ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μελέτη της μεθοδολογίας SCRUM και της εφαρμογής μετρικών

Της φοιτήτριας
Νίκης Τσιτσιρούδη
Αρ. Μητρώου:07/3219

Επιβλέπων καθηγητής
Δεληγιάννης Ιγνάτιος

Θεσσαλονίκη 2012

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία αποτελεί μια προσπάθεια παρουσίασης της μεθοδολογίας Scrum αλλά και των πλεονεκτημάτων των μετρικών λογισμικού μέσα στην συγκεκριμένη μεθοδολογία. Έχει σκοπό να δώσει όλες εκείνες τις απαραίτητες πληροφορίες, που χρειάζεται ένα φοιτητής προκειμένου να την γνωρίσει καλύτερα και ίσως να υιοθετήσει τις αρχές της για την ανάπτυξη κάποιου έργου, που έχει αναλάβει.

Η παρουσίαση της μεθοδολογίας γίνεται σταδιακά, αφού πρώτα αναφερθούν ορισμένοι από τους κυριότερους και βασικότερους όρους της μηχανικής λογισμικού. Η ανάπτυξη του θέματος γίνεται με τέτοιο τρόπο, ώστε να αποσαφηνιστούν τυχόν παρερμηνείες των βασικών όρων της μηχανικής λογισμικού. Έπειτα, αναλύεται η μεθοδολογία, αφού πρώτα αναπτυχθούν τα συστατικά της στοιχεία.

Πιο αναλυτικά στο $1^{ο}$ κεφάλαιο γίνεται μια ιστορική αναδρομή της μηχανικής λογισμικού από την πρώτη εμφάνιση ηλεκτρονικού υπολογιστή μέχρι και σήμερα που οι υπολογιστικές ικανότητες ενός προσωπικού υπολογιστή έχουν πολλαπλασιαστεί. Στο $2^{ο}$ κεφάλαιο αναλύονται οι δύο βασικότερες έννοιες και κατ' επέκταση εργαλεία της μηχανικής λογισμικού, οι μεθοδολογίες και οι μετρικές λογισμικού. Στο $3^{ο}$ κεφάλαιο παρουσιάζεται η μεθοδολογία σχεδιασμού Scrum , καθώς και τα εμπλεκόμενα μέλη και οι διαδικασίες της μεθοδολογίας. Τέλος, στο $4^{ο}$ και τελευταίο κεφάλαιο της πτυχιακής αυτής εργασίας παρουσιάζεται η εφαρμογή των μετρικών λογισμικού στην μεθοδολογία Scrum και τα αποτελέσματα που εξάγει ο σχεδιαστής/προγραμματιστής του έργου για την πρόοδο του.

Κατά την ανάπτυξή της κρίθηκε σκόπιμη η δημιουργία ενός λεξιλογίου όρων για την μεθοδολογία Scrum, οπού θα μπορεί να ανατρέξει ο αναγνώστης προκειμένου να κατανοήσει ευκολότερα κάποιους όρους της, καθώς και την σημασία ορισμένων μεθόδων και διαδικασιών που χρησιμοποιούνται στην εφαρμογή της.

ΠΕΡΙΛΗΨΗ

Η μηχανική λογισμικού είναι κλάδος της επιστήμης των Η/Υ, ο οποίος αναπτύχτηκε και εξελίχθηκε σε σχετικά σύντομο χρονικό διάστημα . Τα πρώτα χρόνια ύπαρξης αυτού του κλάδου η ανάπτυξη ήταν αργή και σε καμία περίπτωση δεν διαφαινόταν η βαρύτητα, που θα προσλάμβανε στην βιομηχανία της πληροφορικής επί των ημερών μας. Κανείς δεν μπορεί να εγγυηθεί, πως στην μετέπειτα πορεία της δεν θα συναντήσει νέες δυσκολίες αλλά και προκλήσεις , τις οποίες θα κληθεί η επιστημονική κοινότητα να διαχειριστεί.

Οι μετρικές και οι μεθοδολογίες σχεδιασμού είναι δύο σημαντικά εργαλεία για τους μηχανικούς, τα οποία τους δίνουν τη δυνατότητα παρακολούθησης της προόδου και της ποιότητας στην ανάπτυξη ενός έργου. Οι ευέλικτες μέθοδοι είναι ιδιαίτερα δημοφιλείς τρόποι ανάπτυξης, οι οποίες υιοθετούνται όλο και από περισσότερους μηχανικούς λογισμικού.

Ιδιαίτερο ενδιαφέρον παρουσιάζει η μεθοδολογία SCRUM, η οποία αποκτά ολοένα και περισσότερους οπαδούς. Πολλοί είναι, πλέον, οι μηχανικοί λογισμικού αλλά και οι εταιρείες ανάπτυξης λογισμικού οι οποίοι εντάσσουν την μεθοδολογία στους τρόπους ανάπτυξης ενός συστήματος λογισμικού. Η μεθοδολογία SCRUM σε συνδυασμό με τις μετρικές λογισμικού μπορεί να προσφέρει στους μηχανικούς λογισμικού πιο ρεαλιστικά αποτελέσματα για την πρόοδο του έργου.

Είναι προφανές πως όλοι στρέφονται στην χρήση νέων μεθόδων σχεδιασμού, όπως η μεθοδολογία SCRUM . Βασική επιδίωξη των μηχανικών λογισμικού είναι η παροχή ενός εύχρηστου και λειτουργικού προγράμματος χωρίς σφάλματα σε όσο το δυνατό πιο σύντομο χρονικό διάστημα.

ABSTRACT

Software engineering is a branch of the field of PC which thrived and evolved in a relatively short period. In the early years of its existence, this new branch was developing too slowly and under no circumstance did it seem that it would be of such importance in the computer industry nowadays. Nobody can guarantee that the evolution of this branch will not find difficulties as well as challenges, which the scientific community will be called to face.

Software metrics/measurement and methodologies are two important design tools for software engineers, which enable them to monitor the progress and the quality of project development. Agility methods are a popular way of development, which are being adopted by more and more software engineers.

Of particular interest is the methodology SCRUM, which is gaining more and more followers. There are a lot of software engineers and companies now that incorporate this methodology on the way they develop a software system. The SCRUM methodology along with software metrics can provide software engineers with more realistic results for the progress of the project.

It is obvious that everybody is turning to the use of new design methods such as the SCRUM methodology. The main goal of software engineering is to provide a practical and functional program without errors in the shortest possible amount of time.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ 2
ΠΕΡΙΛΗΨΗ 3
ABSTRACT 4
ΠΕΡΙΕΧΟΜΕΝΑ 5
ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ 10
ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ 10
ΕΙΣΑΓΩΓΗ 13
ΚΕΦΑΛΑΙΟ 1 15
Η ΕΠΙΣΤΗΜΗ ΤΗΣ ΜΗΧΑΝΙΚΗΣ ΛΟΓΙΣΜΙΚΟΥ 15
ΕΙΣΑΓΩΓΗ 15
ΧΡΟΝΟΛΟΓΙΕΣ ΣΤΑΘΜΟΙ ΣΤΗΝ ΙΣΤΟΡΙΑ ΤΗΣ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ . 15
Η ΠΡΩΤΟΠΟΡΙΑΚΗ ΕΠΟΧΗ (περίπου πρώτο μισό 20ου αιώνα) 15
1945 - 1965: Η ΠΡΟΕΛΕΥΣΗ 17
1965 - 1985: Η ΚΡΙΣΗ ΛΟΓΙΣΜΙΚΟΥ 18
1985 - 1989: THE BULLET SILVER / Η ΑΣΗΜΕΝΙΑ ΣΦΑΙΡΙΑ 19
1990 - 1999:ΑΝΑΔΕΙΞΗ ΤΟΥ ΔΙΑΔΙΚΤΥΟΥ 21
2000 ΕΩΣ ΣΗΜΕΡΑ : ΕΛΑΦΡΕΙΣ ΜΕΘΟΔΟΛΟΓΙΕΣ 23
ΣΥΓΧΡΟΝΕΣ ΤΑΣΕΙΣ ΣΤΗΝ ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ 23
ΜΗΧΑΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ ΣΗΜΕΡΑ 25
ΕΠΙΛΟΓΟΣ 26
ΚΕΦΑΛΑΙΟ 2 28
ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ & ΜΕΘΟΔΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ 28
ΕΙΣΑΓΩΓΗ 28
ΜΕΘΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ ΛΟΓΙΣΜΙΚΟΥ 28
ΓΕΝΝΗΣΗ ΤΩΝ ΜΕΘΟΔΟΛΟΓΙΩΝ ΣΧΕΔΙΑΣΜΟΥ 28
ΠΟΙΕΣ ΕΙΝΑΙ ΜΕΘΟΔΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ 29

ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ.....30
ΜΕΤΡΙΚΕΣ ΤΗΣ ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ..... 32
ΜΕΤΡΙΚΕΣ ΕΡΓΟΥ..... 33
ΜΕΤΡΙΚΕΣ ΔΙΑΔΙΚΑΣΙΑΣ..... 33
ΕΠΙΛΟΓΟΣ.....35
ΚΕΦΑΛΑΙΟ 3..... 37
ΜΕΘΟΔΟΛΟΓΙΑ ΣΧΕΔΙΑΣΜΟΥ SCRUM 37
ΕΙΣΑΓΩΓΗ.....37
Η ΙΣΤΟΡΙΑ ΤΗΣ SCRUM 37
ΟΙ ΡΟΛΟΙ ΣΤΗΝ ΜΕΘΟΔΟΛΟΓΙΑ.....38
ΡΟΛΟΙ ΠΥΡΗΝΑ 39
3.2.2 ΒΟΗΘΗΤΙΚΟΙ ΡΟΛΟΙ..... 42
ΦΑΣΕΙΣ ΚΑΙ ΣΤΑΔΙΑ ΑΝΑΠΤΥΞΗΣ.....42
ΦΑΣΗ ΑΝΑΠΤΥΞΗΣ..... 42
ΣΤΑΔΙΑ ΑΝΑΠΤΥΞΗΣ..... 44
ΑΚΥΡΩΣΗ ΦΑΣΗΣ 48
ΑΝΤΙΚΕΙΜΕΝΑ.....49
ΑΝΕΚΤΕΛΕΣΤΟ ΠΡΟΪΟΝ 49
ΑΝΕΚΤΕΛΕΣΤΗ ΦΑΣΗ..... 50
ΠΡΟΣΑΥΞΗΣΗ 51
ΤΟ ΚΑΘΟΔΙΚΟ ΔΙΑΓΡΑΜΜΑ – BURN DOWN CHART 51
Η ΜΕΘΟΔΟΛΟΓΙΑ SCRUM-BAN 52
ΠΛΕΟΝΕΚΤΗΜΑΤΑ.....54
ΚΙΝΔΥΝΟΙ ΚΑΙ ΛΥΣΕΙΣ.....55
ΕΠΙΛΟΓΟΣ.....57
ΚΕΦΑΛΑΙΟ 4..... 59
ΜΕΘΟΔΟΛΟΓΙΑ ΣΧΕΔΙΑΣΜΟΥ SCRUM ΚΑΙ ΜΕΤΡΙΚΕΣ..... 59

ΕΙΣΑΓΩΓΗ....59
ΜΕΤΡΙΚΕΣ ΠΑΡΑΓΩΓΙΚΟΤΗΤΑΣ....60
ΧΡΟΝΟΣ ΚΥΚΛΟΥ ΕΠΑΝΑΛΗΨΗΣ/ΦΑΣΗΣ.... 60
(In Sprint Cycle Time).... 60
ΠΡΟΟΔΟΣ ΚΑΤΑ ΤΗΝ ΔΙΑΡΚΕΙΑ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ.... 60
(In Sprint Work in Process).... 60
ΟΛΟΚΛΗΡΩΣΗ ΕΠΑΝΑΛΗΨΗΣ/ΦΑΣΗΣ.... 61
(Sprint Completion Bar.).... 61
ΟΛΟΚΛΗΡΩΘΗΚΕ/ΔΕΣΜΕΥΤΗΚΕ.... 61
(Completed vs. Commited (a.k.a. Earned Value)).... 61
ΤΑΧΥΤΗΤΑ.... 61
(Velocity).... 61
ΠΡΟΣΩΠΙΚΗ ΑΞΙΟΛΟΓΗΣΗ ΑΠΟΔΟΣΗΣ.... 62
(Individual Performance Reviews).... 62
ΑΞΙΟΛΟΓΗΣΗ ΠΟΡΩΝ.... 62
(Resource Utilization).... 62
ΜΕΤΡΗΣΗ ΤΗΣ ΑΠΟΔΟΤΙΚΗΣ ΑΞΙΑΣ ΤΗΣ ΟΜΑΔΑΣ.... 62
(Business Value measuring performance).... 62
ΓΡΑΜΜΕΣ ΠΗΓΑΙΟΥ ΚΩΔΙΚΑ.... 63
(Source Lines of Code (SLOC)).... 63
ΜΕΤΡΙΚΕΣ ΠΟΙΟΤΗΤΑΣ....63
ΤΕΧΝΙΚΑ ΣΗΜΕΙΑ ΧΡΕΟΥΣ.... 63
(Technical Debt Point).... 63
ΕΚΤΕΛΕΣΗ ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟΥΝ ΕΛΕΓΧΟΥ.... 64
(Running Automated Test).... 64
ΔΗΜΟΣΙΕΥΣΗ ΤΩΝ ΕΞΕΡΧΟΜΕΝΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ.... 64
(Post Sprint Defect Arrival).... 64

ΔΗΜΟΣΙΕΥΣΗ ΤΩΝ ΕΙΣΕΡΧΟΜΕΝΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ 64
(Post Release Defect Arrival)..... 64
ΒΑΣΙΚΟΤΕΡΑ ΕΛΑΤΤΩΜΑΤΑ ΠΡΟΣ ΕΠΙΔΙΟΡΘΩΣΗ 64
(Root Causes Fixed)..... 64
ΑΡΙΘΜΟΣ ΕΠΑΝΑΛΗΠΤΙΚΩΝ ΔΟΚΙΜΩΝ..... 65
(Number of Test Cycles.)..... 65
ΣΗΜΕΙΑ ΔΙΑΣΦΑΛΙΣΗ ΤΗΣ ΠΟΪΟΤΗΤΑΣ ΠΡΟΙΟΝΤΟΣ..... 65
(Quality assurance Story Points) 65
ΑΡΙΘΜΟΣ ΠΟΙΟΤΙΚΩΝ ΕΛΑΤΤΩΜΑΤΩΝ 65
(Quality Number of Defects) 65
ΕΛΕΓΧΟΣ ΠΟΙΟΤΗΤΑΣ ΜΕΤΑ ΤΗΝ ΑΝΑΠΤΥΞΗ..... 66
(Quality Checked After Development)..... 66
ΜΕΤΡΙΚΕΣ ΠΡΟΒΛΕΠΤΙΚΟΤΗΤΑΣ.....66
ΕΝΙΣΧΥΜΕΝΟ ΚΑΘΟΔΙΚΟ ΔΙΑΓΡΑΜΜΑ 66
(Enhanced Burndown Chart.) 66
ΤΑΧΥΤΗΤΑ..... 67
(Velocity)..... 67
ΚΟΣΤΟΣ ΑΝΑ ΣΗΜΕΙΟ ΙΣΤΟΡΙΑΣ..... 67
(Cost per Story Point) 67
ΩΡΕΣ ΑΝΑ ΣΗΜΕΙΟ ΙΣΤΟΡΙΑΣ..... 68
(Hours per Story Point) 68
ΜΕΤΡΙΚΗ ΕΚΤΙΜΩΜΕΝΟΥ ΧΡΟΝΟΥ ΠΑΡΑΔΟΣΗΣ ΕΡΓΟΥ 69
(Project Estimated time of delivery metric)..... 69
ΜΕΤΡΙΚΕΣ ΑΞΙΑΣ.....69
ΔΗΜΟΣΙΕΥΣΗ ΑΞΙΑΣ ΕΠΙΧΕΙΡΗΣΗΣ 70
(Business Value Delivered)..... 70
ΕΡΕΥΝΑ ΙΚΑΝΟΠΟΙΗΣΗΣ ΠΕΛΑΤΩΝ..... 71

(Customer Satisfaction Survey) 71
ΕΡΕΥΝΑ ΙΚΑΝΟΠΟΙΗΣΗΣ ΕΡΓΑΖΟΜΕΝΩΝ 71
(Employee Satisfaction Survey) 71
ΕΠΙΛΟΓΟΣ 72
ΣΥΜΠΕΡΑΣΜΑΤΑ 73
ΛΕΞΙΚΟ ΟΡΩΝ 75
Ομάδα SCRUM – SCRUM Team 75
Ιδιοκτήτης Προϊόντος – Product Owner 75
SCRUM Master 75
Ομάδα Ανάπτυξης – Development Team_____ 75
Καθοδικό Διάγραμμα Φάσης – Sprint burn down chart 75
Υπόλοιπο ανεκτέλεστο προϊόν – Product backlog 75
Φάση ανεκτέλεστου προϊόντος – Sprint backlog 76
Επαναληπτική φάση - Sprint 76
(Χρήστης) Ιστορία - (User) Story 76
Θέμα – Theme 77
Έπη - Epic 77
Συλλέκτης - Spike 77
Σφαίρα ιχνηλάτησης – Tracer Bullet 78
Σημεία κλίμακας/Προσπάθειας/Σημεία Ιστορίας-Απαιτήσεων – Point Scale/Effort/Story Points 78
Εργασίες - Tasks 78
Ορισμό του Γίνεται – Definition of Done (DoD) 79
Ταχύτητα - Velocity 79
Εμπόδιο - Impediment 79
Sashimi 79
Ανώμαλος Τερματισμός – Abnormal Termination 79

Σχεδιασμός Πόκερ- Planning Poker 80

Μοντέλο V - V-model 80

ΠΑΡΑΠΟΜΠΕΣ 81

ΠΑΡΑΡΤΗΜΑΤΑ 86

ΒΙΒΛΙΟΓΡΑΦΙΑ - ΑΝΑΦΟΡΕΣ 91

Σχετικοί Ιστότοποι 93

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

Πίνακας a Πίνακας κόστους ανά σημείο ιστορίας. 68

Πίνακας b Πίνακας φάσεων ανεκτέλεστου προϊόντος. 76

Πίνακας c Πίνακας ιστοριών χρήστη / περιπτώσεων χρήσης 77

ΕΥΡΕΤΗΡΙΟ ΕΙΚΟΝΩΝ

Εικόνα a Δείγμα διατρητής κάρτας για προγραμματισμό 16

Εικόνα b Υπολογιστής που δουλεύει με διατρητές κάρτες. 17

Εικόνα c Οργανόγραμμα της SCRUM 41

Εικόνα d Πρόοδος των διαδικασιών της SCRUM 44

Εικόνα e Ημερήσιο ραβδόγραμμα με την πρόοδο του έργου με βάση την προτεραιότητα της κάθε εργασίας. 46

Εικόνα f Πίνακας προόδου της SCRUM με χρήση σημειώσεων. 51

Εικόνα g Απλό καθοδικό διάγραμμα της μεθοδολογίας 52

Εικόνα h Διάγραμμα τύπου "Πίτα" που απεικονίζει τα σημεία που έχουν ολοκληρωθεί. 53

Εικόνα i Διάγραμμα τύπου "Πίτα" που απεικονίζει τα σημεία που απομένουν για την ολοκλήρωση του έργου. 54

Εικόνα j Διάγραμμα παρουσίασης αξία και ρίσκου ανάπτυξης μίας απαίτησης.... 56

Εικόνα k Ραβδόγραμμα παρουσίασης του κύκλου ανάπτυξης μίας φάσης. 60

Εικόνα l Διάγραμμα όπου παρουσιάζονται η πολυπλοκότητα ανά κλάση (Complexity/class), ποσοστό των απαιτήσεων που έχουν καλυφτεί(Coverage) και το τεχνικό τμήμα (Technical Dept (s)) 63

Εικόνα m Ενισχυμένο καθοδικό διάγραμμα των σημείων ιστορίας σε σχέση με το χρόνο..... 66

Εικόνα n Διάγραμμα Ταχύτητας. 67

Εικόνα o Διάγραμμα κόστους ανά σημείο ιστορίας..... 68

Εικόνα p Διάγραμμα υπολογισμού παράδοσης έργου. 69

Εικόνα q Διαγράμματα απεικόνισης της αξίας που κερδίζει το έργο και η ομάδα ανάπτυξης. 70

Εικόνα r Καθοδικό διάγραμμα φάσης. 75

Εικόνα s Λίστα με ορισμό του Γίνεται. 79

Εικόνα t Το μοντέλο-V της προόδου ανάπτυξης ενός συστήματος. 80

Εικόνα u Παρουσίαση όλων των μετρικών λογισμικού στον πρόγραμμα Eclipse 86

Εικόνα v Συνολικές Γραμμές Κώδικα ανά κλάση του προγράμματος. 87

Εικόνα w Αποτελέσματα για το πλήθος γραμμών κώδικα (Μ.Ο. ανά μέθοδο)..... 88

Εικόνα x Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος..... 89

Εικόνα y Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος και μιας συγκεκριμένης κλάσης..... 89

Εικόνα z Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος αλλά και όλων των στοιχείων / μεθόδων που περιλαμβάνονται σε όλες τις κλάσεις του προγράμματος λογισμικού. 90

ΕΙΣΑΓΩΓΗ

Αντικείμενο της πτυχιακής αυτής είναι η μελέτη και παρουσίαση της αντικειμενοστραφούς μεθοδολογίας SCRUM, καθώς και η εφαρμογή των μετρικών στις δραστηριότητές της. Λέγοντας δραστηριότητες εννοούνται οι επιμέρους ανάπτυξη των δομών και των λειτουργιών του έργου μέσα από την εξαγωγή των απαιτήσεων χρήστη, η ανάλυσή τους, η σχεδίασή και ανάπτυξη κώδικα. Ωστόσο, κρίθηκε απαραίτητο, αρχικά, να εξεταστούν κάποια επιμέρους στοιχεία της μηχανικής λογισμικού, που βοηθούν στην εξέταση του θέματος.

Στο 1^{ο} κεφάλαιο θα γίνει μια εισαγωγή στην μηχανική λογισμικού. Παρουσιάζεται η «εξέλιξη» της μηχανικής λογισμικού στο πέρασμα των χρόνων.

Στο 2^{ο} κεφάλαιο, παρουσιάζονται ορισμένες μεθοδολογίες σχεδιασμού και αναλύονται οι έννοιες της μετρικής και της μέτρησης. Επιπλέον, η παρουσίαση των μετρικών λογισμικού και μεθοδολογιών γίνεται υπό την μορφή παράθεσης σύντομων παραδειγμάτων για την πιο εύκολη και παραστατική κατανόηση και των δύο.

