι ένα μικρό κομμάτι εργασίας, ώστε να μπορεί να ολοκληρωθεί σε ένα χρονικό περιθώριο τριών ή τεσσάρων ημερών.
- **Να μπορεί να δοκιμαστεί (Testable) :** Μια ιστορία χρήστη θα πρέπει να μπορεί να δοκιμαστεί και να επιβεβαιωθεί.

Μόλις γραφτεί μια ιστορία, ενσωματώνεται στη ροή εργασιών. Κατά τη διάρκεια μιας συνάντησης σπριντ, η ομάδα αποφασίζει ποιες ιστορίες θα αναλύσει σε αυτό το σπριντ. Οι ομάδες συζητούν πάνω στις απαιτήσεις και τη λειτουργικότητα που απαιτεί κάθε ιστορία χρήστη και βαθμολογούν τις ιστορίες με βάση την πολυπλοκότητά τους ή το χρόνο που απαιτείται για την ολοκλήρωσή τους. Οι ομάδες χρησιμοποιούν την τεχνική planning poker για να κάνουν σωστές εκτιμήσεις. Θα δούμε παρακάτω περισσότερα πάνω σε αυτή την τεχνική.

1.8.2.3 Αντίλογος

Η τεχνική των ιστοριών των χρηστών γίνεται ολοένα και δημοφιλέστερη στις εταιρείες που χρησιμοποιούν ευέλικτες μεθοδολογίες. Παρόλα αυτά όμως, δεν συμφωνούν όλοι με την χρηστικότητά τους. Ο Coplien [42] για παράδειγμα είναι αρκετά επικριτικός όσον αφορά τις ιστορίες χρηστών, καθώς πιστεύει ότι οι περισσότερες από αυτές δεν έχουν ούτε χρήστη ούτε ιστορία, ενώ ο απλοϊκός τρόπος χρήσης του ψευδοκώδικα δεν αποτελεί επαγγελματική προσέγγιση. Ένας ακόμη που ασκεί κριτική στη χρήση των ιστοριών χρηστών είναι ο Dan North, ο οποίος πιστεύει ότι ο διαχωρισμός ενός προβλήματος σε μια σειρά από ιστορίες αλλάζει την εστίαση από την επίλυση του προβλήματος στην παράδοση μεγάλου αριθμού ιστοριών.

1.8.3 Εκτίμηση προσπάθειας (Planning Poker)

Η αποτελεσματική εκτίμηση είναι μια από τις πιο δύσκολες προκλήσεις που αντιμετωπίζουν οι προγραμματιστές λογισμικού στη δουλειά τους. Ανεξάρτητα από το μέγεθός της, η ομάδα θα πρέπει να καθορίσει, να εκτιμήσει και να διανέμει τις εργασίες. Καθώς οι ομάδες μεγαλώνουν, καθίσταται ακόμη πιο σημαντικό να οικοδομήσουμε καλές συνήθειες γύρω από το σχεδιασμό και την εκτίμηση της εργασίας. Η έλλειψη προγραμματισμού και εκτίμησης μειώνει την εμπιστοσύνη σε ένα πρόγραμμα, καταστρέφει τις σχέσεις μεταξύ της ομάδας και της επιχείρησης και κάνει την ανάπτυξη πιο δύσκολη για όλους.

Αρχικά ο ιδιοκτήτης του προϊόντος περιγράφει στην ομάδα ανάπτυξης την ιστορία χρήστη. Στη συνέχεια η ομάδα θα έχει την ευκαιρία να υποβάλει ερωτήσεις και να συζητήσει τις τεχνικές πτυχές πάνω στη δοθείσα ιστορία. Κάθε μέλος θα επιλέξει μια κάρτα που αντιστοιχεί στον αριθμό των πόντων που πιστεύουν ότι αντιπροσωπεύει καλύτερα το μέγεθος της ιστορίας χρήστη. Όλα τα μέλη θα πρέπει να δείξουν ταυτόχρονα τις κάρτες τους και τα μέλη με τις υψηλότερες και χαμηλότερες εκτιμήσεις θα πρέπει να δικαιολογήσουν τις επιλογές τους. Μετά την περίοδο αιτιολόγησης, η ομάδα θα ψηφίσει για άλλη μια φορά ώστε να προσπαθήσει να επιτύχει συναίνεση, επαναλαμβάνοντας τη διαδικασία όπως απαιτείται. Ο ρόλος του Scrum Master, ή του συντονιστή της συνάντησης, είναι να διασφαλίσει ότι η συζήτηση είναι δομημένη, ότι επιτυγχάνεται συναίνεση εντός εύλογου χρονικού διαστήματος και ότι δεν ασκείται επιρροή στην απόφαση της ομάδας. Η διαδικασία θα πρέπει να επαναληφθεί έως ότου η ομάδα κρίνει ότι έχουν φτάσει στο προκαθορισμένο χρονικό όριο, το οποίο μπορεί να είναι η διάρκεια ενός σπριντ, προσθέτοντας τα σημεία κάθε ιστορίας χρήστη που αποφασίστηκαν. Όσο περισσότερο συνεργάζεται μια ομάδα, τόσο καλύτερα θα κατανοεί τη διαδικασία και θα αυξάνει την ικανότητα και την ταχύτητά της.

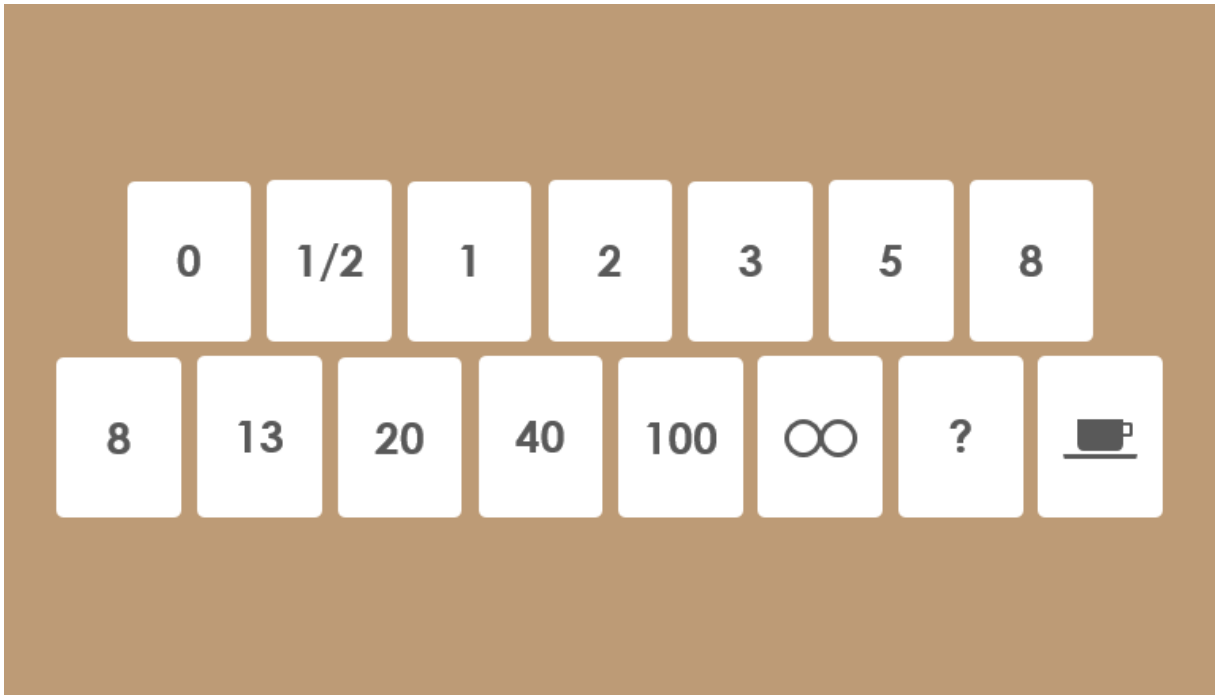
1.8.3.1 Βήματα που ακολουθούνται στην τεχνική του Planning Poker

Μια ομάδα ευέλικτης μεθοδολογίας που επιθυμεί να εκτελέσει μια συνεδρία Planning Poker, θα πρέπει να ακολουθήσει τα παρακάτω βήματα:

- Κάθε μέλος της ομάδας μοιράζεται τις ίδιες κάρτες εκτίμησης προσπάθειας. Κάθε κάρτα έχει έναν αριθμό (οι αριθμοί βασίζονται συνήθως στην ακολουθία Fibonacci,) συν μια κάρτα που απεικονίζει ένα φλιτζάνι καφέ και μία κάρτα με ένα σύμβολο ερωτηματικού. Μέλος της ομάδας θεωρείται οποιοσδήποτε συμβάλει στην ανάπτυξη του έργου.
- Στη συνέχεια, ο ιδιοκτήτης του προϊόντος (ή πιθανώς ένας διαχειριστής προϊόντων) διαβάζει δυνατά κάθε ιστορία χρήστη στην ομάδα.
- Πλέον, όλοι έχουν ακούσει την ιστορία και έτσι η ομάδα ξεκινάει τη συζήτηση. Οι συμμετέχοντες θα πρέπει περιγράψουν πώς οραματίζονται να χειριστούν το έργο, πόσα άτομα

εκτιμούν ότι θα πρέπει να εμπλακούν, ποια σετ δεξιοτήτων θα απαιτηθούν και τι θα συμβεί αν υπάρχουν εμπόδια που επιβραδύνουν την πρόοδο. Επίσης, στο σημείο αυτό, τα μέλη της ομάδας μπορούν να θέσουν τις ερωτήσεις τους σχετικά με την ιστορία χρήστη που εξετάζεται.

- Αφού όλοι καταταλάξουν στην άποψη τους και τους απαντηθούν τυχόν ερωτήσεις, κάθε άτομο επιλέγει κρυφά μια κάρτα, η οποία αντιπροσωπεύει την εκτίμησή του. Όταν όλοι είναι έτοιμοι, όλοι οι συμμετέχοντες αποκαλύπτουν ταυτόχρονα τις κάρτες τους. Όσο υψηλότερη είναι η κάρτα ενός συμμετέχοντα, τόσο πιο δύσκολο είναι να ολοκληρώσει ο συμμετέχων την ιστορία.
- Εάν όλοι οι συμμετέχοντες προτείνουν την ίδια κάρτα, τότε αυτός ο αριθμός θεωρείται το κοινό σημείο και η ομάδα μπορεί να προχωρήσει στην επόμενη ιστορία. Αν όμως τα φύλλα διαφέρουν, τότε η ομάδα συνεχίζει τη συζήτησή της για την ιστορία. Εκείνοι με τις υψηλότερες ή τις χαμηλότερες εκτιμήσεις από την υπόλοιπη ομάδα θα πρέπει να εξηγήσουν τη συλλογιστική τους και να προσπαθήσουν να πείσουν τους συναδέλφους τους να στηρίξουν τη θέση τους. Όταν η ομάδα ολοκληρώσει αυτόν τον νέο γύρο συζήτησης, όλοι θα επανεξετάσουν ξανά τις κάρτες τους και είτε θα διατηρήσουν την προηγούμενη επιλογή τους ή θα επιλέξουν μια νέα. Όλοι οι συμμετέχοντες θα αποκαλύψουν ξανά τις κάρτες τους ταυτόχρονα.
- Επαναλαμβάνεται το προηγούμενο βήμα έως ότου επέλθει η συναίνεση. Όταν γίνει αυτό, η ομάδα περνάει στην επόμενη ιστορία χρήστη.



Σχήμα 16: Τυπικές κάρτες εκτίμησης προσπάθειας

1.8.3.2 Κάρτες εκτίμησης προσπάθειας

Οι κάρτες που χρησιμοποιούνται συχνότερα στο planning poker περιέχουν αριθμούς από την ακολουθία Fibonacci (0, 1, 2, 3, 5, 8, 13, 21, 55, 89) ή μια παρόμοια απλοποιημένη εξέλιξη (0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100). Θεωρείται επίσης κοινή τακτική να χρησιμοποιούνται κάποιες επιπλέον κάρτες- μια κάρτα με ένα ερωτηματικό ώστε να αποδώσει μια αβέβαιη γνώμη ή ότι το στοιχείο δεν ήταν καλά καθορισμένο και πρέπει να γίνει πιο λεπτομερές. Η κάρτα με το σύμβολο του απείρου, αντιπροσωπεύει ένα στοιχείο που είναι πολύ μεγάλο για να ολοκληρωθεί και πρέπει να χωριστεί σε

μικρότερες εργασίες, ενώ η κάρτα που απεικονίζει ένα φλιτζάνι καφέ αποτελεί αίτημα για σύντομη παύση (Σχήμα 16). Κάθε μέλος της ομάδας χρειάζεται ένα πλήρες σύνολο καρτών.

Η ακολουθία Fibonacci είναι μια σειρά αριθμών που προέρχονται προσθέτοντας τους δύο προηγούμενους αριθμούς. Εμφανίζεται συχνά στη φύση, όπως στη σπειροειδή μορφή των λουλουδιών. Πήρε το όνομά της από τον Ιταλό μαθηματικό Leonardo Bonacci, γνωστό ως Fibonacci, αλλά αναγνωρίστηκε για πρώτη φορά στην Ινδία. Η χρήση του στην εκτίμηση είναι σχεδόν αυθαίρετη, αλλά η δημοτικότητά της έχει κάποιες σημαντικές αιτίες:

- **Η διαφορά μεταξύ των αριθμών αυξάνεται με την πάροδο του χρόνου**. Αυτό ταιριάζει με την πρακτική των προγραμματιστών και των υπευθύνων έργων, καθώς γνωρίζουν ότι οι λειτουργίες που είναι πιο περίπλοκες απαιτούν περισσότερο χρόνο για να υλοποιηθούν.
- **Η σχέση μεταξύ των παρακείμενων αριθμών είναι συνεπής**. Οι προγραμματιστές έχουν πολλές λεπτομέρειες με τις οποίες θα πρέπει να ασχοληθούν κατά την εκτίμηση και αν οι παρακείμενοι αριθμοί δεν έχουν συνοχή, τότε προκαλείται σύγχυση.
- **Η ακολουθία Fibonacci είναι σαφώς άσχετη με το χρόνο**. Ενώ οι άνθρωποι έχουν μια αίσθηση για την οποία οι αριθμοί σχετίζονται με το χρόνο, στην πραγματικότητα η ακολουθία Fibonacci διαφέρει από αυτή τη λογική. Αυτό διευκολύνει τη διεξαγωγή συνομιλιών σχετικά με τα σημεία ιστορίας.

1.8.3.3 Πλεονεκτήματα και μειονεκτήματα του Planning Poker

Για τις ευέλικτες ομάδες, η εκτίμηση προσπάθειας αποτελεί ένα εξαιρετικό εργαλείο για την εκτίμηση και την ιεράρχηση των λειτουργιών που πηγάζουν από τις ιστορίες χρηστών. Κάποια από τα οφέλη που αποκομίζουν οι ομάδες με την χρήση αυτής της τακτικής είναι:

- Το πλαίσιο παιχνιδιού ενθαρρύνει τη συνεργασία και τη δημιουργία ομάδων.
- Η απαίτηση συναίνεσης μεταξύ της ομάδας, αντί υπάρχει ένα άτομο το οποίο υπαγορεύει εκτιμήσεις, μπορεί να αυξήσει το ηθικό δίνοντας σε όλους στην ομάδα το δικαίωμα ψήφου.
- Η μορφή της συνομιλίας μπορεί να αποκαλύψει πολύτιμες γνώσεις από μέλη της ομάδας, που σε διαφορετική περίπτωση δεν θα είχαν τρόπο να μοιραστούν τις σκέψεις και τις εμπειρίες τους.

Από την άλλη, η τεχνική αυτή έχει και τα μειονεκτήματά της:

- Η επίτευξη συναίνεσης μπορεί να δώσει στην ομάδα μια ψευδή αίσθηση εμπιστοσύνης. Ενδέχεται να εξακολουθούν να λείπουν σημαντικές πληροφορίες και η συνολική εκτίμηση της ομάδας να παραμένει λανθασμένη.
- Ένα κυρίαρχο άτομο στην ομάδα μπορεί να επηρεάσει αδικαιολόγητα τους άλλους συμμετέχοντες. Εάν δεν δοθεί προσοχή, η ομάδα μπορεί να οδηγηθεί σε εκτιμήσεις που βασίζονται όχι στη συναίνεση, αλλά στη μεμονωμένη βούληση ενός μέλους.
- Συνήθως, η συνολική εκτίμηση μιας ομάδας τείνει να είναι πιο αισιόδοξη από τις επιμέρους εκτιμήσεις των μεμονωμένων μελών της. Έτσι, είναι πιθανό για μια ομάδα να παραπλανηθεί και να υπερεκτιμήσει τις δυνατότητές της, με αποτέλεσμα να φανεί ασυνεπής στο χρονοδιάγραμμα.

Τέλος, ένα ακόμη χαρακτηριστικό της ποιοτικής οργάνωσης που προσφέρει η τεχνική planning poker, είναι η υποχρεωτική χρήση των αξιολογήσεων. Διατηρείται δηλαδή ένα ιστορικό και αυτό επιτρέπει στην ομάδα ανάπτυξης να λαμβάνει υπόψη της τα προηγούμενα επιτεύγματα και λάθη.

Ως αποτέλεσμα, οι πελάτες λαμβάνουν ακριβέστερες εκτιμήσεις για τον χρόνο και τον προϋπολογισμού του έργου. Αυτό διασφαλίζεται από το γεγονός ότι όλοι οι ειδικοί που συμμετέχουν στη διαδικασία της ανάπτυξης ενός προϊόντος, καταλήγουν σε μια κοινή απόφαση.

1.9 Ενοποιημένη Γλώσσα Σχεδίασης Προτύπων (UML)

Η Ενοποιημένη Γλώσσα Μοντελοποίησης (UML) είναι μια γλώσσα μοντελοποίησης γενικής χρήσης. Ο κύριος στόχος της UML είναι να ορίσει έναν τυπικό τρόπο απεικόνισης του τρόπου με τον οποίο έχει σχεδιαστεί ένα σύστημα. Είναι αρκετά παρόμοιο με τα σχεδιαγράμματα που χρησιμοποιούνται σε άλλους τομείς της μηχανικής.

Η UML δεν είναι γλώσσα προγραμματισμού, αλλά αποτελεί περισσότερο οπτική γλώσσα. Τα διαγράμματα UML χρησιμοποιούνται ώστε να απεικονιστούν η συμπεριφορά και η δομή ενός συστήματος. Η UML βοηθά τους μηχανικούς λογισμικού, τους επιχειρηματίες και τους αρχιτέκτονες συστήματος με τη μοντελοποίηση, τον σχεδιασμό και την ανάλυση. Η Ομάδα Διαχείρισης Αντικειμένων (OMG) υιοθέτησε την Ενοποιημένη Γλώσσα Μοντελοποίησης ως πρότυπο το 1997 και έκτοτε βρίσκεται υπό τη διαχείρισή της. Ο Διεθνής Οργανισμός Τυποποίησης (ISO) δημοσίευσε την UML ως εγκεκριμένο πρότυπο το 2005. Η UML αναθεωρείται με την πάροδο των ετών και επανεξετάζεται περιοδικά.

1.9.1 Μοντελοποίηση

Η UML συνδέεται με την αντικειμενοστρεφή σχεδίαση και ανάλυση. Χρησιμοποιεί τη χρήση στοιχείων και σχηματίζει συσχετίσεις μεταξύ τους για να σχηματίσει διαγράμματα. Τα διαγράμματα στο UML μπορούν γενικά να ταξινομηθούν ως:

- **Δομικά μοντέλα UML (Structural):** Απεικονίζουν τα στοιχεία ενός συστήματος που είναι ανεξάρτητα από το χρόνο και που μεταφέρουν τις έννοιες ενός συστήματος και το πώς σχετίζονται μεταξύ τους. Τα στοιχεία σε αυτά τα διαγράμματα προσομοιάζουν στη χρήση με τα ουσιαστικά σε μια φυσική γλώσσα και οι σχέσεις που τα συνδέουν είναι δομικές ή σημασιολογικές σχέσεις. Για παράδειγμα, ένα δομικό διάγραμμα ενός συστήματος ενοικίασης οχημάτων ενδέχεται να περιέχει στοιχεία όπως «Αυτοκίνητο», «Ενοικίαση», «Άδεια Οδήγησης» και «Πιστωτική Κάρτα», καθώς και συνδέσμους που συνδέουν αυτά τα στοιχεία. Υπάρχουν επτά ειδών διαγράμματα που ταξινομούνται ως δομικά. Σε αυτά περιλαμβάνονται τα διαγράμματα κλάσεων, πακέτων και αντικειμένων.
- **Μοντέλα Συμπεριφοράς UML (Behavioral):** Απεικονίζουν τα στοιχεία ενός συστήματος που εξαρτώνται από το χρόνο και μεταφέρουν τις δυναμικές έννοιες του συστήματος, καθώς και τον τρόπο με τον οποίο σχετίζονται αυτά μεταξύ τους. Τα στοιχεία σε αυτά τα διαγράμματα μοιάζουν περισσότερο με τα ρήματα σε μια φυσική γλώσσα και οι σχέσεις που τα συνδέουν συνήθως απεικονίζουν τη ροή του χρόνου. Για παράδειγμα, ένα διάγραμμα συμπεριφοράς ενός συστήματος ενοικίασης οχήματος μπορεί να περιέχει στοιχεία όπως «Κάντε Κράτηση», «Ενοικίαση Αυτοκινήτου» και «Παρέχετε Στοιχεία Πιστωτικής Κάρτας». Και σε αυτήν την κατηγορία ταξινομούνται επτά είδη διαγράμματος, μεταξύ άλλων και τα διαγράμματα περιπτώσεων χρήσης, επικοινωνίας και ακολουθίας.

Σε κάθε περίπτωση, είτε δηλαδή μιλάμε για δομικό μοντέλο ή για μοντέλο συμπεριφοράς, όλα τα μοντέλα θα πρέπει να κατασκευάζονται βάσει των παρακάτω βασικών αρχών:

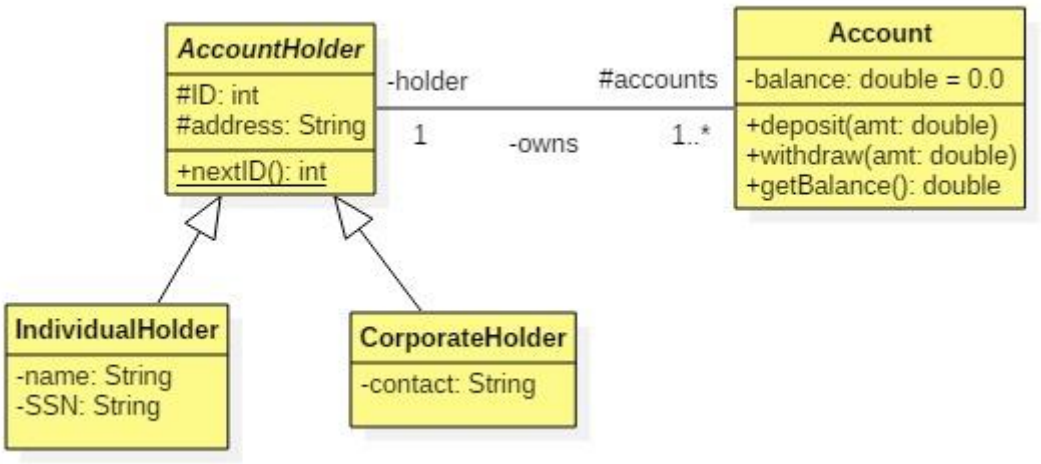
- Η κατασκευή ενός μοντέλου έχει ως κύριο στόχο την κατανόηση του συστήματος.
- Ένα μοντέλο είναι μια απλοποιημένη αναπαράσταση ενός κομματιού του πραγματικού κόσμου.
- Το ίδιο κομμάτι του πραγματικού κόσμου μπορεί να αναπαρασταθεί, ανάλογα με το τι θέλουμε να εξετάσουμε, με διαφορετικά μοντέλα.
- Το ίδιο κομμάτι του πραγματικού κόσμου μπορεί να αναπαρασταθεί σε διαφορετικά επίπεδα λεπτομέρειας.

1.9.2 Διαγράμματα UML

Όπως έχει ήδη αναφερθεί, η UML παρέχει δύο κατηγορίες διαγραμμάτων (δομικά και συμπεριφοράς), με την κάθε κατηγορία να περιέχει επτά είδη διαγραμμάτων. Παρακάτω παρατίθενται και τα δεκατέσσερα είδη διαγραμμάτων:

1.9.2.1 Διαγράμματα δομικών μοντέλων

Α. Διάγραμμα κλάσεων (Class Diagram): Αποτελούν το κύριο δομικό στοιχείο οποιασδήποτε αντικειμενοστρεφούς λύσης. Παρουσιάζει τις κλάσεις ενός συστήματος, τα χαρακτηριστικά και τις λειτουργίες της κάθε κλάσης και τη σχέση μεταξύ τους. Στα περισσότερα εργαλεία μοντελοποίησης, μια κλάση αποτελείται τρία μέρη. Το όνομα στην κορυφή, οι ιδιότητες στη μέση και οι λειτουργίες ή οι μέθοδοι στο κάτω μέρος. Σε ένα μεγάλο σύστημα με πολλές συναφείς κλάσεις, οι κλάσεις ομαδοποιούνται για να δημιουργήσουν διαγράμματα κλάσεων. Διαφορετικές σχέσεις μεταξύ κλάσεων εμφανίζονται με διαφορετικούς τύπους βελών. Στο σχήμα 17 παρουσιάζεται ένα παράδειγμα ενός διαγράμματος κλάσεων:



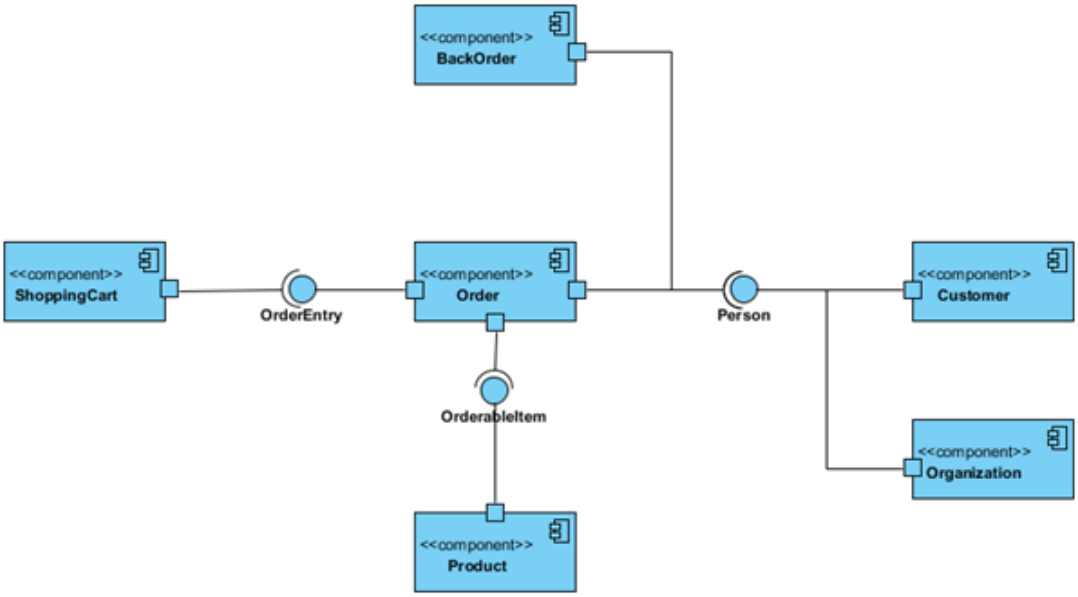
Σχήμα 17: Διάγραμμα κλάσεων

Βλέπουμε στο παραπάνω σχήμα ότι οι κλάσεις συνδέονται μεταξύ τους με διάφορους τρόπους. Στην ουσία, οι σχέσεις στα διαγράμματα κλάσεων αποτελούν διαφορετικούς τύπους λογικών συνδέσεων. Παρακάτω, αναφέρονται οι τύποι λογικών συνδέσεων που χρησιμοποιούνται στα διαγράμματα κλάσεων:

- **Συσχέτιση (Association):** Αποτελεί τον ευρύτερο όρο που περιλαμβάνει σχεδόν οποιαδήποτε λογική σύνδεση ή σχέση μεταξύ κλάσεων. Απεικονίζεται με μια γραμμή που ενώνει τις κλάσεις.

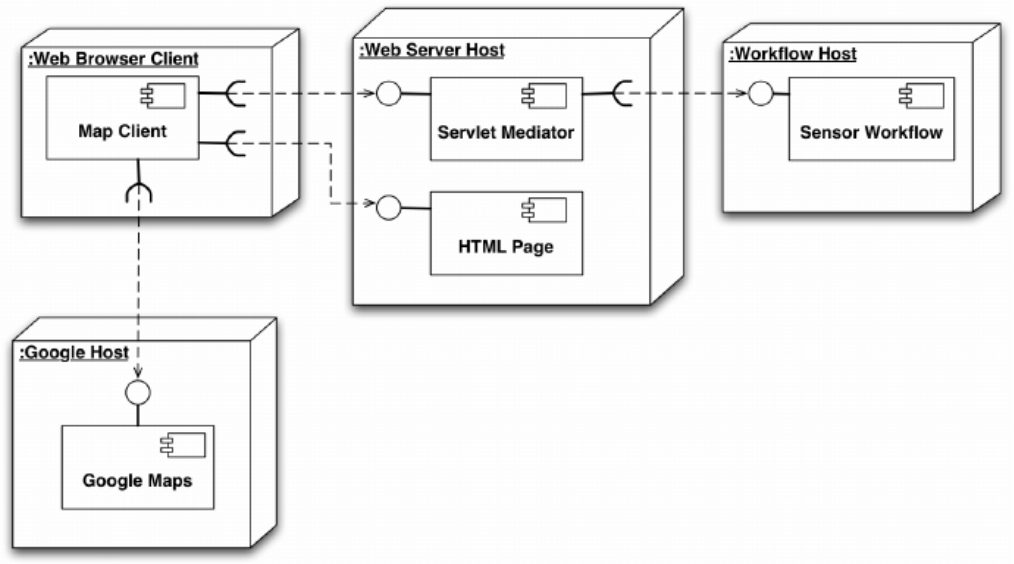
- **Μονής κατεύθυνσης (Directed Association):** Αναφέρεται σε μια κατευθυντική σχέση που αντιπροσωπεύεται από μια γραμμή με ένα βέλος. Το βέλος απεικονίζει την κατευθυνόμενη ροή ανάμεσα στις κλάσεις.
- **Ανακλαστική ένωση (Reflexive Association):** Χρησιμοποιείται όταν μια κλάση έχει πολλαπλές λειτουργίες ή ευθύνες. Για παράδειγμα, ένα μέλος του προσωπικού που εργάζεται σε ένα αεροδρόμιο μπορεί να είναι πιλότος, μηχανικός αεροπορίας, αποστολέας εισιτηρίων, φρουρός ή μέλος του πληρώματος συντήρησης. Απεικονίζεται από μια γραμμή που ξεκινάει και καταλήγει στην ίδια κλάση, χωρίς να συσχετιστεί με άλλη κλάση.
- **Πολλαπλότητα (Multiplicity):** Αποτελεί τον ενεργό λογικό συσχετισμό μιας κλάσης σε σχέση με μια άλλη. Δηλαδή, ένας στόλος μπορεί να περιλαμβάνει πολλά αεροπλάνα, ενώ ένα εμπορικό αεροπλάνο μπορεί να περιέχει μηδέν έως πολλούς επιβάτες. Για παράδειγμα ,η σημείωση «0 .. *» σημαίνει "από κανένα έως πολλά", ενώ αν και στη συσχέτισή τους οι κλάσεις έχουν έναν αστερίσκο (*) λέμε ότι η σχέση είναι «πολλά προς πολλά».
- **Συσσωμάτωση (Aggregation):** Αναφέρεται στο σχηματισμό μιας συγκεκριμένης κλάσης ως αποτέλεσμα μιας άλλης κλάσης η οποία δημιουργείται ως συλλογή. Για παράδειγμα, η κλάση «βιβλιοθήκη» αποτελείται από ένα ή περισσότερα βιβλία, μεταξύ άλλων υλικών. Συνολικά, οι υπαγόμενες κλάσεις δεν εξαρτώνται σε μεγάλο βαθμό από την ευρύτερη. Στο ίδιο παράδειγμα, τα βιβλία θα παραμείνουν σαν κλάση ακόμη και αν διαλυθεί η βιβλιοθήκη. Για να απεικονιστεί η συσσωμάτωση σε ένα διάγραμμα, πρέπει να σχεδιαστεί μια γραμμή από τη ευρύτερη κλάση στην υπαγόμενη κλάση, με ένα σχήμα ρόμβου (μόνο οι γραμμές του ρόμβου, χωρίς γέμισμα) να εφάπτεται στην ευρύτερη κλάση.
- **Σύνθεση (Composition):** Είναι παρόμοια με τη σχέση συσσωμάτωσης- η μόνη διαφορά είναι ότι ο βασικός σκοπός της είναι να δοθεί έμφαση στην εξάρτηση της υπαγόμενης κλάσης. Δηλαδή, η υπαγόμενη κλάση θα εξαλειφθεί και η ίδια αν καταστραφεί η ευρύτερη κλάση. Για παράδειγμα, η τσέπη ενός παντελονιού θα πάψει επίσης να υπάρχει μόλις καταστραφεί το παντελόνι. Για να απεικονιστεί μια σχέση σύνθεσης σε ένα διάγραμμα UML, θα πρέπει να χρησιμοποιηθεί μια γραμμή κατεύθυνσης που συνδέει τις δύο κλάσεις, με ένα γεμάτο σχήμα ρόμβου το οποίο εφάπτεται στη ευρύτερη κλάση.
- **Κληρονομικότητα / Γενίκευση (Inheritance / Generalization):** Αναφέρεται στον τύπο σχέσης όπου μια συνδεδεμένη κλάση είναι παιδί της άλλης, λόγω της ανάληψης των ίδιων λειτουργιών της μητρικής κλάσης. Με άλλα λόγια, η κλάση παιδί είναι ένας συγκεκριμένος τύπος της γονικής κλάσης. Για να εμφανιστεί η κληρονομικότητα σε ένα διάγραμμα UML, θα πρέπει να σχεδιαστεί μια σταθερή γραμμή (όχι διακεκομμένη) από την κλάση παιδί προς την κλάση γονέα, με ένα βέλος να εφάπτεται στην κλάση γονέα.
- **Πραγματοποίηση (Realization):** υποδηλώνει την υλοποίηση της λειτουργικότητας που ορίζεται σε μια κλάση από μια άλλη κλάση. Για να απεικονιστεί η σχέση πραγματοποίησης στο UML θα πρέπει να συνδεθούν με μια διακεκομμένη γραμμή η οποία καταλήγει σε ένα βέλος , το οποίο εφάπτεται στην κλάση που εφαρμόζει τη συνάρτηση.

Β. Διάγραμμα Συστατικών (Component Diagram): Ένα διάγραμμα συστατικών εμφανίζει τη δομική σχέση των συστατικών ενός συστήματος λογισμικού. Αυτού του είδους των διαγραμμάτων χρησιμοποιούνται κυρίως στη σχεδίαση πολύπλοκων συστημάτων τα οποία περιέχουν πολλά στοιχεία. Τα στοιχεία επικοινωνούν μεταξύ τους χρησιμοποιώντας διεπαφές, ενώ οι διεπαφές συνδέονται με τη χρήση συνδέσμων. Στο σχήμα 18 παρουσιάζεται ένα παράδειγμα διαγράμματος συστατικών:



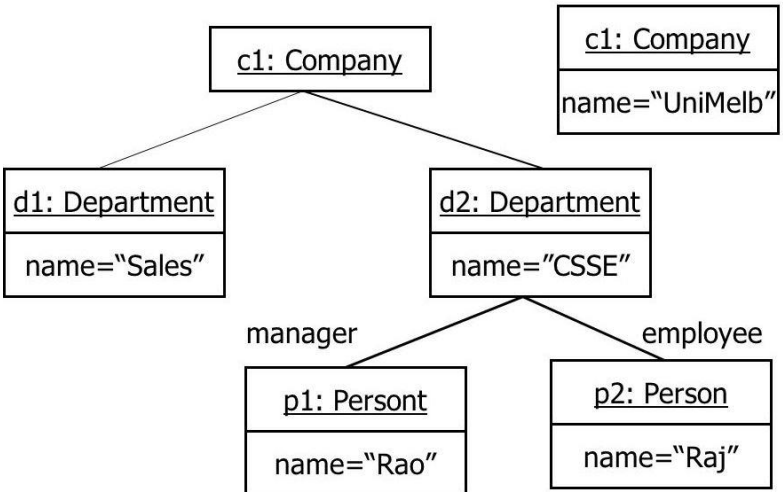
Σχήμα 18: Διάγραμμα συστατικών

Γ. Διάγραμμα ανάπτυξης (Deployment diagram): Ένα διάγραμμα ανάπτυξης απεικονίζει το υλικό του συστήματος και το λογισμικό που υπάρχει σε αυτό το υλικό. Τα διαγράμματα ανάπτυξης είναι χρήσιμα όταν το έργο αναπτύσσεται σε πολλούς υπολογιστές, με κάθε έναν να έχει μια μοναδική διαμόρφωση. Ακολουθεί ένα παράδειγμα διαγράμματος ανάπτυξης (Σχήμα 19):



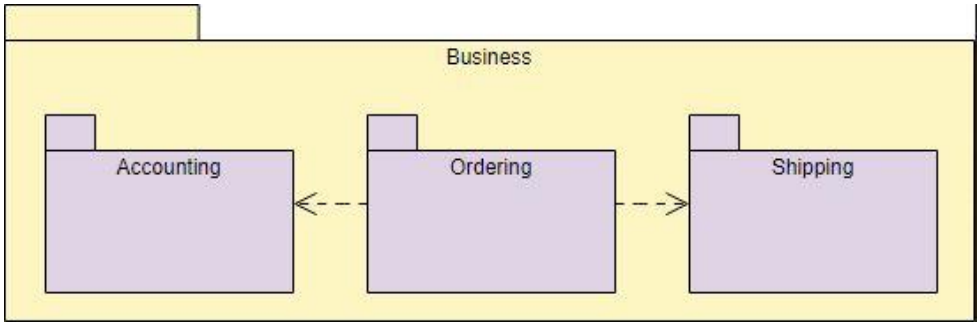
Σχήμα 19: Διάγραμμα ανάπτυξης

Δ. Διάγραμμα αντικειμένων (Object Diagram): Τα διαγράμματα αντικειμένων μοιάζουν πολύ με τα διαγράμματα κλάσης. Όπως τα διαγράμματα τάξης συσχετίζουν τις κλάσεις μεταξύ τους, έτσι και τα διαγράμματα αντικειμένων απεικονίζουν τις σχέσεις μεταξύ των αντικειμένων. Χρησιμοποιούν παραδείγματα του πραγματικού κόσμου και δείχνουν πώς θα μοιάζει ένα σύστημα σε μια δεδομένη στιγμή. Το σχήμα 20 παρουσιάζει ένα παράδειγμα διαγράμματος αντικειμένων:



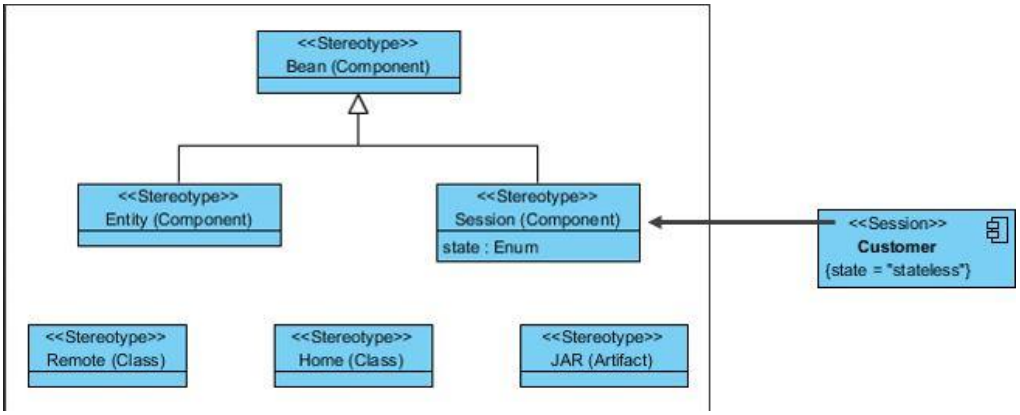
Σχήμα 20: Διάγραμμα αντικειμένων

Ε. Διάγραμμα πακέτων (Package Diagram): Όπως υποδηλώνει το όνομα, ένα διάγραμμα πακέτων δείχνει τις συσχετίσεις μεταξύ διαφορετικών πακέτων σε ένα σύστημα (Σχήμα 21).



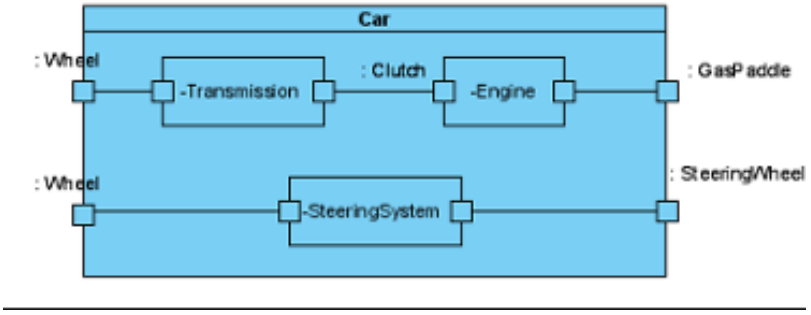
Σχήμα 21: Διάγραμμα πακέτων

ΣΤ. Διάγραμμα προφίλ (Profile Diagram): Το διάγραμμα προφίλ είναι σχετικά ένας νέος τύπος διαγράμματος που εισήχθη στη UML 2. Παρέχει έναν γενικό μηχανισμό επέκτασης για την προσαρμογή των μοντέλων UML για συγκεκριμένους τομείς και πλατφόρμες. Τα προφίλ καθορίζονται χρησιμοποιώντας στερεότυπα, ορισμούς τιμών με ετικέτες και περιορισμούς που εφαρμόζονται σε συγκεκριμένα στοιχεία μοντέλου, όπως κλάσεις, χαρακτηριστικά, λειτουργίες και δραστηριότητες.



Σχήμα 22: Διάγραμμα προφίλ

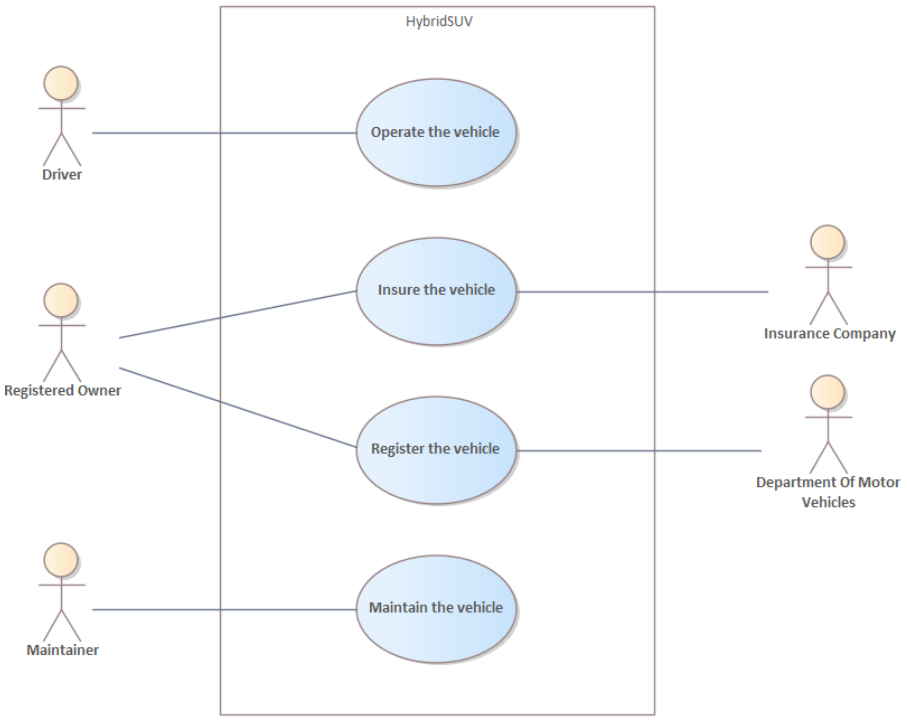
Ζ. Διάγραμμα σύνθετης δομής (Composite Structure Diagram): Τα διαγράμματα σύνθετης δομής χρησιμοποιούνται για να δείξουν την εσωτερική δομή μιας κλάσης.



Σχήμα 23: Διάγραμμα σύνθετης δομής

1.9.2.2 Διαγράμματα μοντέλων συμπεριφοράς

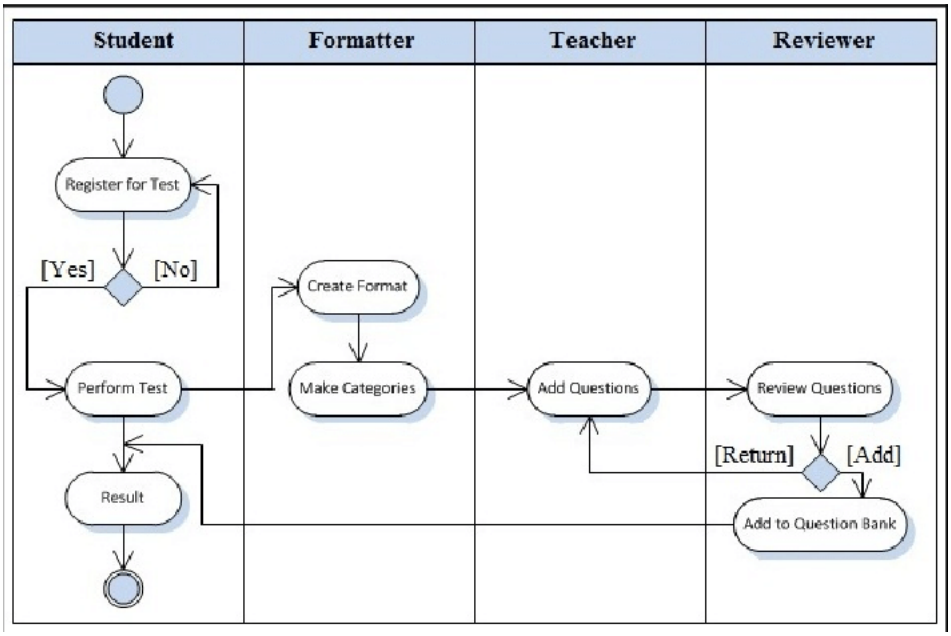
Α. Διάγραμμα περίπτωσης χρήσης (Use Case Diagram): Αποτελεί τον πιο γνωστό τύπο διαγράμματος των μοντέλων συμπεριφοράς UML. Τα διαγράμματα περίπτωσης χρήσης παρέχουν μια γραφική επισκόπηση των παραγόντων που εμπλέκονται σε ένα σύστημα, τις διαφορετικές λειτουργίες που απαιτούνται από αυτούς τους παράγοντες και το πώς αλληλεπιδρούν αυτές οι διαφορετικές λειτουργίες. Η βασική ιδέα της μοντελοποίησης περίπτωσης χρήσης είναι η σχεδίαση ενός συστήματος βασισμένο στην οπτική γωνία του τελικού χρήστη. Ένα διάγραμμα περίπτωσης χρήσης είναι συνήθως απλό- δεν δείχνει κάθε λεπτομέρεια των περιπτώσεων χρήσης, αλλά συνοψίζει μόνο μερικές από τις σχέσεις μεταξύ των περιπτώσεων χρήσης, των φορέων και των συστημάτων. Τέλος, δεν απεικονίζει τη σειρά με την οποία εκτελούνται τα βήματα για την επίτευξη των στόχων της περίπτωσης χρήσης. Το σχήμα 24 παρουσιάζει ένα παράδειγμα διαγράμματος περίπτωσης χρήσης:



Σχήμα 24: Διάγραμμα περίπτωσης χρήσης

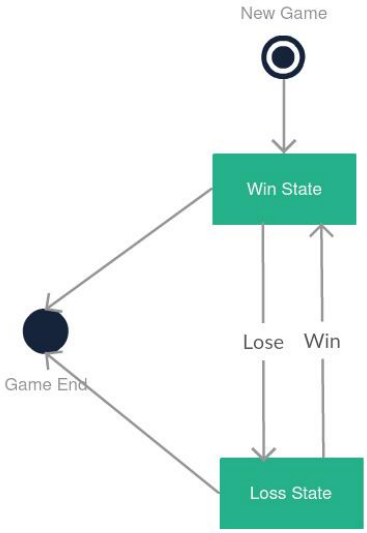
Β. Διάγραμμα δραστηριότητας (Activity Diagram): Τα διαγράμματα δραστηριότητας αναπαριστούν τις ροές δραστηριοτήτων με γραφικό τρόπο (Σχήμα 25). Μπορούν να χρησιμοποιηθούν για να περιγράψουν την επιχειρησιακή ροή ή τη λειτουργική ροή οποιουδήποτε στοιχείου σε ένα σύστημα. Κάποιες φορές, τα διαγράμματα δραστηριότητας χρησιμοποιούνται ως εναλλακτική λύση στα διαγράμματα κατάστασης. Ο σκοπός ενός διαγράμματος δραστηριότητας, εκτός από τη σχεδίαση της ροής δραστηριότητας, είναι και η περιγραφή της ακολουθίας από τη μία δραστηριότητα στην άλλη. Σχεδιάζεται με συγκεκριμένα στοιχεία σύνδεσης, μερικά εκ των οποίων είναι:

- Ενέργεια (Action): Οι χρήστες ή το λογισμικό εκτελούν μια δεδομένη εργασία. Συμβολίζεται με ένα ορθογώνιο το οποίο έχει στρογγυλεμένες άκρες.
- Κόμβος απόφασης (Decision node): Ένας κόμβος στον οποίο υπάρχουν συγκεκριμένοι όροι και απεικονίζεται με ένα ρόμβο. Περιλαμβάνει μία είσοδο και δύο ή περισσότερες εξόδους.
- Ροές ελέγχου (Control flows): Οι συνδέσεις που απεικονίζουν τη ροή μεταξύ των βημάτων στο διάγραμμα.
- Κόμβος έναρξης (Start node): Συμβολίζει την αρχή της δραστηριότητας. Ο κόμβος έναρξης αντιπροσωπεύεται από έναν μαύρο κύκλο.
- Τερματικός κόμβος (End node): Αντιπροσωπεύει το τελικό βήμα στη δραστηριότητα. Ο τελικός κόμβος αντιπροσωπεύεται από δύο ομόκεντρους μαύρους κύκλους.



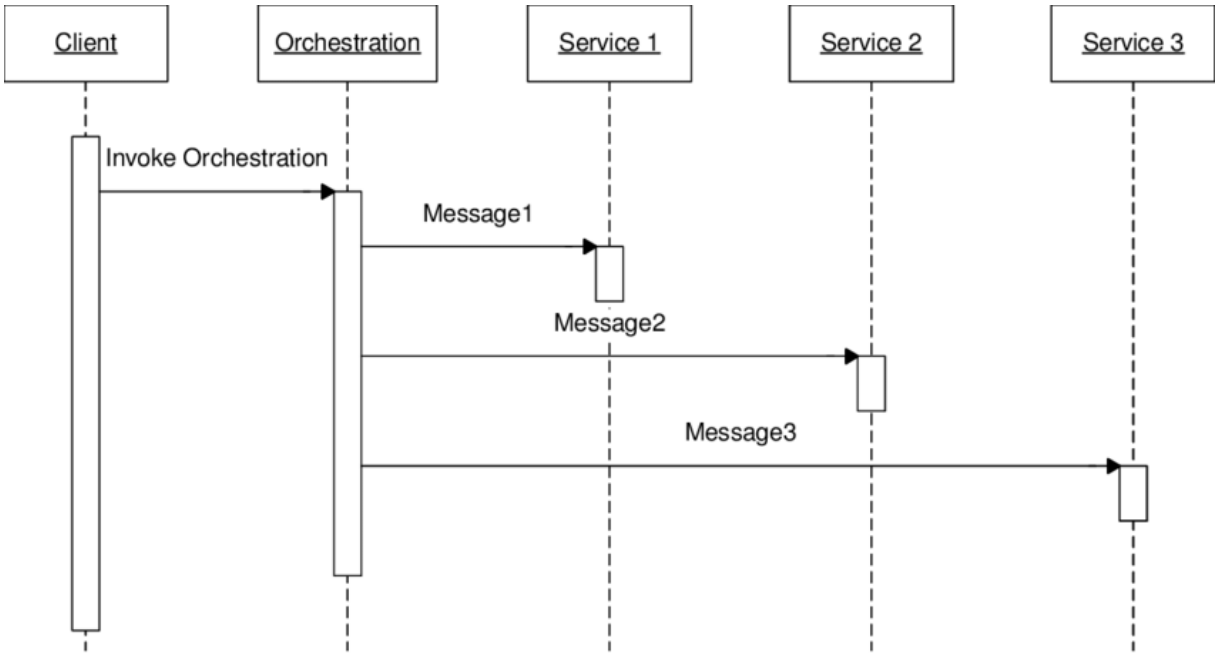
Σχήμα 25: Διάγραμμα δραστηριότητας

Γ. Διάγραμμα κατάστασης (State Machine Diagram): Τα διαγράμματα κατάστασης είναι παρόμοια με τα διαγράμματα δραστηριότητας, αν και διαφέρουν κάπως οι συμβολισμοί και η χρήση. Είναι πολύ χρήσιμα για να περιγράψουν τη συμπεριφορά των αντικειμένων που δρουν με διαφορετικό τρόπο, ανάλογα με την κατάσταση στην οποία βρίσκονται εκείνη τη στιγμή. Το σχήμα 26 απεικονίζει τις βασικές καταστάσεις και ενέργειες ενός διαγράμματος κατάστασης:



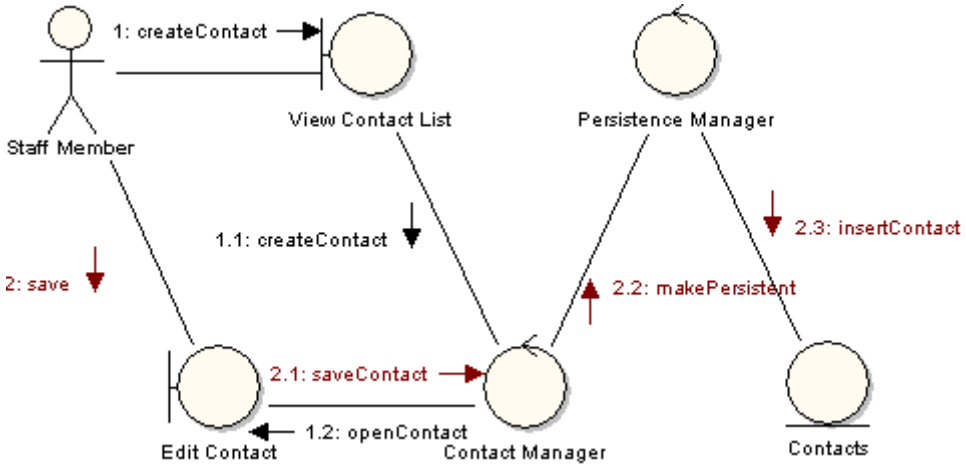
Σχήμα 26: Διάγραμμα κατάστασης

Δ. Διάγραμμα ακολουθίας (Sequence Diagram): Το διάγραμμα ακολουθίας ονομάζεται επίσης και διάγραμμα συμβάντων και παρουσιάζει τη ροή των μηνυμάτων στο σύστημα (Σχήμα 27). Χρησιμοποιείται για την απεικόνιση πολλών δυναμικών σεναρίων. Περιγράφεται η επικοινωνία μεταξύ οποιωνδήποτε δύο χρονικών συμβάντων, με τη σειρά που αυτά τα συμβάντα λαμβάνουν χώρα κατά την εκτέλεση. Στο UML, η χρονική διάρκεια απεικονίζεται από μια κάθετη μπάρα, ενώ η χρονική ροή των μηνυμάτων αντιπροσωπεύεται από μια κατακόρυφη διακεκομμένη γραμμή που εκτείνεται προς στο κάτω μέρος της σελίδας. Τέλος, στο διάγραμμα ακολουθίας ενσωματώνονται οι επαναλήψεις καθώς και οι διακλαδώσεις.



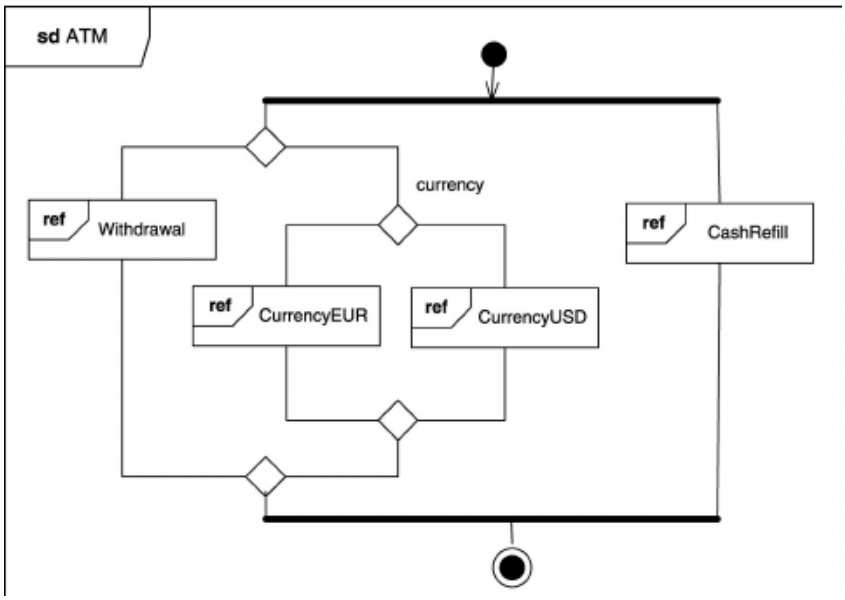
Σχήμα 27: Διάγραμμα ακολουθίας

Ε. Διάγραμμα επικοινωνίας (Communication Diagram): Στη UML 1, τα διαγράμματα επικοινωνίας ονομαζόντουσαν διαγράμματα συνεργασίας. Τα διαγράμματα επικοινωνίας είναι παρόμοια με τα διαγράμματα ακολουθίας, αλλά η εστίαση βρίσκεται στα μηνύματα που μεταδίδονται μεταξύ των αντικειμένων. Οι ίδιες πληροφορίες μπορούν να απεικονιστούν χρησιμοποιώντας ένα διάγραμμα ακολουθίας και διαφορετικά αντικείμενα.



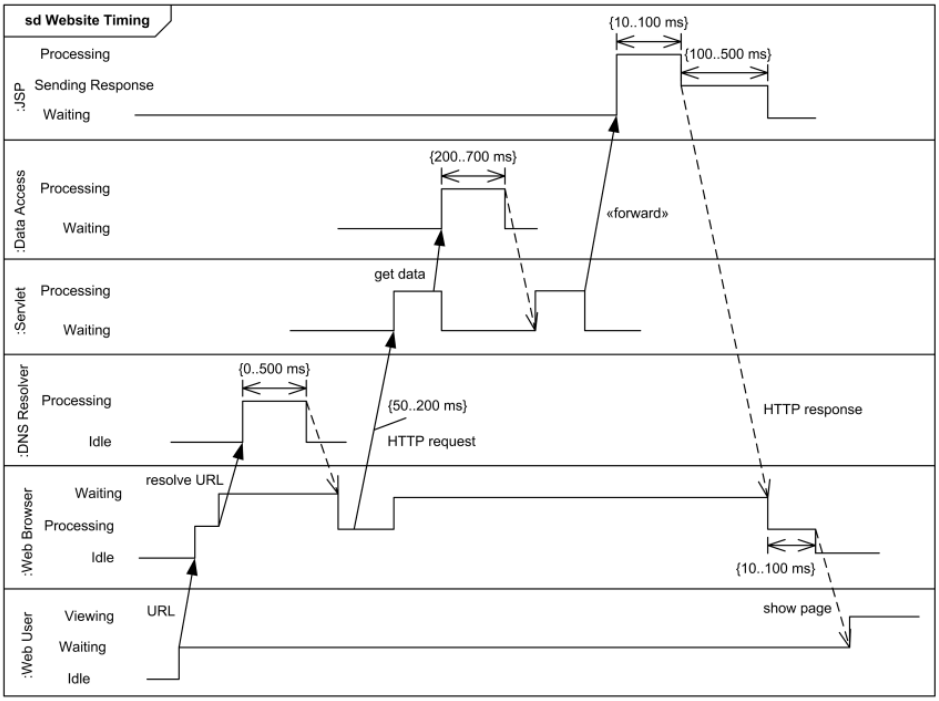
Σχήμα 28: Διάγραμμα επικοινωνίας

ΣΤ. Διάγραμμα επισκόπησης αλληλεπίδρασης (Interaction Overview Diagram): Τα διαγράμματα επισκόπησης αλληλεπίδρασης είναι παρόμοια με τα διαγράμματα δραστηριότητας. Αντί όμως να απεικονίζουν μια ακολουθία διαδικασιών (όπως συμβαίνει στα διαγράμματα δραστηριότητας), τα διαγράμματα επισκόπησης αλληλεπιδράσεων απεικονίζουν μια ακολουθία διαγραμμάτων αλληλεπίδρασης (Σχήμα 29).