Στο 3^{ο} κεφάλαιο, επιλέγεται να παρουσιαστεί και αναλυθεί πλήρως μια από τις πιο ενδιαφέρουσες μεθοδολογίες σχεδιασμού, η SCRUM. Συγκεκριμένα, αναλύονται τα στάδια ανάπτυξης ενός προϊόντος με την συγκεκριμένη μεθοδολογία, ο ρόλος του κάθε εμπλεκόμενου προσώπου, τα αντικείμενα της και μια μεθοδολογία – την SCRUM-ban – που είναι ένα από τα παιδιά της SCRUM.

Τέλος, στο 4^{ο} και τελευταίο κεφάλαιο, παρουσιάζεται η εφαρμογή των μετρικών λογισμικού σε ένα έργο, που αναπτύσσεται με την συγκεκριμένη μεθοδολογία. Η παρουσίαση τους γίνεται με βάση το είδος της μέτρησης που κάνει η κάθε μια, δηλαδή, αν αναφέρονται στην ποιότητα, στην προβλεπτικότητα, την ποσότητα του κώδικα, ή την αξία του έργου.

ΚΕΦΑΛΑΙΟ 1

Η ΕΠΙΣΤΗΜΗ ΤΗΣ ΜΗΧΑΝΙΚΗΣ ΛΟΓΙΣΜΙΚΟΥ

ΕΙΣΑΓΩΓΗ

Με ένα σύντομο ορισμό :

> «Μηχανική (Τεχνολογία) Λογισμικού (SE) είναι η εφαρμογή μιας συστηματικής, πειθαρχημένης, ποσοτικά προσέγγισης στο σχεδιασμό, ανάπτυξη, λειτουργία και συντήρηση του λογισμικού και η μελέτη αυτών των προσεγγίσεων · δηλαδή, η εφαρμογή της μηχανικής στο λογισμικό . »

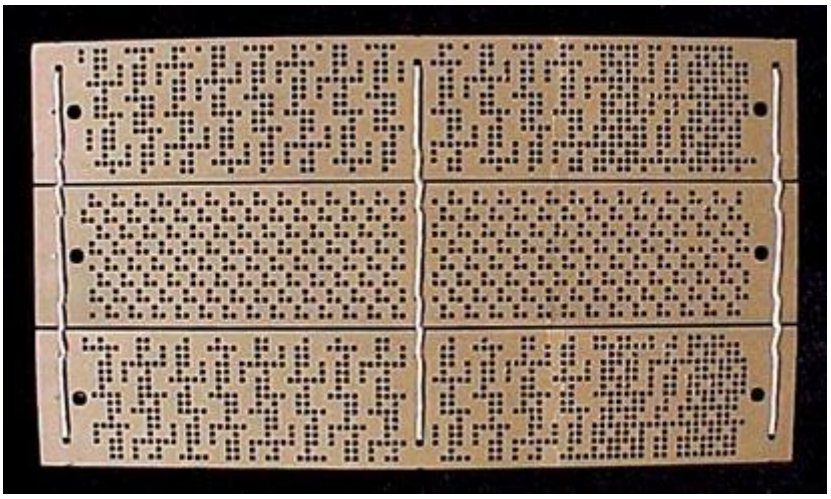
Ο όρος μηχανική λογισμικού χρησιμοποιήθηκε για πρώτη φορά το 1968 σε μια διάσκεψη του ΝΑΤΟ, με βασικό αντικείμενο συζήτησης την «κρίση λογισμικού», που σημειωνόταν την εποχή εκείνη .

ΧΡΟΝΟΛΟΓΙΕΣ ΣΤΑΘΜΟΙ ΣΤΗΝ ΙΣΤΟΡΙΑ ΤΗΣ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ .

Η ΠΡΩΤΟΠΟΡΙΑΚΗ ΕΠΟΧΗ (περίπου πρώτο μισό 20^{ου} αιώνα)

Όταν κάποιος αναφέρεται στην πρωτοποριακή εποχή της μηχανικής λογισμικού, σίγουρα αναφέρεται στην εποχή, όπου νέοι υπολογιστές έβγαιναν σχεδόν κάθε ένα ή δύο χρόνια, καθιστώντας τους υπάρχοντες απαρχαιωμένους, καθώς δεν υπήρχε η δυνατότητα αναβάθμισης. Άνθρωποι οι οποίοι έγραφαν προγράμματα λογισμικού έπρεπε να τα ξαναγράψουν όλα, για να τρέξουν σε αυτές τις νέες μηχανές. Οι προγραμματιστές δεν είχαν δικούς τους προσωπικούς υπολογιστές στα γραφεία τους, με αποτέλεσμα, για να μπορέσουν να αναπτύξουν ένα πρόγραμμα να χρειάζεται να πάνε στην «αίθουσα Η/Υ»[1]. Οι εργασίες πραγματοποιούνταν είτε ύστερα από αίτηση που έκαναν για να αποκτήσουν χρόνο στην «αίθουσα Η/Υ» είτε από το επιχειρησιακό προσωπικό, που προγραμμάτιζε το πρόγραμμα εργασιών της «αίθουσας Η/Υ». Όταν λοιπόν, μπορούσαν να χρησιμοποιήσουν την «αίθουσα Η/Υ», η ανάπτυξη των εργασιών γίνονταν με την τοποθέτηση διάτρητων καρτών σε έναν αναγνώστη καρτών, που

διέθετε ο υπολογιστής για την εισαγωγή δεδομένων και περιμένοντας το αποτέλεσμα της επεξεργασίας τους, που δινόταν από έναν εκτυπωτή.



Εικόνα a Δείγμα διατρητής κάρτας για προγραμματισμό

Το πεδίο της μηχανικής λογισμικού ήταν τόσο νέο, ώστε η ιδέα της διαχείρισής του μέσω προγράμματος ήταν ανύπαρκτη. Η προσπάθεια πρόβλεψης της ημερομηνίας ολοκλήρωσης του έργου ήταν σχεδόν αδύνατη. Το υλικό (hardware) των υπολογιστών ήταν σχεδιασμένο, ώστε να μπορεί να δουλεύει για συγκεκριμένες εφαρμογές. Επιστημονικές και επιχειρηματικές εργασίες απαιτούσαν διαφορετικά μηχανήματα. Συχνά λόγω της ανάγκης να μεταφραστεί κάποιο παλιό λογισμικό (software) για να καλύψει τις ανάγκες των νέων μηχανημάτων, αναπτύσσονταν γλώσσες υψηλής γενιάς όπως η FORTRAN, COBOL, και ALGOL. Πωλητές υλικού έδιναν συστήματα λογισμικού δωρεάν, διότι το υλικό δεν θα μπορούσε να πωληθεί χωρίς λογισμικό. Μερικές εταιρείες πωλούσαν την υπηρεσία της δημιουργίας προσαρμοσμένου λογισμικού, για να ικανοποιούνται οι ανάγκες των πελατών τους, αλλά καμία εταιρεία λογισμικού δεν πωλούσε πακέτα λογισμικού μεμονωμένα.

Εικόνα b Υπολογιστής που δουλεύει με διατρητές κάρτες.

Η έννοια της επαναχρησιμοποίησης έχει αρχίσει πλέον να αναδύεται. Δεδομένου ότι το λογισμικό ήταν ελεύθερο, οργανώσεις χρηστών το μοιράζουν σε όσους το ζητούν. Ομάδες, όπως η επιστημονική ομάδα χρηστών SHARE [2]της IBM, προσφέρουν καταλόγους των επαναχρησιμοποιήσιμων τμημάτων κώδικα/κωδίκων. Η ακαδημαϊκή εκπαίδευση για τους μηχανικούς λογισμικού δεν υπήρχε. Ακόμα, η εκπαίδευση η οποία παρέχονταν ήταν από την στρατιωτική κοινότητα και σε ορισμένους μόνο.

1945 - 1965: Η ΠΡΟΕΛΕΥΣΗ

Για πρώτη φορά στα τέλη της δεκαετίας του 1950 χρησιμοποιείται και ο όρος μηχανική λογισμικού . Οι μηχανικοί λογισμικού την περίοδο αυτή αρχίζουν να γνωρίζουν τους αστικούς (κοινούς υπολογιστές), τους ηλεκτρονικούς υπολογιστές. Ακόμα, όμως, δεν είχαν κατανοήσει τις επιδράσεις που θα είχαν οι καινοτομίες αυτές στον χώρο του software.

Το 1968 στο Garmish της Γερμανίας [3]και το 1969 στην Ρώμη της Ιταλίας [4], το ΝΑΤΟ πραγματοποίησε δύο συνέδρια τα οποία σύμφωνα με κάποιους ήταν και αυτά τα οποία έδωσαν την απαραίτητη ώθηση για την ανάπτυξη του τομέα της μηχανικής λογισμικού. Πολλοί πιστεύουν ότι αυτά τα συνέδρια σηματοδότησαν την επίσημη έναρξη του συγκεκριμένου επαγγέλματος.

1965 - 1985: Η ΚΡΙΣΗ ΛΟΓΙΣΜΙΚΟΥ

Στις δεκαετίες του 1960, 1970 και στις αρχές του 1980 σημειώθηκε μια σχετική στατικότατα, που ονομάστηκε Κρίση λογισμικού. Αυτή οφειλόταν στα λάθη και στα προβλήματα, τα οποία και εντοπίζονταν στην ανάπτυξη των προγραμμάτων. Πολλά ήταν τα έργα εκείνα, τα οποία είχαν βγει εκτός χρονοδιαγράμματος προγραμματισμού, εκτός προϋπολογισμού και περιείχαν αρκετά λάθη στον κώδικα τους που σε ορισμένες περιπτώσεις προκάλεσαν ζημίες στο υλικό του υπολογιστικού συστήματος. Πρέπει να σημειωθεί πως ορισμένα από τα έργα, τα οποία παρουσίασαν σφάλματα και ζημίες στο υλικό, προκάλεσαν και την απώλεια ζωών. Αν και πήρε χρόνο, η κρίση ξεπεράστηκε και μέσα από την επίλυση των ζητημάτων αναπτύχθηκαν νέες δυνατότητες για το software.

Χαρακτηριστικά παραδείγματα προβλημάτων της κρίσης λογισμικού είναι τα ακόλουθα :

1. Αυξημένο κόστος και υπέρβαση του αρχικού προϋπολογισμού του έργου. Το λειτουργικό σύστημα OS/360 είναι ένα κλασικό παράδειγμα. Τη συγκεκριμένη δεκαετία του 1960, και πιο συγκεκριμένα το 1964, το OS/360 ήταν ένα από τα πιο πολύπλοκα λειτουργικά συστήματα, αλλά και από τα πιο μεγάλα (χρειαστήκαν 1000 προγραμματιστές για την ανάπτυξη του). Ο Fred Brooks ισχυρίζεται στο «Ο Μυθικός ανθρώπινος μήνας» (The Mythical Man Month) [5] πως έκανε ένα λάθος πολλών εκατομμυρίων δολαρίων, που τελικά του στοίχησε την ανάπτυξη μίας συνεκτική αρχιτεκτονικής πριν την έναρξη της ανάπτυξης του έργου .
2. Υλικές ζημίες. Όταν κάποιος αναφέρεται σε υλικές ζημίες, εννοεί λάθη στον κώδικα του έργου τα οποία μπορεί να προκαλέσουν ζημίες του υλικού (hardware) του υπολογιστή. Επιπλέον, ένα «κακό» λογισμικό μπορεί να είναι και επιρρεπές σε επιθέσεις κακόβουλων χρηστών ή εισβολέων οι οποίοι θα κλέψουν στοιχεία για τα οποία δεν είναι εξουσιοδοτημένοι να έχουν πρόσβαση και τέτοιου είδους λάθη κοστίζουν χρόνο, χρήμα, και φήμη.
3. Η ζωή και ο θάνατος. Ένα ελαττωματικό λογισμικό μπορεί να αποβεί μοιραίο για τις ζωές ορισμένων ανθρώπων. Για παράδειγμα,

λογισμικό το οποίο χρησιμοποιείται σε μηχανήματα ακτινοθεραπείας απέτυχε καταστροφικά, με αποτέλεσμα να χορηγούνται θανατηφόρες δόσεις ακτινοβολίας στους ασθενείς.

Ο Peter G. Neumann έχει κρατήσει μια λίστα των σύγχρονων προβλημάτων λογισμικού, που έχουν παρουσιαστεί ως σήμερα, καθώς επίσης και τις καταστροφές, τις οποίες αυτά έχουν προκαλέσει . Πλέον, η κρίση του λογισμικού έχει ξεθωριάσει από μια άποψη, διότι η ανάπτυξη του λογισμικού έχει ξεπεράσει τις εγγενείς αδυναμίες, που εμφάνιζαν τα πρώτα συστήματα λογισμικού με την απλότητα τους. Παρ 'όλα αυτά, στο λογισμικό, και ακόμη περισσότερο στο λογισμικό πραγματικού χρόνου, ο κίνδυνος παρουσίασης σφαλμάτων εξακολουθεί να υφίσταται. Τα τελευταία χρόνια ο Michael Jackson A. έχει ασχοληθεί με την περιγραφή της «φύσης» της μηχανικής λογισμικού και έχει προσδιορίσει πως η κύρια πηγή των δυσκολιών για ανάπτυξη ενός καλού ποιοτικά συστήματος λογισμικού είναι η έλλειψη εξειδίκευσης.

1985 - 1989: THE BULLET SILVER / Η ΑΣΗΜΕΝΙΑ ΣΦΑΙΡΙΑ

Για αρκετά μεγάλο διάστημα η εύρεση λύσης στο πρόβλημα της κρίσης λογισμικού ταλάντευε ερευνητές, επιστήμονες και εταιρείες, οι οποίες παρήγαγαν εργαλεία για την ανάπτυξη λογισμικού. Το κόστος για την ιδιοκτησία και τη συντήρηση ενός προγράμματος λογισμικού στη δεκαετία του 1980 ήταν δύο φορές πιο ακριβό από το κόστος της ανάπτυξης του . Με την πάροδο των χρόνων το κόστος αυτό άρχισε να αυξάνεται παραπάνω. Χαρακτηριστικά το 1990 αυξήθηκε κατά 30% σε σύγκριση με το κόστος στην δεκαετία του 1980.

Μέσα από διάφορες στατιστικές έρευνες οι οποίες είχαν γίνει, αποδείχτηκε πως τα συστήματα λογισμικού που είχαν αναπτυχτεί μπορεί να ήταν λειτουργικά, ωστόσο δεν ήταν επιτυχή. Παρατηρήθηκε, επίσης, πως ο μέσος όρος των προγραμμάτων τα οποία είχαν αναπτυχθεί είχε ξεπεράσει το χρονοδιάγραμμα εκπόνησης του έργου, και τα ¾ των μεγάλων έργων λογισμικού, που τελικά παραδόθηκαν στον πελάτη, είτε ήταν αποτυχημένα ως προς τις απαιτήσεις του είτε δεν μπορούν να αξιοποιηθούν καθόλου.

Έργα λογισμικού

Κάθε νέα τεχνολογία και πρακτική από το 1970 και μέχρι το 1990 παρουσιάζονταν ως η λύση του προβλήματος της κρίσης λογισμικού, «Η ασημένια σφαίρα», όπως και αποκαλούσαν τις νέες τεχνολογίες και πρακτικές, αποτελούνταν από εργαλεία, μεθόδους και τυπικές μεθόδους [6], διαδικασίες και αναφορές στον επαγγελματισμό των τεχνικών λογισμικού.

Ως «ασημένιες σφαίρες» θεωρήθηκαν τα εξής:

- Εργαλεία τα οποία αναπτύχτηκαν, όπως ήταν ο δομημένος προγραμματισμός, ο αντικειμενοστραφής (object-oriented) προγραμματισμός [7], εργαλεία CASE [8], Ada [9], τεκμηρίωση [10] και προτυποποίηση [11].
- Μέθοδος. Ορισμένοι ειδήμονες υποστηρίζουν ότι η κρίση λογισμικού οφειλόταν στην έλλειψη μεθόδων των προγραμματιστών.
- Τυπικές μέθοδοι. Κάποιοι πίστευαν ότι εάν εφαρμόζονταν μια επίσημη μεθοδολογία ανάπτυξης για την ανάπτυξη λογισμικού, τότε η παραγωγή του λογισμικού θα γινόταν μια τόσο προβλέψιμη βιομηχανία, όπως και κάποιοι άλλοι κλάδοι της μηχανικής. Υποστήριξαν την άποψη τους αποδεικνύοντας πως όλα τα προγράμματα θα δούλευαν σωστά, αν τις χρησιμοποιούσαν.
- Διαδικασία. Αρκετοί ήταν και εκείνοι, που υποστήριξαν πως η χρήση καθορισμένων διαδικασιών και μεθοδολογιών, όπως το Capability Maturity Model [12] θα επέφερε την λύση στην κρίση λογισμικού .
- Επαγγελματισμός. Υποστηρίχτηκε πως, εάν υπήρχε ένας κώδικας δεοντολογίας, αυτόματα θα οδηγούμασταν και στον επαγγελματισμό.

Το 1986, ο Fred Brooks δημοσίευσε το άρθρο του No Silver Bullet [13], με βασικό επιχείρημα ότι δεν υπάρχει ατομική τεχνολογία ή πρακτική που θα μπορούσε ποτέ να οδηγήσει σε δεκαπλάσια βελτίωση της παραγωγικότητας μέσα σε 10 χρόνια.

Πολλές ήταν οι συζητήσεις, που γίνονταν σχετικά με την «ασημένια σφαίρα» τις επόμενες δεκαετίες. Υποστηρικτές της Ada, των συστατικών, και των διαδικασιών του, συνέχισαν να ισχυρίζονται πως τελικά το αγαπημένο τους εργαλείο θα αποτελούσε την λύση για την κρίση λογισμικού. Τελικά, σχεδόν όλοι

δέχονται ότι υπάρχει η «ασημένια σφαίρα», που θα μπορούσε να φέρει λύση στο πρόβλημα της κρίσης, χωρίς όμως να είναι κάποιος βέβαιος για το ποια θα είναι αυτή. Ωστόσο, οι ισχυρισμοί σχετικά με τις «ασημένιες σφαίρες» εξακολουθούν να ακούγονται και να γίνεται λόγος για αυτές μέχρι και σήμερα.

Από μερικούς το No Silver Bullet [13] μεταφράστηκε ως η αποτυχία της μηχανικής λογισμικού. Ωστόσο, αν ψάξει κάποιος λίγο καλύτερα θα δει πως ο Brooks ισχυρίζεται ότι : «Θα κάνουμε σίγουρα σημαντική πρόοδο κατά τα επόμενα σαράντα χρόνια, και είναι γεγονός ότι η εξέλιξη θα είναι μεγαλύτερης τάξης από ότι περιμέναμε ...».

Η αναζήτηση για ένα μοναδικό κλειδί, που θα οδηγούσε στην επιτυχία ποτέ δεν λειτούργησε. Όλες οι γνωστές τεχνολογίες και πρακτικές έχουν κάνει μόνο μερικές σταδιακές βελτιώσεις στην παραγωγικότητα και στην ποιότητα. Ωστόσο, δεν υπάρχουν ασημένιες σφαίρες για οποιοδήποτε άλλο επιστημονικό κλάδο, άρα ούτε και για την μηχανική λογισμικού. Άλλοι προσπαθούν να βρουν την μαγική συνταγή, ως απόδειξη ότι η τεχνολογία λογισμικού έχει ωριμάσει και τελικά αναγνώρισε ότι τα έργα της θα έχουν επιτυχία λόγω της σκληρής δουλειάς.

Ωστόσο, θα μπορούσε επίσης να ειπωθεί ότι στην πραγματικότητα υπάρχουν, μια σειρά από «ασημένιες σφαίρες», συμπεριλαμβανομένων των μεθοδολογιών, των φυλλομετρητών, κάποιων προσαρμοσμένων προγραμμάτων περιήγησης στον χώρο των μηχανών αναζήτησης, γεννήτριες έκθεσης της βάσης δεδομένων, ολοκληρωμένος σχεδιασμός κωδικοποίησης με συντάκτες, καθώς και καταστήματα, που παράγουν συγκεκριμένο είδος λογισμικού, όπως ιστοσελίδες με πληροφορίες, με ένα τμήμα της κατασκευής της ιστοσελίδας να στοχεύει αποκλειστικά στην ίδια την ανάπτυξης της. Παρ 'όλα αυτά, ο τομέας της μηχανικής λογισμικού φαίνεται υπερβολικά πολύπλοκος και ποικίλος, ώστε μια «ασημένια σφαίρα» να μπορεί να βελτιώσει τα περισσότερα ζητήματα, καθώς κάθε ζήτημα αντιπροσωπεύει μόνο ένα μικρό μέρος του συνόλου των προβλημάτων λογισμικού.

1990 - 1999:ΑΝΑΔΕΙΞΗ ΤΟΥ ΔΙΑΔΙΚΤΥΟΥ

Μέχρι τώρα το διαδίκτυο χρησιμοποιούταν κυρίως από τον στρατό των ΗΠΑ για την διακίνηση πληροφοριών, αλλά και για να καθοριστεί από την

αεροπορία τους ένας τρόπος ελέγχου για βομβαρδιστικά αεροπλάνα και πυραύλους. Το 1976 πραγματοποιείται η πρώτη δορυφορική σύνδεση της Αμερικής με την Ευρώπη. Η σύνδεση πραγματοποιήθηκε στα πλαίσια του προγράμματος SATNET και οι δορυφόροι που χρησιμοποιήθηκαν άνηκαν σε κοινοπραξία της Ευρώπης και των ΗΠΑ για την επέκταση του διαδικτύου εκτός των ΗΠΑ. Με τη σύνδεση των ΗΠΑ και της Ευρώπης μέσω διαδικτύου, μπορεί να πει κάποιος πως σηματοδοτείται η λήξη της στρατιωτικής χρήσης του διαδικτύου και την άνοδο του.

Η άνοδος του, οδήγησε σε ραγδαία αύξηση της ζήτησης για διεθνή προβολή πληροφοριών και συστημάτων ηλεκτρονικού ταχυδρομείου στο διαδίκτυο. Ένας προγραμματιστής έπρεπε να χειριστεί εικόνες, χάρτες, φωτογραφίες και τόσα άλλα, όπως ένα απλό animation, σε τέτοιο βαθμό που δεν είχε σημειωθεί νωρίτερα, με χρήση μερικών γνωστών μεθόδων για την βελτίωση της ποιότητα της εικόνας στην οθόνη ή ακόμη και την αποθήκευση της (όπως η χρήση των μικρογραφιών).