Σχήμα 29: Διάγραμμα επισκόπησης αλληλεπίδρασης

Z. Διάγραμμα χρονισμού (Timing Diagram): Τα διαγράμματα χρονισμού μοιάζουν πολύ με τα διαγράμματα ακολουθίας. Απεικονίζουν τη συμπεριφορά των αντικειμένων σε ένα δεδομένο χρονικό πλαίσιο. Επίσης, τα διαγράμματα χρονισμού, όπως και τα διαγράμματα ακολουθίας, μπορούν να χρησιμοποιηθούν για την περιγραφή των περιορισμών χρόνου (time constraints) αλλά και των περιορισμών διάρκειας (duration constraints). Στο σχήμα 30 παρουσιάζεται ένα διάγραμμα χρονισμού:



Σχήμα 30: Διάγραμμα χρονισμού

1.9.3 Πλεονεκτήματα εφαρμογής της UML

Η UML βοηθά στην οργάνωση, τον σχεδιασμό και την απεικόνιση ενός συστήματος. Επίσης, ως πρότυπο, χρησιμοποιείται ευρέως και γίνεται αποδεκτή ως γλώσσα για την περιγραφή προγραμμάτων. Μπορεί να εφαρμοστεί για διάφορους σκοπούς και η αναγνωσιμότητα και η επαναχρησιμοποίησή της, την καθιστούν ιδανική επιλογή για τους προγραμματιστές. Επιπλέον οφέλη από τη χρήση της UML είναι:

- Απαιτείται λιγότερος χρόνος για την κατανόηση του έργου. Έτσι βελτιώνεται επίσης και η ανίχνευση σφαλμάτων και ελαττωμάτων.
- Συνεισφέρει στη μείωση του χρόνου εκτέλεσης ενός έργου.
- Τα διαγράμματα UML είναι διαθέσιμα ως μέρος της τεκμηρίωσης
- Βελτιώνεται η ποιότητα του τελικού προϊόντος
- Προσφέρει βελτιωμένη επικοινωνία στα μέλη της ομάδας.

1.10 Επίλογος

Σε αυτό το κεφάλαιο παρουσιάστηκε ένα μέρος της θεωρίας της Μηχανικής Λογισμικού, που αποτελεί και το μάθημα για το οποίο δημιουργήθηκε η εφαρμογή. Αρχικά, είδαμε τι είναι λογισμικό, τη φύση του και τις κατηγορίες του. Στη συνέχεια, αναλύθηκε το τι ορίζεται ως Μηχανική

Λογισμικού, έγινε μια ιστορική αναδρομή του όρου και εξετάστηκαν τα χαρακτηριστικά, οι στόχοι και οι ηθικές αρχές που την διέπουν. Ακολούθως επεξηγήθηκε ο όρος ποιότητα λογισμικού, τα χαρακτηριστικά της, τα πρότυπα, τα μοντέλα διασφάλισής της , οι μετρικές και μετρήσεις της και οι οπτικές θεωρήσεις που την αξιολογούν. Κατόπιν αναφερθήκαμε στις ευέλικτες μεθοδολογίες και στο ευέλικτο μανιφέστο. Έπειτα, αναλύθηκε η μεθοδολογία Scrum και τα βασικά χαρακτηριστικά της: οι τρείς πυλώνες και τα τρία βασικά συστατικά της (ομάδες , γεγονότα και αντικείμενα Scrum). Έγινε αναφορά στο μοντέλο Tuckman, το οποίο δίνει την βασική κατεύθυνση την οποία θα πρέπει να ακολουθεί κάθε σπριντ. Στη συνέχεια έγινε εισαγωγή στις ιστορίες χρηστών και στην εκτίμηση προσπαθειών (Planning poker). Τέλος, αναλύθηκε η Ενοποιημένη Γλώσσα Σχεδίασης Προτύπων (UML), η μοντελοποίηση και τα διαγράμματα που προσφέρει.

Κεφάλαιο 2ο: Εφαρμογή Αξιολόγησης Γραπτών Εργασιών

2.1 Εισαγωγή

Όπως έχει ήδη αναφερθεί, ο σκοπός της παρούσας πτυχιακής είναι η ανάπτυξη μιας εφαρμογής αξιολόγησης γραπτών εργασιών για το μάθημα της Μηχανικής Λογισμικού. Σε αυτό το κεφάλαιο θα εξεταστούν οι τεχνικές λεπτομέρειες, οι διάφορες αποφάσεις που έπρεπε να παρθούν, καθώς και η παρουσίαση, βήμα προς βήμα, του τρόπου με τον οποίο αναπτύχθηκε αυτή η εφαρμογή. Αρχικά, θα παρατεθούν οι απαιτήσεις του πελάτη (του επιβλέποντα καθηγητή στην προκειμένη περίπτωση). Θα ακολουθήσει επιγραμματικά αναφορά στην γλώσσα προγραμματισμού που επιλέχθηκε, θα εξηγηθεί το γιατί επιλέχθηκε η συγκεκριμένη γλώσσα και θα δοθούν οι οδηγίες εγκατάστασης. Στη συνέχεια θα αναλυθούν όλα τα στάδια της ανάπτυξης της εφαρμογής, με την παράθεση των αντίστοιχων κομματιών κώδικα. Τέλος, θα παρουσιαστούν τα αποτελέσματα ενός πειραματικού ελέγχου της απόδοσης της εφαρμογής, ώστε να φανεί το αν και πόσο χρόνο εξοικονομεί ο χρήστης με τη χρησιμοποίησή της.

2.2 Απαιτήσεις πελάτη

Ο χρήστης/καθηγητής δέχεται από τους φοιτητές τις εργασίες τους πάνω στο μάθημα της Μηχανικής Λογισμικού σε αρχεία Microsoft Word (αρχεία τύπου doc ή docx). Η δομή των εργασιών αυτών είναι χωρισμένη σε εφτά ενότητες ή ασκήσεις. Η κάθε άσκηση περιέχει την δική της βαρύτητα βαθμολόγησης και τα δικά της λάθη. Μέχρι τώρα, ακολουθείται η χρήση της αντιγραφής και επικόλλησης των λαθών, μια διαδικασία η οποία είναι χρονοβόρα. Εκτός αυτού, μετά το πέρας της βαθμολόγησης, θα πρέπει ο χρήστης να μετρήσει τα λάθη και να υπολογίσει την βαθμολογία. Αυτή δεν είναι μια εύκολη διαδικασία, αφού κάθε λάθος έχει την δική του βαρύτητα και κάθε άσκηση έχει ένα μέγιστο όριο βαθμών που μπορούν να αφαιρεθούν. Αυτά θα πρέπει να αυτοματοποιηθούν από την εφαρμογή.

Η αναλυτική εμφάνιση των λαθών (για κάθε μία άσκηση) και η συνολική απόδοσή του φοιτητή θα πρέπει να εμφανίζονται στο γραπτό που θα του επιστρέφεται, ούτως ώστε ο φοιτητής να μπορεί να γνωρίζει σε ποια από τις ασκήσεις έχει καλή απόδοση και σε ποια όχι. Με αυτόν τον τρόπο, θα είναι σε θέση να καταλάβει που έχει κενά και θα προσπαθήσει να τα καλύψει. Επίσης, η αναλυτική επεξήγηση της βαθμολόγησης θα αποτρέπει φαινόμενα διαμαρτυριών από μέρος των φοιτητών, καθώς πλέον θα φαίνονται όλα ξεκάθαρα.

Όμως, τα στατιστικά στοιχεία είναι απαραίτητα και για τον καθηγητή. Αν έχει τα στοιχεία της συνολικής απόδοσης των φοιτητών, θα είναι σε θέση να αναλύσει σε ποια σημεία χωλαίνει η απόδοσή τους και να βελτιώσει περαιτέρω τον τρόπο διδασκαλίας. Επίσης, τα συνολικά στατιστικά στοιχεία μπορούν να συγκριθούν με αυτά της προηγούμενης χρονιάς ή ακόμη και με την αντίστοιχη σχολή άλλης πόλης που χρησιμοποιεί φυσικά την ίδια εφαρμογή. Μέχρι τώρα η απόδοση των φοιτητών γινόταν με την χειροκίνητη καταγραφή των στοιχείων σε αρχείο Microsoft Excel. Αυτά λοιπόν τα στατιστικά στοιχεία θα πρέπει να αυτοματοποιηθούν από την εφαρμογή.

Καταλήγοντας, οι απαιτήσεις του πελάτη/χρήστη/καθηγητή συνοψίζονται στις εξής:

- **Η εφαρμογή θα πρέπει να μπορεί να ανοίγει και να χειρίζεται αρχεία doc και docx:** Οι εργασίες των φοιτητών παραδίδονται ηλεκτρονικά και με τη συγκεκριμένη διαμόρφωση αρχείου. Θα πρέπει επίσης η εφαρμογή να μπορεί να αποθηκεύει και να εξάγει τα βαθμολογημένα αρχεία, ώστε να είναι δυνατή η επιστροφή τους στους φοιτητές.
- **Κωδικοποίηση των λαθών κάθε ενότητας/άσκησης:** Υπάρχουν συγκεκριμένα λάθη, χωρισμένα σε κατηγορίες ανάλογα με την άσκηση στην οποία ανήκουν. Θα πρέπει να βρίσκονται σε σημείο με εύκολη πρόσβαση και να εισάγουν το μήνυμα του λάθους στο σημείο που θέλουμε (για παράδειγμα στο σημείο του κέρσορα). Ζητείται δηλαδή να τοποθετηθούν σε μια μπάρα μερικά κουμπιά, με το πάτημα των οποίων θα τοποθετείται κείμενο στο ανοιγμένο αρχείο Word.
- **Βαθμολόγηση βάσει κριτηρίων:** Κάθε λάθος θα αφαιρεί από τη βαθμολογία συγκεκριμένο βαθμό. Τα κριτήρια βαθμολόγησης έχουν δοθεί από τον πελάτη/καθηγητή και παρατίθενται στον πίνακα 1. Μπορούμε να δούμε τις εφτά ασκήσεις στις οποίες διαχωρίζεται η εργασία, το ποσοστό που καταλαμβάνει η κάθε μία από αυτές στη βαθμολογία και οι βαθμοί που αφαιρούνται από τα επιμέρους λάθη. Επίσης, στον πίνακα 1 φαίνεται η λεκτική περιγραφή της κάθε ενότητας/άσκησης, καθώς και η λεκτική περιγραφή των λαθών που θα πρέπει να εμφανίζονται στην εργασία του φοιτητή.
- **Στατιστικά φοιτητή:** Απαιτείται η δυνατότητα της αναλυτικής εμφάνισης των λαθών στον φοιτητή ώστε να βελτιωθεί στα σημεία που πάσχει. Επίσης, αν είναι δυνατόν, να εμφανίζει τα ατομικά στατιστικά στοιχεία σε μορφή διαγράμματος ή πίνακα. Θα πρέπει να εμφανίζονται στο τέλος του αρχείου, ενημερώνοντας τον φοιτητή για την επιμέρους και τη συνολική απόδοσή του, καθώς και για την συνολική βαθμολογία του.
- **Συνολικά στατιστικά:** Ο καθηγητής θα ήθελε να έχει μια συνολική εικόνα επάνω στην απόδοση των φοιτητών του. Η εφαρμογή θα πρέπει να παρέχει συνολικά στοιχεία ανά τμήμα, χρονιά ή ακόμη και σχολή, ώστε να είναι σε θέση να έχει στατιστικά στοιχεία της απόδοσης των φοιτητών του. Με αυτά τα στοιχεία στα χέρια του θα είναι σε θέση να βελτιώσει την παράδοση του μαθήματος, εμβαθύνοντας περισσότερο στα σημεία που τα στατιστικά στοιχεία δείχνουν ότι δυσκολεύουν τους φοιτητές. Μέχρι τώρα αυτό γινόταν χειροκίνητα, καταναλώνοντας πολύτιμο χρόνο.

Από τις παραπάνω απαιτήσεις γίνεται εύκολα αντιληπτό ότι η εφαρμογή θα δημιουργηθεί για εξειδικευμένη χρήση (on demand). Δεν αφορά δηλαδή μια εφαρμογή γενικής χρήσης, καθώς για κάθε αλλαγή στα κριτήρια απαιτείται επέμβαση στον κώδικα. Αυτό φυσικά οφείλεται στο γεγονός ότι υπάρχουν συγκεκριμένα κριτήρια αξιολόγησης, τα οποία είναι απόλυτα όσον αφορά τα λάθη και την βαθμολόγηση τους. Σε κάθε περίπτωση όμως, για κάποιον ο οποίος έχει κάποιες γνώσεις πάνω στη γλώσσα VBA δεν θα αντιμετωπίσει κάποιο ιδιαίτερο πρόβλημα στην τροποποίηση της εφαρμογής.

2.3 Visual Basic for Applications (VBA)

Για την ανάπτυξη της εφαρμογής επιλέχθηκε η VBA. Οι λόγοι για τους οποίους έγινε αυτή η επιλογή θα αναλυθούν παρακάτω. Αρχικά όμως, θα επεξηγηθεί τι ακριβώς είναι η VBA, θα παρατεθεί μια σύντομη ιστορική αναδρομή και τα βασικά στοιχεία που εμπεριέχονται σε αυτήν. Στη συνέχεια θα αναλυθεί το πότε χρησιμοποιείται και γιατί επιλέχθηκε για την υλοποίηση της παρούσας εφαρμογής.

Άσκηση	Περιγραφή	Ποσοστό	Βαθμός
1.Α	**Διάγραμμα Ευρωστίας**	**10**	
	Για κάθε κλάση συνοριακή-ελέγχου-οντότητας που λείπει		0,5
	Για κάθε μήνυμα που λείπει		0,5
	Για κάθε λάθος στη ροή μηνυμάτων		0,5
	Για κάθε κλήση περίπτωσης χρήσης που λείπει		0,5
	Για λάθη σύνδεσης		0,5
	Παντελής απουσία σχολιασμού		2
1.Β	**Διάγραμμα Ακολουθίας**	**10**	
	Για κάθε αντικ. οντοτήτων (ΑΟ) του δ. Ευρωστίας που λείπει		0,8
	Για κάθε αντικ. ελέγχου (ΑΕ) του δ. Ευρωστίας που λείπει		0,7
	Για κάθε μήνυμα που λείπει		0,5
	Παντελής απουσία σχολιασμού		2
1.Γ	**Διάγραμμα Κλάσεων**	**10**	
	Για κάθε εννοιολογική κλάση που λείπει		0,5
	Για κάθε χαρακτηριστικό που αναφέρεται ρητά και λείπει		0,5
	Για κάθε συσχέτιση που λείπει		0,5
	Για κάθε σχέση κληρονομικότητας που λείπει		0,5
	Για κάθε πληθυκότητα που λείπει		0,5
	Για κάθε άστοχη γραμμή σχολιασμού		0,5
	Παντελής απουσία σχολιασμού		2
1.Δ	**Υλοποίηση Κλάσεων**	**20**	
	Για κάθε κλάση του διαγράμματος κλάσεων που λείπει από τον κώδικα		0,5
	Για κάθε μέθοδο του διαγράμματος κλάσεων που δεν αντιστοιχεί σε μέθοδο του διαγράμματος ακολουθίας		0,5
	Για κάθε μέθοδο του διαγράμματος κλάσεων που δεν αντιστοιχεί σε μέθοδο του κώδικα και αντίστροφα		0,5
	Για κάθε σχέση του διαγράμματος κλάσεων που δεν αντιστοιχεί σε πεδίο αναφοράς στον κώδικα		0,5
	Μη ορθή αποτύπωση της κληρονομικότητας (costructors, abstract κλπ.)		0,5
1.Ε	**Δημιουργία και επεξεργασία διαδρομής**	**30**	
	Μη υλοποίηση της αναζήτησης γραμμής		3
	Μη υλοποίηση της αναζήτησης στάσης		3
	Μη υλοποίηση υπολογισμού μέσου χρόνου		4
	Μη υλοποίηση της `RetrieveRoute`		10
	Μη υλοποίηση της `ReportRoute`		10
	Για κάθε non private ιδιότητα		0,5
	Για κάθε κλάση χωρίς σχόλια		0,5
	Απουσία αμυντικού προγραμματισμού		1
1.ΣΤ	**Εκτέλεση της εφαρμογής**	**10**	
	Μη ορθή αποτύπωση της ροής του μενού		2
	Ελλείψεις στην παρουσίαση της λειτουργικότητας		5
2	**Συμμόρφωση με τους Κανόνες Συγγραφής**	**10**	
	Σύνολο	**100**	

Πίνακας 1: Κριτήρια αξιολόγησης

2.3.1 Γενικά για την VBA

Η Visual Basic for Applications (VBA) είναι μια γλώσσα προγραμματισμού η οποία βασίζεται σε συμβάντα και υλοποιήθηκε από τη Microsoft για την ανάπτυξη εφαρμογών στην σουίτα του Office. Η VBA βοηθά στην ανάπτυξη αυτοματοποιημένων διαδικασιών, Windows API και λειτουργιών που καθορίζονται από το χρήστη. Επιτρέπει επίσης τον χειρισμό των δυνατοτήτων της διεπαφής του χρήστη των εφαρμογών (όπως δηλαδή που έγινε κατά την υλοποίηση της εφαρμογής).

Η VBA είναι μια διαισθητική και στιβαρή γλώσσα προγραμματισμού, η οποία χρησιμοποιεί αντικειμενοστρεφή σχεδίαση. Παρουσιάζει μεγάλη ομοιότητα με την παλαιότερη και μεγαλύτερη ξαδέλφη της, την Visual Basic (VB). Ωστόσο, αν και οι δύο γλώσσες (VB και VBA) είναι γλώσσες προγραμματισμού που προέρχονται από την BASIC και δημιουργήθηκαν αμφότερες από τη Microsoft, είναι κατά τα άλλα πολύ διαφορετικές. Η VB είναι μια γλώσσα που επιτρέπει στον προγραμματιστή να δημιουργήσει αυτόνομες εκτελέσιμες εφαρμογές και δεν απαιτείται να υπάρχει εγκατεστημένη η σουίτα του Office στον υπολογιστή. Από την άλλη, η VBA δεν μπορεί να δημιουργήσει αυτόνομες εφαρμογές- ο κώδικας τρέχει μέσα σε μια κεντρική εφαρμογή όπως το MS Word ή το MS Excel.

Οι μακροεντολές (macros) είναι το στοιχείο που χρησιμοποιείται περισσότερο από τους προγραμματιστές της VBA. Μια μακροεντολή (μπορεί επίσης να αναφέρεται ως διαδικασία ή υπορουτίνα) είναι μια ομάδα από γραμμές κώδικα που εκτελούν μια σειρά εργασιών ή εντολών εντός ενός στοχευμένου προγράμματος υπολογιστή (γνωστός και ως εφαρμογή). Οι μακροεντολές μπορούν να περιέχουν κώδικα που εκτελεί υπολογισμούς, αντιγραφή και επικόλληση, αλλαγές στη μορφοποίηση κ.α. Με λίγα λόγια, οι μακροεντολές δημιουργούνται για την αυτοματοποίηση των επαναλαμβανόμενων εργασιών ρουτίνας, οι οποίες εκτελούνται σε ελάχιστο χρόνο σε σχέση με το χρονικό διάστημα που απαιτείται για να εκτελεστούν χειροκίνητα (με πληκτρολόγιο και ποντίκι).

2.3.2 Ιστορικά στοιχεία

Το VBA αποτελεί μια σύγχρονη εξέλιξη της προγραμματιστικής γλώσσας BASIC (Beginner's All-purpose Symbolic Instruction Code), η οποία αναπτύχθηκε τη δεκαετία του 1960. Η γλώσσα BASIC χρησιμοποιήθηκε ευρέως για την ανάπτυξη πολλών εφαρμογών λογισμικού κατά τις επόμενες δύο δεκαετίες, γιατί ήταν εύκολη στην εκμάθηση και στην κατανόηση. Με τα χρόνια, η BASIC εξελίχθηκε και βελτιώθηκε ταυτόχρονα με την πρόοδο της τεχνολογίας και των αυξημένων απαιτήσεων των χρηστών, προσφέροντας μεγαλύτερη ευελιξία προγραμματισμού. Το 1985, η Microsoft κυκλοφόρησε μια πολύ πιο σύγχρονη έκδοση της BASIC, η οποία ονομάστηκε QuickBASIC, και διέθετε τις πιο προχωρημένες δυνατότητες που παρείχαν οι γλώσσες προγραμματισμού μέχρι εκείνη τη στιγμή. Το 1992, η Microsoft κυκλοφόρησε τη Visual Basic για τα Windows, μία έκδοση σχεδιασμένη για να λειτουργεί εντός του αναπτυσσόμενου τότε περιβάλλοντος των Windows. Παράλληλα, διάφοροι εκδότες λογισμικού δημιούργησαν τις δικές τους βελτιώσεις και παραλλαγές της BASIC, ώστε να την προσαρμόσουν στις ανάγκες τους. Αυτό είχε σαν αποτέλεσμα ένα ευρύ και μπερδεμένο φάσμα λειτουργικότητας , το οποίο αποτυπωνόταν με μεγάλες διαφορές στις εντολές μεταξύ των εφαρμογών λογισμικού που χρησιμοποιούσαν την BASIC.

Η Microsoft δεν άργησε να αναγνωρίσει την ανάγκη για την ανάπτυξη μιας τυποποιημένης γλώσσας προγραμματισμού για τα προϊόντα λογισμικού της, και έτσι δημιούργησε την Visual Basic για εφαρμογές (VBA). Η VBA κυκλοφόρησε για πρώτη φορά από τη Microsoft μαζί με το Excel 5, την έκδοση του Excel που περιλαμβανόταν στη σουίτα του Office 1995. Έκτοτε, η VBA έχει γίνει η γλώσσα προγραμματισμού και σε άλλες δημοφιλείς εφαρμογές του Office της Microsoft, καθώς και

σε πελάτες εξωτερικού λογισμικού της Microsoft στους οποίους έχει παραχωρηθεί η άδεια χρήσης της VBA.

2.3.3 Τα βασικά στοιχεία της VBA

Τα βασικά στοιχεία της VBA δεν διαφέρουν από αυτά μιας οποιασδήποτε γλώσσας προγραμματισμού. Συγκεκριμένα προγραμματίζοντας σε VBA συναντάμε:

- **Γραμμές κώδικα:** Οι ενέργειες στο VBA πραγματοποιούνται με την εκτέλεση γραμμών κώδικα. Αυτές οι γραμμές κώδικα αποθηκεύονται συνήθως σε μια μονάδα (module) VBA.
- **Module:** Οι λειτουργικές μονάδες (modules) της VBA αποθηκεύονται σε ένα βιβλίο εργασίας του Excel ή σε ένα έγγραφο του Word, ωστόσο μπορούν να προβληθούν και να επεξεργαστούν μόνο με τη χρήση του προγράμματος επεξεργασίας της Visual Basic (Visual Basic Editor - VBE). Ένα VBA module αποτελείται από διαδικασίες
- **Διαδικασίες (Procedures):** Μια διαδικασία είναι βασικά μια ομάδα γραμμών κώδικα που εκτελεί κάποια ενέργεια. Η VBA υποστηρίζει δύο τύπους διαδικασιών: Sub και function. Μια διαδικασία Sub αποτελείται από μια σειρά δηλώσεων και μπορεί να εκτελεστεί ποικιλοτρόπως. Μια συνάρτηση (function) επιστρέφει μια μεμονωμένη τιμή (ή πιθανώς έναν πίνακα) και μπορεί να κληθεί από μια άλλη συνάρτηση.
- **Αντικείμενα (Objects):** Η VBA παρέχει τη δυνατότητα χειρισμού των αντικειμένων που περιέχονται στην εφαρμογή. Για παράδειγμα, το Excel περιλαμβάνει περισσότερες από εκατό κατηγορίες αντικειμένων, τα οποία μπορούν να χρησιμοποιηθούν. Κάποια από αυτά τα αντικείμενα είναι ένα βιβλίο εργασίας, ένα φύλλο εργασίας ή ένα γράφημα.
- **Συλλογές (Collections):** Ένα σύνολο από όμοια αντικείμενα αποτελούν μια συλλογή.
- **Ιεραρχία αντικειμένων (Object hierarchy):** Όταν γίνεται αναφορά σε αντικείμενο που περιέχεται ή αποτελεί μέλος μιας μεγαλύτερης οντότητας, η θέση του στην ιεραρχία αντικειμένων καθορίζεται χρησιμοποιώντας μια τελεία ως διαχωριστικό μεταξύ τους (όπως συμβαίνει και σε πολλές άλλες γλώσσες προγραμματισμού).
- **Ενεργά αντικείμενα (Active objects):** Εάν παραληφθεί μια συγκεκριμένη αναφορά σε ένα αντικείμενο, η εκάστοτε εφαρμογή του Office θα χρησιμοποιήσει τα ενεργά αντικείμενα-αυτά δηλαδή που έχουν focus.
- **Ιδιότητες αντικειμένων (Objects properties):** Τα αντικείμενα έχουν ιδιότητες, ενώ αυτές οι ιδιότητες θεωρούνται ως οι ρυθμίσεις ενός αντικείμενου. Μέσω της VBA είναι δυνατός ο προσδιορισμός και η αλλαγή των ιδιοτήτων ενός αντικειμένου.
- **Μεταβλητές:** Στη VBA δίδεται η δυνατότητα καταχώρησης διαφόρων τύπων τιμών σε μεταβλητές. Όπως και στις υπόλοιπες γλώσσες προγραμματισμού, συστήνεται ανεπιφύλακτα η δόκιμη ονοματοθεσία των μεταβλητών, η οποία θα πρέπει να είναι σχετική με τις τιμές που περιέχει.
- **Μέθοδοι αντικειμένου (Object methods):** Τα αντικείμενα περιέχουν μεθόδους. Έτσι, μια μέθοδος αντικειμένου είναι μια ενέργεια που εκτελείται μαζί με το αντικείμενο.
- **Τυπικές δομές προγραμματισμού:** Το VBA περιλαμβάνει επίσης όλες τις δομές μιας σύγχρονης γλώσσας προγραμματισμού, όπως για παράδειγμα πίνακες, επαναληπτικούς βρόχους κ.α.
- **Συμβάντα (events):** Ορισμένα αντικείμενα αναγνωρίζουν συγκεκριμένα συμβάντα και μπορεί έτσι να γραφεί κώδικας που να εκτελείται όταν συμβαίνει το συμβάν.

2.3.4 Γιατί επιλέχθηκε η VBA για την ανάπτυξη της εφαρμογής

Στη συνέχεια όλων των παραπάνω που έχουν αναφερθεί εξετάζοντας τη VBA, είναι εύκολο να αιτιολογηθούν οι λόγοι για τους οποίους επιλέχθηκε ως η καταλληλότερη γλώσσα ανάπτυξης της εφαρμογής. Αρχικά, η VBA δεν χειρίζεται απλά τα αρχεία doc ή docx, αλλά υλοποιείται μέσω της σουίτας του Office και αποτελεί μέρος της. Δεν χρειάζεται να φορτωθεί το αρχείο doc μέσω μιας ανεξάρτητης εφαρμογής (πιθανότατα μέσα σε μια φόρμα) και να είναι αναγκασμένος ο χρήστης να κινείται με τις μπάρες κίνησης για να δει ολόκληρο το αρχείο στην οθόνη του. Το να μειώσει την εστίαση θα αποτελούσε μια λύση αλλά κάτι τέτοιο θα δυσκόλευε τον χρήστη στην ανάγνωση. Άλλο πρόβλημα που δεν χρειάστηκε να αντιμετωπιστεί είναι οι λειτουργίες όπως «Αποθήκευση», «Αποθήκευση ως», «Εύρεση», «Αντικατάσταση» και πάρα πολλές άλλες οι οποίες είναι ήδη ενσωματωμένες στο Word και δουλεύουν απρόσκοπτα. Επιπλέον δημιουργημένες λειτουργίες θα σήμαιναν πιθανότατα και περισσότερα προβλήματα (bugs) προς επίλυση.

Επιπλέον, η VBA έχει όλα τα δομικά στοιχεία μιας σύγχρονης γλώσσας, ώστε ο προγραμματιστής να είναι σε θέση να δημιουργήσει με ακρίβεια αυτό που επιθυμεί.

Η εφαρμογή της παρούσας πρακτικής αποτελεί στην ουσία ένα πρόσθετο (add-in) του MS Word και όχι μια αυτόνομη εφαρμογή. Ωστόσο, καλύπτει όλες τις απαιτήσεις του πελάτη, με ελάχιστη ανάγκη τεκμηρίωσης και εκπαίδευσης στο χρήστη – όλοι γνωρίζουν να χειρίζονται τη σουίτα του Office.

Μιλώντας όμως για τα διδάγματα της Μηχανικής Λογισμικού, ας δούμε σε ποια σημεία εξυπηρετεί η VBA την εφαρμογή:

- **Εύκολα τροποποιήσιμη:** Η εφαρμογή είναι εύκολα τροποποιήσιμη, αρκεί να υπάρχουν βασικές γνώσεις VBA.
- **Επαναχρησιμοποίηση:** Δεν είναι απαραίτητο να αναπτύξουμε όλες τις λειτουργίες από την αρχή. Ενσωματώνοντας τις έτοιμες λειτουργίες του Word, εξοικονομείται χρόνος και χρήμα.
- **Μείωση της πολυπλοκότητας ανάπτυξης:** Επίσης με την υλοποίηση μέσω του Word και τη χρήση έτοιμων λειτουργιών, η πολυπλοκότητα μειώνεται.
- **Αποτελεσματικότητα:** Οι απαιτήσεις πελάτη καλύπτονται με τον πιο άμεσο και αποτελεσματικό τρόπο.
- **Μείωση χρόνου ανάπτυξης:** Ο προγραμματιστής έχει να ασχοληθεί μόνο με τις απαιτήσεις του πελάτη και όχι με την ανάπτυξη διεπαφής, μενού ή ανεξάρτητης εφαρμογής.
- **Ελαχιστοποίηση του κόστους:** Όλα τα παραπάνω συντελούν στη μείωση του κόστους.
- **Ευκολία στην κατανόηση:** Για την χρήση της εφαρμογής απαιτούνται ελάχιστη τεκμηρίωση και εκπαίδευση, καθώς όλοι γνωρίζουν να χειρίζονται το MS Word.
- **Αξιοπιστία και μελλοντικές εκδόσεις:** Η εφαρμογή μας δεν εμπλέκεται- η βασική δουλειά πέφτει στην Microsoft.

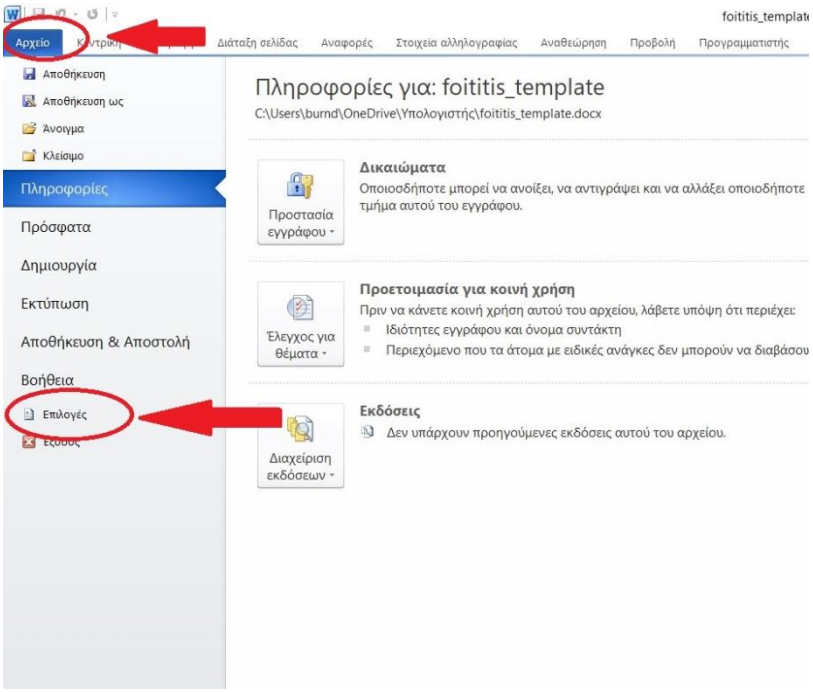
Για όλους λοιπόν τους παραπάνω λόγους, η VBA προκρίθηκε ως η καλύτερη λύση για την υλοποίηση της εφαρμογής που παρουσιάζεται σε αυτήν την πτυχιακή.

2.4 Visual Basic Editor (VBE)

Αναφέρθηκε ότι ο προγραμματισμός με VBA πραγματοποιείται μέσα σε μια μεγαλύτερη εφαρμογή, στην περίπτωσή μας το MS Word. Ο προγραμματισμός λοιπόν γίνεται με την χρήση του συντάκτη VBE, στον οποίο καταχωρείται όλος ο κώδικας. Συγκεκριμένα, ο VBE είναι το εργαλείο που χρησιμοποιείται για τη δημιουργία, τροποποίηση και συντήρηση των διαδικασιών και λειτουργικών μονάδων της VBA σε εφαρμογές του Microsoft Office. Αν και ο VBE περιλαμβάνεται στα

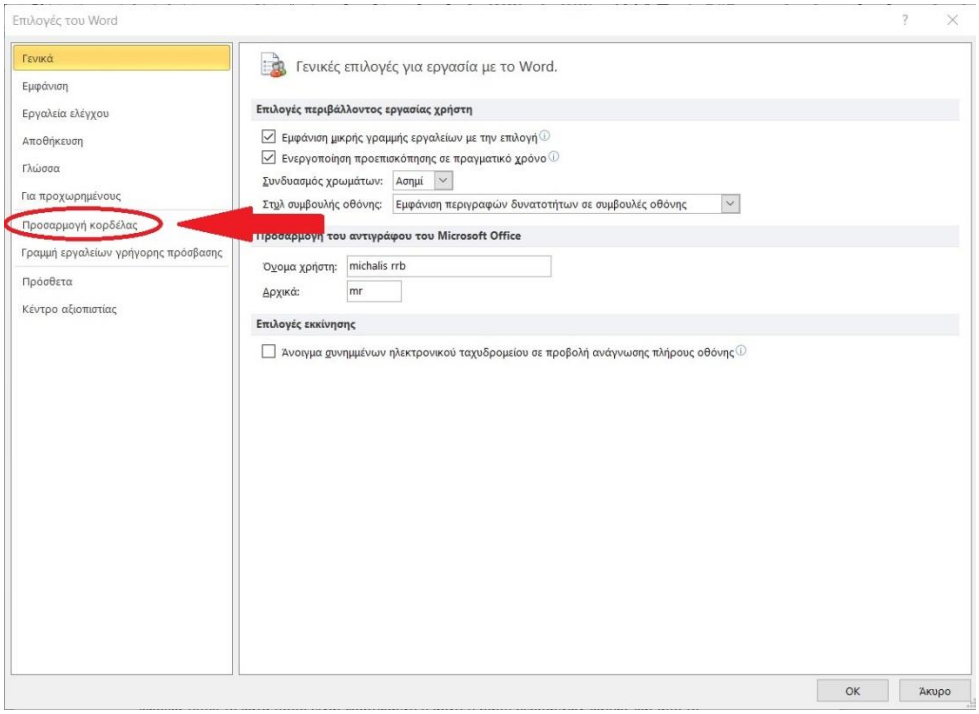
περισσότερα προγράμματα του Office, η εργαστηριακή του ρύθμιση είναι σε θέση απενεργοποιημένος και θα πρέπει να ενεργοποιηθεί για να γίνει προσπελάσιμος. Παρακάτω ακολουθούν τα βήματα που απαιτούνται για να γίνουν προσβάσιμα η καρτέλα «Προγραμματιστής» (Developer) και ο VBE:

- Αρχικά, ανοίγουμε το μενού «Αρχείο» και επιλέγουμε «επιλογές» (εικόνα 1):



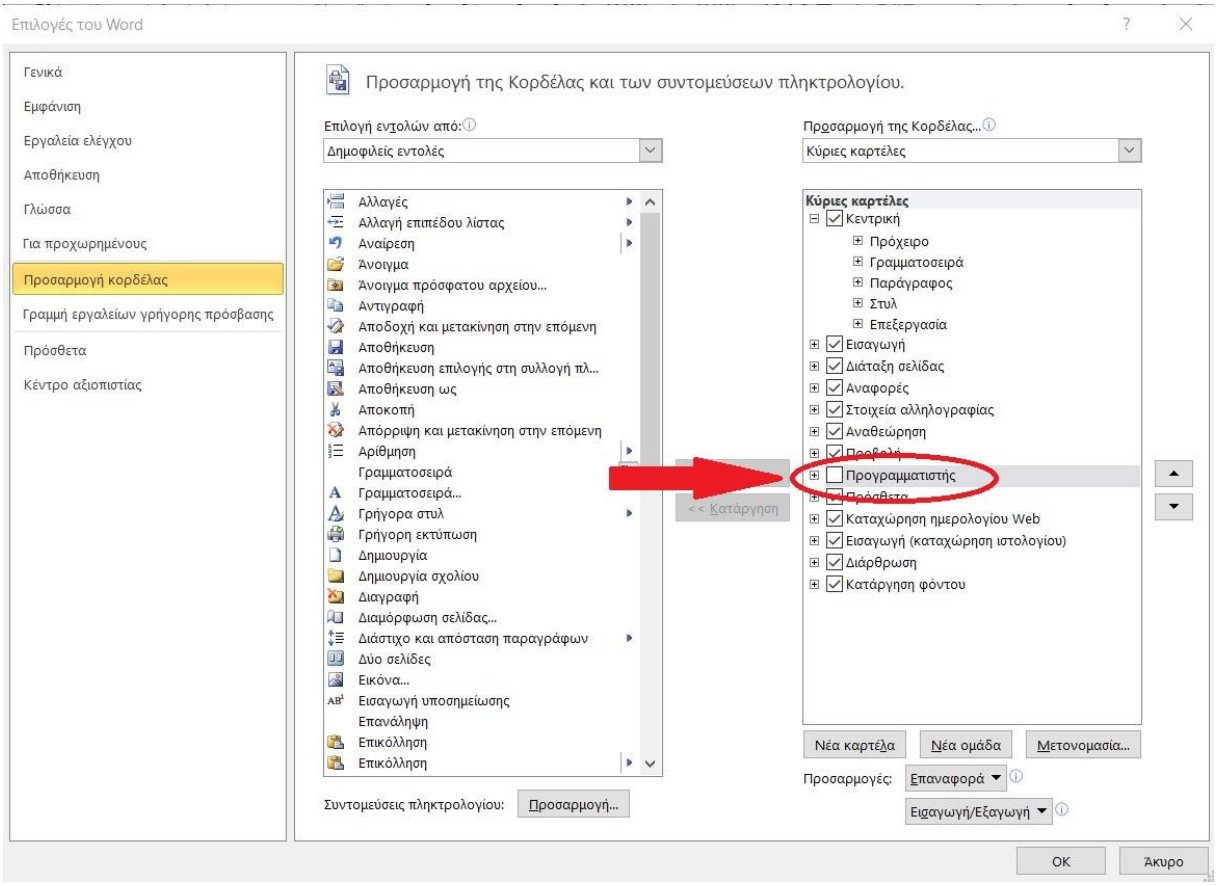
Εικόνα 1

- Εμφανίζεται η καρτέλα «Επιλογές του Word» και επιλέγουμε «Προσαρμογή κορδέλας» (εικόνα 2):



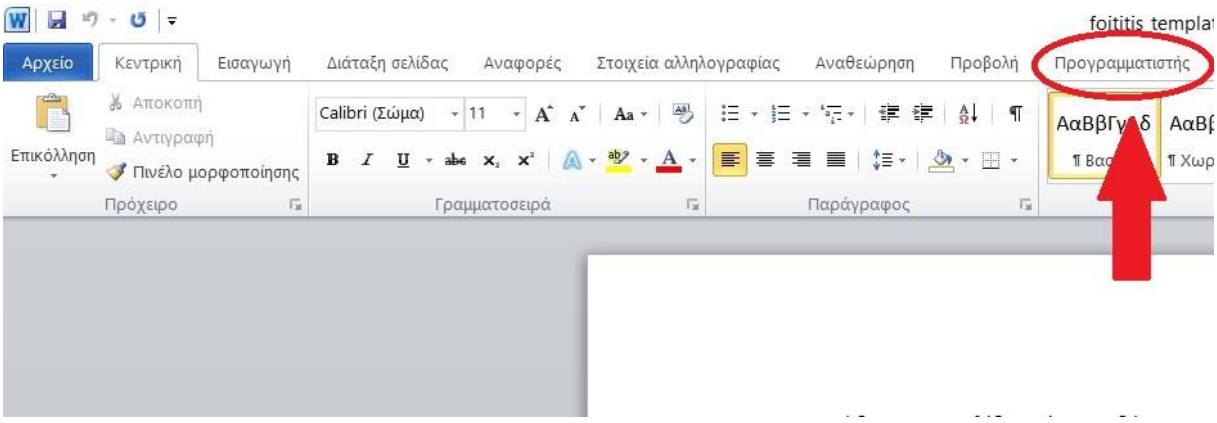
Εικόνα 2

- Έχοντας ανοίξει την καρτέλα «Προσαρμογή της Κορδέλας και των συντομεύσεων πληκτρολογίου», παρατηρούμε στη δεξιά στήλη την απενεργοποιημένη επιλογή «Προγραμματιστής» (εικόνα 3):



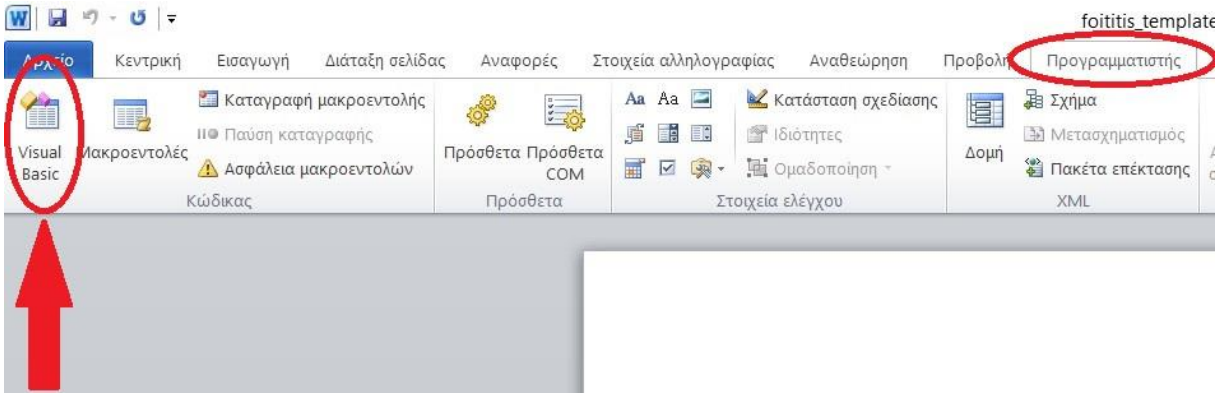
Εικόνα 3

- Επιλέγοντας την καρτέλα «Προγραμματιστής» και πατώντας «ΟΚ», η καρτέλα εμφανίζεται πια δίπλα στις υπόλοιπες (εικόνα 4):



Εικόνα 4

- Από την νεοεισαχθέντα καρτέλα «Προγραμματιστής», αποκτάμε πρόσβαση στον VBE επιλέγοντας το εικονίδιο «Visual Basic» (εικόνα 5):



Εικόνα 5

Πλέον, έχοντας ακολουθήσει όλα τα παραπάνω βήματα, είμαστε σε θέση να καταχωρήσουμε και να εκτελέσουμε κώδικα VBA.

2.5 Εγκατάσταση εφαρμογής

Το πρότυπο του Word που αποτελεί την εφαρμογή μας, μπορεί είτε να εγκατασταθεί μόνιμα στον υπολογιστή μας (οι επιπρόσθετες καρτέλες θα εμφανίζονται σε κάθε έγγραφο Word που θα ανοίγουμε) ή να γίνει μεμονωμένη χρήση με επιλογή προσωρινού προτύπου. Ας δούμε πως επιτυγχάνεται το καθένα από αυτά.

2.5.1 Μόνιμη εγκατάσταση

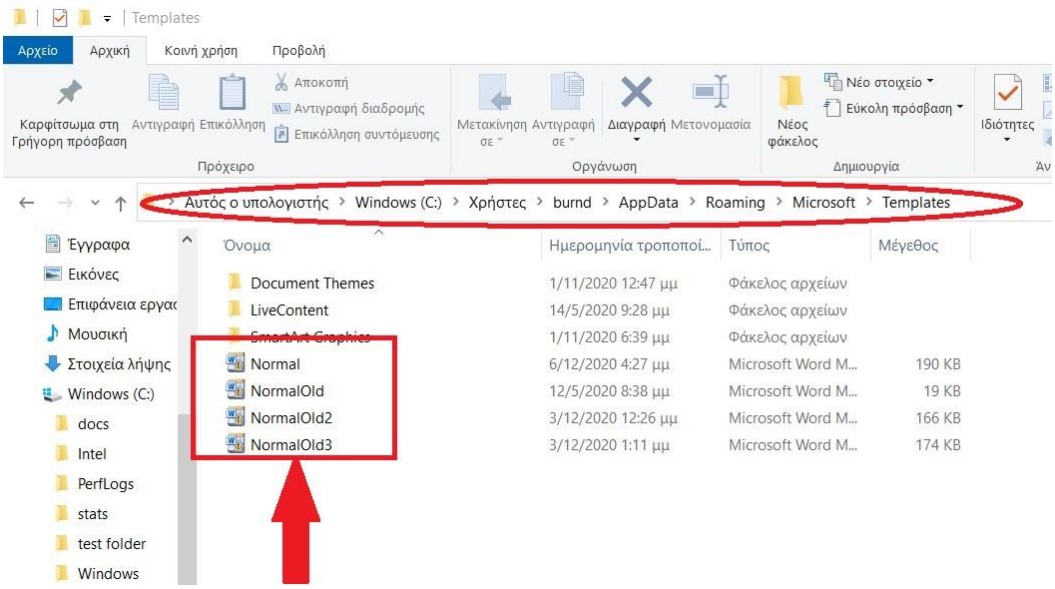
Για μόνιμη εγκατάσταση του προτύπου ακολουθούμε τα ακόλουθα βήματα:

- Αρχικά κλείνουμε όλες τις εφαρμογές που χρησιμοποιούν MS Word, όπως DESS Word Client ή EXEditor.
- Εντοπίστε το υπάρχον αρχείο Normal.dotm. Αυτό το αρχείο αποτελεί το βασικό πρότυπο μορφοποίησης του MS Word. Για μόνιμη εγκατάσταση θα πρέπει να το αντικαταστήσουμε με το πρότυπο της πτυχιακής. Η προεπιλεγμένη τοποθεσία είναι :

  C:\Users\<user-name>\AppData\Roaming\Microsoft\Templates, για την Αγγλική έκδοση και

  C:\ Users\<user-name>\AppData\Roaming\Microsoft\Πρότυπα, για την ελληνική έκδοση.

- Μόλις εντοπιστεί το αρχείο Normal.dotm, θα πρέπει να μετονομαστεί σε κάτι άλλο, για παράδειγμα NormalOld. **ΠΡΟΣΟΧΗ: Συστήνεται ανεπιφύλακτα η μετονομασία του αρχείου Normal.dotm σε κάποιο άλλο όνομα**, διαδικασία που θα πρέπει να επαναλαμβάνεται και σε κάθε έκδοση που τροποποιείται (εικόνα 6). Άπαξ και το εργοστασιακό πρότυπο αντικατασταθεί με τη διαδικασία του overwrite, το νέο πρότυπο καταλαμβάνει τη θέση των εργοστασιακών ρυθμίσεων και δεν υπάρχει τρόπος επαναφοράς. Αν όμως έχουμε αποθηκεύσει το προκαθορισμένο πρότυπο με άλλο όνομα, μπορούμε ανά πάσα στιγμή να επιστρέψουμε στις εργοστασιακές ρυθμίσεις με μια απλή μετονομασία. Στην εικόνα 6, μπορούμε ακόμη να δούμε το μονοπάτι που βρίσκεται το πρότυπο Normal.dotm.



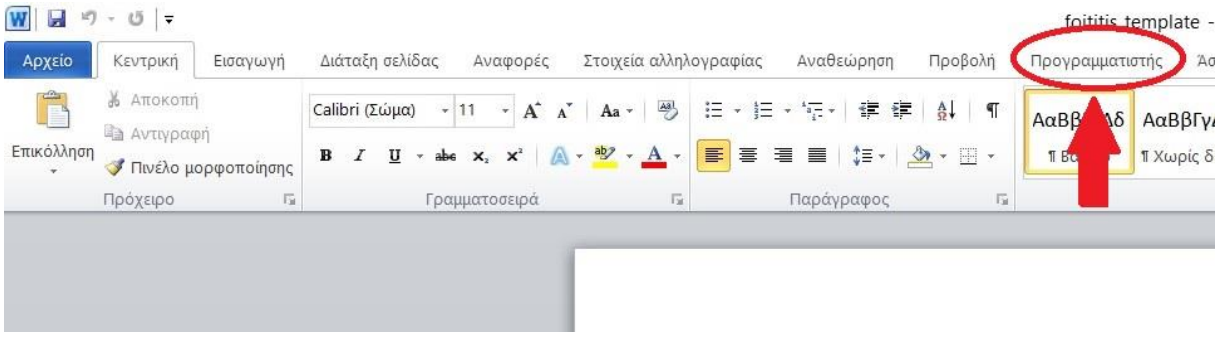
Εικόνα 6

- Μετονομάζουμε το νέο μας πρότυπο σε Normal.dotm και το τοποθετούμε στον φάκελο των προτύπων (το μονοπάτι του φακέλου δόθηκε παραπάνω). Έτσι, το νέο πρότυπο θα αντικαταστήσει το προκαθορισμένο και στο εξής κάθε νέο έγγραφο του MS Word θα ανοίγει με βάση τις ρυθμίσεις του νέου μας προτύπου. Οτιδήποτε δεν έχουμε τροποποιήσει στο νέο μας πρότυπο, θα διατηρήσει τις ρυθμίσεις του εργοστασιακού προτύπου.

2.5.2 Μεμονωμένη χρήση με επιλογή προσωρινού προτύπου

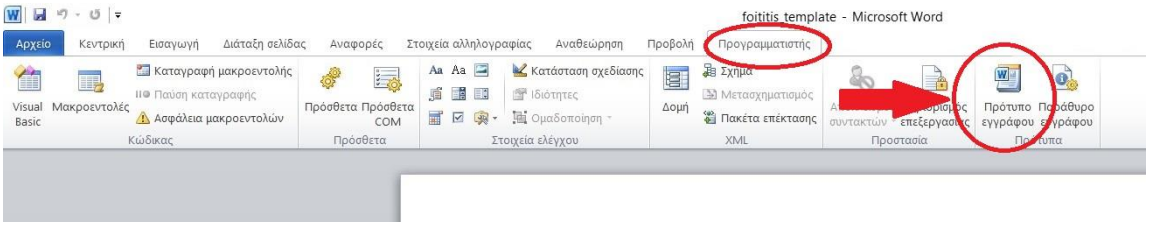
Σε αυτήν την περίπτωση διατηρούμε αναλλοίωτες τις γενικές εργοστασιακές ρυθμίσεις και επιλέγουμε τη εφαρμογή του προτύπου μόνο στο ενεργό έγγραφο του Word. Αυτό επιτυγχάνεται ακολουθώντας τις παρακάτω κινήσεις:

- Στο ενεργό έγγραφο του Word, σε αυτό δηλαδή που θέλουμε να εφαρμόσουμε το πρότυπο, επιλέγουμε την καρτέλα «Προγραμματιστής» (εικόνα 7).



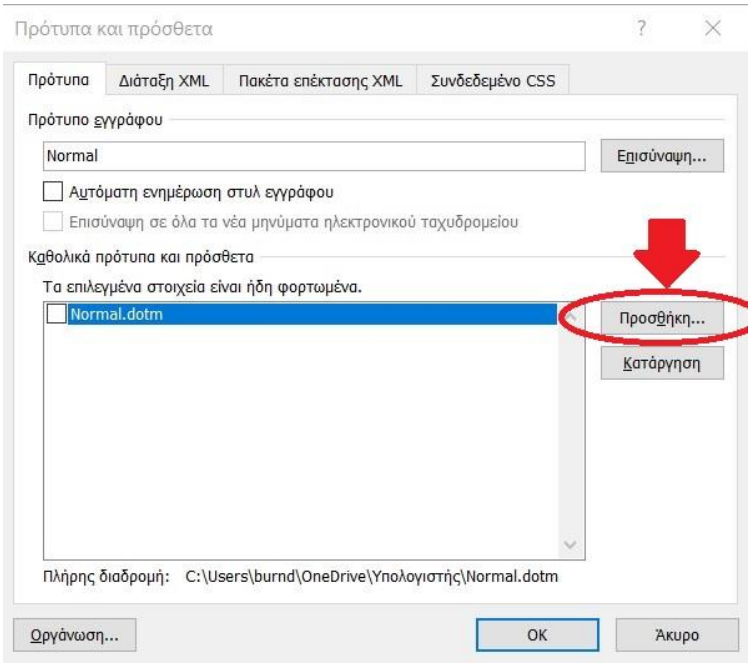
Εικόνα 7

- Στην κορδέλα του συγκεκριμένης καρτέλας θα βρούμε και θα επιλέξουμε το κουμπί «Πρότυπο εγγράφου» (εικόνα 8).



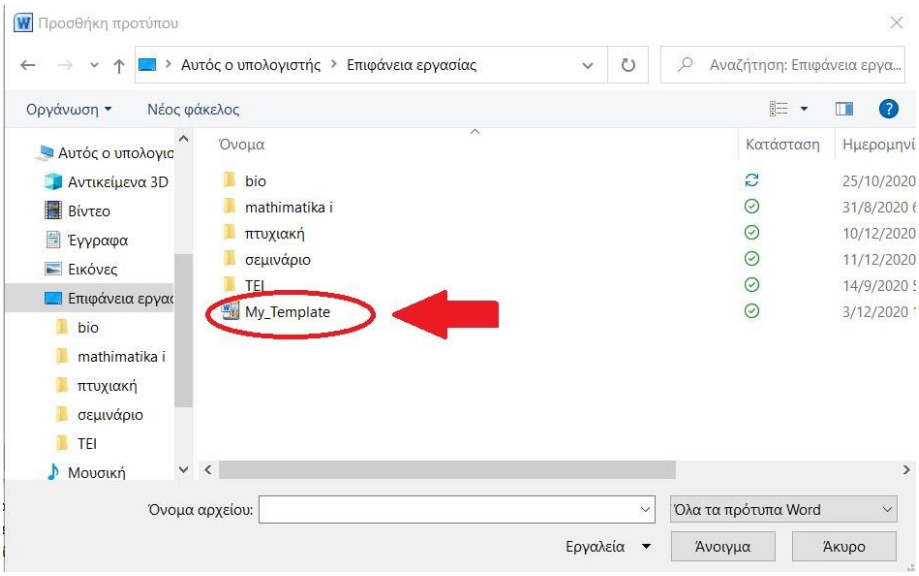
Εικόνα 8

- Θα εμφανιστεί το παράθυρο «Πρότυπα και πρόσθετα». Σε αυτό το παράθυρο θα πρέπει να επιλέξουμε το κουμπί «Προσθήκη» (εικόνα 9).



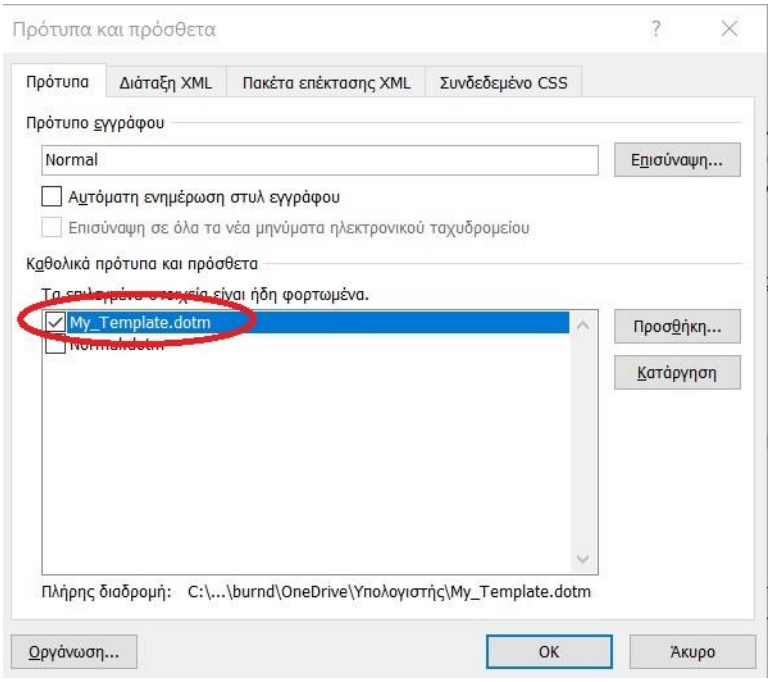
Εικόνα 9

- Θα εμφανιστεί το παράθυρο αναζήτησης (εικόνα 10). Θα εντοπίσουμε και θα επιλέξουμε το πρότυπο που θέλουμε να χρησιμοποιήσουμε.



Εικόνα 10

- Πλέον, το πρότυπό μας θα φαίνεται σαν επιλεγμένο (εικόνα 11). Επιλέγουμε «OK» και οι ρυθμίσεις του προτύπου εμφανίζονται άμεσα στο ενεργό έγγραφο του Word.



Εικόνα 11

Θα πρέπει να σημειωθεί ότι το ενεργό έγγραφο του Word υιοθετεί προσωρινά τις ρυθμίσεις του προτύπου μας. Αν κλείσουμε το έγγραφο και το ανοίξουμε εκ νέου θα δούμε ότι δεν έχει διατηρήσει τις ρυθμίσεις. Έτσι, θα πρέπει ότι θα πρέπει να επαναλάβουμε την παραπάνω διαδικασία ξανά από την αρχή.

2.6 Βήματα ανάπτυξης της εφαρμογής/προτύπου

Αρχικά, έπρεπε να δημιουργηθούν οι καρτέλες οι οποίες αντιστοιχούν στις ασκήσεις (μία καρτέλα για κάθε μία άσκηση). Αυτή η λύση προκρίθηκε γιατί με αυτόν το τρόπο ο χρήστης έχει άμεση πρόσβαση στα λάθη της άσκησης που θέλει, χωρίς να χάνεται σε μια θάλασσα από κουμπιά.

Μέσα σε κάθε καρτέλα υπάρχουν τα κουμπιά με κωδικοποιημένα τα λάθη της αντίστοιχης άσκησης. Τα κουμπιά εκτυπώνουν το λάθος που αντιστοιχεί στο κουμπί στο σημείο που βρίσκεται ο κέρσορας. Η λειτουργικότητα των κουμπιών δεν σταματάει στην εκτύπωση του λάθους αλλά αφαιρεί και τον κατάλληλο βαθμό. Όταν η συνολική βαθμολογία της άσκησης φτάσει στο ανώτερο όριο (όπως αυτά δίδονται από τα κριτήρια αξιολόγησης), τότε η βαθμολογία παραμένει στο ανώτερο όριο βαθμού ασχέτως από το πόσα λάθη θα ακολουθήσουν.

Επιπλέον, υπάρχει μια ξεχωριστή καρτέλα, η οποία περιέχει την βαθμολόγηση του γραπτού και τα στατιστικά στοιχεία. Μόλις ο χρήστης τελειώσει με την αξιολόγηση του γραπτού, πατάει το κουμπί της βαθμολόγησης (Υπολογισμός βαθμού). Ο κέρσορας αλλάζει σελίδα και εκτυπώνει την απόδοση του φοιτητή σε κάθε μία άσκηση. Στη συνέχεια, εκτυπώνεται και η συνολική βαθμολογία του φοιτητή. Αν ο χρήστης επιθυμεί να ενημερώσει με επιπλέον στοιχεία τον φοιτητή, πατώντας το κουμπί «Στατιστικά φοιτητή» θα εκτυπώσει κάτω από την βαθμολογία του έναν πίνακα με το ποσοστό αποπεράτωσης της κάθε άσκησης- το ποσοστό δηλαδή που κατάφερε να ολοκληρώσει επιτυχώς σε κάθε άσκηση. Κάτω από τον πίνακα εμφανίζεται επίσης και ένα διάγραμμα, το οποίο παρουσιάζει γραφικά τα αποτελέσματα που αναγράφονται στον ως άνω αναφερόμενο πίνακα. Τέλος,

ο χρήστης μπορεί να καταγράψει στατιστικά στοιχεία για όλους τους φοιτητές. Για να γίνει αυτό, θα πρέπει όλα τα αρχεία doc να βρίσκονται αποθηκευμένα στον ίδιο φάκελο. Με το πάτημα του κουμπιού «Καθορισμός μονοπατιού (path) φακέλων και αρχείων Excel», το πρόσθετο ζητάει από τον χρήστη να καταχωρίσει το μονοπάτι του φακέλου που βρίσκονται τα αρχεία των ασκήσεων των φοιτητών και το μονοπάτι του αρχείου Excel στο οποίο θα καταχωρηθούν τα δεδομένα. Αν ο χρήστης παραλείψει αυτό το στάδιο και πατήσει κατευθείαν το κουμπί «Συνολικά Στατιστικά Στοιχεία», το σύστημα θα ελέγξει αν έχουν δοθεί. Στην περίπτωση που δεν έχουν καταχωρηθεί, θα ζητηθούν από τον χρήστη εκείνη τη στιγμή. Αν έχουν δοθεί σωστά όλα τα στοιχεία, η εφαρμογή θα σαρώσει τον φάκελο και θα μεταφέρει τα στοιχεία στο Excel. Κάθε φοιτητής θα είναι και μία γραμμή, ενώ οι στήλες είναι οι ασκήσεις. Μόλις ολοκληρωθεί η διαδικασία, τα δεδομένα θα είναι έτοιμα για περαιτέρω επεξεργασία.

Στα επόμενα υποκεφάλαια, ακολουθεί η διαδικασία της ανάπτυξης βήμα προς βήμα.