Η αύξηση της χρήσης των προγραμμάτων περιήγησης, που λειτουργούν με τη γλώσσα HTML, άλλαξε τον τρόπο με τον οποίο οι πληροφορίες απεικονίζονται και ανακτώνται. Οι εκτεταμένες συνδέσεις δικτύου οδήγησαν στην ανάπτυξη των ιών των υπολογιστών, συγκεκριμένα για υπολογιστές με Windows MS, και η μεγάλη εξάπλωση των spam e-mail έγιναν ένα σημαντικό ζήτημα για τον σχεδιασμό συστημάτων ηλεκτρονικού ταχυδρομείου. Επιπλέον, η υπερφόρτωση των μέσων επικοινωνίας απαιτούσε την ανάπτυξη του ημι-αυτόματου προ-ελέγχου προτού, αρχίσει η διακίνηση των πληροφοριών στο μέσο μετάδοσης. Στις μηχανές αναζήτησης πλέον χρησιμοποιήθηκε η λέξη-κλειδί για την αναζήτηση, και πολλά συστήματα λογισμικού έπρεπε να σχεδιαστούν εκ νέου, για να μπορούν να εμφανίζουν δεδομένα τα οποία εξαρτώνται από τη βελτιστοποίηση των τεχνικών των μηχανών αναζήτησης (Search engine optimization -SEO) [14]. Απαιτήθηκαν συστήματα μετάφρασης της ανθρώπινης φυσικής γλώσσας, για να γίνει η μετάφραση της ροής πληροφοριών σε πολλές ξένες γλώσσες, σχεδιάστηκαν συστήματα λογισμικού που αξιοποιούν multi-language χρήση (βάση καταχώρησης πολλαπλών γλωσσών), που λειτουργεί συμφώνα με τις αρχές και τους κανόνες που χρησιμοποιεί ο άνθρωπος, για να μεταφράσει.

2000 ΕΩΣ ΣΗΜΕΡΑ : ΕΛΑΦΡΕΙΣ ΜΕΘΟΔΟΛΟΓΙΕΣ

Με την αυξανόμενη ζήτηση λογισμικών από πολλές μικρότερες οργανώσεις, εμφανίστηκε η ανάγκη για φθηνές λύσεις λογισμικού, που οδήγησε στην ανάπτυξη των απλουστευμένων και ταχύτερων μεθοδολογιών. Η χρήση της ταχείας κατασκευής πρωτοτύπων εξελίχθητε σε μια ελαφριά μεθοδολογία, όπως η Extreme Programming (XP) [15], η οποία προσπάθησε να απλοποιήσει πολλές περιοχές της μηχανικής λογισμικού, συμπεριλαμβανομένων των απαιτήσεων για τη συλλογή και τον έλεγχο της αξιοπιστίας και για την καλλιέργεια, από έναν μεγάλο αριθμό μικρών συστημάτων λογισμικού. Πολύ μεγάλα συστήματα λογισμικού εξακολουθούν να χρησιμοποιούν σε μεγάλο βαθμό τεκμηριωμένες μεθόδους, με πολλούς τόμους στο σύνολο τεκμηρίωσης. Όμως, μικρότερα συστήματα είχαν μια απλούστερη, ταχύτερη εναλλακτική προσέγγιση για τη διαχείριση της ανάπτυξης και της συντήρησης των υπολογισμών του λογισμικού και των αλγορίθμων για αποθήκευση, ανάκτηση και απεικόνιση των πληροφοριών.

ΣΥΓΧΡΟΝΕΣ ΤΑΣΕΙΣ ΣΤΗΝ ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ

Η μηχανική λογισμικού ακόμη εξελίσσεται και αποτελεί την ανάπτυξη μιας νέας μεθόδου. Οι κατευθύνσεις στις οποίες η τεχνολογία λογισμικού είναι υπό ανάπτυξη περιλαμβάνουν:

Πτυχές- Aspects

Βοηθούν τους μηχανικούς λογισμικού να ασχολούνται με ποιοτικά χαρακτηριστικά, με την παροχή εργαλείων για να προσθέσουν ή να αφαιρέσουν τον κωδικό boilerplate [16] από πολλές περιοχές του πηγαίου κώδικα. Οι πτυχές περιγράφουν τον τρόπο με τον οποίο όλα τα αντικείμενα ή οι λειτουργίες θα πρέπει να συμπεριφέρονται σε συγκεκριμένες περιστάσεις. Για παράδειγμα, μπορεί να προσθέσει θέματα εντοπισμού σφαλμάτων, την καταγραφή, τον έλεγχο ή το κλείδωμα σε όλα τα αντικείμενα των συγκεκριμένων τύπων. Οι ερευνητές εργάζονται στην εύρεση τρόπου, ώστε να γίνει κατανοητός ο τρόπος χρήσης τους

για θέματα σχεδιασμού κώδικα γενικής χρήσης. Σχετικές έννοιες περιλαμβάνουν παραγωγική προγραμματισμού και πρότυπα.

Ευέλικτο - Agile

Το ευέλικτο λογισμικό καθοδηγεί την ανάπτυξη των έργων λογισμικού, που εξελίσσονται γρήγορα με την αλλαγή των προσδοκιών και την ανταγωνιστικότητα των αγορών. Οι υποστηρικτές της μεθόδου αυτής πιστεύουν ότι τα έγγραφα με γνώμονα τις διαδικασίες (όπως TickIT [17], CMM [12]και ISO 9000) είναι «απαρχαιωμένα» και ξεπερασμένα. Μερικοί πιστεύουν ότι οι εταιρείες και οι οργανισμοί πλέον καταργούν πολλές από τις θέσεις εργασίας που μπορεί να καθοδηγούνται από «περίπλοκες» διαδικασίες, και στην θέση τους ενσωματώνουν νέες τεχνικές.

Πειραματική - Experimental

Πειραματική τεχνολογία λογισμικού είναι ένας κλάδος της μηχανικής λογισμικού που ενδιαφέρεται για την εκπόνηση πειραμάτων σχετικά με το λογισμικό, τη συλλογή δεδομένων από τα πειράματα, καθώς και για την επινόηση νόμων και θεωριών που προκύπτουν από αυτά τα δεδομένα. Οι υποστηρικτές αυτής της μεθόδου πιστεύουν ότι η φύση του λογισμικού είναι τέτοια, που μπορούμε να προωθήσουμε τη γνώση σχετικά με το λογισμικό μόνο μέσα από τεκμηριωμένα πειράματα.

Οδηγούμενο Μοντέλο - Model-driven

Μοντέλο με γνώμονα το σχεδιασμό και την ανάπτυξη κειμένου και γραφικών, ως κύριο αντικείμενο ενδιαφέροντος είναι το design. Τα εργαλεία ανάπτυξης του είναι διαθέσιμα για τον μετασχηματισμό του μοντέλου και την παραγωγή κώδικα με καλά οργανωμένα κομμάτια, που χρησιμεύουν ως βάση για την παραγωγή ολοκληρωμένων εφαρμογών.

Σειρές προϊόντων λογισμικού - Software product lines

Σειρές προϊόντων λογισμικού είναι ένας συστηματικός τρόπος για να παράγονται οικογένειες συστημάτων λογισμικού, αντί να δημιουργηθεί μια σειρά από εντελώς μεμονωμένα προϊόντα. Ίσως ένα από τα πιο χαρακτηρίστηκα παράδειγμα σειράς προϊόντων λογισμικού το γνωστό σε όλους Microsoft Office της εταιρείας Microsoft Inc. Η μέθοδος αυτή δίνει εκτεταμένη έμφαση στην συστηματική επαναχρησιμοποίηση κώδικα, και προσπαθεί να εκβιομηχανίσει τη διαδικασία ανάπτυξης λογισμικού.

ΜΗΧΑΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ ΣΗΜΕΡΑ

Σήμερα το επάγγελμα του Μηχανικού Λογισμικού προσπαθεί ακόμη να καθορίσει τα όρια του καθώς και το περιεχόμενο του. Το Σώμα Μηχανικών Λογισμικού της Γνώσης SWEBOK [18] έχει καταθέσει ένα πρότυπο ISO κατά τη διάρκεια του 2006 (ISO / IEC TR 19759) στο οποίο καθορίζει τα χαρακτηριστικά γνωρίσματα, τα οποία και πρέπει να έχει κάποιος για να χαρακτηριστεί Μηχανικός Λογισμικού.

Το 2006, το περιοδικό Money Magazine και το Salary.com αξιολόγησαν την μηχανική λογισμικού ως τον κλάδο με την καλύτερη δουλειά στην Αμερική όσον αφορά την ανάπτυξη, τις αμοιβές, τα επίπεδα του άγχους, την ευελιξία στις ώρες και το εργασιακό περιβάλλον, τη δημιουργικότητα, και πόσο εύκολο είναι να εισέλθουν και να προσχωρήσει κανείς στον τομέα αυτό.

ΕΠΙΛΟΓΟΣ

Είναι προφανές πως το επάγγελμα του μηχανικού λογισμικού είναι ακόμη σε πολύ «νεαρή» ηλικία. Μπορεί να πει κάποιος πως βρίσκεται στα πρώτα στάδια της εξέλιξης του και πως όσο μεγαλώνει τόσο περισσότερα πράγματα θα γνωρίσουν και θα αντιμετωπίσουν οι μηχανικοί λογισμικού αλλά και όλοι εκείνοι οι οποίοι ασχολούνται με την χρήση προγραμμάτων λογισμικού.

Πρέπει να τονιστεί ωστόσο πως η εξέλιξη της μηχανικής ήταν ραγδαία, και αυτό χάρη στην ραγδαία εξέλιξη των υπολογιστών και νέων αναγκών, που προέκυψαν για τεχνολογικά μέσα. Ίσως το κομβικό σημείο για την ανάπτυξη της μηχανικής λογισμικού ήταν η ανάπτυξη του διαδικτύου και οι νέες απαιτήσεις των χρηστών του.

Ιδιαίτερη έμφαση πρέπει να δοθεί στο γεγονός πως ταυτόχρονα με την ανάπτυξη της μηχανικής λογισμικού ραγδαία είναι και η ανάπτυξη των γλωσσών προγραμματισμού, που αποτελούν και το βασικό εργαλείο του κάθε μηχανικού. Έτσι από την ALGOL και την FORTRAN στην BASIC, την C, την C++ και την JAVA και ένα σύνολο αντικειμενοστραφών γλωσσών προγραμματισμού.

ΚΕΦΑΛΑΙΟ 2

ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ & ΜΕΘΟΔΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ

ΕΙΣΑΓΩΓΗ

Ορισμός Μεθοδολογίας Ανάπτυξης Λογισμικού :

> «Μεθοδολογία ανάπτυξης λογισμικού συστήματος ή μεθοδολογία ανάπτυξης στην μηχανική λογισμικού είναι ένα πλαίσιο που χρησιμοποιείται για την κατασκευή, το σχεδιασμό, και τον έλεγχο της διαδικασίας ανάπτυξης ενός συστήματος λογισμικού.»

Ορισμός Ποιότητας Λογισμικού :

> «Η Ποιότητα ενός προϊόντος ή μιας υπηρεσίας αντικατοπτρίζει την ικανοποίηση των απαιτήσεων του πελάτη, ακόμα και αυτών που δεν έχουν εκφραστεί.»

Από τις αρχές της δεκαετίας του 1940, όταν εμφανίστηκαν οι πρώτοι ψηφιακοί υπολογιστές, η συγγραφή προγραμμάτων εξελίχτηκε σε ένα επάγγελμα το οποίο ασχολούταν με την εύρεση νέων τρόπων για την μεγιστοποίηση της ποιότητας του λογισμικού αλλά και τον τρόπο με τον οποίο αυτό μπορεί να επιτευχθεί.

ΜΕΘΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ ΛΟΓΙΣΜΙΚΟΥ

ΓΕΝΝΗΣΗ ΤΩΝ ΜΕΘΟΔΟΛΟΓΙΩΝ ΣΧΕΔΙΑΣΜΟΥ

Από πολλούς υποστηρίζεται ότι το πλαίσιο μεθοδολογιών σχεδιασμού (γνωστό και ως SDM – Software Development Methodology [19]) μέχρι και την δεκαετία του 1960 δεν είχε εμφανιστεί. Ωστόσο, σύμφωνα με τον Elliott (2004) ο κύκλος ζωής ανάπτυξης ενός συστήματος μπορεί να θεωρηθεί από τα παλαιότερα πλαίσια ανάπτυξης μεθοδολογιών για πληροφοριακά συστήματα. Η βασική ιδέα του κύκλου ζωής ανάπτυξης ενός συστήματος, ώστε συνεχιστεί η ανάπτυξη των πληροφοριακών συστημάτων αλλά με πιο σκόπιμο, μεθοδικό και δομημένο τρόπο,

επιβάλλοντας σε κάθε κύκλο ζωής ανάπτυξης από την σύλληψη της ιδέας του προγράμματος μέχρι και την περαίωση τους, ήταν να ακολουθείται αυστηρά και διαδοχικά, το εκάστοτε πλαίσιο ανάπτυξης. Βασικός στόχος του πλαισίου την δεκαετία του 1960 ήταν «να αναπτυχθεί ένα μεγάλης κλίμακας επιχειρηματικό λειτουργικό σύστημα, σε μία εποχή οπού οι επιχειρήσεις και το μέγεθος των επιχειρήσεων μεγάλωνε. Οι δραστηριότητες των πληροφοριακών συστημάτων περιστρέφονταν γύρω από την επεξεργασία μεγάλου όγκου δεδομένων από πολλές ρουτίνες και μεθόδους».

ΠΟΙΕΣ ΕΙΝΑΙ ΜΕΘΟΔΟΛΟΓΙΕΣ ΣΧΕΔΙΑΣΜΟΥ

Κατά καιρούς έχουν δοθεί πολλές διαφορετικές περιγραφές σχετικά με το τι είναι μέθοδος και τι μεθοδολογία. Παρακάτω, γίνεται μια προσπάθεια να δοθούν ορισμένοι πιο σωστοί ορισμοί για τους όρους αυτούς.

- Μεθοδολογία (ή μέθοδος) είναι μια συγκεκριμένη συλλογή από αρχές ή / και πρακτικές.
- Οικογένεια Μεθοδολογιών είναι μια σειρά από εναλλακτικές μεθόδους, που υπάρχουν παράλληλα με κάθε άλλη μέθοδο.
- Πλαίσιο είναι ο σκελετός μίας μεθόδου που πρέπει να προσαρμοστεί ή ακόμη και να αυξηθεί πριν από τη χρήση του.
- Μοντέλο είναι μια περιγραφή των μεθόδων, η οποία πρέπει να εφαρμοστεί στην μέθοδο που θα χρησιμοποιηθεί.

Για την πλειοψηφία του κόσμου των προγραμματιστών ωστόσο, μια μεθοδολογία είναι μία κατηγορία, η οποία αρχικοποιείται πριν την έναρξη ενός έργου. Μια οικογένεια μεθοδολογιών είναι συνώνυμο των διάφορων κατηγοριών αλλά ταυτόχρονα αποτελεί και το μέτρο σύγκρισης τους. Ένα πλαίσιο μεθοδολογίας μπορεί να πει κάποιος πως είναι όπως μια αφηρημένη κλάση, η οποία δεν μπορεί να υλοποιηθεί, αν πρώτα δεν κληρονομηθεί και δεν επεκταθεί. Ένα μοντέλο μεθοδολογίας είναι σαν τις διεπαφές που περιέχουν απλά μια περιγραφή και πρέπει να εφαρμοστούν σε μία ή περισσότερες κατηγορίες .

Οι μεθοδολογίες οι οποίες υπάρχουν και είναι γνωστές σήμερα είναι συνολικά ενενήντα τέσσερις. Από αυτές ορισμένες είναι πιο γνωστές και χρησιμοποιούνταν ή χρησιμοποιούνται ακόμη από τους προγραμματιστές, αυτές λοιπόν οι μεθοδολογίες είναι :

1. Extreme Programming ή XP. Είναι μια ευέλικτη μεθοδολογία ανάπτυξης λογισμικού που αναπτύχθηκε/δημιουργήθηκε από τον Kent Beck. Πρόκειται για ένα σύνολο από τις βέλτιστες πρακτικές οι οποίες εφαρμόστηκαν σε ακραίες καταστάσεις. Όπως και κάποιες άλλες ευέλικτες μεθοδολογίες έτσι και η XP, αφορά τις συνεχιζόμενες αλλαγές των απαιτήσεων, κάτι άκρως φυσικό και αναμενόμενο για την ανάπτυξη λογισμικού. Οι επιλογές των πρακτικών της XP, που θα χρησιμοποιηθούν, έχουν να κάνουν και με τις δραστηριότητες του προγραμματιστή ως προς την ανάπτυξη του έργου.
2. Rational Unified Process (RUP). Η RUP είναι ένα πλαίσιο τεχνολογίας λογισμικού, που δημιουργήθηκε και συντηρείται από τους ανθρώπους της Rational Software (η οποία και σήμερα ανήκει στην IBM), και τον Philippe Kruchten. Πρόκειται για ένα εμπορικό προϊόν το οποίο έχει παραδοθεί με μία πιο λεπτομερή έκδοση της Unified Software Development Process.
3. Waterfall. Πρόκειται για ένα μοντέλο που ασχολείται με την διαδοχική προσέγγιση της ανάπτυξης, στην οποία η ροή ανάπτυξης θεωρείται σταθερή προς τα κάτω σε όλες τις φάσεις των απαιτήσεων ανάλυσης, σχεδιασμού, υλοποίησης, ελέγχου, ολοκλήρωσης και συντήρησης. Με αυτό το μοντέλο το έργο χωρίζεται σε φάσεις, οι οποίες δεν είναι ανεξάρτητες μεταξύ τους. Δίνεται ιδιαίτερη έμφαση στον προγραμματισμό και στο χρονοδιάγραμμα του έργου. Πραγματοποιούνται τακτικοί έλεγχοι, τόσο με το τέλος μια φάσης όσο και πριν την έναρξη της επόμενης, για να διαπιστωθεί, αν έχουν καλυφθεί ή όχι οι απαιτήσεις του χρήστη.

ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ

Ο Lord Kelvin αναφέρει πως :

> "Όταν μπορείς να μετρήσεις αυτό για το οποίο μιλάς και να το εκφράσεις σε αριθμούς, τότε ξέρεις κάτι σχετικά με αυτό. Όταν δε

μπορείς να το μετρήσεις και να το εκφράσεις σε αριθμούς, η γνώση σου για αυτό είναι φτωχή και μη ικανοποιητική."

Τα λόγια του ισχύουν και για την ποιότητα λογισμικού και αναφέρονται στη μέτρηση των χαρακτηριστικών που περιγράφουν τα κατώτερα επίπεδα αφαίρεσης των μοντέλων ποιότητας. Στην Τεχνολογία Λογισμικού οι όροι «μέτρηση» (measure) και «μετρική» (metric) χρησιμοποιούνται πολλές φορές ως συνώνυμες.

Ως Μέτρηση (Measure) μπορεί να ορισθεί η αντιστοίχηση ενός μεγέθους με μία τιμή . Δηλαδή, με τον όρο μέτρηση ορίζεται η διαδικασία με την οποία αριθμοί ή σύμβολα αντιστοιχίζονται με κάποιες ιδιότητες ορισμένων οντοτήτων του πραγματικού κόσμου, με σκοπό να μπορούν να τις περιγράψουν με έναν προκαθορισμένο σύνολο κανόνων. Στον αντίποδα, ο DeMacro αναφέρει ότι δεν είναι εφικτό να ελέγξεις ότι δεν μπορείς να μετρήσεις. Άρα, είναι ακόμη πιο αδύνατον να επιδιώκει κανείς να εξασφαλίσει την ποιότητα για κάτι, το οποίο δεν μπορεί να μετρήσει.

Ως μετρική (metric), ορίζεται η ποσοτική μέτρηση του βαθμού στον οποίο ένα σύστημα ή τμήμα του συστήματος έχει ένα χαρακτηριστικό. Ο Fenton ορίζει τη μετρική ως μία εμπειρική και αντικειμενική αντιστοίχηση ενός αριθμού ή συμβόλου σε μία οντότητα με σκοπό να χαρακτηρίσει ένα συγκεκριμένο χαρακτηριστικό της.

Ο Edward Berard ορίζει τη μετρική ως μία μονάδα μέτρησης και αναφέρει ότι ο όρος χρησιμοποιείται για να δηλώσει ένα σύνολο μετρήσεων, που πραγματοποιούνται πάνω σε συγκεκριμένο αντικείμενο ή διαδικασία.

Πολύ εύκολα μπορεί να γίνει αντιληπτό πως, όταν κάποιος αναφέρεται στην μετρική, ουσιαστικά αναφέρεται σε μια «μεθοδολογία» μέτρησης, που στην πράξη κάνει μια αντιστοίχηση μεταξύ μίας τιμής και κάποιας ιδιότητας του αντικειμένου. Για κάθε μετρική υπάρχουν πέντε βαθμίδες μέτρησης:

1. Η ονομαστική (Nominal), όπου μεταξύ των δεδομένων η μόνη ιδιότητα που ισχύει είναι η ιδιότητα της ισότητας,
2. Η τακτική (Ordinal), στην οποία η ιδιότητα μεταξύ των δεδομένων είναι η ισότητα και η διάταξη,
3. Η διαστήματος (Interval), όπου οι αποστάσεις που υπάρχουν μεταξύ των δεδομένων αποτελούν πληροφορία,
4. Η αναλογική (Ratio), στην οποία έχουν νόημα και οι αναλογίες μεταξύ των δεδομένων και

5. Η απόλυτη (Absolute), στην οποία επιτρέπεται και η αρίθμηση των δεδομένων.

Οι μετρικές λογισμικού χωρίζονται σε τρείς βασικές κατηγορίες οι οποίες είναι :

1. Μετρικές Προϊόντος, οι οποίες σχετίζονται με το τελικό προϊόν και τον πηγαίο κώδικα, αλλά και τις δηλώσεις ελέγχου που γίνονται. Οι Μετρικές Προϊόντος χωρίζονται σε δύο υποκατηγορίες :
    a. Τις εσωτερικές μετρικές οι οποίες μετρούν των αριθμό γραμμών του προϊόντος (LOC), τον χρόνος εκτέλεσης του προγράμματος, τα λάθη, που μπορεί να υπάρχουν στον κώδικα
    b. Τις εξωτερικές μετρικές οι οποίες μετρούν την λειτουργικότητα του προϊόντος (Functionality), την ποιότητα (Quality), την πολυπλοκότητα (Complexity), την αποτελεσματικότητα (Efficiency), την Αξιοπιστία (Reliability), και την συντηρησιμότητα (Maintainability) του .
2. Μετρικές Έργου ασχολούνται με τον καθορισμό της στρατηγικής που θα ακολουθηθεί, σχετίζονται άμεσα με την τακτική εκτέλεσης του υπό ανάπτυξη έργου, καθώς, επίσης, καθορίζουν και την ροή των τεχνικών που θα χρησιμοποιηθούν.
3. Μετρικές Διαδικασίας σχετίζονται με την μέτρηση των διαδικασιών κατασκευής του τελικού προϊόντος, δηλαδή τον σχεδιασμό, την συγγραφή του κώδικα και τους πόρους, που απαιτούνται να χρησιμοποιηθούν προκειμένου να ολοκληρωθεί η ανάπτυξη του έργου.