2.6.1 XML

Η δημιουργία των επιπλέον καρτελών έγινε με χρήση της XML. Συγκεκριμένα χρησιμοποιήθηκε ένα δωρεάν, ανοιχτού κώδικα πρόγραμμα, το Office RibbonX Editor. Με τη βοήθεια αυτού του προγράμματος, ανοίχτηκε το πρότυπο και επιτράπηκε η συγγραφή του κώδικα της XML.

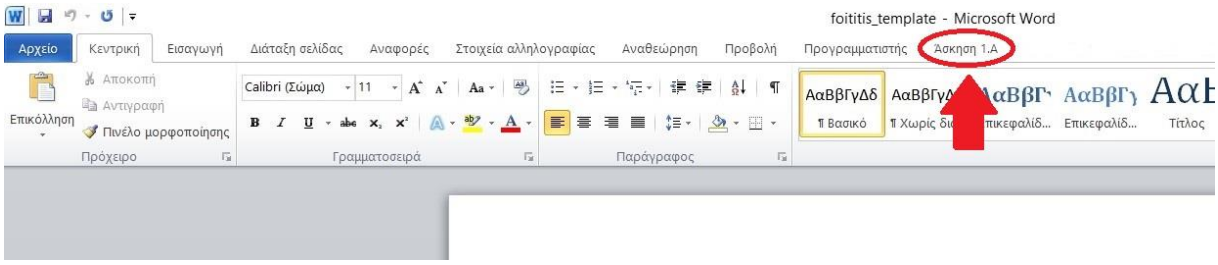
Παρακάτω είναι ο κώδικας που χρησιμοποιήθηκε για την δημιουργία της πρώτης πρόσθετης καρτέλας (εικόνα 12):

customUI.xml

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
    <ribbon startFromScratch="false">
        <tabs>
            <tab id="_1a" label="Άσκηση 1.Α">
                <group id="diagramma_eurostias" label="Διάγραμμα Ευρωστίας">
                    <button id="_1a_1" label="Λείπει κλάση συνοριακή-ελέγχου-οντότητας" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro" />
                    <button id="_1a_2" label="Λείπει μήνυμα" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro2" />
                    <button id="_1a_3" label="Λάθος στη ροή μηνυμάτων" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro3" />
                    <button id="_1a_4" label="Λείπει κλήση περίπτωσης χρήσης" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro4" />
                    <button id="_1a_5" label="Λάθος σύνδεσης" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro5" />
                    <button id="_1a_6" label="Παντελής απουσία σχολιασμού" image="_x0031_A" size="large" onAction="ThisDocument.MyMacro6" />
                </group>
                <group id="undo1" label="Undo">
                    <button id="Undo" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro7" />
                </group>
            </tab>
```

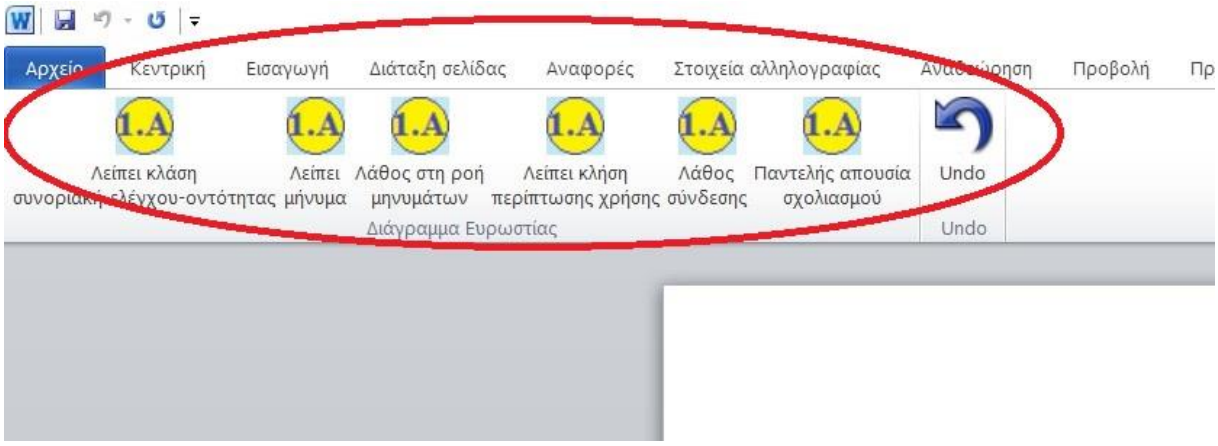
Εικόνα 12

Ως αποτέλεσμα του παραπάνω κώδικα XML εμφανίστηκε η καρτέλα με το όνομα «Άσκηση 1.Α» (εικόνα 13).



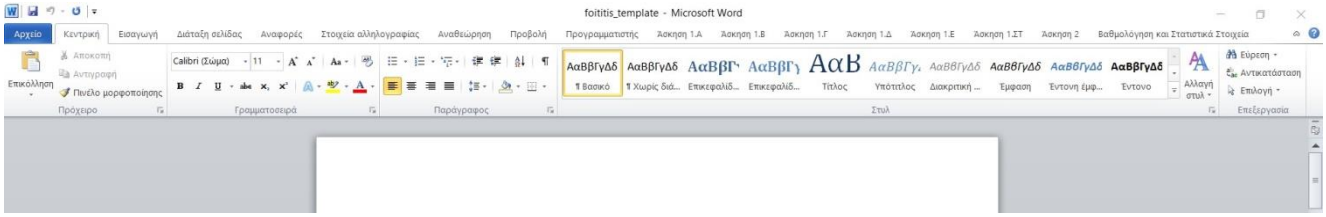
Εικόνα 13

Εκτός όμως από την καρτέλα, ο κώδικας της εικόνας 12 περιέχει και την δημιουργία των κουμπιών με τα αντίστοιχα λάθη. Σημειώστε ότι τα κουμπιά χωρίζονται μεταξύ τους σε δυο ομάδες- συγκεκριμένα στην ομάδα «<group id="diagramma_eurostias" label="Διάγραμμα Ευρωστίας">» που περιέχει τα κουμπιά με τα λάθη και στην ομάδα <group id="undo1" label="Undo"> που περιέχει το κουμπί αναίρεσης (εικόνα 14).



Εικόνα 14

Συνεχίζοντας με την XML κατασκευάστηκαν όλες οι καρτέλες και τα κουμπιά που περιέχονται σε αυτές, καθώς και η επιπλέον καρτέλα της βαθμολόγησης και στατιστικών στοιχείων. Το τελικό αποτέλεσμα φαίνεται στην εικόνα 15.



Εικόνα 15

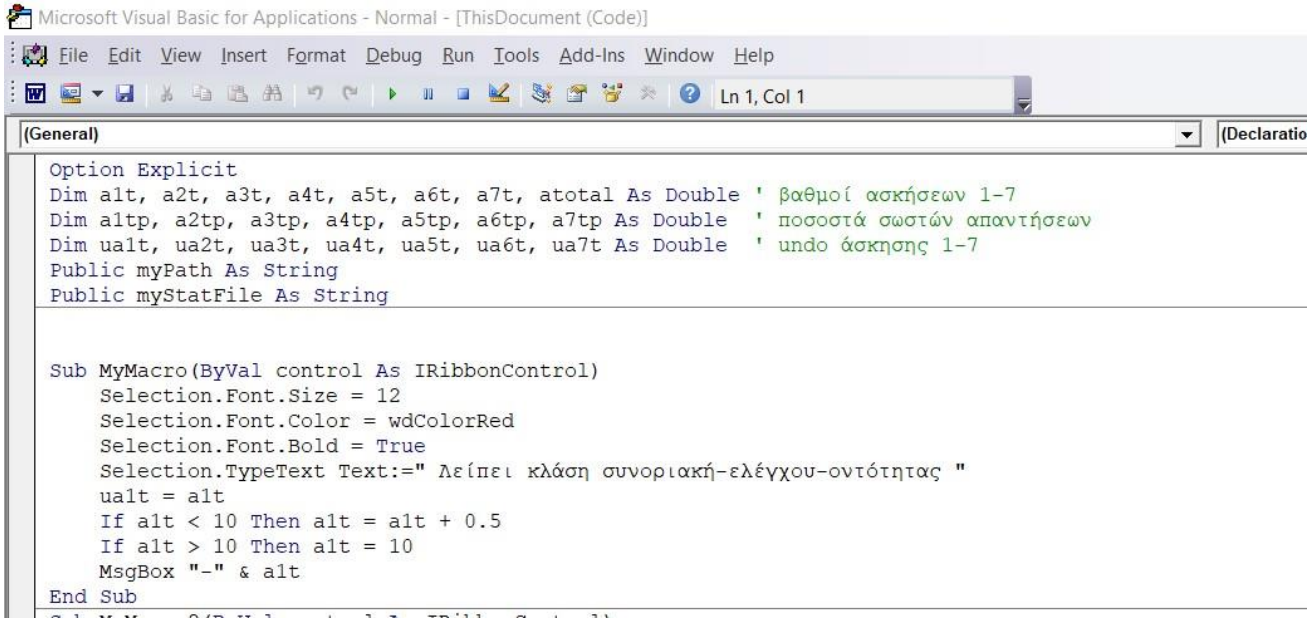
Φυσικά, όλα τα παραπάνω αποτελούν μόνο δομικές τροποποιήσεις και δεν παρέχουν καμία λειτουργικότητα. Αυτή θα δοθεί στη συνέχεια, με προγραμματισμό μέσω της γλώσσας VBA. Επίσης, θα πρέπει να σημειωθεί ότι τα εικονίδια δημιουργήθηκαν με το πρόγραμμα Ζωγραφικής των Windows και εισήχθησαν μέσω XML. Ο πλήρης κώδικας XML παρατίθεται στο Παράρτημα Α: Κώδικας ανάπτυξης.

2.6.2 Εισαγωγή λαθών άσκησης

Από το σημείο αυτό, ξεκινά ο προγραμματισμός με VBA. Όπως είδαμε στο σχετικό κεφάλαιο, όλος ο κώδικας καταχωρείται στον συντάκτη VBE. Ανοίγουμε επομένως τον συντάκτη (με τον τρόπο που αναφέρθηκε παραπάνω) και προχωράμε στον προγραμματισμό με κώδικα.

Η πρώτη εργασία είναι η προσθήκη λειτουργικότητας στα κουμπιά που δημιουργηθήκαν με XML. Θέλουμε το κάθε κουμπί να εκτυπώνει το αντίστοιχο μήνυμα στο κείμενο, στο σημείο που βρίσκεται ο κέρσορας. Παράλληλα, να γίνεται ο υπολογισμός του βαθμού που αφαιρείται με κάθε λάθος. Δημιουργείται η μεταβλητή της συνολικής βαθμολογίας της κάθε άσκησης, η οποία θα χρησιμοποιηθεί αργότερα για τον υπολογισμό της συνολικής βαθμολογίας.

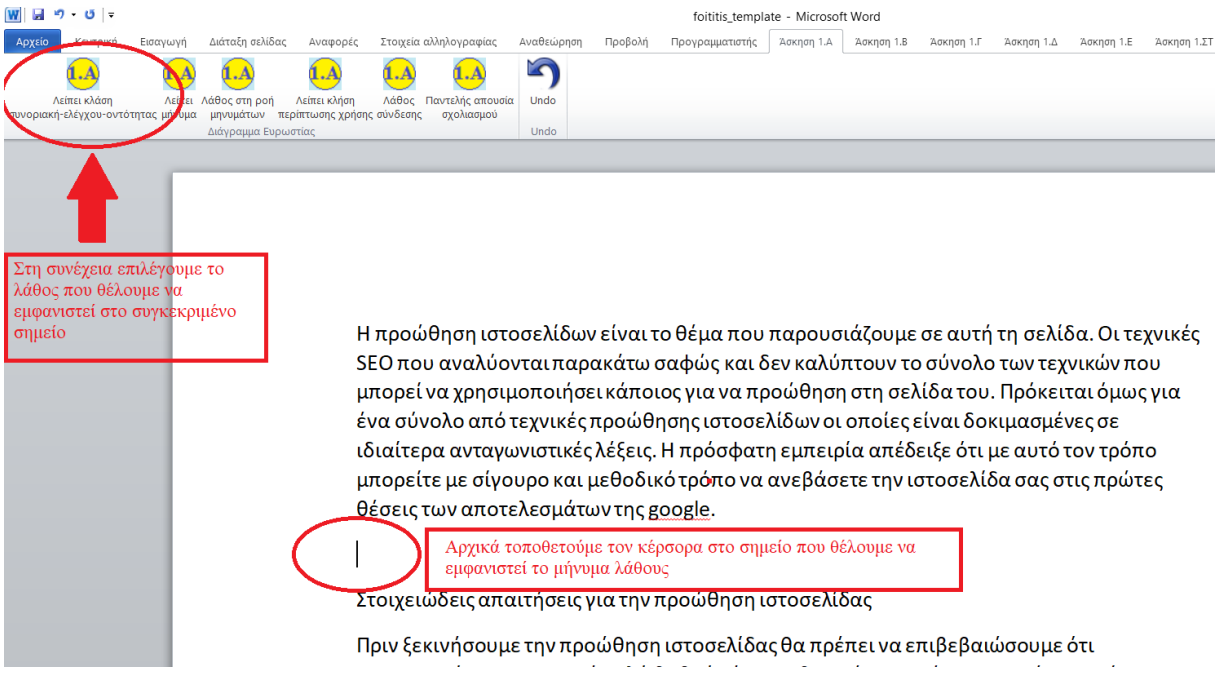
Με παράδειγμα το πρώτο κουμπί της Άσκησης 1.Α, το κουμπί δηλαδή με το λάθος «Λείπει κλάση συνοριακή-ελέγχου-οντότητας», ο κώδικας VBA που υλοποιεί όλα τα παραπάνω φαίνεται στην εικόνα 16.



Εικόνα 16

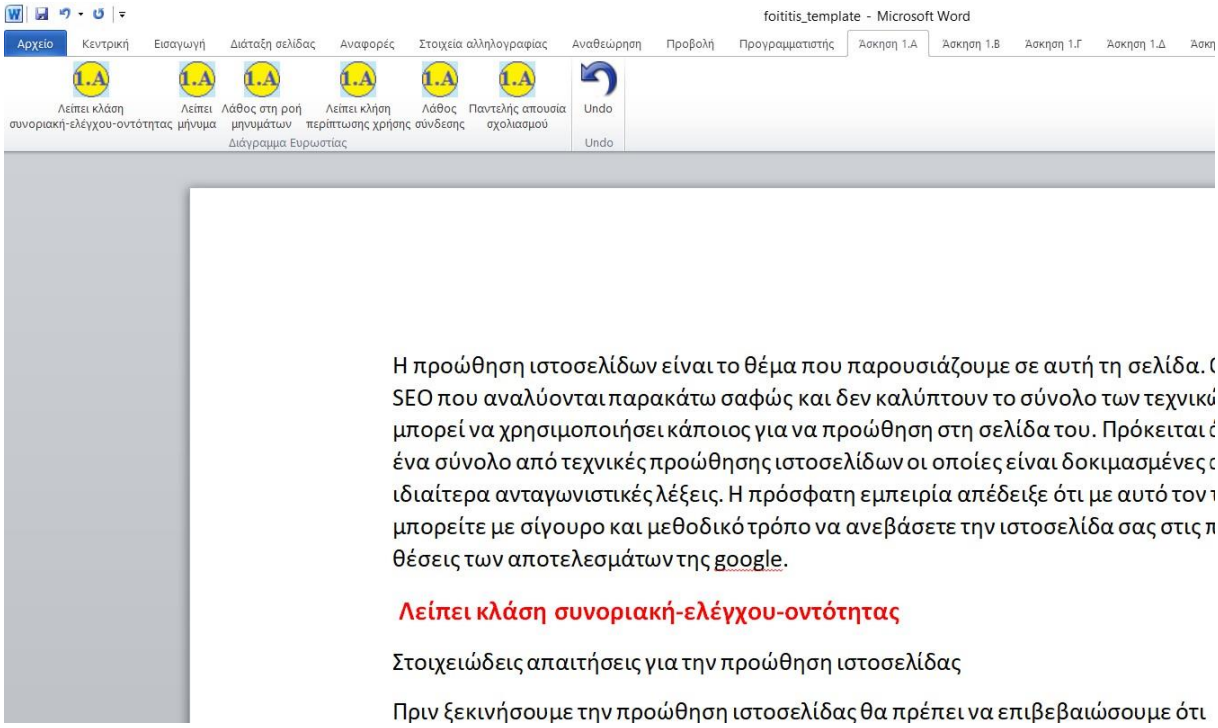
Στον κώδικα της εικόνας 16 είναι εύκολο να διακρίνουμε τους παραμέτρους του μηνύματος λάθους, όπως για παράδειγμα το μέγεθος της γραμματοσειράς, το χρώμα της γραμματοσειράς, η επιλογή της έντονης γραφής και το περιεχόμενο του μηνύματος. Φαίνεται τέλος και ο χειρισμός της βαθμολογίας – στη συγκεκριμένη περίπτωση αφαιρεί 0,5 από τη βαθμολογία, ενώ ο μέγιστος βαθμός που μπορεί να αφαιρεθεί από αυτήν την άσκηση είναι 10. Μπορούμε επίσης να δούμε τις δηλώσεις των μεταβλητών, οι οποίες φυσικά γίνονται άπαξ στην αρχή του κώδικα.

Ας δούμε τώρα το πως υλοποιείται ο παραπάνω κώδικας στο Word. Η εικόνα 17 παρουσιάζει τον τρόπο με τον οποίο πραγματοποιούμε την εισαγωγή λαθών.



Εικόνα 17

Υποθέτοντας ότι ο χρήστης εντοπίζει ένα λάθος στην εργασία του φοιτητή, για να καταχωρήσει το συγκεκριμένο λάθος θα πρέπει αρχικά να τοποθετήσει τον κέρσορα στο σημείο που θέλει να καταχωρηθεί το μήνυμα λάθους. Συστήνεται πάντα ο κέρσορας να τοποθετείται μετά το τέλος μιας πρότασης ή παραγράφου. Αν τυπωθεί το μήνυμα λάθους ενδιάμεσα σε μια πρόταση ή ακόμη χειρότερα ανάμεσα σε μια λέξη, η εισαγωγή του λάθους θα είναι αντιαισθητική και το μήνυμα δυσνόητο. Στη συνέχεια, ο χρήστης θα πρέπει να επιλέξει το κουμπί με το κατάλληλο λάθος- στην περίπτωσή μας το κουμπί «Λείπει κλάση συνοριακή-ελέγχου-οντότητας». Το αποτέλεσμα φαίνεται στην εικόνα 18.

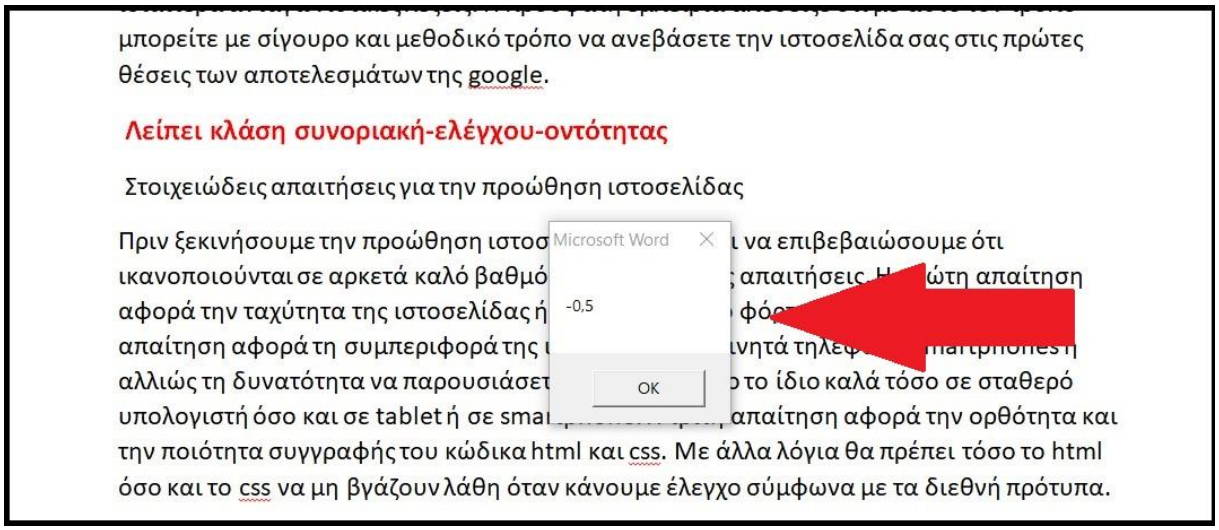


Εικόνα 18

Παράλληλα, όπως έχει ήδη αναφερθεί, αφαιρείται ο αντίστοιχος βαθμός του λάθους. Η ίδια λογική με την παραπάνω επαναλαμβάνεται για όλα τα κουμπιά σε όλες τις καρτέλες. Ο πλήρης κώδικας της λειτουργικότητας των κουμπιών παρατίθεται στο Παράρτημα Α: Κώδικας ανάπτυξης.

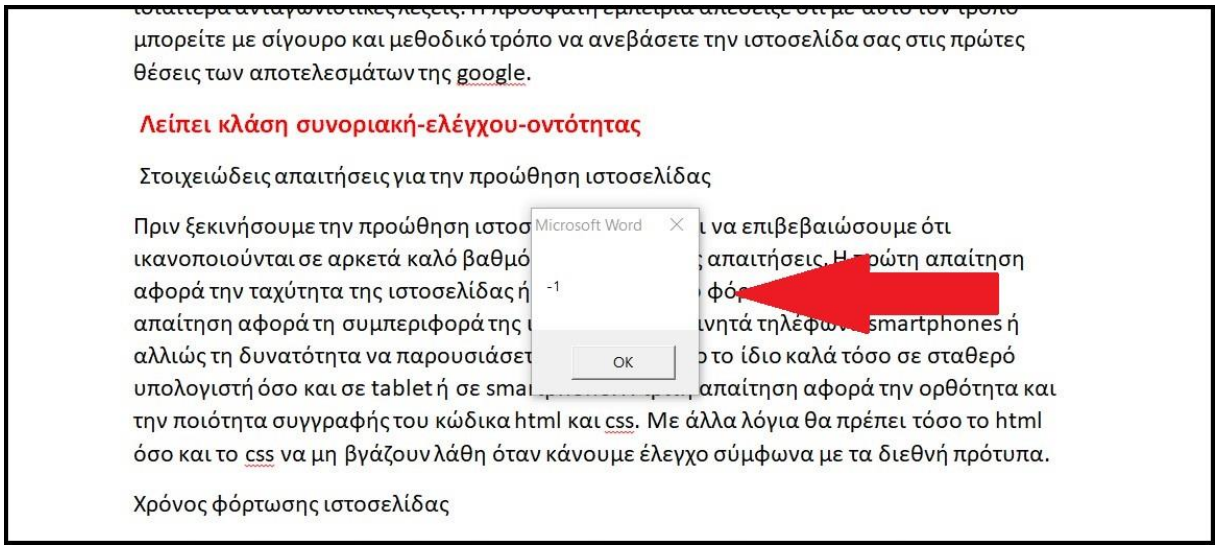
2.6.3 Αναίρεση/ Undo

Κατά τη ροή της ανάπτυξης του πρόσθετου, παρατηρήθηκε ένα πρόβλημα με τη λειτουργία της αναίρεσης του Word. Συγκεκριμένα, όταν γινόταν επιλογή ενός λάθους και μετά χρήση της λειτουργίας αναίρεσης, το Word αναιρούσε την πληκτρολόγηση αλλά όχι όμως και την βαθμολόγηση. Με την προσθήκη ενός message box, η δυσλειτουργία γίνεται εμφανής στις εικόνες 19 και 20. Στην εικόνα 19, έχει καταχωρηθεί το μήνυμα λάθους και μέσω ενός message box μας ενημερώνει ότι αφαιρέθηκε 0,5 από τη βαθμολογία . Στη συνέχεια πατάμε το κουμπί της αναίρεσης του Word. Προσωρινά φαίνονται όλα εντάξει καθώς έχει αναιρεθεί η πληκτρολόγηση.



Εικόνα 19

Στην εικόνα 20 εντοπίζεται το πρόβλημα καθώς, παράλληλα με την εκ νέου καταχώρηση του λάθους, εμφανίζεται το message box το οποίο αποδεικνύει την δυσλειτουργία. Η βαθμολογία είναι πλέον 1 ενώ θα έπρεπε να είναι 0,5. Είναι λοιπόν ξεκάθαρο ότι κράτησε τη βαθμολογία και από το αναιρεμένο λάθος.



Εικόνα 20

Αυτό βεβαίως, δεν θα μπορούσε να γίνει αποδεκτό καθώς τυχόν λάθος στη βαθμολόγηση θα έπληττε την αξιοπιστία της εφαρμογής (και την αξιοπιστία του καθηγητή). Η λύση στο πρόβλημα θα αποτελούσε η κατασκευή μιας προσαρμοσμένης εσωτερικής λειτουργίας αναίρεσης, η οποία θα χρησιμοποιείται αντί της γενικής λειτουργίας αναίρεσης του Word.

Κρίθηκε σκόπιμο να τοποθετηθεί ένα επιπλέον κουμπί σε κάθε καρτέλα, μιας και η τοποθέτηση ενός μόνο γενικού κουμπιού στο μενού της γραμμής εργαλείων γρήγορης πρόσβασης δεν θα ήταν λειτουργική (σε εκείνο το σημείο βρίσκεται και η γενική λειτουργία αναίρεσης του Word). Επιπλέον, είναι ευκολότερο για τον χρήστη να θυμηθεί ότι θα πρέπει να επιλέξει την εσωτερική λειτουργία της αναίρεσης, όταν αυτή βρίσκεται με τα υπόλοιπα κουμπιά που χρησιμοποιεί. Επίσης, ο παραπάνω

σχεδιασμός της τοποθέτησης κάνει ευκολότερη τη δουλειά του προγραμματιστή, καθώς με αυτόν το τρόπο δεν θα χρειαστεί καμιά τροποποίηση στις υπάρχουσες μεταβλητές.

Η τρίτη εναλλακτική, η δημιουργία δηλαδή μιας ξεχωριστής καρτέλας η οποία θα περιέχει το κουμπί της αναίρεσης κρίθηκε ως αντιπαραγωγική. Ο χρήστης θα έπρεπε να κάνει δυο κινήσεις για την αναίρεση, ενώ με την άλλη σχεδίαση το αποτέλεσμα επιτυγχάνεται με μία μόλις κίνηση.

Η δημιουργία των κουμπιών της αναίρεσης σε κάθε καρτέλα έγινε με χρήση της XML. Στην εικόνα 12 μπορούμε να δούμε ότι μετά τη δημιουργία του group των κουμπιών, ακολουθεί και το group που περιλαμβάνει το κουμπί της αναίρεσης. Επιπροσθέτως, στην εικόνα 21 βλέπουμε τον κώδικα VBA, ο οποίος δίνει λειτουργικότητα στο προσαρμοσμένο κουμπί της αναίρεσης.

```
Sub MyMacro12(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
       a2t = ua2t
    End If
End Sub
```

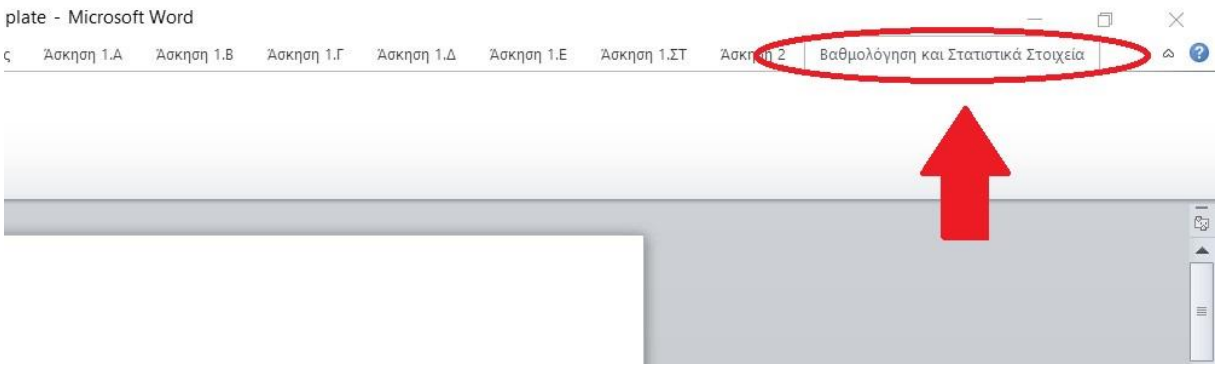
Εικόνα 21

Πλέον, μαζί με το κείμενο αφαιρείται και η δοθείσα βαθμολογία. Σε αντίστοιχο έλεγχο, παρόμοιο με αυτόν που κάναμε πριν με τις εικόνες 19 και 20, θα δούμε ότι πλέον όλα δουλεύουν όπως πρέπει.

Στη συνέχεια, μετά την επιτυχή ολοκλήρωση των διαδικασιών της εισαγωγής των λαθών, της καταγραφής της βαθμολογίας και της προσαρμοσμένης λειτουργίας της αναίρεσης, προχωράμε στο επόμενο βήμα ανάπτυξης : τη λειτουργία βαθμολόγησης.

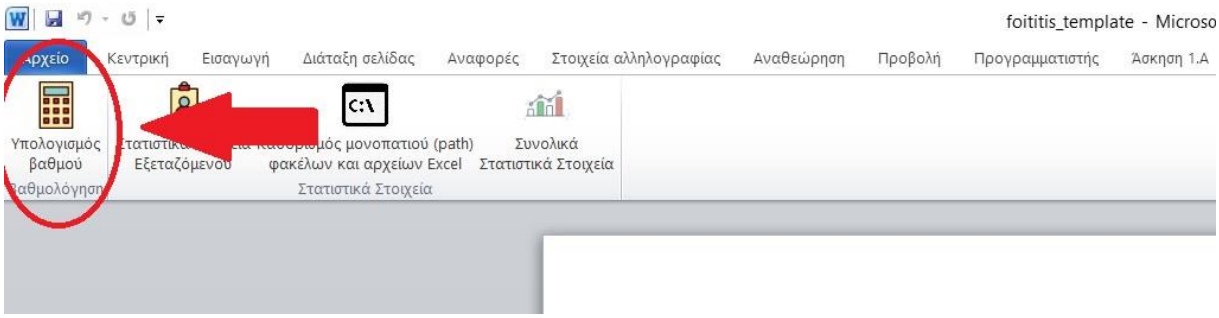
2.6.4 Βαθμολόγηση

Μετά την ολοκλήρωση της διαδικασίας της εύρεσης και καταγραφής των λαθών, ο χρήστης μπορεί να προχωρήσει στην έκδοση της βαθμολογίας. Η βαθμολόγηση πραγματοποιείται μέσω της καρτέλας «Βαθμολόγηση και Στατιστικά Στοιχεία» (εικόνα 22):



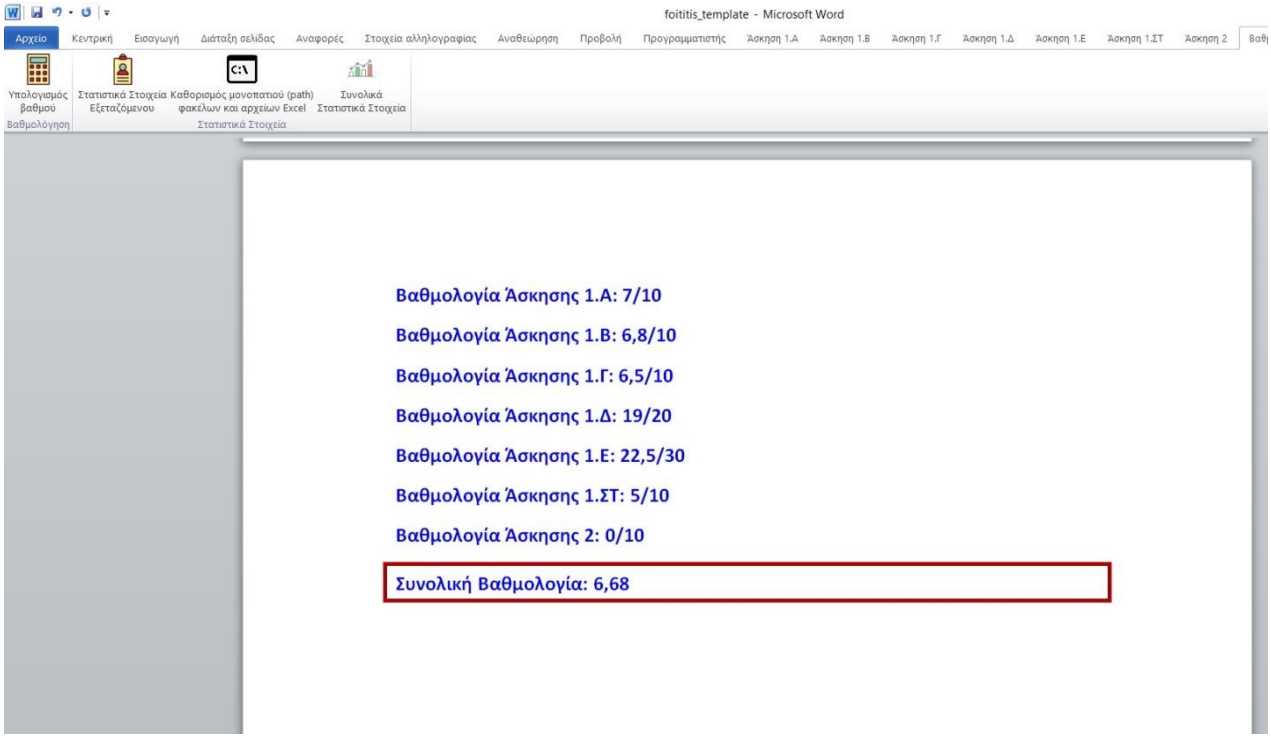
Εικόνα 22

Από αυτήν την καρτέλα επιλέγουμε το κουμπί «Υπολογισμός βαθμού» (εικόνα 23):



Εικόνα 23

Μόλις πατηθεί το κουμπί «Υπολογισμός βαθμού» θα εκτυπώσει την επίδοση του φοιτητή για κάθε επιμέρους άσκηση και την συνολική του βαθμολογία (Εικόνα 24). Τα παραπάνω εκτυπώνονται στην επόμενη της τελευταίας σελίδας και έτσι δεν έχει καμιά σημασία που βρίσκεται ο κέρσορας. Σε αντίθεση δηλαδή με τα μηνύματα λάθους, η βαθμολογία δεν εκτυπώνεται στο σημείο που βρίσκεται ο κέρσορας αλλά πάντα σε νέα σελίδα.



Εικόνα 24

Στην εικόνα 24 φαίνεται ο τρόπος με τον οποίο εκτυπώνεται η βαθμολογία του φοιτητή. Παρατηρούμε ότι κάθε άσκηση έχει την δική της επίδοση ώστε ο φοιτητής να γνωρίζει σε ποια σημεία χρειάζεται να βελτιωθεί και σε ποια σημεία η απόδοσή του ήταν ικανοποιητική. Τέλος, η συνολική βαθμολογία δεν στρογγυλοποιείται αυτόματα και μπορεί να τροποποιηθεί. Αφήνεται δηλαδή στην ευχέρεια του καθηγητή για το αν θα στρογγυλοποιήσει την τελική βαθμολογία προς τα πάνω, προς τα κάτω, ή ακόμη και να τροποποιήσει τον βαθμό κατά βούληση. Για παράδειγμα, αυτή η επιλογή επιτρέπει στον καθηγητή να βοηθήσει βαθμολογικά έναν φοιτητή με υποδειγματική απόδοση στην τάξη.

Ο κώδικας VBA που δίνει λειτουργικότητα στο κουμπί της βαθμολόγησης φαίνεται στην εικόνα 25:

```
Sub MyMacro41(ByVal control As IRibbonControl)
    MsgBox "Υπολογισμός Βαθμολογίας"
    a1tp = (10 - a1t) / 10
    a1tp = Round(a1tp, 2)
    a2tp = (10 - a2t) / 10
    a2tp = Round(a2tp, 2)
    a3tp = (10 - a3t) / 10
    a3tp = Round(a3tp, 2)
    a4tp = (20 - a4t) / 20
    a4tp = Round(a4tp, 2)
    a5tp = (30 - a5t) / 30
    a5tp = Round(a5tp, 2)
    a6tp = (10 - a6t) / 10
    a6tp = Round(a6tp, 2)
    a7tp = (10 - a7t) / 10
    a7tp = Round(a7tp, 2)

    Selection.EndOf Unit:=wdStory, Extend:=wdMove
    Selection.InsertBreak Type:=wdPageBreak
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorBlue
    Selection.Font.Bold = True
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Α: " & 10 - a1t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Β: " & 10 - a2t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Γ: " & 10 - a3t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Δ: " & 20 - a4t & "/20" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Ε: " & 30 - a5t & "/30" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.ΣΤ: " & 10 - a6t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 2: " & 10 - a7t & "/10" & Chr(13)
    atotal = (100 - (a1t + a2t + a3t + a4t + a5t + a6t + a7t)) / 10

    With Selection.Borders
         .Enable = True
         .OutsideLineStyle = wdLineStyleEmboss3D
         .OutsideColor = wdColorRed
         .Shadow = True
         .InsideLineStyle = wdLineStyleEmboss3D
         .InsideColor = wdColorRed
    End With