ΜΕΤΡΙΚΕΣ ΤΗΣ ΠΟΙΟΤΗΤΑ ΛΟΓΙΣΜΙΚΟΥ

Οι μετρικές προϊόντος σχετίζονται με το προϊόν το οποίο αναπτύσσεται, για παράδειγμα τον πηγαίο κώδικα που έχει γραφτεί ή τις δηλώσεις ελέγχου που περιέχονται μέσα στον κώδικα. Χωρίζονται σε δύο επιπλέον κατηγορίες :

i. Τις εσωτερικές μετρικές (Αριθμός Γραμμών (LOC), Χρόνος Εκτέλεσης, Λάθη του Κώδικα) και
ii. Τις εξωτερικές μετρικές (Λειτουργικότητα (Functionality), Ποιότητα (Quality), Πολυπλοκότητα (Complexity), αποτελεσματικότητα (Efficiency), Αξιοπιστία (Reliability), Συντηρησιμότητα (Maintainability))

Ενδιαφέρον παρουσιάζει η σχέση μεταξύ των εσωτερικών και των εξωτερικών μετρικών, διότι οι εσωτερικές μετρικές ερμηνεύονται σε εξωτερικές, ώστε να γίνει πιο εύκολη η μέτρηση τους.

Οι εσωτερικές μετρικές σε σχέση με τις εξωτερικές είναι πιο εύκολο να αυτοματοποιηθούν. Για παράδειγμα, είναι ευκολότερο να μετράς γραμμές κώδικα από να προσπαθείς να μετρήσεις, πόσο ευχαριστημένος είναι ο τελικός χρήστης από το αποτέλεσμα των υπηρεσιών, που ζήτησε και την λειτουργικότητα που του προσφέρει το συγκεκριμένο πρόγραμμα λογισμικού.

Οι εσωτερικές μετρικές έχουν ένα ακόμη πλεονέκτημα. Αυτό είναι πως η συχνότητα λαθών είναι μικρότερη , γιατί μετρούν καθορισμένα χαρακτηριστικά λογισμικού, όπως το σύνολο των γραμμών κώδικα.

Αντίθετα, με τις εσωτερικές μετρικές, οι εξωτερικές μετρικές είναι υψηλού επιπέδου και βρίσκονται πιο κοντά στην έννοια της ποιότητας και τα αποτελέσματα τους είναι άμεσα ερμηνεύσιμα.

ΜΕΤΡΙΚΕΣ ΕΡΓΟΥ

Οι μετρικές έργου χρησιμοποιούνται για να καθοριστεί η στρατηγική, που θα χρησιμοποιηθεί, και σχετίζεται με την τακτική εκτέλεσης του έργου και καθορίζει τη ροή του και τις τεχνικές, που θα ακολουθηθούν.

ΜΕΤΡΙΚΕΣ ΔΙΑΔΙΚΑΣΙΑΣ

Οι μετρικές διαδικασίας μετρούν τη διαδικασία κατασκευής ενός προϊόντος, δηλαδή το σχεδιασμό, τη συγγραφή κώδικα, και τους απαιτούμενους πόρους για την σωστή ανάπτυξη του έργου. Για παράδειγμα, σε μια εταιρεία, οι μετρικές προϊόντος καθορίζονται από ένα άτομο. Ο συνδυασμός του ατόμου με τις μετρικές προϊόντος αποτελούν τις μετρικές του έργου, που κοινοποιούνται στην ομάδα ανάπτυξης του έργου. Ο συνδυασμός τώρα της ομάδας ανάπτυξης και των μετρικών έργου αναπτύσσει τις μετρικές διαδικασίας, που ανήκουν σε όλη την εταιρία. Το βασικότερο ερώτημα είναι πώς μία εταιρία μπορεί να συνδυάσει τις μετρικές από διαφορετικά άτομα-εργαζόμενους ή και διαφορετικά έργα, ώστε να παραγάγει τις μετρικές διαδικασίας;

Για παράδειγμα, εργαζόμενοι από δύο διαφορετικές ομάδες καταγράφουν και κατηγοριοποιούν όλα τα λάθη που προκύπτουν κατά τη διάρκεια μίας διαδικασίας. Οι μετρήσεις αυτές συνδυάζονται, για να προκύψουν τα αποτελέσματα: η ομάδα Α βρήκε χ αριθμό από λάθη κατά τη διαδικασία, ενώ η ομάδα Β ψ αριθμό λαθών. Για να βγει συμπέρασμα για το, ποια ομάδα είναι πιο αποτελεσματική, θα πρέπει να ληφθούν υπόψη και άλλες παράμετροι, όπως το μέγεθος ή η πολυπλοκότητα των έργων και οι μετρήσεις αυτές να κοινοποιηθούν.

ΕΠΙΛΟΓΟΣ

Είναι πασιφανές, πως οι μετρικές λογισμικού και οι μεθοδολογίες λογισμικού είναι δύο βασικές έννοιες στην μηχανική λογισμικού. Αν δεν επιλεχθεί η κατάλληλη μεθοδολογία σχεδιασμού, τότε η ανάπτυξη του έργου δεν θα είναι σωστή και το αποτέλεσμα δεν θα είναι αυτό, που ο πελάτης θα έχει ζητήσει. Αυτό μπορεί να το καταλάβει κάποιος, αν χρησιμοποιήσει τις μετρικές λογισμικού, για να μπορέσει να εξάγει αποτελέσματα σχετικά με το προϊόν, το οποίο παρήγαγε. Με τις μετρικές λογισμικού μπορεί να λάβει αποτελέσματα από τα οποία να αξιολογήσει και το τελικό προϊόν, ως προς την πολυπλοκότητα του, τα σφάλματα του, τις γραμμές του κώδικα που έχει γράψει, τον αριθμό των διαδικασιών που χρησιμοποιεί σε όλη την έκταση του έργου, αλλά εν μέρει και την ομάδα ανάπτυξης, από των αριθμό των λαθών που προέκυψαν στην ανάπτυξη του τμήματος, που της ανατέθηκε. Όσο αναφορά την αξιολόγηση της ομάδα θα πρέπει, πάντοτε, να λαμβάνονται υπόψη και παράγοντες που αφορούν την δυσκολία ανάπτυξης ενός συγκεκριμένου κομματιού, αλλά την «υπολογιστική ισχύ» της συγκεκριμένης ομάδας ανάπτυξης.

ΚΕΦΑΛΑΙΟ 3

ΜΕΘΟΔΟΛΟΓΙΑ ΣΧΕΔΙΑΣΜΟΥ SCRUM

ΕΙΣΑΓΩΓΗ

(Στο κεφάλαιο που ακολουθεί χρησιμοποιούνται όροι των οποίων η έννοια αναλύεται και εξηγείται σε ειδικό Λεξιλόγιο όρων, που έχει αναπτυχθεί και βρίσκεται στην σελίδα 69)

Στο προηγούμενο κεφάλαιο αναφέρθηκαν ορισμένες μεθοδολογίες σχεδιασμού, όπως η Rational Unified Process (RUP), Extreme Programming (XP) και η Waterfall (Καταρράκτη). Σκοπίμως, όμως, δεν αναφέρθηκε μια που για πολλούς η μελέτη της παρουσιάζει ιδιαίτερο ενδιαφέρον. Αυτή, λοιπόν, η μεθοδολογία είναι η SCRUM.

Πρόκειται για μία επαναληπτική και αυξητική μεθοδολογία ανάπτυξης λογισμικού για την διαχείριση έργων ή ακόμη την ανάπτυξη μίας εφαρμογής. Η SCRUM είναι μια μεθοδολογία η οποία συνέβαλλε στην ενίσχυση του ενδιαφέροντος του σχεδιαστή μια εφαρμογής, αλλά ταυτόχρονα αμφισβήτησε και τις συμβατικές ιδέες για την διαχείριση των έργων. Η χρήση της επικεντρώνεται κυρίως για την διαχείριση προγραμμάτων / έργων, των οποίων η μελλοντική διαχείριση του προγράμματος είναι σε πολλές περιπτώσεις ανέφικτη. Στην SCRUM οι εμπειρικοί μηχανισμοί ελέγχου είναι αυτοί που χρησιμοποιούνται έναντι των παραδοσιακών εντολών ελέγχου για την διαχείριση του έργου, όπου οι μηχανισμοί ανατροφοδότησης, που χρησιμοποιεί, αποτελούν την βασική τεχνική ελέγχου του πυρήνα του έργου. Πρόκειται για μία εντελώς καινούργια προσέγγιση για τον σχεδιασμό και την διαχείριση έργων, φέρνοντας πια την εξουσία για την λήψη των απαραίτητων αποφάσεων στο επίπεδο της λειτουργίας.

Η ΙΣΤΟΡΙΑ ΤΗΣ SCRUM

Η μεθοδολογία περιγράφηκε για πρώτη φόρα το 1986 από τους Hirotaka Takeuchi και Ikujiro Nonaka. Η περιγραφή που της δόθηκε ήταν μιας νέας

προσέγγισης για την εμπορική ανάπτυξη κάποιου προϊόντος, που θα συνέβαλε στην αύξηση της ταχύτητας και της ευελιξίας, και όλα αυτά με βάση την μελέτη κάποιων περιπτώσεων χρήσης που έγιναν από εταιρείες παραγωγής αυτοκινήτων, φωτοτυπικών μηχανημάτων αλλά και εκτυπωτών. Την προσέγγιση αυτή αποκάλεσαν ολιστική, μιας και στην ισχύουσα διαδικασία παραγωγής επικάλυπτε η μία την άλλη με αποτέλεσμα να μην υπάρχει σταθερή πρόοδος.

Αν παρομοιάσει κανείς την διαδικασία παραγωγής με ένα παιχνίδι ποδοσφαίρου, τότε η μεθοδολογία SCRUM θα είναι ο διαιτητής του παιχνιδιού, ο οποίος ορίζει τον τρόπο με τον οποίο θα συνεχιστεί το παιχνίδι/παραγωγή μετά από μία παράβαση. Στις αρχές του 1990 ο Ken Schwber και οι Jeff Sutherland, μαζί με τον John Scumniotales και τον Jeff McKenna ήταν οι πρώτοι που χρησιμοποίησαν την μεθοδολογία στις εταιρείες τους ,Advanced Development Methods και Easel Corporation αντίστοιχα, και της προσέδωσαν τον όρο / την ονομασία SCRUM.

Το 1995, οι Sutherland και Schwaber παρουσίασαν από κοινού ένα έγγραφο στο εργαστήριο Business Object Design and Implementation το οποίο πραγματοποιήθηκε ως μέρος του OOPSLA '95 [20] στο Ώστιν του Τέξας. Τότε έγινε η πρώτη δημόσια παρουσίαση του, που περιέγραφε τη μεθοδολογία. Τα χρόνια που ακολούθησαν οι δύο τους συνεργάστηκαν και συγχώνευσαν/συνένωσαν τόσο τα συγγράμματα τους, όσο και τις εμπειρίες τους για τις βέλτιστες/ευέλικτες πρακτικές, οι οποίες ήταν γνωστές, αλλά πιο συγκεκριμένα για την μεθοδολογία SCRUM. Το 2001 ο Schwaber ξεκινάει μια συνεργασία με τον Mike Beedle με βασικό σκοπό την περιγραφή της νέας αυτής μεθοδολογίας στο βιβλίο Agile Software Development with Scrum.

Αν και το όνομα της μεθοδολογίας δεν προέρχεται από κάποιο αρκτικόλεξο, ορισμένες εταιρείες ψάχνουν μέχρι και σήμερα να βρουν από πού πραγματικά μπορεί να προέρχεται το όνομα της.

ΟΙ ΡΟΛΟΙ ΣΤΗΝ ΜΕΘΟΔΟΛΟΓΙΑ

Στην μεθοδολογία υπάρχουν τρεις «βασικοί» ρόλοι ή ρόλοι πυρήνα αλλά και μία σειρά από «βοηθητικούς» βασικούς , οι οποίοι και κατανέμονται στα μέλη της ομάδας προγραμματισμού. Συχνά οι «βασικοί» ρόλοι αναφέρονται ως «γουρούνια», ενώ οι δευτερεύοντες ρόλοι αναφέρονται ως «κότες» (οι

προσδιορισμοί αυτοί προέκυψαν από την ιστορία «Το κοτόπουλο και το γουρούνι») .

ΡΟΛΟΙ ΠΥΡΗΝΑ

Οι ρόλοι πυρήνα είναι εκείνοι που δεσμεύονται με την SCRUM. Ουσιαστικά, είναι αυτοί που παράγουν το προϊόν. Αποτελούν την ομάδα της μεθοδολογίας.

Στους ρόλους πυρήνα περιλαμβάνονται τα ακόλουθα πρόσωπα :

A. Ο ιδιοκτήτης προϊόντος, ο οποίος αντιπροσωπεύει τα ενδιαφερόμενα μέρη και αποτελεί την «φωνή» του πελάτη. Το πρόσωπο που θα καταλάβει την συγκεκριμένη θέση είναι υπεύθυνο για την εξασφάλιση, πως η ομάδα θα δίνει αξία στην επιχείρηση. Ο ιδιοκτήτης του προϊόντος γράφει τα πελατοκεντρικά στοιχεία δίνοντας προτεραιότητα σε ορισμένα και προσθέτοντας τα στο Υπόλοιπο ανεκτέλεστο προϊόν – Product backlog. Η ομάδα θα πρέπει να έχει οπωσδήποτε κάποιον ιδιοκτήτη προϊόντος, αυτός ο ρόλος μπορεί να δοθεί σε κάποιον από την ομάδα ανάπτυξης. Ο «κάτοχος» του συγκεκριμένου ρόλου δεν μπορεί ταυτόχρονα να κατέχει και τον ρόλο του SCRUM Master.

B. Η Ομάδα Ανάπτυξης – Development Team είναι υπεύθυνη για να την παράδοση εκτελέσιμων αρχείων στο τέλος κάθε φάσης της διαδικασίας. Συνήθως αποτελείται από 3-9 άτομα τα οποία έχουν τις ικανότητες/δεξιότητες για να κάνουν διάφορες εργασίες (ανάπτυξη, σχεδιασμό, δοκιμή, τεχνικές επικοινωνίας, κλπ). Η Ομάδα ανάπτυξης είναι μια «ανεξάρτητη» οργάνωση και μπορεί να διασυνδεθεί με τις υπόλοιπες οργανώσεις διαχείρισης και ανάπτυξης του έργου.

C. Scrum Master είναι υπεύθυνος για την πρόοδο της SCRUM, δηλαδή, είναι αυτός ο οποίος θα υποδείξει τον τρόπο με τον οποίο θα μπορέσει η ομάδα να προσπεράσει εμπόδια, τα οποία μπορεί να παρουσιαστούν, ώστε η ομάδα να μπορέσει να φτάσει στην επίτευξη του στόχου αλλά και στην παράδοση των απαραίτητων παραδοτέων. Ο Scrum Master δεν είναι ο ηγέτης της ομάδας, αλλά αποτελεί τον ενδιάμεσο μεταξύ της ομάδας και όλων εκείνων των επιρροών που συντελούνται γύρω τους και μπορούν να της αποσπάσουν την προσοχή. Επιπλέον, επιδιώκει πως η διαδικασία θα χρησιμοποιείται σύμφωνα με όσα αυτή ορίζει, άρα είναι ο ελεγκτής τήρησης των κανόνων της μεθοδολογίας. Μία ακόμη αρμοδιότητα του είναι να

προστατεύει την ομάδα ανάπτυξης από εξωγενείς παράγοντες που μπορούν άμεσα να επηρεάσουν την παραγωγικότητα της και την βοηθά επικεντρωθεί στο έργο της. Πολλές φορές εσφαλμένα έχει συσχετιστεί ο ρόλος του SCRUM Master με αυτόν του Project Manager. Η βασική διαφορά μεταξύ των δύο αυτών ρόλων είναι πως ο ρόλος του Project Manager μπορεί να ανατεθεί σε κάποιον ο οποίος δεν θα έχει πολλές αρμοδιότητες στην ανάπτυξη του έργου αλλά δεν μπορεί να συμβεί το ίδιο και με την θέση του Scrum Master.

a. Οι υπηρεσίες του Scrum Master προς τον ιδιοκτήτη του προϊόντος. Ο SCRUM Master εξυπηρετεί τον ιδιοκτήτη με τους ακόλουθους τρόπους :

   i. Στην εύρεση τεχνικών για την αποτελεσματική διαχείριση των ανεκτέλεστων προϊόντων

   ii.Στον τρόπο επικοινωνίας για να γίνει σαφές το όραμα, οι στόχοι, και τα στοιχεία των ανεκτέλεστων προϊόντων από την Ομάδα Ανάπτυξης

   iii. Στην διδασκαλία της ομάδας SCRUM ώστε να δημιουργήσουν σαφείς και συνοπτικές αναλύσεις σχετικά με τα ανεκτέλεστα στοιχεία. Επιπλέον, στην κατανόηση του μακροπρόθεσμου σχεδιασμού του προϊόντος σε ένα εμπειρικό περιβάλλον

   iv. Στην κατανόηση και την εξάσκηση της ευελιξίας της ομάδας ,

   Και τέλος

   v. Στην διευκόλυνση της εκτέλεσης της μεθοδολογίας σε γεγονότα όπως ζητείται ή απαιτείται.

b. Οι υπηρεσίες του Scrum Master προς την ομάδα ανάπτυξης. Ο Scrum Master εξυπηρετεί την ομάδα ανάπτυξης με τους παρακάτω τρόπους :

   i. Εκπαιδεύει την Ομάδα Ανάπτυξης του έργου, ώστε να λειτουργεί εν μέρει αυτόνομα και να έχει πολλαπλή λειτουργικότητα

   ii.Μέσα από την διδασκαλία εκπαίδευσης οδηγεί την Ομάδα Ανάπτυξης στη δημιουργία υψηλής αξίας προϊόντων

   iii. Κατάργηση των εμποδίων που προκύπτουν κατά την ανάπτυξη του έργου και επηρεάζουν την ομάδα ανάπτυξης

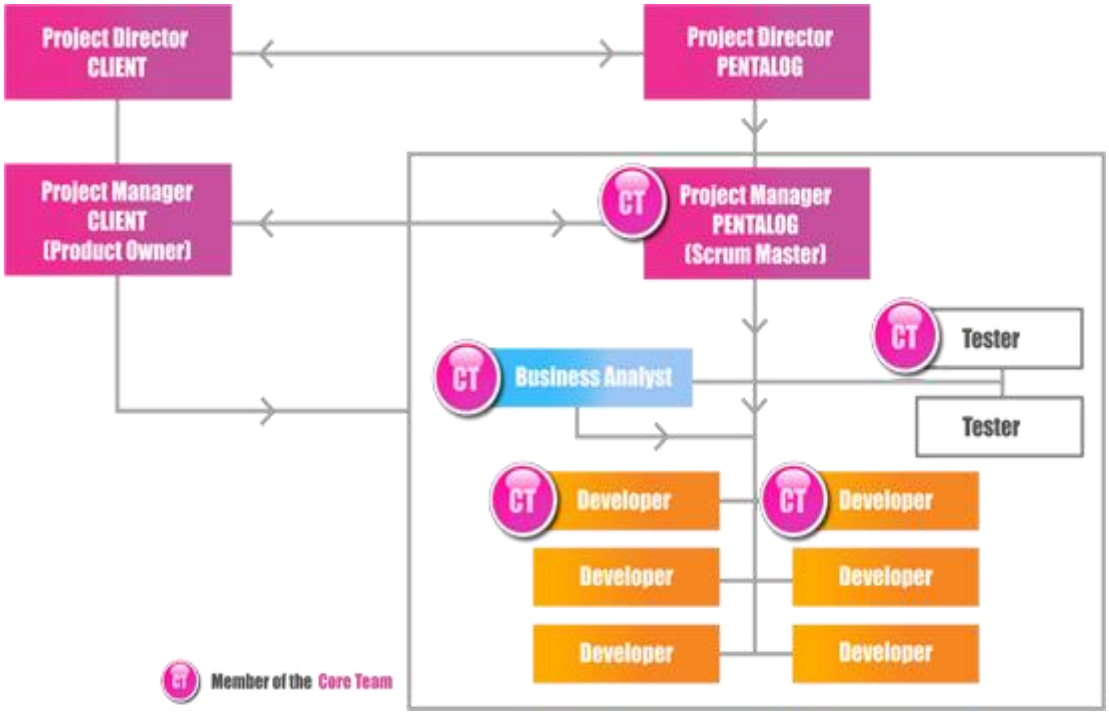
iv. Διευκόλυνση στη συνέχιση εκτέλεσης της μεθοδολογίας Scrum μέσα από γεγονότα όπως αυτά ζητήθηκαν,

Και τέλος

v. Εκπαίδευση της Ομάδα Ανάπτυξης του, ώστε να εργάζονται σε περιβάλλοντα στα οποία η Scrum δεν έχει ακόμη υιοθετηθεί και κατανοηθεί πλήρως.

c. Οι υπηρεσίες του Scrum Master προς τον οργανισμό . Ο Scrum Master εξυπηρετεί την οργάνωση με τρόπους όπως :

i. Οδηγώντας και καθοδηγώντας την οργάνωση να υιοθετήσει την μεθοδολογία Scrum
ii. Σχεδιάζοντας εφαρμογές Scrum εντός του οργανισμού
iii. Βοηθώντας τους εργαζόμενους και τα ενδιαφερόμενα μέλη να κατανοήσουν και να θεσπίσουν την χρήση της Scrum και την εμπειρική ανάπτυξη του προϊόντος
iv. προκαλεί αλλαγές που αυξάνουν την παραγωγικότητα της ομάδας, και,
v. Έρχεται σε συνεργασία με άλλες ομάδες ανάπτυξης και Scrum Masters, για να αυξήσει την αποτελεσματικότητα της εφαρμογής της Scrum στην οργάνωση.



Εικόνα c Οργανόγραμμα της SCRUM

3.2.2 ΒΟΗΘΗΤΙΚΟΙ ΡΟΛΟΙ

Στους βοηθητικούς ρόλους περιλαμβάνονται οι ομάδες εκείνες που δεν έχουν κάποιο επίσημο ρόλο και η συμμετοχή τους στις διαδικασίες της μεθοδολογίας είναι σπάνια. Παρόλα αυτά, πρέπει να αναφερθούν και τα πρόσωπα που περιλαμβάνονται :

1. Οι ενδιαφερόμενοι. Στην ομάδα αυτή περιλαμβάνονται οι πελάτες και οι προμηθευτές. Ουσιαστικά περιλαμβάνει όλους εκείνους, που ενεργοποιούν το έργο και όλους εκείνους για τους οποίους προορίζεται το έργο και ωφελούνται από την παραγωγή του. Οι συγκεκριμένοι εμπλέκονται μόνο κατά την διάρκεια παράδοσης των εκτελέσιμων αρχείων κώδικα.
2. Managers οι οποίοι είναι όλοι εκείνοι, που ελέγχουν το περιβάλλον εργασίας της ομάδας ανάπτυξης.