    Selection.TypeText Text:="Συνολική Βαθμολογία: " & atotal
End Sub
```

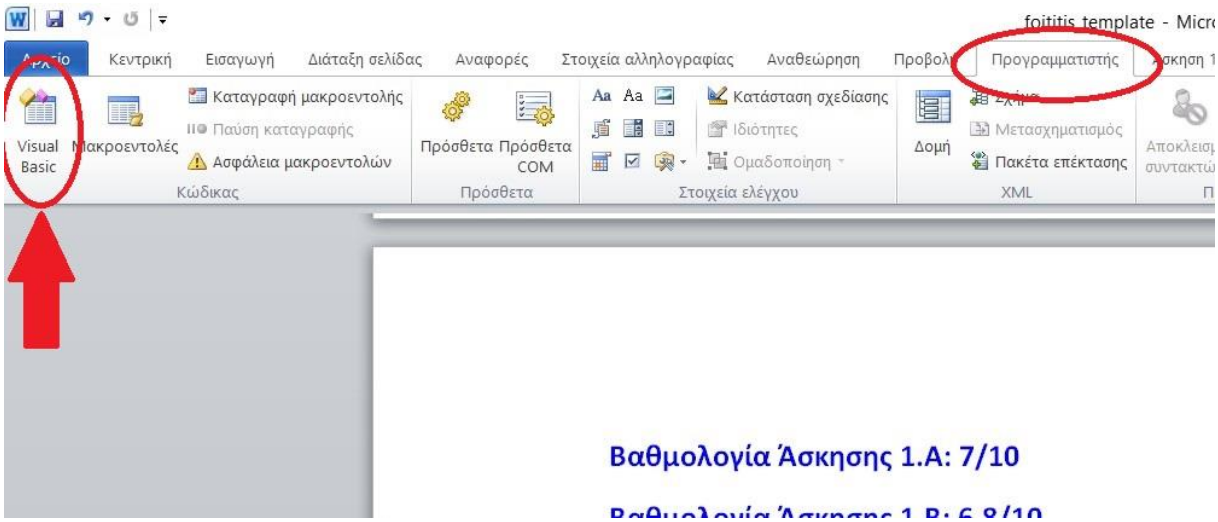
Εικόνα 25

Έχοντας βαθμολογήσει την απόδοση του φοιτητή επιμέρους και συνολικά, μπορούμε να συνεχίσουμε στο επόμενο βήμα, που είναι η προσθήκη των στατιστικών στοιχείων του φοιτητή.

2.6.5 Στατιστικά στοιχεία φοιτητή

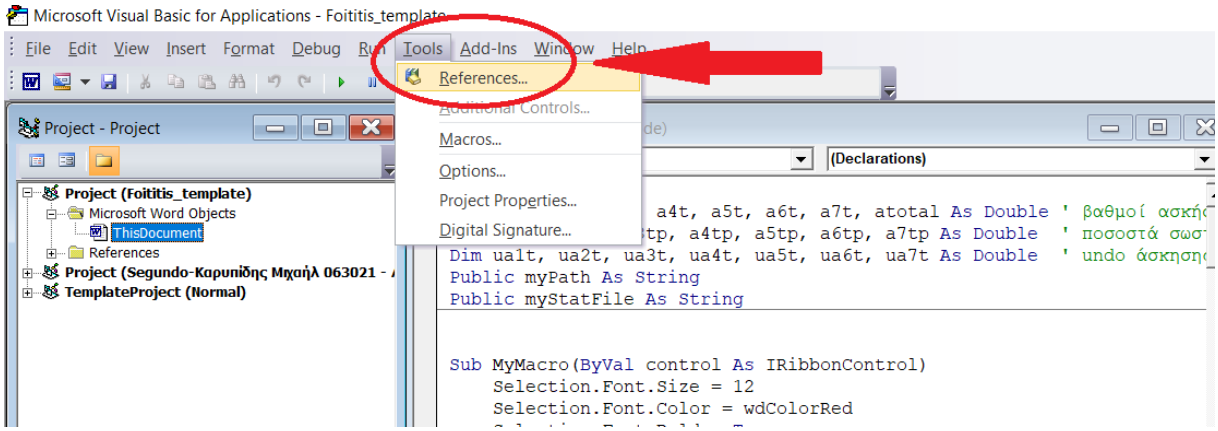
Για την εμφάνιση των στατιστικών στοιχείων του φοιτητή (όπως και για την λειτουργία των συνολικών στατιστικών) θα πρέπει να κληθεί ένα φύλλο εργασίας του Excel, το οποίο είναι απαραίτητο για την κατασκευή ενός διαγράμματος. Έτσι, πριν προχωρήσουμε παρακάτω, θα πρέπει πρώτα να προσθέσουμε στο Word μια αναφορά της βιβλιοθήκης των αντικειμένων του Excel. Για να επιτευχθεί αυτό, θα πρέπει να ακολουθήσουμε τα εξής βήματα:

- Αρχικά, επιλέγουμε το κουμπί «Visual Basic» από την καρτέλα «Προγραμματιστής» (Εικόνα 26):



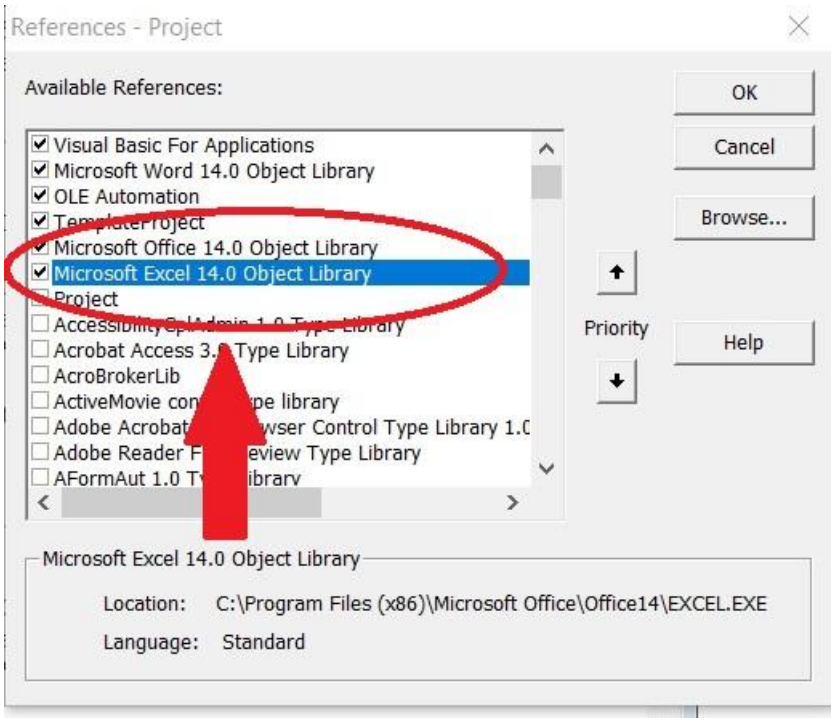
Εικόνα 26

- Ανοίγει ο συντάκτης VBE και από την καρτέλα «Tools» επιλέγουμε «References», όπως φαίνεται στην εικόνα 27:



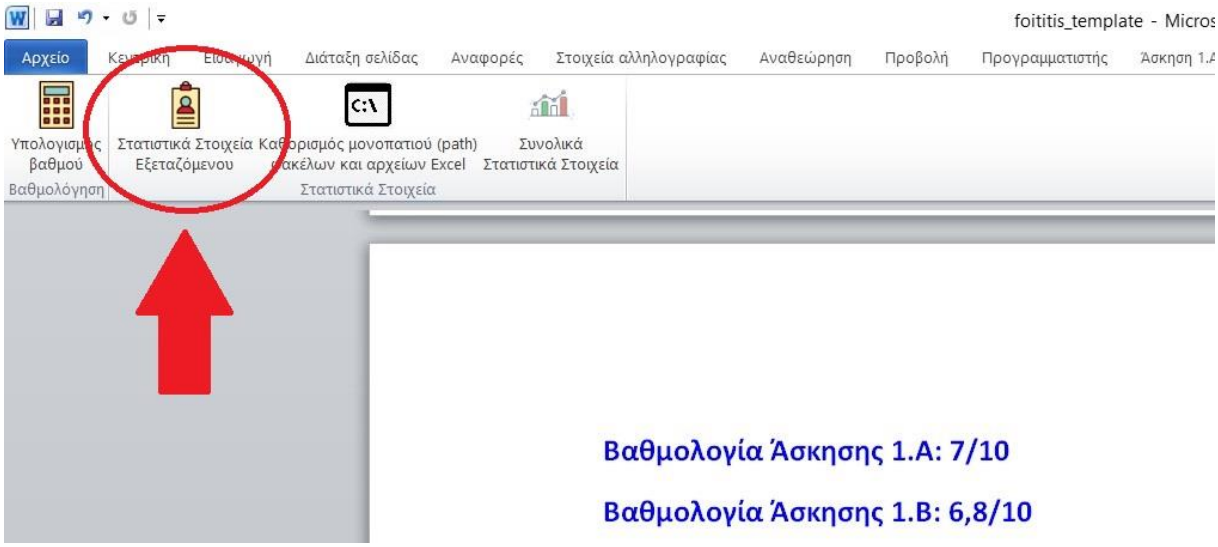
Εικόνα 27

- Εμφανίζεται το παράθυρο «References Project» μέσα στο οποίο ψάχνουμε να βρούμε την επιλογή «Microsoft Excel 14.0 Object Library». Μόλις βρεθεί, φροντίζουμε να είναι επιλεγμένο το κουτάκι που φαίνεται στα αριστερά πριν την επιλογή (Εικόνα 28). Έτσι ενεργοποιούμε την βιβλιοθήκη αντικειμένων του Excel και μπορούμε να συνεχίσουμε. Πρέπει επίσης να σημειωθεί ότι υπάρχουν διάφορες εκδόσεις αυτών των βιβλιοθηκών- μπορεί για παράδειγμα στο δικό σας σύστημα να υπάρχει η έκδοση Microsoft Excel 15.0 Object Library, ή κάποια άλλη. Επιλέξτε την πιο πρόσφατη που διαθέτετε.



Εικόνα 28

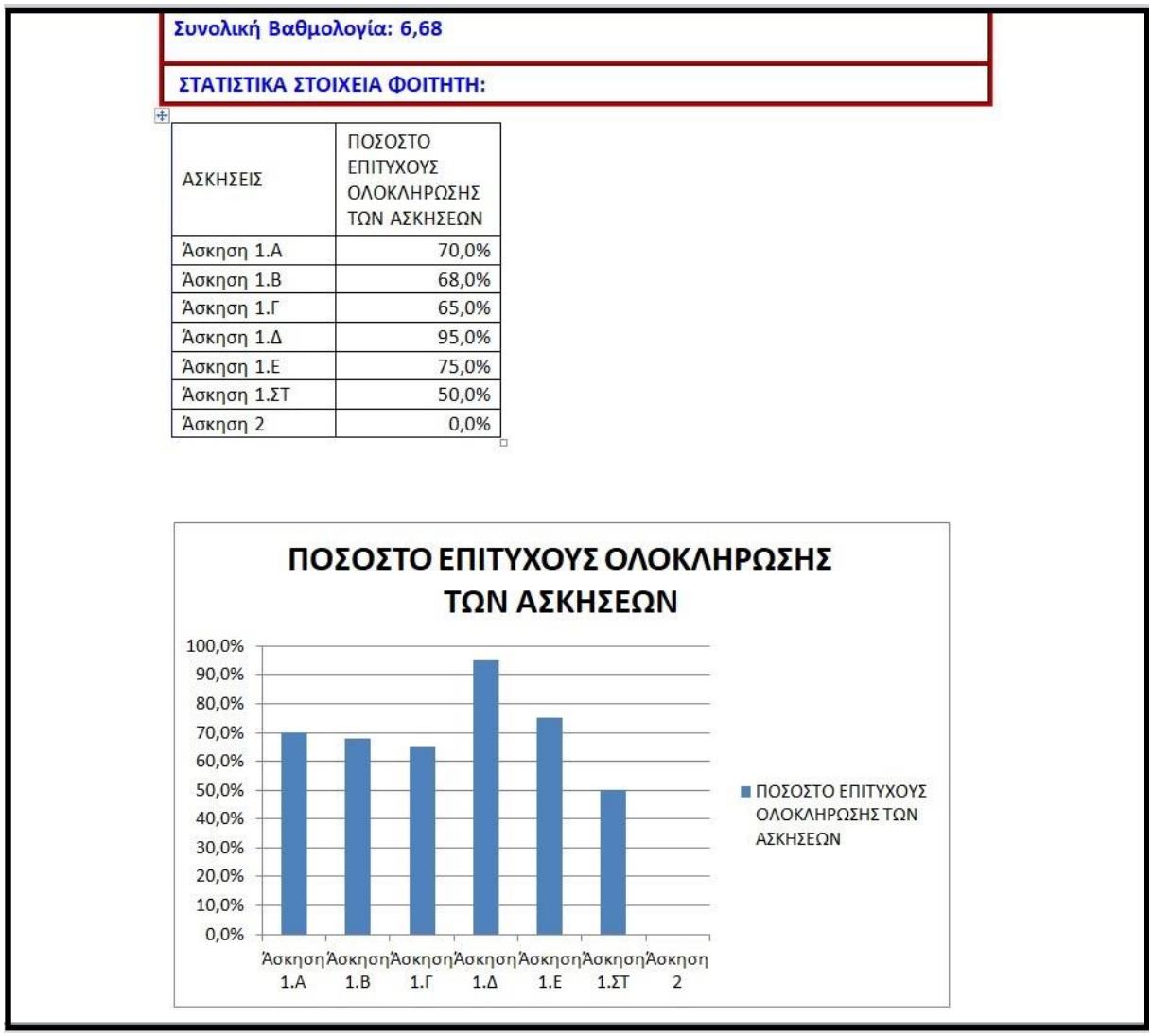
Αφού λοιπόν έχουμε ολοκληρώσει τις παραπάνω οδηγίες και έχουμε προσθέσει επιτυχώς τη βιβλιοθήκη αντικειμένων του Excel, μπορούμε να προχωρήσουμε στην επόμενη λειτουργία. Για να εμφανίσουμε τα στατιστικά στοιχεία του φοιτητή χρησιμοποιούμε το κουμπί «Στατιστικά Στοιχεία Εξεταζόμενου», το οποίο βρίσκεται επίσης στην καρτέλα «Βαθμολόγηση και Στατιστικά Στοιχεία» (Εικόνα 29):



Εικόνα 29

Πατώντας λοιπόν το κουμπί των στατιστικών του φοιτητή, συμπληρώνονται κάτω από την βαθμολογία του ένας πίνακας με τις ασκήσεις και το ποσοστό επιτυχούς κάλυψής τους.

Επιπροσθέτως, εμφανίζεται και ένα διάγραμμα που παρουσιάζει με γραφικό τρόπο το ποσοστό περάτωσης της κάθε άσκησης (Εικόνα 30):



Εικόνα 30

Η επιλογή της βαθμολόγησης είναι ανεξάρτητη από αυτήν των στατιστικών στοιχείων του φοιτητή, δηλαδή μπορεί να χρησιμοποιηθεί ασχέτως αν ακολουθήσει εκτύπωση των στατιστικών στοιχείων ή όχι. Χρησιμοποιήθηκαν δυο διαφορετικά κουμπιά για αυτές τις λειτουργίες ώστε να είναι στην ευχέρεια του χρήστη το αν, πέρα από τη βαθμολόγηση, επιθυμεί να εκτυπώσει και τα στατιστικά του φοιτητή. Προσοχή όμως: η αντίστοιχη σχέση δεν ισχύει. Η λειτουργία των στατιστικών του φοιτητή εξαρτάται άμεσα από τα αποτελέσματα της βαθμολόγησης και έτσι, αν αυτή δεν προηγηθεί, δεν μπορούν να υπολογιστούν τα στατιστικά στοιχεία του φοιτητή.

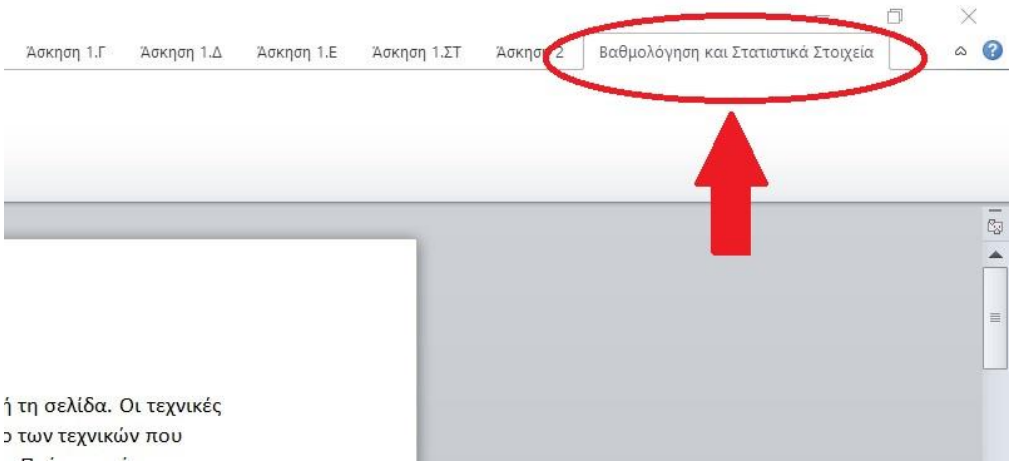
Το ίδιο ισχύει και με την σχέση στατιστικών φοιτητή και συνολικών στατιστικών. Τα στατιστικά φοιτητή είναι ανεξάρτητα από τα συνολικά στατιστικά, μπορούν δηλαδή να χρησιμοποιηθούν αυτόνομα. Τα συνολικά στατιστικά όμως εξαρτώνται άμεσα από τα στατιστικά στοιχεία του φοιτητή και συγκεκριμένα από τον πίνακα με τα ποσοστά επίδοσης, καθώς από εκεί αντλούν τα στοιχεία τους. Αυτός ακριβώς είναι και ο λόγος που εφαρμόζεται αυτόματη στρογγυλοποίηση στα ποσοστά του πίνακα.

Τέλος, ο κώδικας της λειτουργίας των στατιστικών φοιτητή αποτελείται από αρκετές γραμμές κώδικα και έτσι είναι αδόκιμο να παρατεθεί εδώ. Άλλωστε, όπως και ο υπόλοιπος κώδικας, παρατίθεται στο Παράρτημα Α: Κώδικας Ανάπτυξης.

2.6.6 Συνολικά στατιστικά στοιχεία

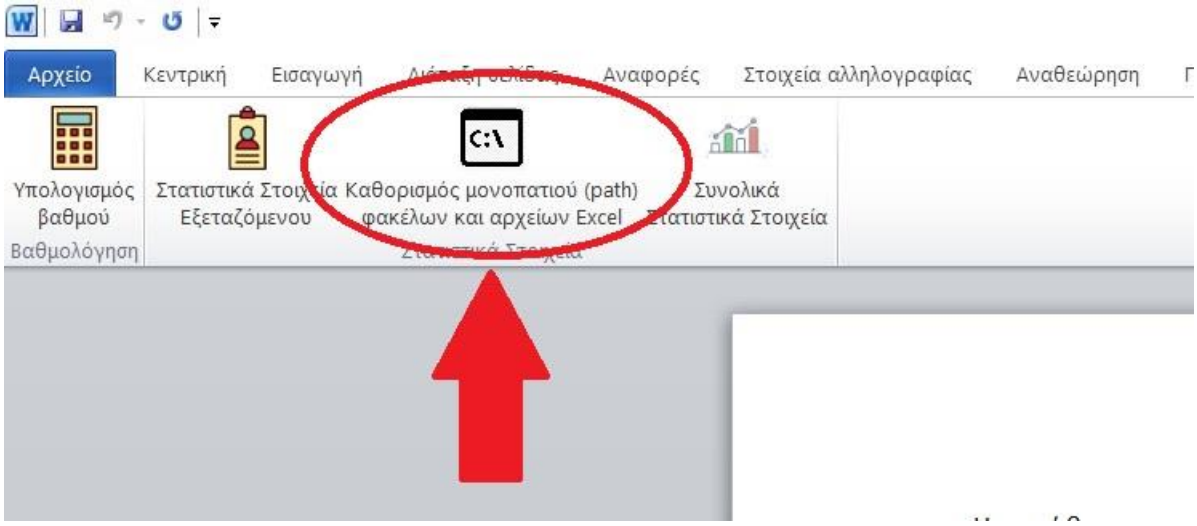
Τα συνολικά στατιστικά αποτελούν τη συλλογή των στοιχείων, τα οποία αντλούνται από ένα σύνολο εγγράφων Word που βρίσκονται σε έναν συγκεκριμένο φάκελο. Όπως έχει αναφερθεί πιο πάνω, τα στοιχεία αυτά αντλούνται από τον πίνακα που παρέχουν τα στατιστικά του φοιτητή. Συγκεκριμένα, σαρώνονται ένα προς ένα τα αρχεία Word που βρίσκονται σε έναν φάκελο και καταγράφονται τα στοιχεία τους σε ένα βιβλίο εργασίας Excel. Ο τρόπος καταχώρησης στο Excel είναι: μια σειρά ανά φοιτητή και επτά στήλες που αντιπροσωπεύουν τις ασκήσεις.

Η διαδικασία είναι ανεξάρτητη από το ενεργό έγγραφο του Word, δηλαδή δεν έχει σημασία ποιο έγγραφο είναι ενεργό. Ξεκινάμε επιλέγοντας την καρτέλα « Βαθμολόγηση και Στατιστικά Στοιχεία» (Εικόνα 31):



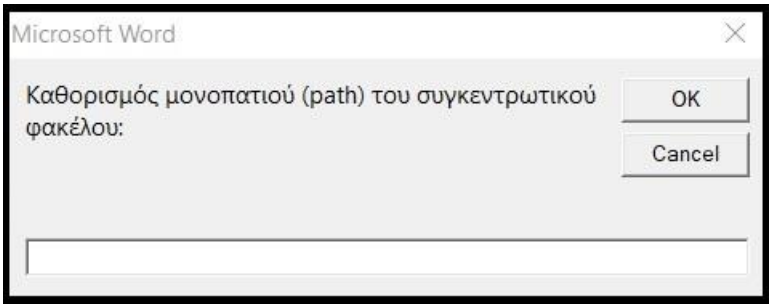
Εικόνα 31

Στη συνέχεια πατάμε το κουμπί «Καθορισμός μονοπατιού (path) φακέλων και αρχείων Excel», όπως αυτό φαίνεται στην εικόνα 32:



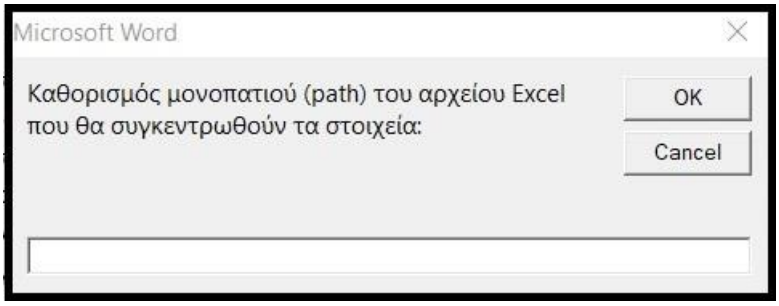
Εικόνα 32

Εμφανίζεται ένα Input Box (εικόνα 33), το οποίο μας ζητάει να εισάγουμε το μονοπάτι του φακέλου στο οποίο περιέχονται τα έγγραφα του Word και από τα οποία θέλουμε να αντλήσουμε τα στοιχεία. Εισάγουμε το μονοπάτι και πατάμε OK.



Εικόνα 33

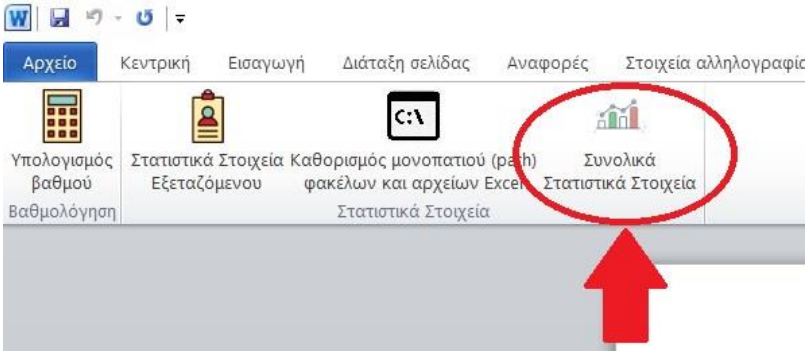
Μόλις πατήσουμε το OK εμφανίζεται ένα νέο Input Box (εικόνα 34), το οποίο μας προτρέπει να εισάγουμε το μονοπάτι του αρχείου Excel στο οποίο θέλουμε να κρατήσουμε τα στατιστικά. Εισάγουμε το μονοπάτι και πατάμε OK. Σημείωση: Το αρχείο Excel δεν θα πρέπει να είναι ανοιχτό. Η συνάρτηση συμπεριλαμβάνει και το άνοιγμά του.



Εικόνα 34

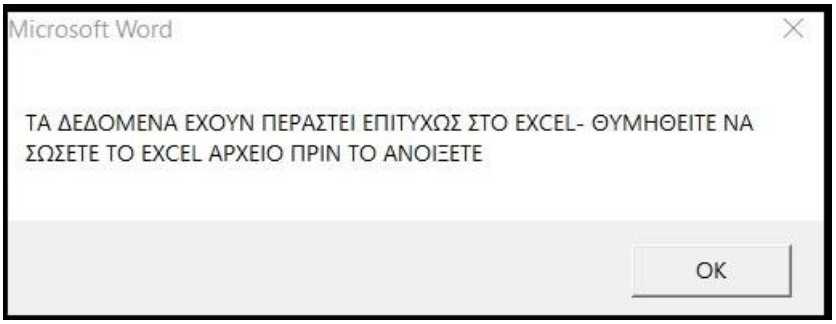
Με το πάτημα του OK, το Input box εξαφανίζεται. Δεν υπάρχει κάποιο μήνυμα επιβεβαίωσης.

Εφόσον έχουμε τελειώσει με τη διαδικασία καθορισμού του μονοπατιού, μπορούμε να προχωρήσουμε στη καταγραφή των στοιχείων. Πατάμε το κουμπί « Συνολικά Στατιστικά Στοιχεία» (Εικόνα 35):



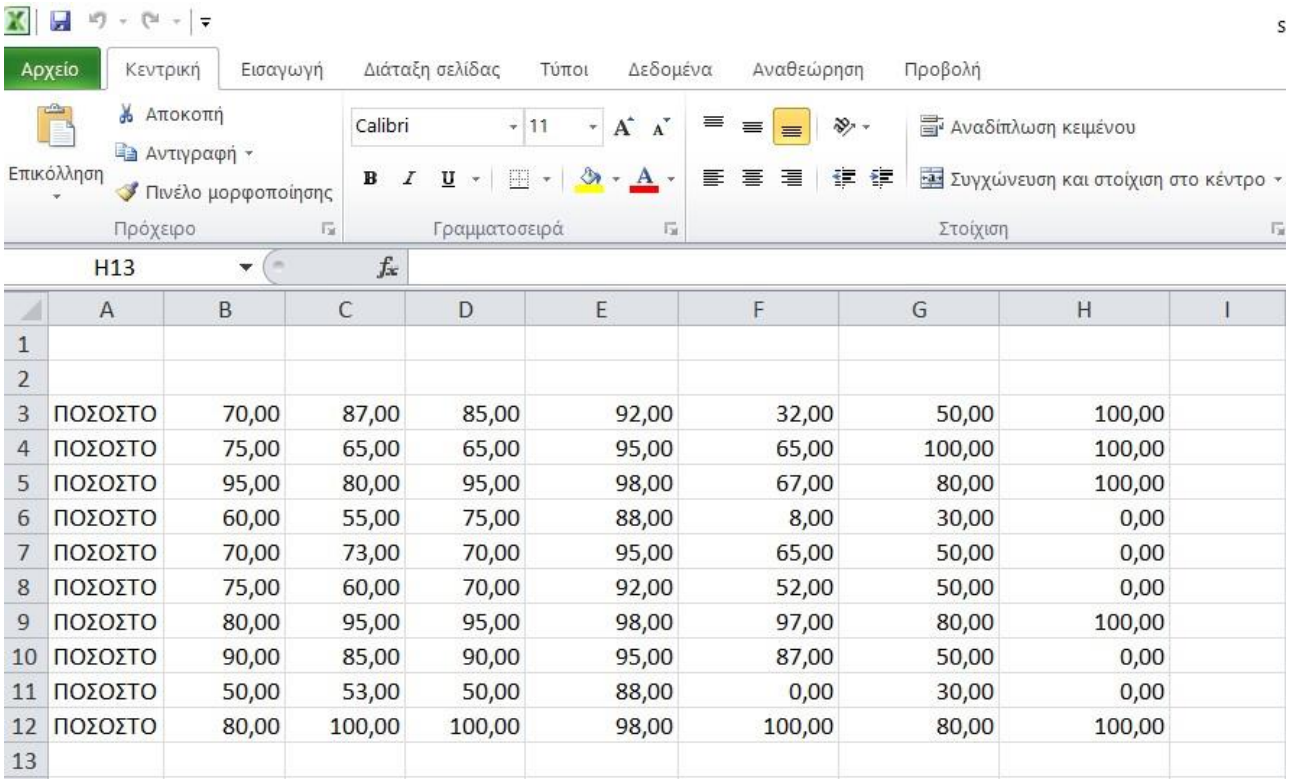
Εικόνα 35

Στο σημείο αυτό η εφαρμογή θα ελέγξει αν έχουν δοθεί τα απαραίτητα για τη διαδικασία μονοπάτια. Αν δεν έχουν δοθεί, θα ζητηθεί η εισαγωγή τους σε αυτό το σημείο (πρακτικά θα καλέσει τη συνάρτηση του καθορισμού μονοπατιού). Μόλις δοθούν θα συνεχιστεί η διαδικασία, θα σαρωθούν δηλαδή τα αρχεία. Με το πέρας της σάρωσης και της άντλησης των στοιχείων θα εμφανιστεί ένα μήνυμα επιβεβαίωσης (Εικόνα 36):



Εικόνα 36

Το ίδιο μήνυμα συνιστά την αποθήκευση του αρχείου του Excel πριν από το άνοιγμα, ώστε αυτό να προλάβει να ενημερωθεί. Ελαχιστοποιώντας όλα τα ενεργά παράθυρα, θα βρούμε το μήνυμα αποθήκευσης του Excel. Επιβεβαιώνουμε και ανοίγουμε το αρχείο. Η μορφή του θα είναι παραπλήσια με αυτήν της εικόνας 37:



	A	B	C	D	E	F	G	H	I
1									
2									
3	ΠΟΣΟΣΤΟ	70,00	87,00	85,00	92,00	32,00	50,00	100,00	
4	ΠΟΣΟΣΤΟ	75,00	65,00	65,00	95,00	65,00	100,00	100,00	
5	ΠΟΣΟΣΤΟ	95,00	80,00	95,00	98,00	67,00	80,00	100,00	
6	ΠΟΣΟΣΤΟ	60,00	55,00	75,00	88,00	8,00	30,00	0,00	
7	ΠΟΣΟΣΤΟ	70,00	73,00	70,00	95,00	65,00	50,00	0,00	
8	ΠΟΣΟΣΤΟ	75,00	60,00	70,00	92,00	52,00	50,00	0,00	
9	ΠΟΣΟΣΤΟ	80,00	95,00	95,00	98,00	97,00	80,00	100,00	
10	ΠΟΣΟΣΤΟ	90,00	85,00	90,00	95,00	87,00	50,00	0,00	
11	ΠΟΣΟΣΤΟ	50,00	53,00	50,00	88,00	0,00	30,00	0,00	
12	ΠΟΣΟΣΤΟ	80,00	100,00	100,00	98,00	100,00	80,00	100,00	
13									

Εικόνα 37

Παρατηρώντας τα στοιχεία μπορούμε να διαπιστώσουμε ότι κάθε σειρά είναι και ένας φοιτητής, ενώ οι στήλες (Range B-H) απεικονίζουν τις επιμέρους ασκήσεις. Τα κελιά περιέχουν το ποσοστό επιτυχούς ολοκλήρωσης της συγκεκριμένης άσκησης, από κάθε φοιτητή. Με αυτά τα στοιχεία ο

χρήστης μπορεί να καταλήξει σε χρήσιμα συμπεράσματα, ανάλογα με τη σύσταση του φακέλου. Για παράδειγμα, αν ο φάκελος περιέχει τα στοιχεία ενός τμήματος, ο χρήστης μπορεί να συγκρίνει την απόδοση του τμήματος σε σχέση με άλλο τμήμα. Μπορεί ακόμη να συγκεντρώσει τις εργασίες όλων των τμημάτων που διδάσκει σε έναν φάκελο, ώστε να λάβει τη γενική εικόνα της απόδοσης των φοιτητών του. Έτσι, θα είναι σε θέση να αναδιαμορφώσει επιτυχώς την παράδοση του μαθήματος, εστιάζοντας περισσότερο στα σημεία που δυσκολεύονται οι φοιτητές. Μπορεί να συγκρίνει την απόδοση των φοιτητών ανά έτος, ώστε να διαπιστώσει αν υπάρχει βελτίωση στην απόδοση ή όχι σε βάθος χρόνου. Μπορεί, αν θέλει, να συγκεντρώσει όλες τις εργασίες κάθε έτους και κάθε τμήματος σε έναν φάκελο, ώστε να έχει τα συνολικά στατιστικά.

Με το βήμα της άντλησης των συνολικών στατιστικών, ολοκληρώθηκε η ανάπτυξη της εφαρμογής. Όπως και στη προηγούμενη περίπτωση, δεν κρίνεται σκόπιμη η παράθεση του κώδικα ανάπτυξης στο σημείο αυτό καθώς αποτελείται από πολλές γραμμές. Θα συμπεριλαμβάνεται και αυτός στον κώδικα που παρατίθεται στο Παράρτημα Α: κώδικας ανάπτυξης.

2.7 Έλεγχος Απόδοσης

Η ιδέα κάποιου υποτυπώδους έστω ελέγχου της απόδοσης προέκυψε από την ανάγκη διασφάλισης της ποιότητας της εφαρμογής. Όπως αναφέρθηκε στο θεωρητικό σκέλος αυτής της πτυχιακής, η αποδοτικότητα είναι ένα από τα βασικά χαρακτηριστικά ενός ποιοτικού λογισμικού. Αυτό σημαίνει ότι θα πρέπει να βρεθεί ένας τρόπος να μετρηθεί (και να αποδειχθεί), ακόμη και σε περιπτώσεις που αυτό φαίνεται ξεκάθαρα. Ο τρόπος που προκρίθηκε για την μέτρηση της απόδοσης σε αυτήν την εφαρμογή, είναι η μέτρηση του χρόνου που θα χρειαστεί ένα πλήθος χρηστών για να ολοκληρώσει την αξιολόγηση ενός γραπτού, αρχικά χωρίς, και κατόπιν με τη χρήση του λογισμικού. Καταγράφοντας τους χρόνους και μελετώντας την οποιαδήποτε διαφορά που τυχόν θα εμφανιστεί, θα προκύψουν χρήσιμα συμπεράσματα ως προς την αποδοτικότητα της εφαρμογής.

2.7.1 Στατιστικό δείγμα

Εκ των πραγμάτων, το στατιστικό δείγμα δεν θα μπορούσε να είναι μεγάλο καθώς, σε αντίθεση με τη συμπλήρωση ενός ερωτηματολογίου, η συγκεκριμένη διαδικασία δεν είναι δυνατόν να ολοκληρωθεί μέσα σε λίγα λεπτά. Αντιθέτως, χρειάζεται εκτεταμένη επεξήγηση για το τι απαιτείται να κάνουν οι χρήστες, μία τουλάχιστον δοκιμαστική χρήση και διπλή καταμέτρηση των πεπραγμένων τους.

Το στατιστικό δείγμα λοιπόν, αποτελείται από εννέα (9) χρήστες: Δυο (2) εξ αυτών είναι φοιτητές, έξι (6) υπηρετούν ως διοικητικοί υπάλληλοι σε δημόσια υπηρεσία και μια (1) είναι καθηγήτρια σε δημόσιο τεχνικό λύκειο (ΕΠΑΛ).

Ακόμη και αν το δείγμα δεν είναι εκτεταμένο, αυτό δεν εμποδίζει την εξαγωγή κάποιων βασικών συμπερασμάτων σχετικά με την αποδοτικότητα της εφαρμογής. Τέλος, μεγάλη προσπάθεια και χρόνος καταναλώθηκε ώστε το δείγμα να μην παρουσιάζει ομοιογένεια.

2.7.2 Όροι και συνθήκες του ελέγχου

Φυσικά, ο έλεγχος απόδοσης της εφαρμογής θα πρέπει να πραγματοποιηθεί κάτω από τους ίδιους όρους και συνθήκες για όλους. Σε περίπτωση που έστω και μία από τις ακόλουθες συνθήκες δεν μπορούσε να ισχύσει, ο χρήστης αποκλειόταν από το στατιστικό δείγμα:

- Όλοι οι συμμετέχοντες πρέπει να έχουν εμπειρία χρήσης και καλή γνώση της σουίτας Office (κυρίως του Word και του Excel).

- Προηγείται πλήρης επεξήγηση της εφαρμογής σε κάθε χρήστη ξεχωριστά, έως ότου να κατανοήσει απόλυτα τη διαδικασία και τις απαιτήσεις της. Δίνονται παραδείγματα από τον επόπτη.
- Δίνεται χρόνος στον χρήστη για μια δοκιμαστική χρήση της εφαρμογής, ώστε να εξοικειωθεί μαζί της.
- Σε όλους του συμμετέχοντες δίνεται το ίδιο κείμενο.
- Παραχωρούνται σε όλους τους συμμετέχοντες η λίστα των λαθών σε ηλεκτρονική μορφή (σε αρχείο pdf), ώστε να μπορούν να χρησιμοποιήσουν τη λειτουργία αντιγραφή/επικόλληση (υποθέτουμε ότι αυτός ήταν ο τρόπος προσθήκης των λαθών προ της εφαρμογής).
- Δίνεται ο τύπος υπολογισμού της βαθμολογίας σε Excel (και πάλι υποθέτουμε ότι αυτός ήταν ο τρόπος με τον οποίο ο καθηγητής υπολόγιζε την βαθμολογία). Εννοείται ότι αν κάποιος από τους συμμετέχοντες προτιμήσει να υπολογίσει την βαθμολογία στο χαρτί, είναι ελεύθερος να το κάνει. Για λόγους ευχρηστίας, προτιμήθηκε να μην συμπεριληφθεί η συνθήκη του ανώτατου ορίου απώλειας βαθμών ανά άσκηση.
- Ο αριθμός των λαθών που θα πρέπει να εισαχθούν στο κείμενο ορίστηκαν στα επτά (7). Είναι ξεκάθαρος ο λόγος που έγινε αυτό: Αν τα λάθη δεν είναι ισάριθμα, θα οδηγηθούμε σε λάθος μετρήσεις.
- Κάθε λάθος θα εισάγεται στο τέλος της κάθε παραγράφου. Δεν θα μπορεί ο χρήστης να βάλει, για παράδειγμα, και τα επτά λάθη στην αρχή του κειμένου, καθώς κάτι τέτοιο θα αφαιρούσε τον ρεαλισμό από την προβλεπόμενη χρήση της εφαρμογής. Αυτό σημαίνει ότι το κείμενο που θα δοθεί θα πρέπει να έχει τουλάχιστον επτά παραγράφους.
- Η επιλογή των λαθών δεν είναι τυχαία. Θα επιλέγεται το πρώτο από κάθε ομάδα. Αυτό γίνεται για να αποφευχθούν εσκεμμένες επιλογές κοντινών λαθών, ή ακόμη και του ίδιου λάθους επτά φορές από τους χρήστες, στην προσπάθειά τους (συνειδητά ή ασυνείδητα) να επιτύχουν καλύτερο χρόνο.
- Η χρονομέτρηση πραγματοποιείται πάντα από τον ίδιο επόπτη και με την ίδια συσκευή χρονομέτρησης. Εννοείται ότι ο χρονομέτρης δεν μπορεί να συμμετάσχει στη διαδικασία ως χρήστης.
- Ο επόπτης –και μόνο αυτός- είναι που καταχωρεί τα δεδομένα. Αυτή η συνθήκη, καθώς και η προηγούμενη, είναι απαραίτητες για λόγους αξιοπιστίας του ελέγχου.

Όλες οι παραπάνω συνθήκες θα πρέπει να ικανοποιούνται για να εισαχθεί κάποιος χρήστης στο στατιστικό δείγμα. Από τη συγκεκριμένη διαδικασία αποκλείστηκαν δύο υποψήφιοι, καθώς κρίθηκε ότι δεν διαθέτουν ικανοποιητικές γνώσεις πάνω στη σουίτα του Office. Τον ρόλο του επόπτη ανέλαβε ο συντάκτης αυτής της πτυχιακής.

2.7.3 Αποτελέσματα

Οι επιδόσεις μετρήθηκαν και καταγράφηκαν σε Excel. Αρχικά, το πρώτο στάδιο για όλους ήταν η καταμέτρηση χωρίς τη χρήση της εφαρμογής. Από αυτήν λοιπόν η καταμέτρηση, η οποία πραγματοποιήθηκε με τους όρους και τις συνθήκες που αναλύθηκαν παραπάνω, προέκυψαν τα αποτελέσματα που παρατίθενται στον πίνακα 2:

	ΧΕΙΡΟΚΙΝΗΤΑ- ΧΩΡΙΣ ΧΡΗΣΗ ΕΦΑΡΜΟΓΗΣ		
ΧΡΗΣΤΗΣ	ΧΡΟΝΟΣ ΕΙΣΑΓΩΓΗΣ ΛΑΘΩΝ	ΧΡΟΝΟΣ ΥΠΟΛΟΓΙΣΜΟΥ ΒΑΘΜΟΛΟΓΙΑΣ	ΣΥΝΟΛΙΚΟΣ ΧΡΟΝΟΣ
ΦΟΙΤ 1	01:03:66	00:41:99	01:45:65
ΦΟΙΤ 2	01:05:12	00:35:01	01:40:22
Δ/Υ 1	01:10:67	00:55:24	02:05:91
Δ/Υ 2	01:08:55	00:37:96	01:46:51
Δ/Υ 3	01:12:32	01:02:45	02:14:77
Δ/Υ 4	01:07:09	01:04:89	02:11:98
Δ/Υ 5	01:16:84	00:48:63	02:05:47
Δ/Υ 6	01:10:93	00:42:76	01:53:69
ΚΑΘΗΓ 1	01:08:68	00:59:47	02:08:15

Πίνακας 2: Μετρήσεις χωρίς χρήση εφαρμογής

Βλέπουμε ότι η διαδικασία χωρίστηκε σε δυο χρονικές περιόδους: Στον χρόνο που χρειάστηκαν οι χρήστες να εισάγουν τα προσυμφωνημένα λάθη στο κείμενο και στον χρόνο που χρειάστηκαν για τον υπολογισμό της βαθμολογίας. Η τελευταία στήλη είναι το άθροισμα των δυο χρόνων και αποτελεί το συνολικό χρόνο που απαιτήθηκε για την ολοκλήρωση της διαδικασίας.

Στο δεύτερο στάδιο μετρήθηκαν και καταγράφηκαν οι αποδόσεις των χρηστών με τη χρήση της εφαρμογής. Τα στοιχεία που προέκυψαν φαίνονται στον πίνακα 3:

	ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΑ- ΧΡΗΣΗ ΕΦΑΡΜΟΓΗΣ		
ΧΡΗΣΤΗΣ	ΧΡΟΝΟΣ ΕΙΣΑΓΩΓΗΣ ΛΑΘΩΝ	ΧΡΟΝΟΣ ΥΠΟΛΟΓΙΣΜΟΥ ΒΑΘΜΟΛΟΓΙΑΣ	ΣΥΝΟΛΙΚΟΣ ΧΡΟΝΟΣ
ΦΟΙΤ 1	00:26:53	0	00:26:53
ΦΟΙΤ 2	00:30:52	0	00:30:52
Δ/Υ 1	00:35:94	0	00:35:94
Δ/Υ 2	00:33:12	0	00:33:12
Δ/Υ 3	00:36:46	0	00:36:46
Δ/Υ 4	00:32:81	0	00:32:81
Δ/Υ 5	00:37:02	0	00:37:02
Δ/Υ 6	00:33:52	0	00:33:52
ΚΑΘΗΓ 1	00:34:27	0	00:34:27

Πίνακας 3: Μετρήσεις με χρήση εφαρμογής

Παρατηρούμε ότι η στήλη του χρόνου υπολογισμού της βαθμολογίας είναι μηδενική, καθώς δεν μπορεί να μετρηθεί ο χρόνος πατήματος ενός κουμπιού (τουλάχιστον όχι με συμβατικά μέσα). Έτσι ο συνολικός χρόνος συμπίπτει με τον χρόνο που απαιτήθηκε για την εισαγωγή των λαθών στο κείμενο.

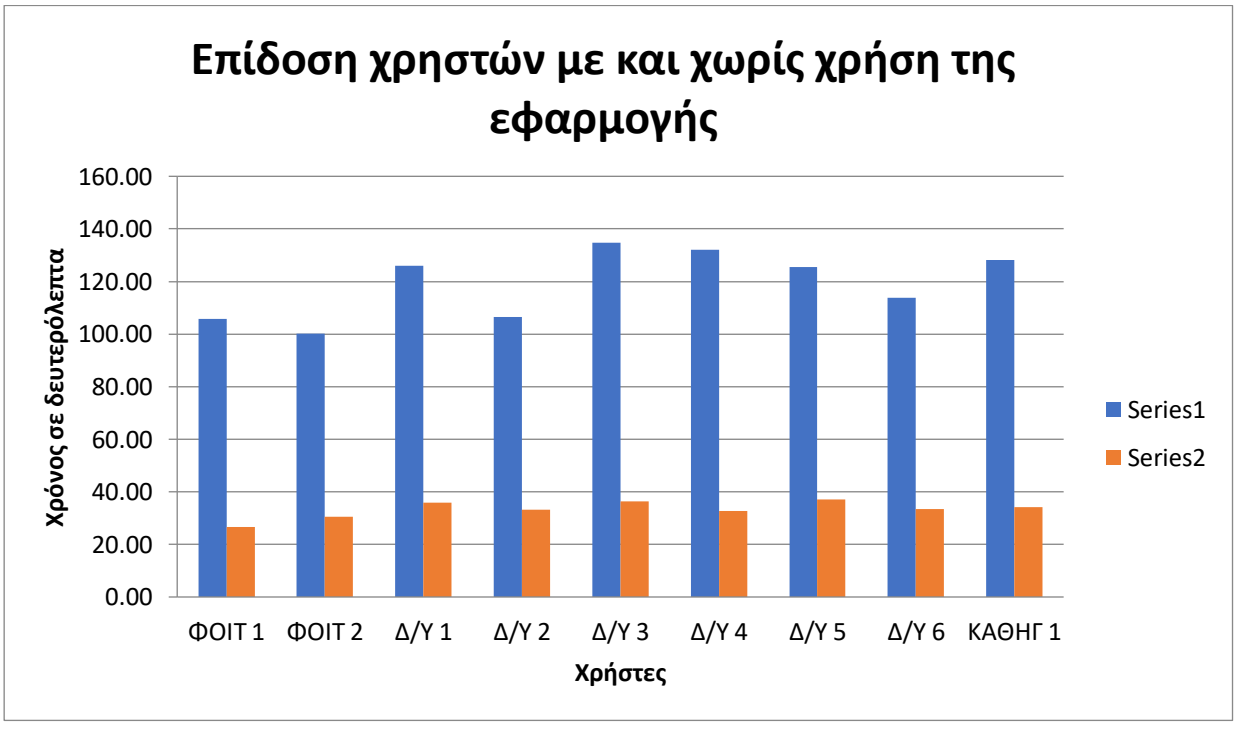
Με την επεξεργασία των δυο προηγούμενων πινάκων προέκυψαν χρήσιμα συμπεράσματα. Αυτά απεικονίζονται στον πίνακα 4:

ΧΡΗΣΤΗΣ	ΔΙΑΦΟΡΑ ΧΡΟΝΟΥ	ΠΟΣΟΣΤΟ ΒΕΛΤΙΩΣΗΣ
ΦΟΙΤ 1	01:19:12	74,9
ΦΟΙΤ 2	01:09:70	69,5
Δ/Υ 1	01:29:97	71,4
Δ/Υ 2	01:13:39	68,9
Δ/Υ 3	01:38:31	72,9
Δ/Υ 4	01:39:17	75,1
Δ/Υ 5	01:28:45	70,5
Δ/Υ 6	01:20:17	70,5
ΚΑΘΗΓ 1	01:33:88	73,2

Πίνακας 4: Συμπεράσματα του ελέγχου απόδοσης

Στον πίνακα 4, βλέπουμε την χρονική διαφορά που προέκυψε από την αφαίρεση του χρόνου που απαιτήθηκε με τη χρήση της εφαρμογή από τον χρόνο που απαιτήθηκε για την χειροκίνητη ολοκλήρωση. Βλέπουμε ότι σε όλους το χρήστες η διαφορά είναι πάνω από ένα λεπτό, σε ορισμένους μάλιστα η διαφορά ξεπερνάει το ενάμιση λεπτό. Το ποσοστό βελτίωσης εμφανίζεται στην τρίτη και τελευταία στήλη και βλέπουμε ότι η απόδοση όλων είναι περίπου στο 70%. Ο μέσος όρος του ποσοστού της βελτίωσης ανέρχεται στο 71,87%.

Στην εικόνα 38 απεικονίζεται γραφικά η διαφορά της επίδοσης για κάθε χρήστη:



Εικόνα 38

Η σειρά ένα απεικονίζει τον χρόνο που απαιτήθηκε χειροκίνητα και η σειρά 2 με την αυτοματοποίηση που προσφέρει η εφαρμογή. Ο κάθετος άξονας απεικονίζει τον χρόνο σε δευτερόλεπτα ενώ στον οριζόντιο άξονα βρίσκονται οι χρήστες. Είναι ξεκάθαρο και πέραν κάθε αμφιβολίας η βοήθεια που προσφέρει η εφαρμογή στους χρήστες.

Στην παρακάτω εικόνα απεικονίζονται γραφικά τα ποσοστά αυτής της βελτίωσης:

Εικόνα 39

Αν και οι παραπάνω μετρήσεις αποδεικνύουν ξεκάθαρα την αποδοτικότητα της εφαρμογής - πράγμα που ήταν και ο αρχικός σκοπός μας-, σε καμία περίπτωση δεν διατείνονται ότι οι συνθήκες του ελέγχου της απόδοσης προσομοιάζουν απόλυτα με τις πραγματικές συνθήκες της βαθμολόγησης. Η βαθμολόγηση είναι πολύ πιο σύνθετη διαδικασία από την απλή εκτέλεση ορισμένων εργασιών, όπως έγινε στον έλεγχο της απόδοσης που πραγματοποιήθηκε παραπάνω. Η εφαρμογή από την άλλη όμως, δημιουργήθηκε ώστε να βοηθήσει τον καθηγητή και όχι να τον αντικαταστήσει.

Τέλος, η χρήση μεγαλύτερου στατιστικού δείγματος , το οποίο θα αποτελούταν αποκλειστικά από καθηγητές, σε πραγματικές εργασίες φοιτητών και για μεγάλο χρονικό διάστημα (για παράδειγμα ενός έτους) θα παρουσίαζε σαφώς λεπτομερέστερα δεδομένα για ανάλυση. Δυστυχώς, δεν ήταν δυνατόν να καλυφθούν οι παραπάνω απαιτήσεις. Σε κάθε περίπτωση όμως τα αποτελέσματα που προέκυψαν, έστω και με τις περιορισμένες δυνατότητες που υπήρχαν, θεωρούνται ενδεικτικά της απόδοσης της εφαρμογής.

2.8 Επίλογος

Σε αυτό το κεφάλαιο εστιάσαμε στην εφαρμογή και στα τεχνικά χαρακτηριστικά της. Αρχικά αναλύθηκαν οι απαιτήσεις του πελάτη, πάνω στις οποίες βασίστηκε η ανάπτυξη της εφαρμογής, και ο τρόπος του σχεδιασμού των λύσεων. Στη συνέχεια παρατέθηκαν κάποια στοιχεία σχετικά με τη γλώσσα προγραμματισμού που επιλέχθηκε, δηλαδή την VBA, και επεξηγήθηκαν οι λόγοι που οδήγησαν σε αυτήν την επιλογή. Έγινε αναφορά στον συντάκτη της VBA , τον VBE, που πάνω σε αυτόν συντάσσεται ο κώδικας. Δόθηκαν επίσης και οι οδηγίες ενεργοποίησής του. Ακολούθως αναλύθηκαν οι δυο τρόποι εγκατάστασης: ο μόνιμος τρόπος με την αντικατάσταση του εργοστασιακού προτύπου και η μεμονωμένη φόρτωση και χρήση του προτύπου από τον χρήστη. Αργότερα αναλύθηκε η ανάπτυξη της εφαρμογής βήμα προς βήμα, εξηγώντας παράλληλα τη λογική που ακολουθήθηκε. Αναλύθηκαν τα δομικά κομμάτια που δημιουργήθηκαν με XML, όπως και τα λειτουργικά που αναπτύχθηκαν με VBA. Το κεφάλαιο ολοκληρώνεται με τον έλεγχο απόδοσης που επιχειρήθηκε, καταγράφοντας την απόδοση των χρηστών και εξάγοντας χρήσιμα συμπεράσματα.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Vermaat, Misty E., *Microsoft Office 2013 Introductory*. Cengage Learning, p.IT3. 2014.

[2] Kruchten P., *The Nature of Software What's so special about software engineering?* , International Conference « The Sciences of Design », Lyon (France), March 15-16, 2002.

[3] Brooks, F., *The Mythical Man-Month*, Anniversary Edition. Boston, MA, USA: Addison Wesley Longman Inc. 1995.

[4] Recommended Practice for Architectural Description of Software-Intensive Systems, in The Institute of Electrical and Electronics Engineers (IEEE) Standards Board, (IEEE-Std-1471-2000, September 2000).

[5] I. Sommerville, *Software engineering (8th ed.)* (Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006).

[6] C. Ghezzi, M. Jazayeri and D. Mandrioli, *Fundamentals of Software Engineering, second edn.* (Prentice Hall, September 2002).

[7] W. Emmerich, *Engineering Distributed Objects* (John Wiley and Sons Ltd, 2000).

[8] Maurice V. Wilkes. *Memoirs of a Computer Pioneer (History of Computing)*, 1985.

[9] A. Oettinger, *"President's Letter to the ACM Membership,"* Comm. ACM, vol. 9, no. 8, 1966.

[10] G. Boole, *The Laws of Thought*, Walton and Maberly, 1854.

[11] W.J. Eckert, *Punched Card Methods in Scientific Computing*, Thomas J. Watson Astronomical Computing Bureau, Columbia Univ., 1940.

[12] J. von Neumann, *First Draft of a Report on the EDVAC*, US Army Ordinance Department and Univ. Pennsylvania Moore School of Electrical Eng., 1945.

[13] E. W. Dijkstra. *Notes on structured programming*. In Structured Programming. O.-J. Dahl, E. W. Dijkstra and C.A.R. Hoare, Acad. Press, 1972.

[14] C.A.R. Hoare. *Notes on data structuring*. In Structured Programming. O.-J. Dahl, E. W. Dijkstra and C.A.R. Hoare, Acad. Press, 1972.

[15] N. Wirth. *The Programming Language Pascal*. Acta Informatica 1, (1971) 35 – 63.

[16] E. W. Dijkstra, *Cooperating sequential processes*. Sept. 1965. Reprinted in Programming Languages, F. Genuys, Ed., Acad. Press, New York, 1968, 43-112.

[17] C.A.R. Hoare. *Communicating sequential processes Comm. ACM*, 21, 8 (August 1978) pp. 666 - 677.

[18] Booch, Grady (1993). *Object-oriented Analysis and Design with Applications (2nd ed.).* Redwood City: Benjamin Cummings. ISBN 0-8053-5340-2.

[19] Kruchten, Philippe (1995, November). *Architectural Blueprints — The "4+1" View Model of Software Architecture*. IEEE Software 12 (6), pp. 42-50.

[20] Mills, H.; M. Dyer; R. Linger (September 1987). "*Cleanroom Software Engineering*", IEEE Software. 4 (5): 19–25. doi:10.1109/MS.1987.231413. S2CID 383170.

[21] David F. Rico, *Short history of software methods*, 2005, pp.1.

[22] Douglas T. Ross, John B. Goodenough, C.A. Irvine, *Software Engineering: Process, Principles, and Goals*, May 1975, pp. 17-27, vol. 8.

[23] G. J. Myers, "*Characteristics of Composite Design*," Datamation, vol. 19, pp. 100-102, September 1973.

[24] Donald Gotterbarn, Keith Willam Miller, Simon Rogerson, Steven Barber, *Software Engineering Code of Ethics and Professional Practice* ,Science and Engineering Ethics, Volume 7, Issue 2, 2001.

[25] Crosby P.; (1979), *Quality is Free*; New York, McGraw Hill, 1979.

[26] Bevan N., Azuma M. (1997), *Quality in Use: Incorporating Human Factors into the Software Engineering Lifecycle*, 3rd international Software Engineering Standards Symposium (ISESS '97), 1997, pp 169.

[27] ISO9126, International Standards Organization. *Software Engineering-Product Quality-Part 1: Quality Model, ISO/IEC Standard 9126-1*. (2001) International Organization for Standardization/International Electrotechnical Commission: Geneva Switzerland.

[28] Garvin D., (1984), What does product quality really mean? , Sloane Management Review, pp 25-43.

[29] McCall J., Richards P., Walters G.; (1977); "*Factors in Software Quality, Vols I, II, III*", US Rome Air Development Center Reports NTIS AD/A-049 014,015, 055, 1977.

[30] Boehm, B.: *Characteristics of software quality* (1978).

[31] Dromey, G.: *A Model for Software Product Quality*. IEEE Transactions on Software Engineering 146, 21 (1995).

[32] S. S. Stevens, *On the Theory of Scales of Measurement*, Science, vol. 103, pp. 677-680, 1946.

[33] L. Finkelstein, *Theory and Philosophy of Measurement*, in Theoretical Fundamentals, vol. 1, Handbook of Measurement Science, P. H. Sydenham, Ed. Chichester: John Wiley & Sons, 1982, pp. 1-30.

[34] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, 2nd Edition Revised ed. Boston: PWS Publishing, 1997.

[35] Rising, L., & Janoff, N. S., *The Scrum software development process for small teams*, IEEE software, 17(4), (pp. 26-32), 2000.

[36] Glaiel, F., Moulton, A., & Madnick, S., *Agile project dynamics: A system dynamics investigation of agile software development methods*. Working Paper CISL# 2013= 05). Cambridge, MA: MIT, 2013.

[37] Cho, J., *Issues and Challenges of agile software development with SCRUM*, Issues in Information Systems, 9(2), (pp. 188-195), 2008.

[38] Cohen, David, Mikael Lindvall, and Patricia Costa., *An introduction to agile methods*, Advances in computers, 62, (pp. 1-66), 2004.

[39] B W Tuckman, *Developmental Sequence in Small Groups*, Psychological Bulletin 63, 1965.

[40] Tsui, F., & Karam, O., *Essentials of Software Engineering*, Sudbury, MA: Jones and Bartlett Publishers, 2007.

[41] Cohn, M., *User Stories Applied for Agile Software Development*, Boston, MA: Pearson Education, 2004.

[42] Coplien, J., *Lean Deconstruction*, Lean Magazine , 18-19, 2009.

ΠΑΡΑΡΤΗΜΑ Α : ΚΩΔΙΚΑΣ ΑΝΑΠΤΥΞΗΣ

XML

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
<ribbon startFromScratch="false">
<tabs>
<tab id="_1a" label="Άσκηση 1.Α">
<group id="diagramma_eurostias" label="Διάγραμμα Ευρωστίας">
<button id="_1a_1" label="Λείπει κλάση συνοριακή-ελέγχου-οντότητας" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro" />
<button id="_1a_2" label="Λείπει μήνυμα" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro2" />
<button id="_1a_3" label="Λάθος στη ροή μηνυμάτων" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro3" />
<button id="_1a_4" label="Λείπει κλήση περίπτωσης χρήσης" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro4" />
<button id="_1a_5" label="Λάθος σύνδεσης" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro5" />
<button id="_1a_6" label="Παντελής απουσία σχολιασμού" image="_x0031_A" size="large"
onAction="ThisDocument.MyMacro6" />
</group>
<group id="undo1" label="Undo">
<button id="Undo" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro7"/>
</group>
</tab>
<tab id="_1b" label= "Άσκηση 1.Β">
<group id="diagramma_akolouthias" label="Διάγραμμα Ακολουθίας">
<button id="_1b_1" label="Λείπει αντικ. οντοτήτων (ΑΟ) του δ. Ευρωστίας" image="_x0031_B" size="large"
onAction="ThisDocument.MyMacro8" />
<button id="_1b_2" label="Λείπει αντικ. ελέγχου (ΑΕ) του δ. Ευρωστίας " image="_x0031_B" size="large"
onAction="ThisDocument.MyMacro9" />
<button id="_1b_3" label="Λείπει μήνυμα" image="_x0031_B" size="large"
onAction="ThisDocument.MyMacro10" />
<button id="_1b_4" label="Παντελής απουσία σχολιασμού" image="_x0031_B" size="large"
onAction="ThisDocument.MyMacro11" />
</group>
<group id="undo2" label="Undo">
<button id="Undo2" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro12" />
</group>
</tab>
<tab id="_1c" label= "Άσκηση 1.Γ">
<group id="diagramma_klaseon" label="Διάγραμμα Κλάσεων">
<button id="_1c_1" label="Λείπει εννοιολογική κλάση" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro13" />
<button id="_1c_2" label="Λείπει χαρακτηριστικό που αναφέρεται ρητά" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro14" />
<button id="_1c_3" label="Λείπει συσχέτιση" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro15" />
<button id="_1c_4" label="Λείπει σχέση κληρονομικότητας" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro16" />
<button id="_1c_5" label="Λείπει πληθυκότητα" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro17" />
<button id="_1c_6" label="Άστοχη γραμμή σχολιασμού" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro18" />
<button id="_1c_7" label="Παντελής απουσία σχολιασμού" image="_x0031_C" size="large"
onAction="ThisDocument.MyMacro19" />
</group>
<group id="undo3" label="Undo">
```

<button id="Undo3" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro20" />
</group>
</tab>
<tab id="_1d" label= "Άσκηση 1.Δ">
<group id="ylopoiisi_klaseon" label="Υλοποίηση Κλάσεων">
<button id="_1d_1" label="Η κλάση του διαγράμματος κλάσεων λείπει από τον κώδικα " image="_x0031_d" size="large" onAction="ThisDocument.MyMacro21" />
<button id="_1d_2" label="Η μέθοδος του διαγράμματος κλάσεων δεν αντιστοιχεί σε μέθοδο του διαγράμματος ακολουθίας" image="_x0031_d" size="large" onAction="ThisDocument.MyMacro22" />
<button id="_1d_3" label="Η μέθοδος του διαγράμματος κλάσεων δεν αντιστοιχεί σε μέθοδο του κώδικα και αντίστροφα" image="_x0031_d" size="large" onAction="ThisDocument.MyMacro23" />
<button id="_1d_4" label="Η σχέση του διαγράμματος κλάσεων δεν αντιστοιχεί σε πεδίο αναφοράς στον κώδικα" image="_x0031_d" size="large" onAction="ThisDocument.MyMacro24" />
<button id="_1d_5" label="Μη ορθή αποτύπωση της κληρονομικότητας" image="_x0031_d" size="large" onAction="ThisDocument.MyMacro25" />
</group>
<group id="undo4" label="Undo">
<button id="Undo4" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro26" />
</group>
</tab>
<tab id="_1e" label= "Άσκηση 1.Ε">
<group id="dimioyrgia_epexergasias_diadromis" label="Δημιουργία και επεξεργασία διαδρομής">
<button id="_1e_1" label="Μη υλοποίηση της αναζήτησης γραμμής" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro27" />
<button id="_1e_2" label="Μη υλοποίηση της αναζήτησης στάσης" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro28" />
<button id="_1e_3" label="Μη υλοποίηση υπολογισμού μέσου χρόνου" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro29" />
<button id="_1e_4" label="Μη υλοποίηση της RetrieveRoute" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro30" />
<button id="_1e_5" label="Μη υλοποίηση της ReportRoute" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro31" />
<button id="_1e_6" label="Κάθε non private ιδιότητα" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro32" />
<button id="_1e_7" label="Κλάση χωρίς σχόλια" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro33" />
<button id="_1e_8" label="Απουσία αμυντικού προγραμματισμού" image="_x0031_e" size="large" onAction="ThisDocument.MyMacro34" />
</group>
<group id="undo5" label="Undo">
<button id="Undo5" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro35" />
</group>
</tab>
<tab id="_1st" label= "Άσκηση 1.ΣΤ">
<group id="ektelesi_efarmogis" label="Εκτέλεση της εφαρμογής">
<button id="_1st_1" label="Μη ορθή αποτύπωση της ροής του μενού" image="_x0031_st" size="large" onAction="ThisDocument.MyMacro36" />
<button id="_1st_2" label="Ελλείψεις στην παρουσίαση της λειτουργικότητας" image="_x0031_st" size="large" onAction="ThisDocument.MyMacro37" />
</group>
<group id="undo6" label="Undo">
<button id="Undo6" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro38" />
</group>
</tab>
<tab id="_2" label= "Άσκηση 2">
<group id="symorfosi_kanones_syggrafis" label="Συμμόρφωση με τους Κανόνες Συγγραφής">
<button id="_2_1" label="Μη συμμόρφωση με τους Κανόνες Συγγραφής" image="_x0032_" size="large" onAction="ThisDocument.MyMacro39" />
</group>
<group id="undo7" label="Undo">

```
<button id="Undo7" label="Undo" image="Undo" size="large" onAction="ThisDocument.MyMacro40" />
</group>
</tab>
<tab id="stats" label="Βαθμολόγηση και Στατιστικά Στοιχεία">
<group id="vathnologisi" label="Βαθμολόγηση">
<button id="vathmos" label="Υπολογισμός βαθμού" image="grades" size="large"
onAction="ThisDocument.MyMacro41" />
</group>
<group id="statistika" label="Στατιστικά Στοιχεία">
<button id="personal" label="Στατιστικά Στοιχεία Εξεταζόμενου" image="student" size="large"
onAction="ThisDocument.MyMacro42" />
<button id="path" label="Καθορισμός μονοπατιού (path) φακέλων και αρχείων Excel" image="d" size="large"
onAction="ThisDocument.MyMacro43" />
<button id="general" label="Συνολικά Στατιστικά Στοιχεία" image="stats" size="large"
onAction="ThisDocument.MyMacro44" />
</group>
</tab>
</tabs>
</ribbon>
</customUI>
```

VBA

```
Option Explicit
Dim a1t, a2t, a3t, a4t, a5t, a6t, a7t, atotal As Double ' βαθμοί ασκήσεων 1-7
Dim a1tp, a2tp, a3tp, a4tp, a5tp, a6tp, a7tp As Double  ' ποσοστά σωστών απαντήσεων
Dim ua1t, ua2t, ua3t, ua4t, ua5t, ua6t, ua7t As Double  ' undo άσκησης 1-7
Public myPath As String
Public myStatFile As String


Sub MyMacro(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Λείπει κλάση συνοριακή-ελέγχου-οντότητας "
   ua1t = a1t
   If a1t < 10 Then a1t = a1t + 0.5
   If a1t > 10 Then a1t = 10
   MsgBox "-" & a1t
End Sub
Sub MyMacro2(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Λείπει μήνυμα "
   ua1t = a1t
   If a1t < 10 Then a1t = a1t + 0.5
   If a1t > 10 Then a1t = 10
   MsgBox "-" & a1t
End Sub

Sub MyMacro3(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Λάθος στη ροή μηνυμάτων "
   ua1t = a1t
   If a1t < 10 Then a1t = a1t + 0.5
   If a1t > 10 Then a1t = 10
```

```
    MsgBox "-" & a1t
End Sub

Sub MyMacro4(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει κλήση περίπτωσης χρήσης "
    ua1t = a1t
    If a1t < 10 Then a1t = a1t + 0.5
    If a1t > 10 Then a1t = 10
    MsgBox "-" & a1t
End Sub

Sub MyMacro5(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λάθος σύνδεσης "
    ua1t = a1t
    If a1t < 10 Then a1t = a1t + 0.5
    If a1t > 10 Then a1t = 10
    MsgBox "-" & a1t
End Sub

Sub MyMacro6(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Παντελής απουσία σχολιασμού "
    ua1t = a1t
    If a1t < 10 Then a1t = a1t + 2
    If a1t > 10 Then a1t = 10
    MsgBox "-" & a1t
End Sub

Sub MyMacro7(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a1t = ua1t
    End If
End Sub


Sub MyMacro8(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει αντικ. οντοτήτων (ΑΟ) του δ. Ευρωστίας "
    ua2t = a2t
    If a2t < 10 Then a2t = a2t + 0.8
    If a2t > 10 Then a2t = 10
    MsgBox "-" & a2t
End Sub

Sub MyMacro9(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει αντικ. ελέγχου (ΑΕ) του δ. Ευρωστίας "
    ua2t = a2t
```

```
    If a2t < 10 Then a2t = a2t + 0.7
    If a2t > 10 Then a2t = 10
    MsgBox "-" & a2t
End Sub

Sub MyMacro10(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει μήνυμα "
    ua2t = a2t
    If a2t < 10 Then a2t = a2t + 0.5
    If a2t > 10 Then a2t = 10
    MsgBox "-" & a2t
End Sub

Sub MyMacro11(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Παντελής απουσία σχολιασμού "
    ua2t = a2t
    If a2t < 10 Then a2t = a2t + 2
    If a2t > 10 Then a2t = 10
    MsgBox "-" & a2t
End Sub

Sub MyMacro12(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a2t = ua2t
    End If
End Sub

Sub MyMacro13(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει εννοιολογική κλάση "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro14(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει χαρακτηριστικό που αναφέρεται ρητά "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro15(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει συσχέτιση "
```

```
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro16(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει σχέση κληρονομικότητας "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro17(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Λείπει πληθυκότητα "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro18(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Άστοχη γραμμή σχολιασμού "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 0.5
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro19(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Παντελής απουσία σχολιασμού "
    ua3t = a3t
    If a3t < 10 Then a3t = a3t + 2
    If a3t > 10 Then a3t = 10
    MsgBox "-" & a3t
End Sub

Sub MyMacro20(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a3t = ua3t
    End If
End Sub

Sub MyMacro21(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
```

```
    Selection.TypeText Text:=" Κλάση του διαγράμματος κλάσεων που λείπει από τον κώδικα "
    ua4t = a4t
    If a4t < 20 Then a4t = a4t + 0.5
    If a4t > 20 Then a4t = 20
    MsgBox "-" & a4t
End Sub

Sub MyMacro22(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Μέθοδος του διαγράμματος κλάσεων που δεν αντιστοιχεί σε μέθοδο του
διαγράμματος ακολουθίας"
    ua4t = a4t
    If a4t < 20 Then a4t = a4t + 0.5
    If a4t > 20 Then a4t = 20
    MsgBox "-" & a4t
End Sub

Sub MyMacro23(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Μέθοδος του διαγράμματος κλάσεων που δεν αντιστοιχεί σε μέθοδο του κώδικα
και αντίστροφα "
    ua4t = a4t
    If a4t < 20 Then a4t = a4t + 0.5
    If a4t > 20 Then a4t = 20
    MsgBox "-" & a4t
End Sub

Sub MyMacro24(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Σχέση του διαγράμματος κλάσεων που δεν αντιστοιχεί σε πεδίο αναφοράς στον
κώδικα"
    ua4t = a4t
    If a4t < 20 Then a4t = a4t + 0.5
    If a4t > 20 Then a4t = 20
    MsgBox "-" & a4t
End Sub

Sub MyMacro25(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Μη ορθή αποτύπωση της κληρονομικότητας (costructors, abstract κλπ.) "
    ua4t = a4t
    If a4t < 20 Then a4t = a4t + 0.5
    If a4t > 20 Then a4t = 20
    MsgBox "-" & a4t
End Sub


Sub MyMacro26(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a4t = ua4t
    End If
End Sub
```

```
Sub MyMacro27(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Μη υλοποίηση της αναζήτησης γραμμής "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 3
   If a5t > 30 Then a5t = 30
   MsgBox "-" & a5t
End Sub

Sub MyMacro28(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Μη υλοποίηση της αναζήτησης στάσης "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 3
   If a5t > 30 Then a5t = 30
   MsgBox "-" & a5t
End Sub
Sub MyMacro29(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Μη υλοποίηση υπολογισμού μέσου χρόνου "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 4
   If a5t > 30 Then a5t = 30
   MsgBox "-" & a5t
End Sub
Sub MyMacro30(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Μη υλοποίηση της RetrieveRoute "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 10
   If a5t > 30 Then a5t = 30
   MsgBox "-" & a5t
End Sub

Sub MyMacro31(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Μη υλοποίηση της ReportRoute "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 10
   If a5t > 30 Then a5t = 30
   MsgBox "-" & a5t
End Sub
Sub MyMacro32(ByVal control As IRibbonControl)
   Selection.Font.Size = 12
   Selection.Font.Color = wdColorRed
   Selection.Font.Bold = True
   Selection.TypeText Text:=" Κάθε non private ιδιότητα "
   ua5t = a5t
   If a5t < 30 Then a5t = a5t + 0.5
```

```
    If a5t > 30 Then a5t = 30
    MsgBox "-" & a5t
End Sub
Sub MyMacro33(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Κλάση χωρίς σχόλια "
    ua5t = a5t
    If a5t < 30 Then a5t = a5t + 0.5
    If a5t > 30 Then a5t = 30
    MsgBox "-" & a5t
End Sub

Sub MyMacro34(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Απουσία αμυντικού προγραμματισμού "
    ua5t = a5t
    If a5t < 30 Then a5t = a5t + 1
    If a5t > 30 Then a5t = 30
    MsgBox "-" & a5t
End Sub

Sub MyMacro35(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a5t = ua5t
    End If
End Sub

Sub MyMacro36(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Μη ορθή αποτύπωση της ροής του μενού "
    ua6t = a6t
    If a6t < 10 Then a6t = a6t + 2
    If a6t > 10 Then a6t = 10
    MsgBox "-" & a6t
End Sub
Sub MyMacro37(ByVal control As IRibbonControl)
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Ελλείψεις στην παρουσίαση της λειτουργικότητας "
    ua6t = a6t
    If a6t < 10 Then a6t = a6t + 5
    If a6t > 10 Then a6t = 10
    MsgBox "-" & a6t
End Sub

Sub MyMacro38(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a6t = ua6t
    End If
End Sub

Sub MyMacro39(ByVal control As IRibbonControl)
    Selection.Font.Name = "Times New Roman"
```

```
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorRed
    Selection.Font.Bold = True
    Selection.TypeText Text:=" Μη συμμόρφωση με τους κανόνες συγγραφής  "
    ua7t = a7t
    If a7t < 10 Then a7t = a7t + 10
    If a7t > 10 Then a7t = 10
    MsgBox "-" & a7t
End Sub

Sub MyMacro40(ByVal control As IRibbonControl)
    If ActiveDocument.Undo = True Then
      a7t = ua7t
    End If
End Sub

Sub MyMacro41(ByVal control As IRibbonControl)
    MsgBox "Υπολογισμός Βαθμολογίας"
    a1tp = (10 - a1t) / 10
    a1tp = Round(a1tp, 2)
    a2tp = (10 - a2t) / 10
    a2tp = Round(a2tp, 2)
    a3tp = (10 - a3t) / 10
    a3tp = Round(a3tp, 2)
    a4tp = (20 - a4t) / 20
    a4tp = Round(a4tp, 2)
    a5tp = (30 - a5t) / 30
    a5tp = Round(a5tp, 2)
    a6tp = (10 - a6t) / 10
    a6tp = Round(a6tp, 2)
    a7tp = (10 - a7t) / 10
    a7tp = Round(a7tp, 2)

    Selection.EndOf Unit:=wdStory, Extend:=wdMove
    Selection.InsertBreak Type:=wdPageBreak
    Selection.Font.Size = 12
    Selection.Font.Color = wdColorBlue
    Selection.Font.Bold = True
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Α: " & 10 - a1t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Β: " & 10 - a2t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Γ: " & 10 - a3t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Δ: " & 20 - a4t & "/20" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.Ε: " & 30 - a5t & "/30" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 1.ΣΤ: " & 10 - a6t & "/10" & Chr(13)
    Selection.TypeText Text:="Βαθμολογία Άσκησης 2: " & 10 - a7t & "/10" & Chr(13)
    atotal = (100 - (a1t + a2t + a3t + a4t + a5t + a6t + a7t)) / 10

    With Selection.Borders
        .Enable = True
        .OutsideLineStyle = wdLineStyleEmboss3D
        .OutsideColor = wdColorRed
        .Shadow = True
        .InsideLineStyle = wdLineStyleEmboss3D
        .InsideColor = wdColorRed
    End With

    Selection.TypeText Text:="Συνολική Βαθμολογία: " & atotal
End Sub

Sub MyMacro42(ByVal control As IRibbonControl)
```

```
Dim oChart As Chart
Dim oXls As Excel.Worksheet

Set oChart = ActiveDocument.Shapes.AddChart.Chart
Set oXls = oChart.ChartData.Workbook.Worksheets(1)

   oXls.ListObjects("Πίνακας1").Resize oXls.Range("A1:B8")

   oXls.Range("A1").FormulaR1C1 = "ΑΣΚΗΣΕΙΣ"
   oXls.Range("A2").FormulaR1C1 = "Άσκηση 1.Α"
   oXls.Range("A3").FormulaR1C1 = "Άσκηση 1.Β"
   oXls.Range("A4").FormulaR1C1 = "Άσκηση 1.Γ"
   oXls.Range("A5").FormulaR1C1 = "Άσκηση 1.Δ"
   oXls.Range("A6").FormulaR1C1 = "Άσκηση 1.Ε"
   oXls.Range("A7").FormulaR1C1 = "Άσκηση 1.ΣΤ"
   oXls.Range("A8").FormulaR1C1 = "Άσκηση 2"
   oXls.Range("B1").FormulaR1C1 = "ΠΟΣΟΣΤΟ ΕΠΙΤΥΧΟΥΣ ΟΛΟΚΛΗΡΩΣΗΣ ΤΩΝ ΑΣΚΗΣΕΩΝ "
   oXls.Range("B2").FormulaR1C1 = a1tp
   oXls.Range("B3").FormulaR1C1 = a2tp
   oXls.Range("B4").FormulaR1C1 = a3tp
   oXls.Range("B5").FormulaR1C1 = a4tp
   oXls.Range("B6").FormulaR1C1 = a5tp
   oXls.Range("B7").FormulaR1C1 = a6tp
   oXls.Range("B8").FormulaR1C1 = a7tp
   oXls.Range("C1:D5").Cells.Clear

   Columns("B").NumberFormat = "0.0%"

   With Columns("A:B")
         .ColumnWidth = 15
         .WrapText = True
         .VerticalAlignment = xlCenter
   End With
   With oXls.Range("A1:B8")
         .Borders(xlDiagonalDown).LineStyle = xlNone
         .Borders(xlDiagonalUp).LineStyle = xlNone
   End With
   With oXls.Range("A1:B8").Borders(xlEdgeTop)
         .LineStyle = xlContinuous
         .Weight = xlThin
         .ColorIndex = xlAutomatic
   End With
   With oXls.Range("A1:B8").Borders(xlEdgeBottom)
         .LineStyle = xlContinuous
         .Weight = xlThin
         .ColorIndex = xlAutomatic
   End With
   With oXls.Range("A1:B8").Borders(xlEdgeRight)
         .LineStyle = xlContinuous
         .Weight = xlThin
         .ColorIndex = xlAutomatic
   End With
   With oXls.Range("A1:B8").Borders(xlInsideVertical)
         .LineStyle = xlContinuous
         .Weight = xlThin
         .ColorIndex = xlAutomatic
   End With
   With oXls.Range("A1:B8").Borders(xlInsideHorizontal)
         .LineStyle = xlContinuous
```

```
            .Weight = xlThin
            .ColorIndex = xlAutomatic
    End With




    With oChart.Parent
            .Top = InchesToPoints(6)
            .Left = InchesToPoints(0)
            .Width = 400
            .Height = 250
            oChart.Axes(xlValue).MinimumScale = 0
            oChart.Axes(xlValue).MaximumScale = 1
    End With

    oXls.Range("A1:B8").Copy
    ActiveDocument.Activate
    Selection.EndKey Unit:=wdStory
    Selection.TypeText Text:=Chr(13) & " ΣΤΑΤΙΣΤΙΚΑ ΣΤΟΙΧΕΙΑ ΦΟΙΤΗΤΗ:"
    Selection.EndKey Unit:=wdStory
    Selection.PasteAndFormat (wdFormatOriginalFormatting)
    Selection.Borders.Enable = False
    oChart.ChartData.Workbook.Application.Quit

End Sub
Sub MyMacro43(Optional ByVal control As IRibbonControl)
    myPath = InputBox("Καθορισμός μονοπατιού (path) του συγκεντρωτικού φακέλου: ")
    myStatFile = InputBox("Καθορισμός μονοπατιού (path) του αρχείου Excel που θα συγκεντρωθούν τα
στοιχεία: ")
End Sub
Sub MyMacro44(ByVal control As IRibbonControl)
    MsgBox "Συνολικά στατιστικά στοιχεία"
    If myPath = "" Or myStatFile = "" Then
    MsgBox "Το μονοπάτι φακέλου ή το μονοπάτι του Excel αρχείου δεν έχει δωθεί. Παρακαλώ, προσθέστε τα
μονοπάτια καθώς η εισαγωγή τους είναι απαραίτητη για την διαδικασία ", vbCritical
    Call MyMacro43
    Else: MsgBox myPath & "     " & myStatFile
    End If
    Call Module3.WorkOnAWorkbook(myPath, myStatFile)
End Sub

Sub LoopAllSubFolders(ByVal folderpath As String)

Dim fileName As String
Dim fullFilePath As String
Dim numFolders As Long
Dim folders() As String
Dim i As Long

If Right(folderpath, 1) <> "\" Then folderpath = folderpath & "\"
fileName = Dir(folderpath & "*.*", vbDirectory)
Dim doc As Word.Document
Set doc = ActiveDocument

While Len(fileName) <> 0

    If Left(fileName, 1) <> "." Then
```

```
        fullFilePath = folderpath & fileName

        If (GetAttr(fullFilePath) And vbDirectory) = vbDirectory Then
            ReDim Preserve folders(0 To numFolders) As String
            folders(numFolders) = fullFilePath
            numFolders = numFolders + 1

        End If

    End If

    fileName = Dir()

Wend

For i = 0 To numFolders - 1

    LoopAllSubFolders folders(i)

Next i

End Sub

Sub WorkOnAWorkbook(ByVal folderpath As String, xlsxFile As String)


Dim folder As String
Dim oXL As Excel.Application
Dim oWB As Excel.Workbook
Dim oSheet As Excel.Worksheet
Dim oRng As Excel.Range
Dim ExcelWasNotRunning As Boolean
Dim WorkbookToWorkOn As String
Dim ii As Integer

Dim wdDoc         As Object
Dim wdFileName    As Variant
Dim TableNo       As Integer  'number of tables in Word doc
Dim iTable        As Integer  'table number index
Dim iRow          As Long     'row index in Excel


Dim fileName As String
Dim fullFilePath As String
Dim numFolders As Long
Dim folders() As String
Dim i As Integer

Dim my_tmp_string As String
Dim my_tmp_string_len As Long

folder = folderpath
'specify the workbook to work on
WorkbookToWorkOn = xlsxFile

'Start a new instance of Excel
Set oXL = New Excel.Application

'Open the workbook
Set oWB = oXL.Workbooks.Open(fileName:=WorkbookToWorkOn)
```

```
oWB.Worksheets(1).Activate

 ii = 1



If Right(folderpath, 1) <> "\" Then folderpath = folderpath & "\"
fileName = Dir(folderpath & "*.*", vbDirectory)
Dim doc As Word.Document
Set doc = ActiveDocument

While Len(fileName) <> 0

    If Left(fileName, 1) <> "." Then

        fullFilePath = folderpath & fileName

        If (GetAttr(fullFilePath) And vbDirectory) = vbDirectory Then
            ReDim Preserve folders(0 To numFolders) As String
            folders(numFolders) = fullFilePath
            numFolders = numFolders + 1
        Else


"""""""""""""enarxi antigrafis"""""""""""""""
wdFileName = fullFilePath
MsgBox (wdFileName)
   Set wdDoc = GetObject(wdFileName)   'open Word file

   With wdDoc
      TableNo = 1
      With .Tables(TableNo)
         'copy cell contents from Word table cells to Excel cells
         For iRow = 1 To .Rows.Count

my_tmp_string = .Cell(iRow, 2).Range.Text
my_tmp_string_len = Len(my_tmp_string)
my_tmp_string_len = my_tmp_string_len - 1
my_tmp_string = Left(my_tmp_string, my_tmp_string_len)

x11 = InStr(my_tmp_string, ",")
If (x11 > 1) Then
my_tmp_string = Left(my_tmp_string, x11 - 1)
End If


oWB.Worksheets(1).Cells(ii, iRow).Value = my_tmp_string

         Next iRow
      End With
   End With


    wdDoc.Close
"""""""""""telos antigrafis"""""""""""


        End If
```

```
    End If

    fileName = Dir()

ii = ii + 1
Wend

For i = 0 To numFolders - 1

    LoopAllSubFolders folders(i)

Next i


oXL.Quit
MsgBox "ΤΑ ΔΕΔΟΜΕΝΑ ΕΧΟΥΝ ΠΕΡΑΣΤΕΙ ΕΠΙΤΥΧΩΣ ΣΤΟ EXCEL- ΘΥΜΗΘΕΙΤΕ ΝΑ ΣΩΣΕΤΕ ΤΟ
EXCEL ΑΡΧΕΙΟ ΠΡΙΝ ΤΟ ΑΝΟΙΞΕΤΕ"


'Make sure you release object references.
Set oRng = Nothing
Set oSheet = Nothing
Set oWB = Nothing
Set oXL = Nothing

Exit Sub

End Sub
```