ΦΑΣΕΙΣ ΚΑΙ ΣΤΑΔΙΑ ΑΝΑΠΤΥΞΗΣ

ΦΑΣΗ ΑΝΑΠΤΥΞΗΣ

Η φάση ανάπτυξης είναι η βασική μονάδα στην ανάπτυξη ενός έργου σύμφωνα με την SCRUM. Η φάση ανάπτυξης μπορεί να διαρκέσει από μια εβδομάδα έως και έναν μήνα και περιορίζεται εντός συγκεκριμένης διάρκειας, η οποία αναμένεται να είναι σταθερού μήκους.

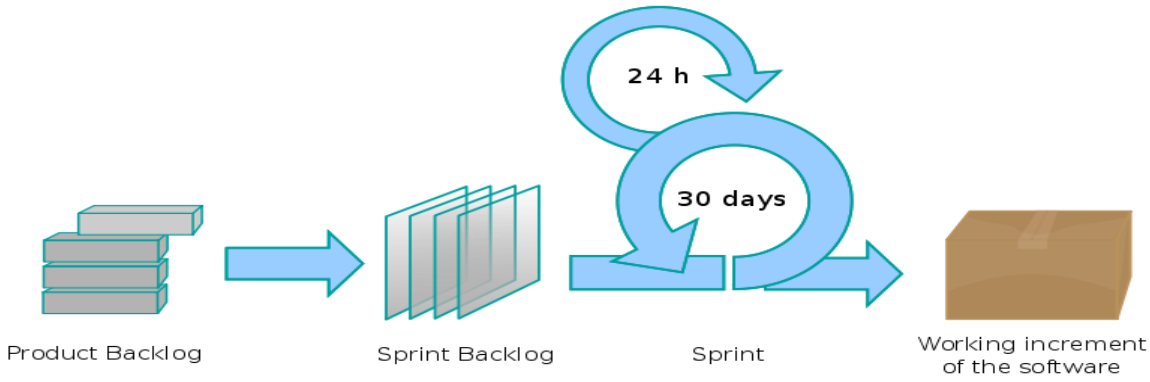
Πριν από κάθε φάση έχει προηγηθεί μια συνάντηση προγραμματισμού, όπου εντοπίζονται οι εργασίες που πρέπει να ολοκληρωθούν για τη συγκεκριμένη φάση ανάπτυξης και εκτιμάται μια χρονική δέσμευση για την εκπόνηση του συγκεκριμένου στόχου. Με το πέρας της συγκεκριμένης συνάντησης μπορεί είτε να ακολουθήσει μια συνάντηση επανεξέτασης, είτε μια αναδρομική συνάντηση επανεξέτασης των στόχων, όπου αξιολογείται η πρόοδος ανάπτυξης του έργου και εντοπίζονται τυχόν λάθη για την αντιμετώπιση τους στην επόμενη φάση ανάπτυξης.

Κατά την διάρκειά της, η ομάδα δημιουργεί ένα τελικό τμήμα του προϊόντος (εκτελέσιμο), που περιέχει τις μέχρι τώρα απαιτήσεις, που έχουν αναπτυχθεί και

αναλυθεί. Το σύνολο των χαρακτηριστικών που οδηγούν σε μία φάση προέρχονται από την συσσώρευση των απαιτήσεων, που δεν έχουν αναλυθεί. Ουσιαστικά, πρόκειται για μια λίστα η οποία περιλαμβάνει τις απαιτήσεις του πελάτη, όπου αυτές είναι ταξινομημένες με σειρά προτεραιότητας, δηλαδή με τέτοιο τρόπο, ώστε να υλοποιηθούν πρώτα οι πιο σημαντικές και απαραίτητες απαιτήσεις σύμφωνα πάντα με την προσωπική κρίση του πελάτη. Το σύνολο του ανεκτέλεστου έργου το οποίο θα πάει στην επόμενη φάση (δηλαδή στους στόχους περαίωσης της επόμενης φάσης) καθορίζεται κατά την διάρκεια του σχεδιασμού της. Στη συνάντηση αυτή ο ιδιοκτήτης του προϊόντος ενημερώνει την ομάδα για τα στοιχεία του ανεκτέλεστου προϊόντος, που επιθυμεί να ολοκληρωθούν πρώτα (στοιχεία που κατά την κρίση του ιδιοκτήτη είναι υψηλότερης σημασίας). Η ομάδα συνεχίζει να καθορίζει όλα εκείνα τα στοιχεία για τα οποία μπορεί να εξασφαλίσει την ολοκλήρωση τους κατά την διάρκεια της επόμενης φάσης. Επιπλέον, καθορίζονται τα μέλη που θα συμμετάσχουν, για να ολοκληρωθούν οι απαιτήσεις στην τρέχουσα φάση. Η φάση του ανεκτέλεστου έργου είναι ιδιοκτησία της ομάδας ανάπτυξης, και κανένας δεν επιτρέπεται να την επεξεργαστεί εκτός από την ίδια. Οι στόχοι μιας φάσης ανάπτυξης δεν μπορούν να αλλάξουν κατά την διάρκειά της. Το σημείο λήξης της κάθε φάσης τίθεται την κατάλληλη χρονική περίοδο έτσι ώστε, η φάση να έχει ολοκληρωθεί. Σε περίπτωση που δεν έχουν καλυφθεί όλες οι απαιτήσεις για οποιοδήποτε λόγο, δε μπορεί πλέον να συνεχιστεί η ανάπτυξη τους και ανήκουν στο ανεκτέλεστο τμήμα του έργου. Με την ολοκλήρωση μιας φάσης η ομάδα παραδίδει ένα δείγμα του έργου που περιέχει τις απαιτήσεις, που μέχρι τώρα έχουν υλοποιηθεί.

Η μεθοδολογία επιτρέπει την δημιουργία αυτόνομων ομάδων και αυτό επιτυγχάνεται με την ενθάρρυνση και παρότρυνση των μελών τους να εγκαθίστανται στον ίδιο χώρο, ώστε να μπορούν να επικοινωνούν μεταξύ τους για την ανάπτυξη του τμήματος του προϊόντος που έχουν αναλάβει. Βασική αρχή της μεθοδολογίας είναι η αναγνώριση τόσο των νέων απαιτήσεων των πελατών, που προκύπτουν κατά την ανάπτυξη του έργου, όσο και των απρόβλεπτων νέων απαιτήσεων. Έτσι, η SCRUM υιοθετεί μια πιο εμπειρική προσέγγιση αυτών, αποδεχόμενη πως ορισμένα προβλήματα μπορεί να μην κατανοηθούν πλήρως ή ορθά κατά την πρώτη ανάλυση τους . Για αυτόν το λόγο η μεθοδολογία εστιάζει την προσοχή της στην εύρεση τρόπων για τη μεγιστοποίηση της ικανότητας της

ομάδας να ανταποκριθεί, να αναγνωρίσει και να καλύψει έγκαιρα τις απαιτήσεις που προκύπτουν εκ των υστέρων.



Εικόνα d Πρόοδος των διαδικασιών της SCRUM

Η SCRUM, όπως και άλλες ευέλικτες μέθοδοι, μπορεί να υλοποιηθεί μέσα από τη χρήση ενός μεγάλου φάσματος εργαλείων. Πολλές είναι οι εταιρείες που χρησιμοποιούν εργαλεία, όπως τα υπολογιστικά φύλλα, για να μπορέσουν να χτίσουν και να διατηρήσουν αντικείμενα του έργου, τα οποία δεν έχουν αναλυθεί ακόμα. Επίσης, υπάρχουν ορισμένα εργαλεία ανοιχτού κώδικα και ιδιόκτητα πακέτα, τα οποία είναι αφιερωμένα στη διαχείριση των προϊόντων στο πλαίσιο της συγκεκριμένης διαδικασίας. Από την άλλη πλευρά, υπάρχουν και οργανισμοί οι οποίοι εφαρμόζουν τη μεθοδολογία χωρίς να χρησιμοποιούν εργαλεία, ενώ ταυτόχρονα διατηρούν τα αντικείμενα του έργου σε έντυπη μορφή (χαρτιά, πίνακες, σημειώσεις).

ΣΤΑΔΙΑ ΑΝΑΠΤΥΞΗΣ

Το υπόβαθρο της μεθοδολογία SCRUM είναι η σταδιακή ανάπτυξη μιας εφαρμογής λογισμικού, διατηρώντας παράλληλα έναν διαφανή κατάλογο των νέων απαιτήσεων αλλά και των διορθώσεων που προκύπτουν και πρέπει να εφαρμοστούν. Πρόκειται για τακτικές παραδόσεις εκτελέσιμου κώδικα, περίπου κάθε τέσσερις εβδομάδες, όπου ο πελάτης/τελικός χρήστης λαμβάνει μια τέλεια λειτουργικά εφαρμογή, όπου κάθε νέο εκτελέσιμο τμήμα ,που παραδίδεται, περιλαμβάνει όλο και περισσότερες ανάγκες/απαιτήσεις του που έχουν καλυφθεί.

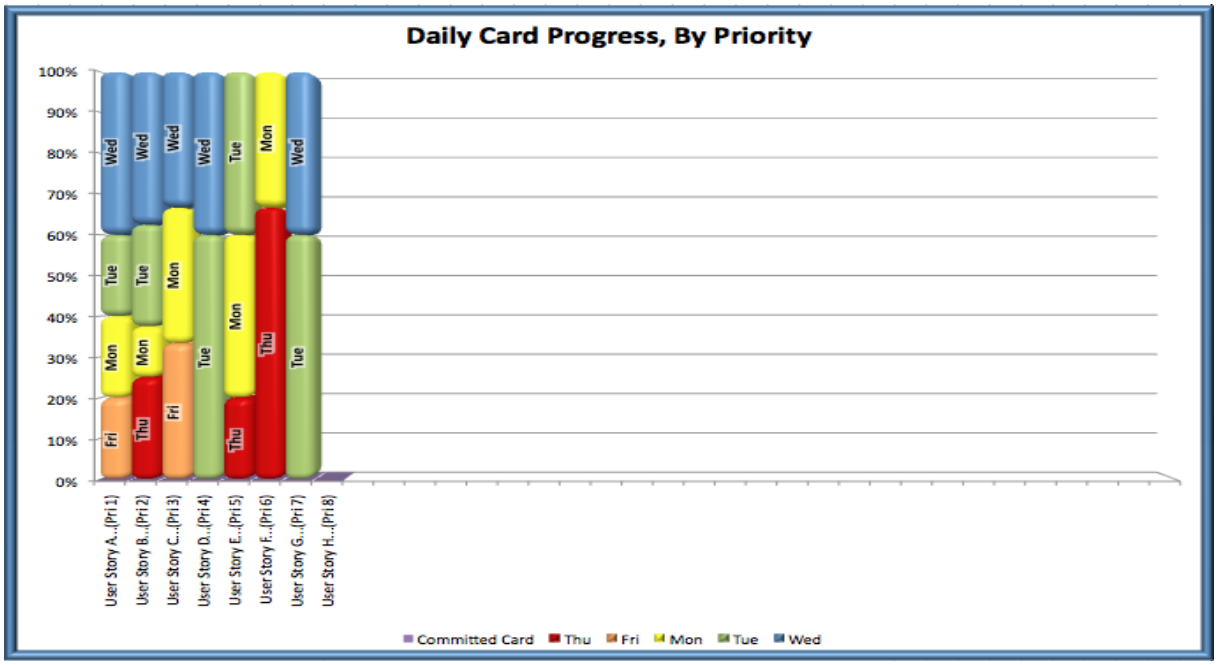
Οι αναβαθμίσεις του προγράμματος μπορούν να ενσωματωθούν πιο εύκολα με την χρήση του Μοντέλο V - V-model .

Για την επίτευξη της σωστής ανάπτυξης του έργου, η μεθοδολογία περιλαμβάνει μια σειρά από συναντήσεις.

1. ***Καθημερινή Συνάντηση.*** Κάθε μέρα κατά την διάρκεια της φάσης ανάπτυξης πραγματοποιούνται συναντήσεις, όπου εξετάζεται το στάδιο προόδου του έργου. Αυτό αποκαλείται daily SCRUM ή daily standup. Σε αυτήν τη συνάντηση καθορίζονται οι κατευθυντήριες γραμμές ανάπτυξης που πρέπει να ακολουθηθούν από την ομάδα ανάπτυξης. Μία καθημερινή συνάντηση έχει διάρκεια το πολύ 15 λεπτά και όλα τα μέλη της ομάδας ανάπτυξης έρχονται προετοιμασμένα για τις αλλαγές/ενημερώσεις, που έχουν πραγματοποιήσει. Ακόμη, και αν κάποια από τα μέλη της ομάδας λείπουν, η συνάντηση θα πραγματοποιηθεί. Επίσης, η ώρα και η τοποθεσία των συναντήσεων πρέπει να είναι σταθερή, και να μην αλλάζει. Στη συνάντηση μπορεί να προσέλθει ο καθένας αλλά τον λόγο μπορούν να έχουν μόνο τα μέλη του πυρήνα. Κατά την διάρκειά της κάθε μέλος απαντά σε τρία βασικά ερωτήματα :
   - «Τί έχετε κάνει από εχθές;»
   - «Τι σκοπεύετε να κάνετε σήμερα;»
   - «Υπάρχουν εμπόδια;»

   Κάθε εμπόδιο το οποίο εντοπίζεται σε μια συνάντηση πρέπει να τεκμηριωθεί/αναλυθεί από τον SCRUM Master και με το πέρας της συνάντησης ο ίδιος να εργαστεί για την επίλυσή του. Πρέπει να τονιστεί πως κατά την διάρκεια της συγκεκριμένης συνάντησης δεν πρέπει να πραγματοποιούνται εκτενείς συζητήσεις για κάθε θέμα, που προκύπτει.
2. ***Σύντομη αξιολόγηση ανεκτέλεστου προϊόντος.*** Η ομάδα ανάπτυξης πρέπει κατά την διάρκεια μιας επανάληψης να κάνει αξιολόγηση του ανεκτέλεστου προϊόντος. Πρόκειται για μια διαδικασία εκτίμησης των εκκρεμοτήτων που υπάρχουν, με σκοπό τη βελτιστοποίηση των κριτηρίων αποδοχής των επιμέρους εργασιών. Σε περίπτωση που η έκταση των εκκρεμοτήτων αφορά μεγάλα τμήματα, τότε αυτά πρέπει να χωριστούν σε μικρότερα. Αυτές οι

συναντήσεις δεν επιτρέπεται να έχουν διάρκεια παραπάνω από μια ώρα, ενώ ταυτόχρονα δεν πρέπει να περιλαμβάνουν τον διαχωρισμό των εργασιών σε μικρότερα τμήματα. Η ομάδα είναι αυτή που αποφασίζει των αριθμό των συναντήσεων που πρέπει να πραγματοποιηθούν μέσα στην εβδομάδα και για τον καλύτερο υπολογισμό τους χρησιμοποιείται μια μέθοδος που λέγεται σχεδιασμός πόκερ (planning poker)



Εικόνα e Ημερήσιο ραβδόγραμμα με την πρόοδο του έργου με βάση την προτεραιότητα της κάθε εργασίας.

3. ***SCRUM of SCRUMS.*** Πραγματοποιείται κάθε μέρα μετά την καθιερωμένη καθημερινή συνάντηση. Σκοπός αυτών των συναντήσεων είναι να δοθεί έμφαση στους τομείς επικάλυψης και ολοκλήρωσης. Για αυτόν το λόγο κάθε ομάδα ορίζει έναν «αντιπρόσωπο», που παρακολουθεί την πρόοδο τους. Η ημερήσια αυτή ατζέντα είναι η ίδια με αυτή της ημερήσιας συνάντησης και σκοπό έχει να δώσει απάντηση στα ακόλουθα ερωτήματα :

- «Τι έχει κάνει η ομάδα από την τελευταία συνάντηση μέχρι τώρα;»
- «Τι πρέπει να έχει κάνει η ομάδα μέχρι την επόμενη συνάντηση;»

- «Υπάρχει κάτι που να επιβραδύνει την ομάδα ή που να την εμποδίζει να συνεχίσει την ανάπτυξη του έργου;»
- «Υπάρχει κάτι έτοιμο το οποίο να μπορέσει να τροφοδοτήσει μια άλλη ομάδα με εργασία;»

4. ***Συνάντηση φάσης σχεδιασμού.*** Η συγκεκριμένη συνάντηση πραγματοποιείται στην αρχή κάθε κύκλου επανάληψης (περίπου κάθε εφτά με τριάντα ημέρες). Σε αυτήν τη συνάντηση επιλέγεται ποια εργασία θα πραγματοποιηθεί και προετοιμάζεται η φάση του ανεκτέλεστου σχεδιασμού, καθώς και τα απαραίτητα στοιχεία για την εκπόνηση της, όπως ο χρόνος που απαιτείται για την ολοκλήρωση του. Επιπλέον, προσδιορίζεται το ποσοστό του έργου, το οποίο μπορεί να ολοκληρωθεί κατά την διάρκεια της τρέχουσας φάσης. Για την ολοκλήρωση της συγκεκριμένης φάσης δίνεται μια προθεσμία οκτώ ωρών, όπου κατά τις τέσσερις πρώτες ώρες η ομάδα συζητά, για να βρει και να ιεραρχήσει το ανεκτέλεστο τμήμα του προϊόντος, ενώ τις επόμενες τέσσερις η ομάδα ανάπτυξης κατακερματίζει το ανεκτέλεστο τμήμα του έργου, που έχει αναλάβει. Στο τέλος αυτής της συνάντησης, βοηθούν και δύο άλλες συναντήσεις, η συνάντηση αναθεώρησης της φάσης και η αναδρομική συνάντηση.

5. ***Συνάντηση αναθεώρησης φάσης.*** Στη συνάντηση αυτή εξετάζεται κατά πόσο έχει ολοκληρωθεί ή όχι ένα κομμάτι του έργου. Αν έχει ολοκληρωθεί, τότε αυτό παρουσιάζεται στους ενδιαφερόμενους (αυτό το κομμάτι είναι γνωστό στο ευρύ κοινό ως demo). Τα τμήματα που δεν έχουν ολοκληρωθεί ακόμα, δεν παρουσιάζονται. Για την εκπόνηση της συγκεκριμένης συνάντησης δίνεται ένα περιθώριο τεσσάρων ωρών.

6. ***Αναδρομική συνάντηση.*** Τα μέλη της βρίσκονται και συζητούν σχετικά με προβλήματα, τα οποία παρουσιάστηκαν σε προηγούμενες φάσεις του έργου. Σκοπός αυτών των συναντήσεων είναι η βελτίωση των διεργασιών, όπου τίθενται δύο ερωτήσεις «Τι δεν πήγε καλά κατά τη διάρκεια της συγκεκριμένης φάσης;», «Τι είναι αυτό το οποίο θα μπορούσαμε να βελτιώσουμε στην επόμενη φάση;». Η συνάντηση αυτή έχει διάρκεια το πολύ τέσσερις ώρες και στην διεξαγωγή της βοηθάει ο SCRUM Master.

ΑΚΥΡΩΣΗ ΦΑΣΗΣ

Ο ιδιοκτήτης του προϊόντος μπορεί να κρίνει σκόπιμο, πως μια φάση ανάπτυξης του πρέπει να ακυρωθεί, πριν παραδοθεί. Μόνο ο ιδιοκτήτης του προϊόντος έχει την δυνατότητα να ακυρώσει μια φάση, μπορεί να το πράξει υπό την επιρροή που του ασκούν τα ενδιαφερόμενα μέρη, η Ομάδα Ανάπτυξης ή ο SCRUM Master.

Η απόφαση για να ακυρωθεί μια φάση θα παρθεί, αν ο στόχος της συγκεκριμένης φάσης καθίσταται άνευ αντικειμένου. Αυτό μπορεί να συμβεί τόσο αν υπάρξουν αλλαγές κατεύθυνσης της εταιρείας, όσο και αν διαφοροποιηθούν οι υπάρχουσες συνθήκες αγοράς αλλά ακόμη, και αν παρουσιαστεί αλλαγή της τεχνολογίας που χρησιμοποιείται. Κρίνεται σκόπιμο μία φάση να ακυρωθεί, όταν οι απαιτήσεις που θα αναπτύσσονταν σε αυτή, κριθούν μη χρήσιμες για την λειτουργία του. Όμως, λόγω της σύντομης διάρκειας που έχει η κάθε φάση ανάπτυξης, μια ακύρωση σπάνια γίνεται αισθητή από τα εμπλεκόμενα μέλη.

Όταν μία φάση ακυρωθεί, είναι απαραίτητο να αναθεωρηθεί κάθε ολοκληρωμένο και "Έτοιμο" Τμήμα Ανεκτέλεστου προϊόντος. Σε περίπτωση που ένα μέρος του έργου χρειαστεί να αφαιρεθεί, ο ιδιοκτήτης του προϊόντος πρέπει να αποδεχτεί τις αλλαγές που έγιναν. Το υπόλοιπο Ανεκτέλεστο προϊόντων επανεξετάζεται, για να επανέλθει στην λίστα αναθεωρημένων Ανεκτέλεστων προϊόντων περιλαμβάνοντας τις αλλαγές που σημειώθηκαν. Οι εργασίες, που πραγματοποιήθηκαν μέχρι εκείνη την στιγμή, υποβιβάζονται, για αυτό πρέπει να υπολογίζεται εκ νέου ο χρόνος ολοκλήρωσης του έργου.

Η ακύρωση μιας φάσης καταναλώνει τους διαθέσιμους πόρους, αφού όλοι οι υπόλοιποι πόροι έχουν χρησιμοποιηθεί ήδη σε μία άλλη φάση-δομή. Ο σχεδιασμός των συνεδριάσεων πρέπει να ξεκινήσει έχοντας νέα δομή. Φάσεις που έχουν ακυρωθεί μπορεί να δημιουργήσουν προβλήματα στην ομάδα της Scrum ,για την οποία είναι κάτι πολύ ασυνήθιστο , καθιστώντας ίσως την συνέχιση της ανάπτυξης του έργου δύσκολη.

ΑΝΤΙΚΕΙΜΕΝΑ

ΑΝΕΚΤΕΛΕΣΤΟ ΠΡΟΪΟΝ

Όταν αναφέρεται κάποιος στο ανεκτέλεστο υπόλοιπο προϊόν, αναφέρεται σε μία λίστα η οποία περιέχει τις απαιτήσεις του πελάτη για ορθή λειτουργία του προϊόντος, που επιθυμεί να λάβει. Περιέχει στοιχεία, τα οποία έχει εντοπίσει ο ιδιοκτήτης με βάση κάποιες εκτιμήσεις κινδύνων, όπως η επιχειρηματική αξία, οι εξαρτήσεις του προϊόντος από άλλα, ημερομηνίες οι οποίες και έχουν σημασία για την σωστή λειτουργία του κ.α.. Τα χαρακτηριστικά αυτά προσθέτονται στην λίστα των απαιτήσεων του ανεκτέλεστου προϊόντος με την μορφή μίας ιστορίας. Αυτό το ανεκτέλεστο προϊόν είναι το «τι» πρόκειται να «χτιστεί», και το οποίο ταξινομείται,\ σύμφωνα με μία λίστα, που περιέχει όλα αυτά τα που πρέπει να προστεθούν. Η λίστα αυτή είναι προσβάσιμη από οποιοδήποτε, αλλά ο ιδιοκτήτης είναι ο τελικός υπεύθυνος για προσθήκη των απαιτήσεων/ιστοριών στην λίστα του ανεκτέλεστου προϊόντος για την ομάδα ανάπτυξης. Το ανεκτέλεστο προϊόν περιέχει κατά προσέγγιση εκτιμήσεις για την επιχειρηματική αξία και την προσπάθεια ανάπτυξης του. Αυτές οι αξίες αναφέρονται στις διάφορες απαιτήσεις/ιστορίες με την χρήση μια ολοκληρωμένης ακολουθίας Fibonacci. Οι εκτιμήσεις αυτές μπορούν να βοηθήσουν τον ιδιοκτήτη να κάνει μια πιο ολοκληρωμένη εκτίμηση για το χρονοδιάγραμμα ολοκλήρωσης του έργου αλλά μπορεί να τον βοηθήσει να επηρεάσει και την προσθήκη νέων απαιτήσεων. Για παράδειγμα, αν η προσθήκη «ορθογραφικού έλεγχου» και η «υποστήριξη πίνακα» προσδίδουν στο έργο την ίδια επιχειρηματική αξία, αυτό το οποίο χρειάζεται μικρότερη προσπάθεια ανάπτυξης πιθανότατα να έχει και υψηλότερη προτεραιότητα στην λίστα απαιτήσεων, γιατί και η ROI (Return On Investment – Απόδοση της Επένδυσης) είναι υψηλότερη.

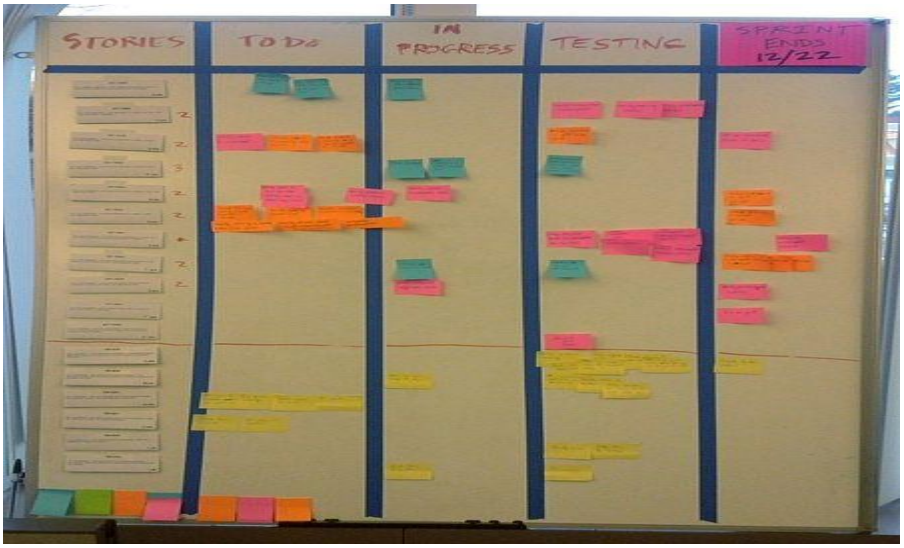
Το υπόλοιπο ανεκτέλεστο προϊόν, και η επιχειρηματική του αξία του κάθε στοιχείου που περιέχει είναι ευθύνη του ιδιοκτήτη του προϊόντος. Η εκτιμώμενη προσπάθεια για την ολοκλήρωση κάθε υπόλοιπου ανεκτέλεστου τμήματος καθορίζεται από την ομάδα ανάπτυξης.

ΑΝΕΚΤΕΛΕΣΤΗ ΦΑΣΗ

Η ανεκτέλεστη φάση είναι μια λίστα με τις εργασίες, τις οποίες πρέπει να αντιμετωπίσει η ομάδα ανάπτυξης κατά την διάρκεια της επόμενης φάσης ανάπτυξης. Ο κατάλογος αυτός περιλαμβάνει εργασίες για την ομάδα ανάπτυξης που έχουν επιλεχθεί από την κορυφή του πίνακα με τα ανεκτέλεστα προϊόντα. Επιλέγεται τέτοιος αριθμός χαρακτηριστικών/απαιτήσεων, ώστε προσδιοριστεί η εργασία της ομάδας ανάπτυξης για την επόμενη φάση. Η επιλογή των χαρακτηριστικών/απαιτήσεων γίνεται από την ομάδα ανάπτυξης απαντώντας στο ερώτημα «Τι μπορούμε να κάνουμε εμείς για αυτό;» προσθέτοντας χαρακτηριστικά/ιστορίες στην συγκεκριμένη φάση. Η ομάδα ανάπτυξης πρέπει να έχει κατά νου την διάρκεια των προηγούμενων φάσεων (συνολικά σημεία χαρακτηριστικών τα οποία έχουν ολοκληρωθεί στις τελευταίες φάσεις), και όταν επιλέγει χαρακτηριστικά/ιστορίες για μια νέα φάση ανεκτέλεστου προϊόντος να την ακολουθήσει ως κατευθυντήρια γραμμή για την προσπάθεια ολοκλήρωσης της.

Τα χαρακτηριστικά/ιστορίες αναλύονται εκ νέου σε εργασίες από τις ομάδες ανάπτυξης, ως μια βέλτιστη πρακτική, των οποίων η ολοκλήρωση της ανάλυσης πρέπει να είναι μεταξύ τεσσάρων και δεκαέξι ωρών εργασίας. Μετά από αυτό, η ομάδα ανάπτυξης έχει πλέον κατανοήσει τις λεπτομέρειες των απαιτήσεων, και αφού γνωρίζει τι πρέπει να κάνουν, το κάθε μέλος τη ομάδας μπορεί να πάρει από την λίστα με τις εργασίες μία, για να την εκτελέσει. Οι εργασίες για την φάση των ανεκτέλεστων τμημάτων δεν έχουν καταχωρηθεί, και πιθανότατα οι εργασίες να υπογραφούν από τα μέλη της ομάδας κατά την διάρκεια της ημερήσιας συνάντησης, πάντοτε τηρώντας την προτεραιότητα των δεξιοτήτων των μελών της.

Το ανεκτέλεστο υπόλοιπο είναι ιδιοκτησία της ομάδας ανάπτυξης και περιλαμβάνει όλες τις εκτιμήσεις, που παρέχονται από την ομάδα ανάπτυξης. Συχνά, ένα επιπλέον διοικητικό συμβούλιο των μελών ανάπτυξης του έργου, χρησιμοποιείται για να δει και να αλλάξει την κατάσταση των εργασιών της τρέχουσας φάσης, για παράδειγμα «για να γίνει/να προγραμματιστεί», «σε εξέλιξη» και «έχει γίνει/ολοκληρωθεί».



Εικόνα f Πίνακας προόδου της SCRUM με χρήση σημειώσεων.

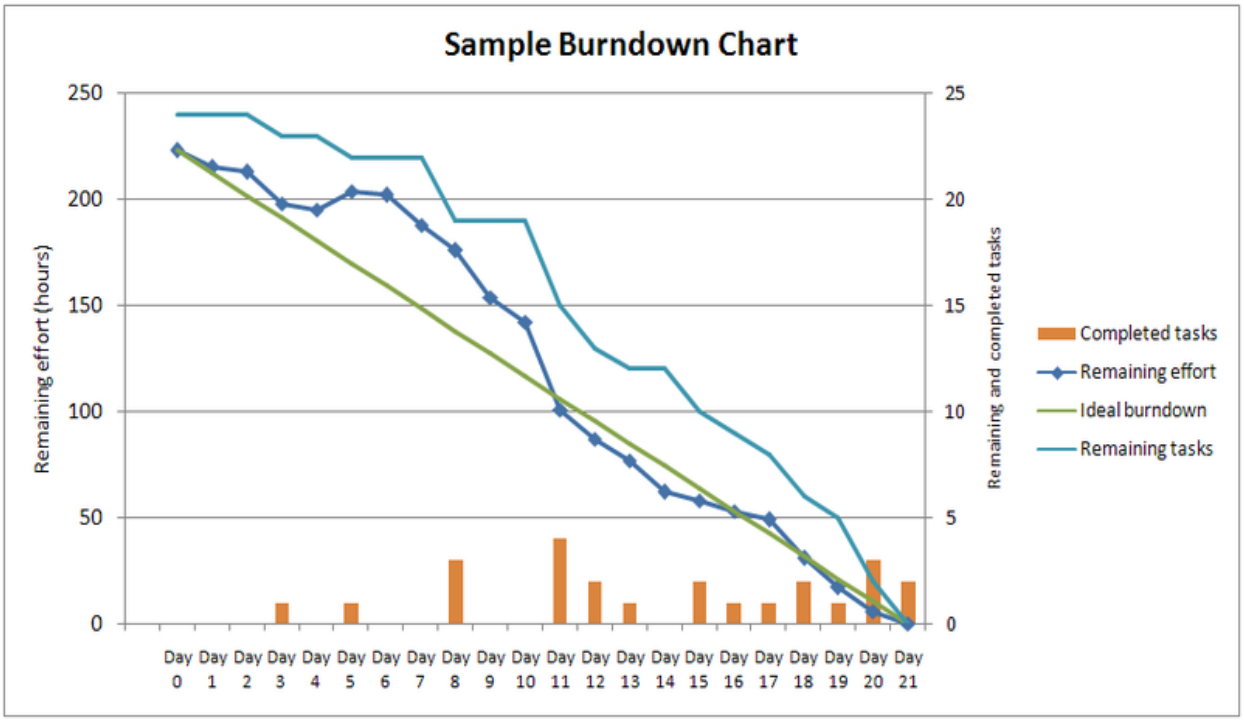
ΠΡΟΣΑΥΞΗΣΗ

Η προσαύξηση αναφέρεται στο άθροισμα των ανεκτέλεστων τμημάτων του προϊόντος που μπορεί να έχουν ολοκληρωθεί κατά την διάρκεια μιας επαναληπτικής φάσης αλλά και από όλες τις προηγούμενες. Με το πέρας κάθε επαναληπτικής φάσης, η προσαύξηση γίνεται με όλα εκείνα τα ανεκτέλεστα τμήματα, τα οποία υποστηρίζει η ομάδα σχεδιασμού πως έχουν ολοκληρωθεί. Για αυτό τον λόγο θα πρέπει να είναι σε καλή κατάσταση λειτουργίας, ανεξάρτητα από το αν ο ιδιοκτήτης του προϊόντος αποφασίσει, αν θα «χρησιμοποιήσει» τελικά ή όχι την συγκεκριμένη απαίτηση.

ΤΟ ΚΑΘΟΔΙΚΟ ΔΙΑΓΡΑΜΜΑ – BURN DOWN CHART

Σε κάθε επαναληπτική φάση δημιουργείται ένα διάγραμμα το οποίο παρουσιάζει το υπόλοιπο από το ανεκτέλεστο προϊόν. Το διάγραμμα αυτό ενημερώνεται κάθε μέρα και αποτελεί μια απλή προβολή της προόδου ανάπτυξης του έργου. Επιπρόσθετα, παρέχει την δυνατότητα μιας άμεσης απεικόνισης μιας αναφοράς. Υπάρχουν ορισμένοι ακόμη τύποι διαγραμμάτων, όπως το διάγραμμα με όνομα «η απελευθέρωση του καθοδικού διαγράμματος – the release burn down chart», που παρουσιάζει το μέγεθος/την ποσότητα εργασίας που πραγματοποιήθηκε, και το σχεδιάγραμμα με τίτλο «η εναλλακτική απελευθέρωση του καθοδικού διαγράμματος – alternative release burn down chart» το οποίο φαίνεται να είναι το ίδιο αλλά στην πραγματικότητα παρουσιάζει ξεκάθαρα όλες

εκείνες τις αλλαγές, που έγιναν σε επίπεδο εφαρμογής, για να «αφήσει το περιεχόμενο» του να επανέλθει στην αρχική του κατάσταση.



Εικόνα g Απλό καθοδικό διάγραμμα της μεθοδολογίας

Η ΜΕΘΟΔΟΛΟΓΙΑ SCRUM-BAN

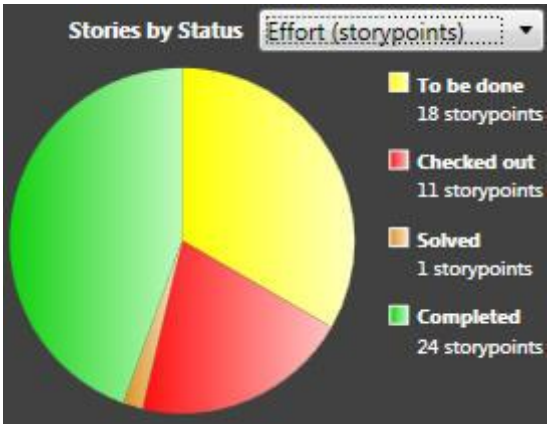
Η μεθοδολογία SCRUM-ban είναι ένα μοντέλο παραγωγής λογισμικού που έχει ως βάση την μεθοδολογία SCRUM και Kanban. Οι απαγορεύσεις SCRUM ενδείκνυνται για εργασίες συντήρησης προϊόντων ή για έργα με συχνές και απρόσμενες ιστορίες χρηστών ή ακόμη για λάθη προγραμματισμού. Σε τέτοιες περιπτώσεις, οι χρονικά περιορισμένες φάσεις του μοντέλου της SCRUM δεν έχουν καμία αξιολόγηση από τον χρήστη, αλλά μπορούν να εφαρμοστούν στις ημερήσιες συναντήσεις άλλες πρακτικές, ανάλογα με την ομάδα και την κατάσταση της. Η οπτικοποίηση των σταδίων εργασίας και οι περιορισμοί για τις ταυτόχρονες ημιτελείς ιστορίες και ελαττώματα είναι γνωστά από το μοντέλο του Kanban. Κάνοντας χρήση των συγκεκριμένων μεθόδων, η ροή των εργασιών της ομάδας κατευθύνεται με τέτοιο τρόπο, ώστε να τους επιτρέπεται να ελαχιστοποιήσουν τον χρόνο παράδοσης του έργου, για κάθε ιστορία χρήστη ή σφάλματος προγραμματισμού και αυτό γίνεται αφού πρώτα εξασφαλιστεί πως η ομάδα θα ασχολείται με αυτά συνεχώς.

Η συγκεκριμένη μεθοδολογία, για να δείξει το κάθε στάδιο της εργασίας, υποδεικνύει στις ομάδες που εργάζονται στον ίδιο χώρο να χρησιμοποιούν σημειώματα ή έναν μεγάλο πίνακα, όπου αναφέρονται τα τμήματα του έργου, που έχουν ολοκληρωθεί. Στην περίπτωση των αποκεντρωμένων ομάδων, όπως Assembla, ScrumWorks, Rational Team Concert ή JIRA σε συνδυασμό με την GreenHopper, τα στάδια εργασίας μπορούν να χρησιμοποιηθούν, για να απεικονιστούν οι ιστορίες των χρηστών, τα ελαττώματα και τα καθήκοντα της κάθε ομάδας και αυτά χωρίζονται σε διακριτές φάσεις.

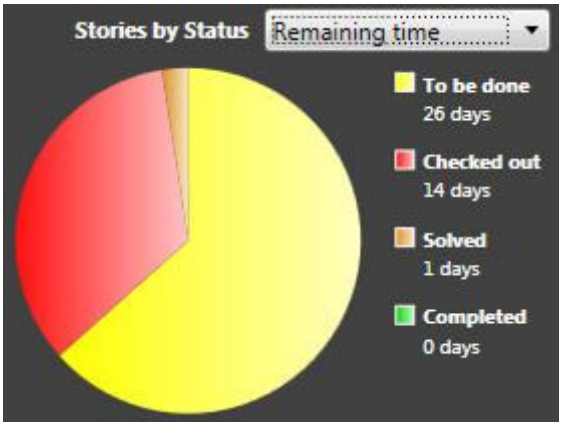
Τα στάδια στην πιο απλή τους μορφή χωρίζονται στις εξής κατηγορίες :

- «Δεν έχει ξεκινήσει»
- «Συνεχίζεται»
- «Ολοκληρώθηκε»

Ωστόσο, αν η ομάδα το επιθυμεί μπορεί να προσθέσει στάδια εργασίας, όπως «Ορίζεται», «Σχεδιάστηκε», «Δοκιμάστηκε» ή «Παραδόθηκε». Αυτές οι πρόσθετες φάσεις μπορούν να βοηθήσουν την ομάδα να κατανοήσει καλύτερα το ανεκτέλεστο έργο, ειδικότερα, αν η ομάδα δυσκολεύεται να το αναπτύξει και αν οι οριακές τιμές του έργου είναι χαμηλές. Μια πιο συγκεκριμένη κατανομή καθηκόντων στα μέλη της ομάδας τους δίνει την ευκαιρία να ειδικευτούν σε μια πιο συγκεκριμένη φάση.



Εικόνα h Διάγραμμα τύπου "Πίτα" που απεικονίζει τα σημεία που έχουν ολοκληρωθεί.



Εικόνα i Διάγραμμα τύπου "Πίτα" που απεικονίζει τα σημεία που απομένουν για την ολοκλήρωση του έργου.

Δεν υπάρχουν κάποια συγκεκριμένα όρια για τις οριακές τιμές του ημιτελούς έργου. Αντ' αυτού, κάθε ομάδα πρέπει να τις καθορίσει ξεχωριστά, για μία δοκιμή και για ένα λάθος. Μία τιμή αποτελέσματος είναι πολύ μικρή από εργαζόμενους σε αδράνεια, ενώ πολύ υψηλή όταν οι εργαζόμενοι συσσωρεύουν πολλές ημιτελείς εργασίες ταυτόχρονα, το οποίο με την σειρά του οδηγεί στην ολοκλήρωση της φάσης. Ένας εμπειρικός κανόνας λέει πως ένα μέλος μίας ομάδας δεν πρέπει να έχει ταυτόχρονα περισσότερες από δύο επιλεγμένες εργασίες, καθώς και πως δεν θα πρέπει όλα τα μέλη της ομάδας να έχουν ταυτόχρονα δύο εργασίες.

Οι διαφορές της SCRUM με Kanban είναι πως στη πρώτη το έργο χωρίζεται σε φάσεις, που έχουν μια ορισμένη χρονική διάρκεια, ενώ στην δεύτερη η ροή ανάπτυξης είναι συνεχής. Αυτό μπορεί να γίνει αντιληπτό από τους πίνακες απεικόνισης των σταδίων εργασίας, οπού στην SCRUM με το τέλος μίας φάσης αδειάζουν. Στην Kanban όλες οι εργασίες σημειώνονται στον ίδιο πίνακα. Η SCRUM επίσης, επιδιώκει οι ομάδες της να έχουν πολύπλευρη τεχνογνωσία, ενώ η Kanban κάνει όσο το δυνατόν πιο εξειδικευμένες και λειτουργικές ομάδες.

Από τότε που η SCRUM-Ban είναι ένα νέο μοντέλο ανάπτυξης, δεν υπάρχει υλικό αναφοράς για την χρήση της. Από την άλλη η Kanban εφαρμόζεται από την Microsoft και την Corbis.

ΠΛΕΟΝΕΚΤΗΜΑΤΑ

Η SCRUM διαφέρει από άλλες μεθόδους ανάπτυξης. Αυτό συμβαίνει γιατί με τα πλεονεκτήματα τα οποία διαθέτει, μετατρέπει σε πιο ρεαλιστική μορφή τις σημερινές ανάγκες των ιδιοκτητών προϊόν. Έτσι τα πλεονεκτήματα της είναι :

- Επαναληπτική και αυξητική μέθοδος: η μέθοδος αυτή επιτρέπει να αποφευχθεί το "φαινόμενο σήραγγας", δηλαδή, το γεγονός ότι ο ιδιοκτήτης του προϊόντος βλέπει μόνο το αποτέλεσμα κατά την τελική παράδοση του, και τίποτα ή σχεδόν τίποτα κατά τη διάρκεια όλης της φάσης ανάπτυξης. Η μέθοδος επιτυγχάνεται με χρήση του μοντέλου V-κύκλο εξελίξεις.
- Μέγιστη προσαρμοστικότητα για την ανάπτυξη προϊόντων και της εφαρμογής: η διαδοχική σύνθεση του περιεχομένου της φάσης επιτρέπει να προσθέσετε μια τροποποίηση ή ένα χαρακτηριστικό, το οποίο δεν είχε αρχικά προγραμματιστεί. Αυτό είναι ακριβώς αυτό που καθιστά αυτήν την μέθοδο "ευέλικτη".
- Συμμετοχική μέθοδος: κάθε μέλος της ομάδας καλείται να εκφράσει τις απόψεις του και μπορεί να συμβάλει σε όλες τις αποφάσεις, που ελήφθησαν σχετικά με το έργο. Ως εκ τούτου, μεγαλύτερη συμμετοχή των μελών της ομάδας ανάπτυξης άρα περισσότερα κίνητρα για εργασία.
- Ενίσχυση της επικοινωνίας: δουλεύοντας στο ίδιο δωμάτιο ή με ανάπτυξη που συνδέεται με διάφορα μέσα επικοινωνίας. Τα μέλη της ομάδας μπορούν εύκολα να επικοινωνούν και να ανταλλάσσουν απόψεις σχετικά με τα εμπόδια τα οποία συναντούν, προκειμένου να εξαλειφθούν όσο το δυνατόν νωρίτερα.
- Μεγιστοποίηση της συνεργασίας: καθημερινή επικοινωνία μεταξύ του πελάτη και της ομάδας Pentalog τους επιτρέπει να συνεργάζονται πιο στενά.
- Αύξηση της παραγωγικότητας: διότι αφαιρεί ορισμένους "περιορισμούς" από τις κλασσικές μεθόδους, όπως η τεκμηρίωση ή η υπερβολική τυποποίηση. Η SCRUM επιτρέπει η παραγωγικότητα της ομάδας να αυξηθεί

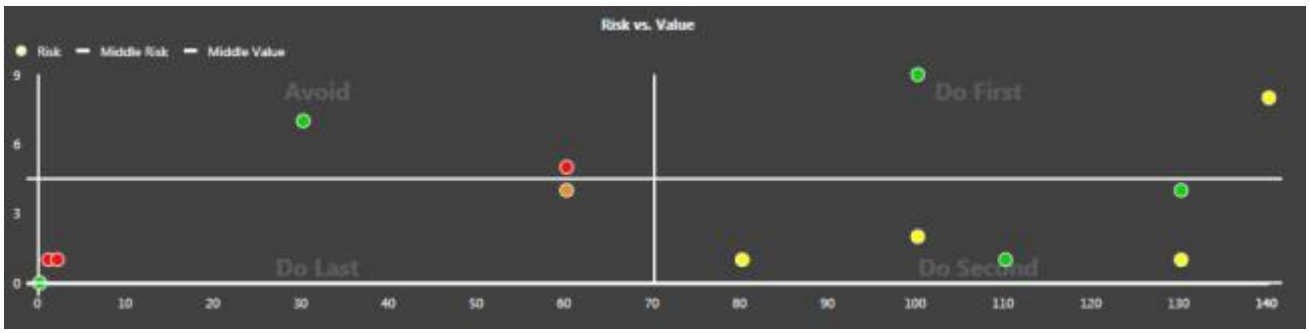
ΚΙΝΔΥΝΟΙ ΚΑΙ ΛΥΣΕΙΣ

Η μεθοδολογία δεν προσφέρει μια καθολική απάντηση σε όλα τα προβλήματα που είναι συνυφασμένα με την ανάπτυξη λογισμικού. Οι ομάδες πρέπει να δώσουν προσοχή στους κινδύνους παρακάτω:

- Μέγεθος ομάδας: το μέγεθος της ομάδας συνήθως περιορίζεται σε 7 ή 10 άτομα και μπορεί να γίνει ένα εμπόδιο, αν αυτή υπερβαίνει τις συστάσεις

αυτές. Στην τελευταία περίπτωση, η διοργάνωση συναντήσεων καθίσταται αδύνατη και επηρεάζονται τα ίδια τα θεμέλια της μεθόδου. Η λύση είναι να δημιουργήσει ένα Scrum of Scrums. Αυτό συνίσταται στην διαίρεση του έργου σε ομάδες των κατάλληλων μεγεθών και προσθέτοντας έναν Scrum Master από ένα υψηλότερο επίπεδο το οποίο συγκεντρώνει τους Scrum Masters κάθε Scrum.

- Πολλαπλές αιτήσεις: Αιτήσεις μπορούν να μεταδοθούν μέσω διαφόρων διαύλων για ένα έργο. Μάλιστα μερικές φορές μπορεί να είναι δύσκολο να διαχειριστούν όλες οι αιτήσεις εξαιτίας των αντιφατικών πτυχών τους. Αυτές οι αντιφάσεις μπορεί να επιβραδύνουν τη διαδικασία επικύρωσης παράδοσης. Για να λυθεί αυτό το πρόβλημα, που είναι ζωτικής σημασίας, χρησιμοποιείται ένα εργαλείο διαχείρισης των αιτήσεων, το οποίο είναι μια βασική επιλογή με Pentalog έργου.
- Ποιοτική ανάπτυξη: Όσο περισσότερο αυξάνεται ο αριθμός των ομάδων, τόσο πιο δύσκολη είναι να ασχοληθεί με την ποιότητα. Ο κανόνας αυτός ισχύει κατά μείζονα λόγο, όταν το έργο κατανέμεται μεταξύ πολλών κέντρων. Οι κυριότεροι κίνδυνοι σχετίζονται με την ποιότητα κώδικα και τον αριθμό των σφαλμάτων που εντοπίστηκαν κατά την ένταξη ενός τμήματος κώδικα στο υπόλοιπο έργο. Γι 'αυτό, είναι σημαντικό να έχουμε μια αυστηρή πολιτική ποιότητας και ένα σχέδιο για την ποιότητα του έργου που ορίζει επακριβώς τους κανόνες του έργου. Οι συχνοί έλεγχοι κώδικα και η εφαρμογή των δεικτών που μετρούν την απόδοση των προγραμματιστών, χρησιμοποιούνται για να επιτρέψουν την ελαχιστοποίηση του κινδύνου αυτού.



Εικόνα j Διάγραμμα παρουσίασης αξία και ρίσκου ανάπτυξης μίας απαίτησης.

ΕΠΙΛΟΓΟΣ

Η SCRUM είναι μια ευέλικτη μεθοδολογία σχεδιασμού. Σκοπός της ανάπτυξης της είναι να επιταχύνει την ανάπτυξη ενός έργου μέσα από την χρήση μεθόδων και πρακτικών, όπου ο κάθε συμμετέχοντας θα έχει ένα ρόλο και ένα σύνολο δραστηριοτήτων και ενεργειών, τα οποία θα πρέπει να ολοκληρώσει σε συγκεκριμένο χρονικό διάστημα.

Για να είναι επιτυχής η ανάπτυξη του έργου, όλα γίνονται υπό την καθοδήγηση ενός συγκεκριμένου ανθρώπου. Αυτός καθημερινά δίνει στον καθένα τις εργασίες, τις οποίες θα πρέπει να φέρει σε πέρας. Μέσα από τις συναντήσεις μπορεί να ελέγχεται η πρόοδος του έργου και να δίνονται εκ νέου οδηγίες σε όλους, όσοι συμβάλλουν στην ανάπτυξη του σε περίπτωση που υπάρξουν αλλαγές στην επιχειρησιακή λογική.

ΚΕΦΑΛΑΙΟ 4

ΜΕΘΟΔΟΛΟΓΙΑ ΣΧΕΔΙΑΣΜΟΥ SCRUM ΚΑΙ ΜΕΤΡΙΚΕΣ

ΕΙΣΑΓΩΓΗ

(Στο κεφάλαιο που ακολουθεί χρησιμοποιούνται όροι των οποίων η έννοια αναλύεται και εξηγείται σε ειδικό Λεξιλόγιο όρων το οποίο έχει αναπτυχθεί και βρίσκεται στην σελίδα 69)

Σε αυτό το κεφάλαιο θα παρουσιαστεί η εφαρμογή ορισμένων μετρικών στην μεθοδολογία SCRUM, καθώς και τα συμπεράσματα που μπορεί να προκύψουν από τα εξαγόμενα αποτελέσματα της εφαρμογής τους στο έργο. Ωστόσο, πριν από την παρουσίασης τους κρίνεται σκόπιμο να δοθούν δύο ορισμοί για το πότε μία μετρική είναι «καλή» και πότε «κακή». Ουσιαστικά, πότε μία μέτρηση είναι καλή και πότε όχι. Επιπλέον, στο συγκεκριμένο κεφάλαιο γίνεται συχνή χρήση των φράσεων καλός δείκτης και κακός δείκτης. Οι δύο αυτές φράσεις αναφέρονται στα αποτελέσματα της εφαρμογής αλλά και στο ποσοστό σύμφωνα το οποίο αυτά τα αποτελέσματα είναι τα επιθυμητά ή όχι.

Μία μέτρηση λέγεται «καλή», όταν χρησιμοποιείται για να βοηθήσει την ομάδα να προσδιορίσει το σημείο στο οποίο βρίσκεται, το όποιο είναι από τα σημαντικότερα αποτελέσματα και συμπεράσματα που εξάγονται, ώστε να την χρησιμοποιήσει ως οδηγό προκειμένου να επιθεωρήσει και να προσαρμόσει τις διαδικασίες για βελτίωση στην διάρκεια του χρόνου.

Αντίθετα, μία μετρική είναι κακή, όταν χρησιμοποιείται ως μια άκαμπτη γραμμή στην άμμο, για την μικρο-διαχείριση της απόδοσης ενός μέλους της ομάδας στην πάροδο του χρόνου. Το κυριότερο είναι πως μία «κακή» μετρική μπορεί να επηρεάσει άμεσα το ανθρώπινο δυναμικό και να τους δημιουργήσει κακή ψυχολογία.

ΜΕΤΡΙΚΕΣ ΠΑΡΑΓΩΓΙΚΟΤΗΤΑΣ

ΧΡΟΝΟΣ ΚΥΚΛΟΥ ΕΠΑΝΑΛΗΨΗΣ/ΦΑΣΗΣ (In Sprint Cycle Time)

Αποτελεί την μέτρηση του κύκλου εργασιών των ιστοριών σε όλη την διάρκεια μιας φάσης, όπως μετράται σε μια ιστορία, από όταν ξεκίνησε μέχρι αυτή να ολοκληρωθεί. Ένας συντομότερος χρόνος κύκλου είναι ένας καλός δείκτης.

Εικόνα k Ραβδόγραμμα παρουσίασης του κύκλου ανάπτυξης μίας φάσης.

ΠΡΟΟΔΟΣ ΚΑΤΑ ΤΗΝ ΔΙΑΡΚΕΙΑ ΤΗΣ ΔΙΑΔΙΚΑΣΙΑΣ (In Sprint Work in Process)

Είναι μέτρηση της συνεχούς ροής εργασίας κατά τη διάρκεια της διαδικασίας ανάπτυξης μιας φάσης, όπου σχετίζεται με τη ροή (απόδοση) της ομάδας. Η περισσότερη δουλειά στη διαδικασία είναι μια κακή ένδειξη. Λιγότερη εργασία στη διεργασία είναι ένας καλός δείκτης. Αυτό προϋποθέτει πως λιγότερη εργασία στην συγκεκριμένη φάση, θα αποδώσει κώδικα υψηλότερου επιπέδου.

ΟΛΟΚΛΗΡΩΣΗ ΕΠΑΝΑΛΗΨΗΣ/ΦΑΣΗΣ
(Sprint Completion Bar.)

Πρόκειται για την δημιουργία ενός πίνακα με 4 στήλες, όπου με πράσινο σημειώνονται οι εργασίες που έχουν χαρακτηρισμό «Ολοκληρώθηκε», με κίτρινο όσες έχουν χαρακτηρισμό «Δεν δοκιμάστηκε», με πορτοκαλί εκείνες που έχουν χαρακτηρισμό «Δεν ολοκληρώθηκε» και τέλος με κόκκινο εκείνες που έχουν χαρακτηριστεί ως «Απόβλητα». Κάθε στοιχείο παρουσιάζεται σε ένα ξεχωριστό χρώμα. Η σωρευμένη ράβδος δείχνει μια φάση εργασίας πάνω σε μια άλλη φάση. Έγινε περισσότερο με την μορφή ράβδων, γιατί είναι ένας καλός παραστατικός δείκτης. Ένας μεγάλος δείκτη μπορεί να αποτελείται από τμήματα τα οποία έχουν χαρακτηριστεί «Δεν Δοκιμάστηκε» ή «Δεν Κωδικοποιήθηκε», γιατί η ολοκλήρωση τους είναι δύσκολο και συνιστούν μια κακή ένδειξη για την ομάδα.

ΟΛΟΚΛΗΡΩΘΗΚΕ/ΔΕΣΜΕΥΤΗΚΕ
(Completed vs. Commited (a.k.a. Earned Value))

Μετράει την πληρότητα του έργου έναντι της αρχικής εκτίμησης της φάσης και μετριέται σε ποσοστά. Χρησιμοποιείται για να δούμε ,εάν η ομάδα έχει καταφέρει να ολοκληρώσει με επιτυχία ή όχι τον στόχο της φάσης. Σε περίπτωση όπου η ομάδα έχει ολοκληρώσει περισσότερο από το 80% της διεργασίας που ανέλαβε, τότε πετυχαίνει τον στόχο της φάσης, ενώ στην περίπτωση που είναι λιγότερο αποτυγχάνει. Συχνά, γίνεται κατάχρηση διότι, αν μια ομάδα πιεστεί, θα έχει πτωτική τάση είτε για να είναι ασφαλής στη δέσμευσή της (π.χ. μείωση της παραγωγικότητας) είτε για να αποκρύψει ημιτελή τμήματα του έργου. Αυτή η μετρική μπορεί να χρησιμοποιηθεί αποτελεσματικά, αν ο όρος "αποτυχία" αφαιρείται και η ομάδα δεν ωθείται/πιέζεται προκειμένου να ολοκληρώσει ένα ορισμένο ποσοστό του έργου.

ΤΑΧΥΤΗΤΑ
(Velocity)

Η ταχύτητα μετριέται με τον αριθμό των σημείων του έργου τα οποία έχουν ολοκληρωθεί μέσα στην φάση. Εάν χρησιμοποιηθεί ως μέτρο παραγωγικότητας, αρχικά μπορεί να οδηγήσει στην αύξηση της παραγωγικότητα της αλλά στην πορεία να της ασκήσει πίεση και να μειώσει την παραγωγικότητα της. Η χρήση της

ταχύτητας δεν είναι ορθός τρόπος σύγκρισης των ομάδων. Θεωρείται μια μέτρηση προβλεπτικότητας.

ΠΡΟΣΩΠΙΚΗ ΑΞΙΟΛΟΓΗΣΗ ΑΠΟΔΟΣΗΣ (Individual Performance Reviews)

Οι πλειοψηφία των εταιρειών χρησιμοποιεί τις αξιολογήσεις των επιδόσεων για την μέτρηση της απόδοσης των ομάδων. Αυτό έχει ως αποτέλεσμα να οδηγούνται σε ατομικές, αντι-ομαδικές συμπεριφορές, οι οποίες έχουν αρνητική επιρροή στην παραγωγικότητα της ομάδας.

ΑΞΙΟΛΟΓΗΣΗ ΠΟΡΩΝ (Resource Utilization)

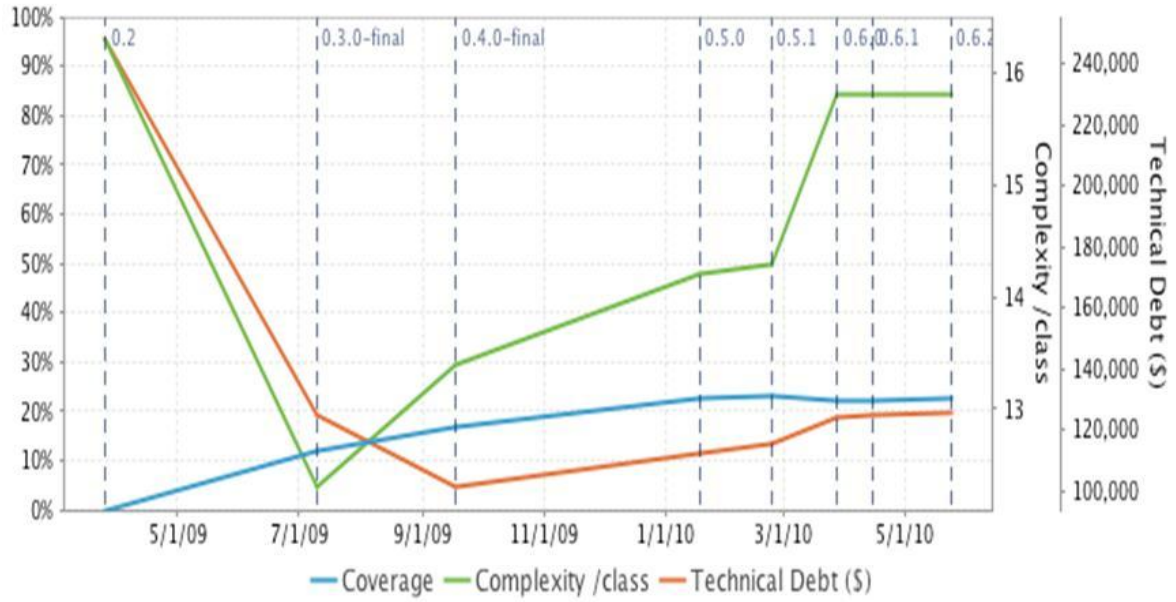
Πρόκειται για την μέτρηση του ποσοστού για τον παράγοντα της απασχόλησης. Οι ομάδες που ωθούνται στην ολοκληρωτική χρήση ενός πόρου, τείνουν να βελτιστοποιήσουν και τις ομάδες τους σε τοπικό επίπεδο, που συνεπάγεται πως ολόκληρο το σύστημα είναι υπο-βελτιωμένο. Αυτό αποτελεί και μια προσπάθεια, για να αυξηθεί η απόδοση της ομάδας άρα και η υπο-βελτίωση του ατόμου.

ΜΕΤΡΗΣΗ ΤΗΣ ΑΠΟΔΟΤΙΚΗΣ ΑΞΙΑΣ ΤΗΣ ΟΜΑΔΑΣ (Business Value measuring performance)

Αποτελεί μία προσπάθεια για την μέτρηση της παραγωγικότητας μέσω των μετρήσιμων αξιών της επιχείρησης για μία τοπική προσέγγιση προτεραιότητας του ανεκτέλεστου προϊόντος. Με την κατάλληλη προτεραιότητα, ιστορίες με υψηλή αξία –προτεραιότητα, θα βρίσκονται στην κορυφή των καθυστερήσεων του προϊόντος, άρα θα παρέχουν στην φάση υψηλότερη αξία. Σε περίπτωση που η τιμή της παραγωγικότητας χρησιμοποιείται, τότε το αποτέλεσμα είναι να παρουσιάζεται δυσλειτουργία, όταν η τιμή αυτή είναι μικρή. Όμως, όταν η πολυπλοκότητα των ιστοριών είναι υψηλότερη, τότε αυτές εισάγονται στο ανεκτέλεστο προϊόν. Αντίθετα, η μέτρηση αυτή είναι πιο συνετό να χρησιμοποιείται, για να επικυρώνεται η ιεραρχία των καθυστερήσεων και όχι για την απόδοση της ομάδας.

ΓΡΑΜΜΕΣ ΠΗΓΑΙΟΥ ΚΩΔΙΚΑ
(Source Lines of Code (SLOC))

Οι γραμμές πηγαίου κώδικα χρησιμοποιούνται για τον προσδιορισμό της παραγωγικότητας. Ένας μεγάλος αριθμός γραμμών δεν σημαίνει απαραίτητα και υψηλότερη παραγωγικότητα, αλλά μόνο παραπάνω κώδικα, ο οποίος μπορεί και να μην χρειάζεται. Χρησιμοποιώντας την συγκεκριμένη μετρική έρχεται σε αντίθεση με τις αρχές του «όσο το δυνατό πιο απλό» και «επαναπροσδιορισμό», για να παραμείνει ο κώδικας απλουστευμένος. Μια παραγωγική διαδικασία μπορεί να μειώσει τον πραγματικά παραγόμενο κώδικα, και δεν θα τον αυξήσει.



Εικόνα Ι Διάγραμμα όπου παρουσιάζονται η πολυπλοκότητα ανά κλάση (Complexity/class), ποσοστό των απαιτήσεων που έχουν καλυφτεί(Coverage) και το τεχνικό τμήμα (Technical Dept (s)) .

ΜΕΤΡΙΚΕΣ ΠΟΙΟΤΗΤΑΣ

ΤΕΧΝΙΚΑ ΣΗΜΕΙΑ ΧΡΕΟΥΣ
(Technical Debt Point)

Τα τεχνικά σημεία μετρούν τον όγκο και την δυναμικότητα του χρέους, για να καθοριστεί η ποιότητα εξέλιξης του προϊόντος. Μια αύξηση του αριθμού των σημείων αυτών αποτελεί μια κακή ένδειξη. Ένας μικρός αριθμός σημείων είναι ένας καλός δείκτης για την εταιρεία. Αυτό μπορεί να σημειωθεί με ένα τεχνικό όριο

του χρέους. Αν η οφειλή υπερβεί μία ορισμένη τιμή, τότε η ομάδα θα πρέπει να τεθεί σε αναγκαστική μείωση του χρέους της. Μπορεί να χρησιμοποιηθεί και σε ένα ραβδόγραμμα, όπου στοιβάζονται η μία φάση πάνω από την άλλη παρουσιάζοντας τις τεχνικές χρέους έναντι των νέων ιστοριών.

ΕΚΤΕΛΕΣΗ ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟΥΝ ΕΛΕΓΧΟΥ
(Running Automated Test)

Συμβάλει στην μέτρηση της λειτουργικότητας των εκτελέσιμων αρχείων μετά από κάθε επαναληπτική φάση. Μια υψηλή τιμή αυτού του δείκτη είναι προτιμότερη από μια χαμηλή. Δημιουργήθηκε και μελετήθηκε εκτενώς από τον Ron Jeffries. Σε συνδυασμό με χρήση μετρικών κάλυψης κώδικα, μπορεί να οδηγήσει στις μετρικές παραγωγικότητας και συμπεριφοράς.

ΔΗΜΟΣΙΕΥΣΗ ΤΩΝ ΕΞΕΡΧΟΜΕΝΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ
(Post Sprint Defect Arrival)

Μέτρηση των ελαττωμάτων που παρουσιάστηκαν μετά την ανάπτυξη της αρχικής φάσης. Ένας μεγάλος αριθμός για την συγκεκριμένη μετρική δεν είναι αποδεκτός , ενώ ένας μικρότερος είναι. Αποτελεί σημαντική ένδειξη της ποιότητας του προϊόντος, μιας και μπορεί να προβλέψει την πορεία της ποιότητας με βάση την στοιχειώδη ποιότητα προηγούμενων επαναληπτικών φάσεων. Δεν αποτελεί μετρική μόνο για την ποιότητα προϊόντος, αλλά και της ομάδας.

ΔΗΜΟΣΙΕΥΣΗ ΤΩΝ ΕΙΣΕΡΧΟΜΕΝΩΝ ΑΠΟΤΕΛΕΣΜΑΤΩΝ
(Post Release Defect Arrival)

Με την συγκεκριμένη υπολογίζονται τα ελαττώματα τα οποία παρουσιάστηκαν στο προϊόν, από την στιγμή που αυτό παραδόθηκε στους πελάτες/ιδιοκτήτες του. Το επιθυμητό αποτέλεσμα για την συγκεκριμένη μετρική θα πρέπει να είναι χαμηλό. Σε αντίθετη περίπτωση δείχνει πως η ποιότητα του προϊόντος υστερεί και δεν θα πρέπει να μετρηθεί μέχρι να απαλειφθεί.

ΒΑΣΙΚΟΤΕΡΑ ΕΛΑΤΤΩΜΑΤΑ ΠΡΟΣ ΕΠΙΔΙΟΡΘΩΣΗ
(Root Causes Fixed)

Μέτρηση του αριθμού των ελαττωμάτων για τα οποία έχει καθοριστεί η αιτία. Στη συγκεκριμένη, αν ο δείκτης του αποτελέσματος είναι υψηλός, τότε ο δείκτης γίνεται αποδεκτός. Λειτουργεί καλύτερα σε ομάδες, όπου η ομαδικότητα διατηρείται για την επιδιόρθωση ελαττωμάτων, και όχι μόνο για την αντιμετώπιση ορισμένων συμπτωμάτων.

ΑΡΙΘΜΟΣ ΕΠΑΝΑΛΗΠΤΙΚΩΝ ΔΟΚΙΜΩΝ
(Number of Test Cycles.)

Μετράει τον αριθμό των επαναλήψεων που έχουν γίνει για μια ιστορία ή μια φάση μεταξύ της ανάπτυξης και της δοκιμής. Για την συγκεκριμένη, ένας μικρότερος αριθμός είναι και ο καλύτερος δείκτης. Η εφαρμογή της μετρικής θα μπορούσε να οδηγήσει σε λιγότερες επαναλήψεις ανάδρασης/ανατροφοδότησης.

ΣΗΜΕΙΑ ΔΙΑΣΦΑΛΙΣΗ ΤΗΣ ΠΟΪΟΤΗΤΑΣ ΠΡΟΙΟΝΤΟΣ
(Quality assurance Story Points)

Μετράει ξεχωριστά τα τεχνικά και ποιοτικά σημεία μίας ιστορίας. Επιτρέπει στις ομάδες διασφάλισης των σημείων να προβλέψουν καλύτερα την παραγωγικότητα και την απόδοση τους. Ωστόσο, αυτό χωρίζει της ομάδες των μηχανικών και της ποιότητας να έρχονται αντίθετες με την βασική αρχή της μεθοδολογίας SCRUM για την δημιουργία διατμηματικών ομάδων. Επίσης, περιορίζει την συνομιλία για μηχανική/δοκιμή η οποία είναι κρίσιμη για τα αποτελέσματα της ομάδας της SCRUM.

ΑΡΙΘΜΟΣ ΠΟΙΟΤΙΚΩΝ ΕΛΑΤΤΩΜΑΤΩΝ
(Quality Number of Defects)

Μετρική για την ποιότητα και την παραγωγικότητα που προκύπτει από τον αριθμό των ελαττωμάτων, που βρίσκονται. Ένα υψηλό αποτέλεσμα θα αύξανε το μέτρο της ποιότητας και της παραγωγικότητας, αλλά και σε αυτή την περίπτωση έρχεται σε αντίθεση με την αρχή της ομαδικότητας της SCRUM. Είναι προτιμότερο να ενθαρρύνεται η διασφάλιση των σημείων και οι προγραμματιστές να εργάζονται για την τρέχουσα ιστορία και όχι για τον έλεγχο της ποιότητας όσων δεν έχουν αναπτυχτεί.

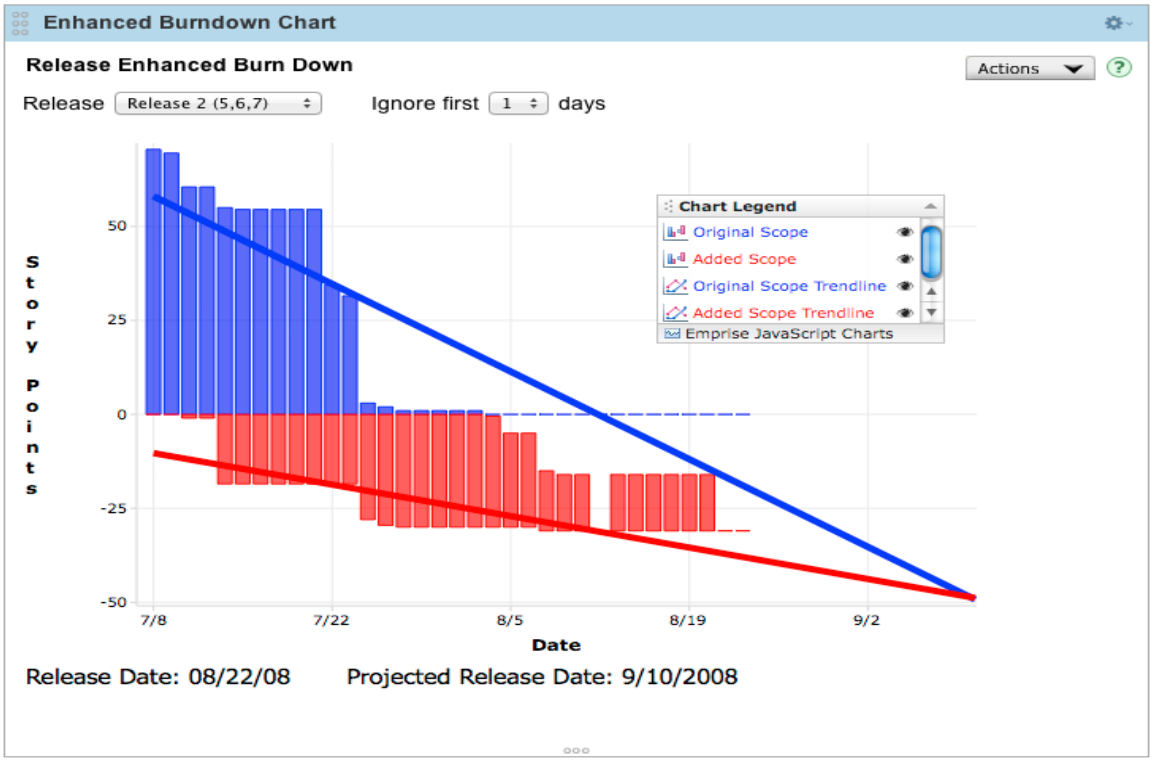
ΕΛΕΓΧΟΣ ΠΟΙΟΤΗΤΑΣ ΜΕΤΑ ΤΗΝ ΑΝΑΠΤΥΞΗ
(Quality Checked After Development)

Στην πραγματικότητα δεν είναι μια μετρική και για αυτό δεν συνιστάται η χρήση της.

ΜΕΤΡΙΚΕΣ ΠΡΟΒΛΕΠΤΙΚΟΤΗΤΑΣ

ΕΝΙΣΧΥΜΕΝΟ ΚΑΘΟΔΙΚΟ ΔΙΑΓΡΑΜΜΑ
(Enhanced Burndown Chart.)

Μετρική τόσο για τον υπολογισμό της ταχύτητας όσο και για την αλλαγή του πεδίου εφαρμογής σε όλη την έκδοση του προϊόντος, και αποτελεί το μέτρο για να διαπιστωθεί ή να προβλεφτεί η επίτευξη ή μη του στόχου. Μπορεί να συνδυαστεί με τις μετρικές του κόστους, για να προβλεφτεί το κόστος παραγωγής του συγκεκριμένου έργου.

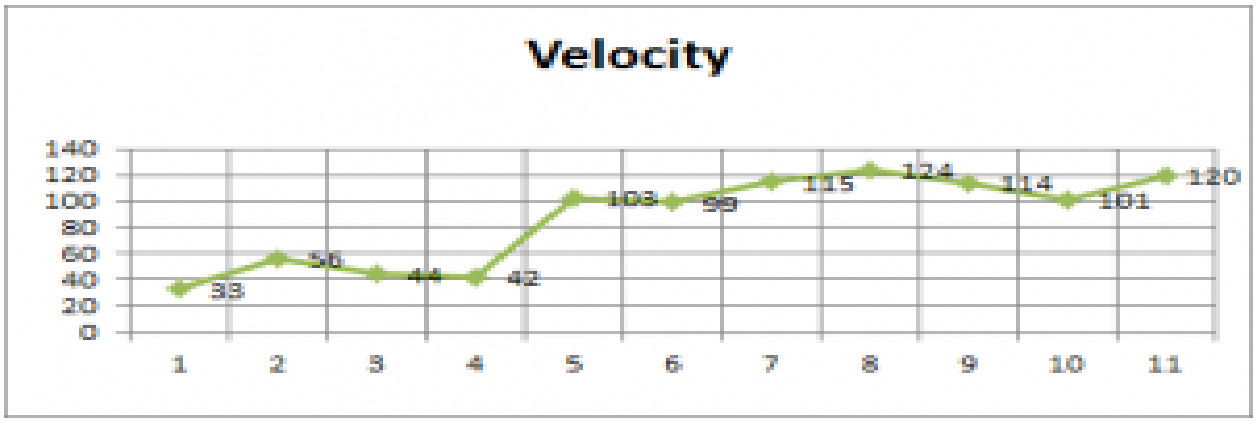


Εικόνα m Ενισχυμένο καθοδικό διάγραμμα των σημείων ιστορίας σε σχέση με το χρόνο.

ΤΑΧΥΤΗΤΑ
(Velocity)

Αποτελεί υποσύνολο της προηγούμενης μετρικής.

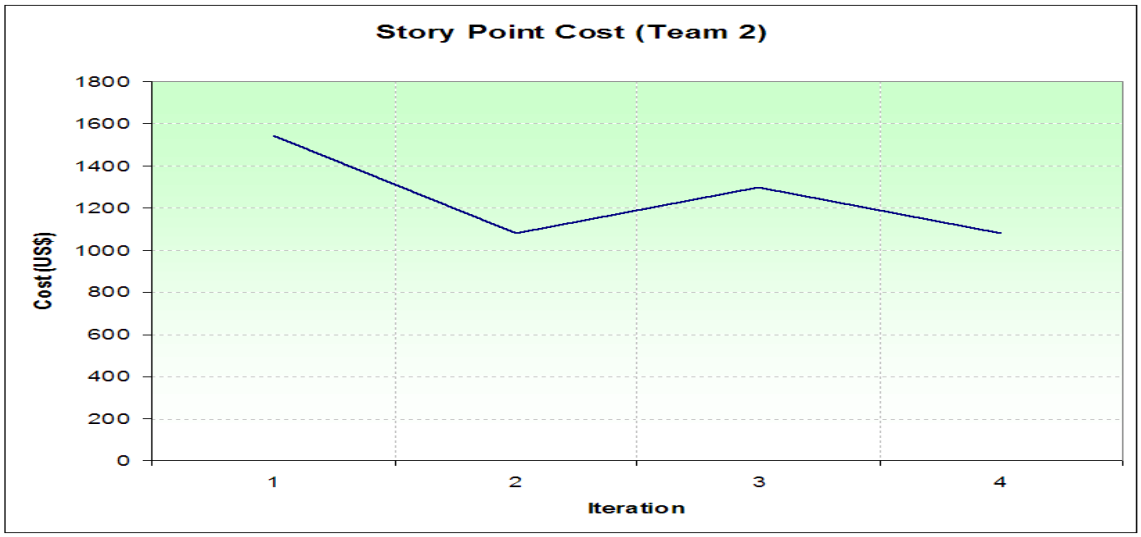
$$\text{Velocity} = \frac{\text{Sum of all 'sprint velocities'}}{\text{'Number of sprints}}$$



Εικόνα n Διάγραμμα Ταχύτητας.

ΚΟΣΤΟΣ ΑΝΑ ΣΗΜΕΙΟ ΙΣΤΟΡΙΑΣ
(Cost per Story Point)

Μετρική που υπολογίζεται σε € ανά σημείο ιστορίας, γνωρίζοντας τον αριθμό των ατόμων της κάθε μίας, το μήκος της επαναληπτικής διαδικασίας και η ταχύτητα της ιστορίας καθορίζεται με βάση το κόστος της ιστορίας ανά σημείο. Μπορεί να χρησιμοποιηθεί για τον υπολογισμό των κεφαλαίων αντί των κεφαλαιοποιημένων εξόδων, που χρησιμοποιεί η εταιρεία στον τομέα της πληροφορικής. Με τον διαχωρισμό των σημείων που είναι κεφάλαια και όχι κεφαλαιοποίηση του έργου. Είναι καλύτερο να χρησιμοποιεί ένα μέσο όρο για το κόστος και ταχύτητα χρησιμοποιώντας έναν αριθμό φάσεων.



Εικόνα ο Διάγραμμα κόστους ανά σημείο ιστορίας

Πίνακας α Πίνακας κόστους ανά σημείο ιστορίας.

Sprint	Velocity	Cost (US$)	Story Point Cost
1	7	55900	7986
2	16	55400	3463
3	17	47900	2818
4	21	52500	2500
5	26	55300	2127
6	29	49400	1703
7	17	49200	2894
8	27	51600	1911
9	12	43700	3642
		Average	3227

ΩΡΕΣ ΑΝΑ ΣΗΜΕΙΟ ΙΣΤΟΡΙΑΣ (Hours per Story Point)

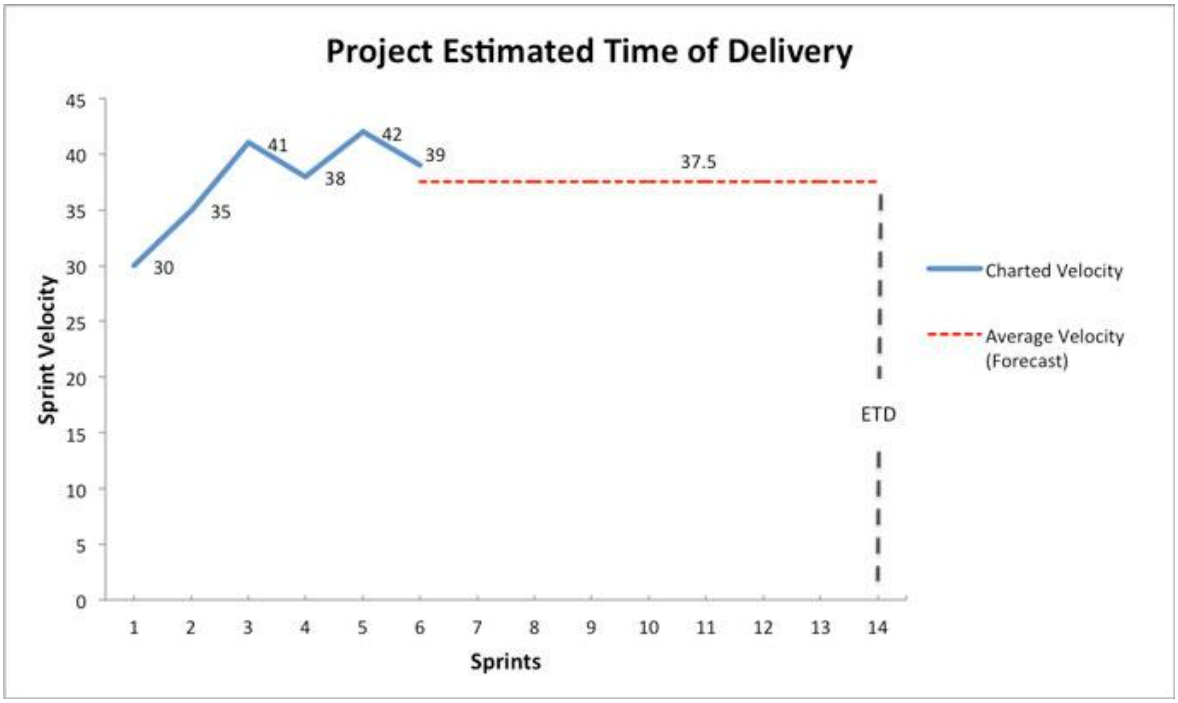
Μέτρηση του μέσου αριθμού των ωρών που εκτιμάται να ξοδευτούν ανά σημείο ιστορίας. Χρησιμοποιείται, για να προσδιορίσει, εάν η εκτιμώμενη ώρα εργασίας μπορεί να εφαρμοστεί σε ένα σημείο και να αλλάξει στην πάροδο του χρόνου. Για παράδειγμα, μια ομάδα που ξεκινά στις 10 ώρες ανά σημείο κατά μέσο όρο σπαταλάει 20 ώρες ανά σημείο και αυτό δείχνει ότι η αξία του σημείου μειώνεται με την πάροδο του χρόνου. Μπορεί εύκολα να γίνει κατάχρηση της μέτρησης, είτε από την ομάδα στην προσπάθεια της για εκτίμηση σημείων χρησιμοποιώντας την αντίστροφη αυτής της μέτρησης ή από τους ηγέτες πιέζοντας τις ομάδες να μειώσουν την προβλεπόμενη ώρα τους ανά σημείο

(υποθέτοντας ότι την κάνουν πιο αποτελεσματική). Θα πρέπει να χρησιμοποιείται μόνο για την αξιολόγηση της προβλεπτικότητας.

ΜΕΤΡΙΚΗ ΕΚΤΙΜΩΜΕΝΟΥ ΧΡΟΝΟΥ ΠΑΡΑΔΟΣΗΣ ΕΡΓΟΥ
(Project Estimated time of delivery metric)

Υπολογισμός προκύπτει από την διαίρεση του συνολικού αριθμού των υπόλοιπων σημείων ιστορίας στο ανεκτέλεστο υπόλοιπο προϊόν με «Μέση ταχύτητα», αν έχετε τρέξει λιγότερο από τέσσερις περίπου επαναλήψεις, τότε αυτή η μετρική πρόκειται να είναι αμφίβολη στην καλύτερη περίπτωση. Δημιουργείται στο τέλος κάθε φάσης και μας δίνει μια κατά προσέγγιση εκτίμηση (όχι εγγυήσεις) του, πότε ολόκληρη η καθυστέρηση (ή μέρος αυτής) θα ολοκληρωθεί με βάση τη μέση ταχύτητα της ομάδας, που λαμβάνονται σε μία συγκεκριμένη χρονική στιγμή.

$$\text{ETD} = \frac{\text{Total number of remaining story points in the product backlog}}{\text{'Average velocity'}}$$

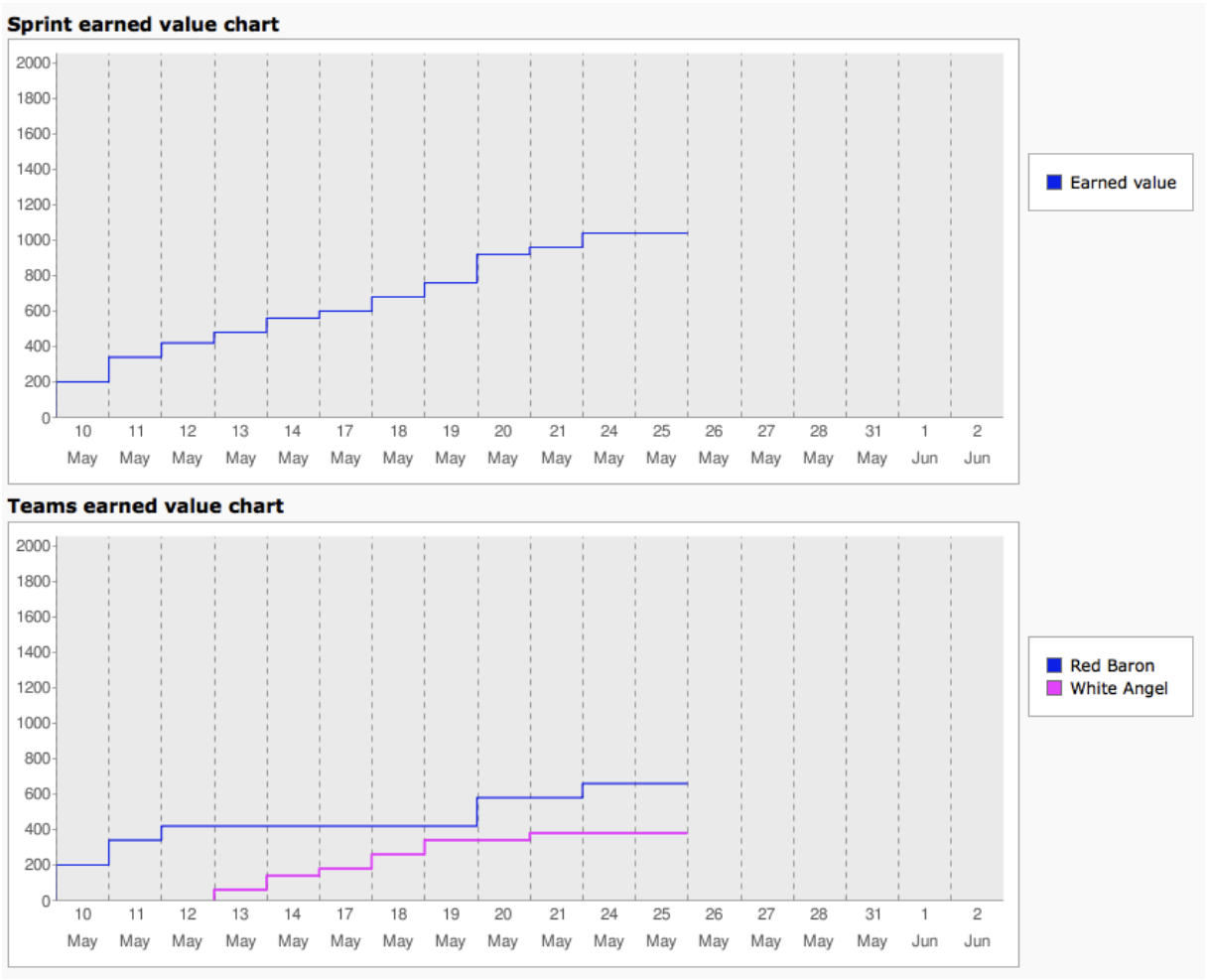


Εικόνα ρ Διάγραμμα υπολογισμού παράδοσης έργου.

ΜΕΤΡΙΚΕΣ ΑΞΙΑΣ

ΔΗΜΟΣΙΕΥΣΗ ΑΞΙΑΣ ΕΠΙΧΕΙΡΗΣΗΣ
(Business Value Delivered)

Μέτρηση της αξίας της επιχείρησης που παραδίδεται σε κάθε επαναληπτική φάση, με βάση την εκχώρηση μιας επιχειρηματικής αξίας για κάθε στοιχείο των καθυστερήσεων. Ο υψηλότερος αριθμός είναι ένας καλός δείκτης. Αυτό μπορεί επίσης, να αποτελέσει ένα κίνητρο για τους εργαζόμενους, διότι οι ομάδες θέλουν να προσφέρουν αξία και να αισθάνονται καλά. Όπως αναφέρθηκε παραπάνω, αυτό μπορεί να κάνει την ομάδα δυσλειτουργική, αν αυτή προσπαθεί να διατηρήσει την επιχειρηματική αξία κατά τη διάρκεια των πολλαπλών επαναλήψεων, διότι στην καθυστέρηση έχει δοθεί η σωστή προτεραιότητα, και η επιχειρηματική αξία θα πρέπει να είναι να δείξει μείωση κατά τη διάρκεια της αποτελεσματικότητας.



Εικόνα q Διαγράμματα απεικόνισης της αξίας που κερδίζει το έργο και η ομάδα ανάπτυξης.

ΕΡΕΥΝΑ ΙΚΑΝΟΠΟΙΗΣΗΣ ΠΕΛΑΤΩΝ
(Customer Satisfaction Survey)

Αποτελεί μέτρηση ανατροφοδότησης ποσοτικά και / ή ποιοτικά από τους πελάτες και άλλα ενδιαφερόμενα μέλη μέσω τακτικών ερευνών, που πραγματοποιούνται περίπου ανά τρίμηνο. Διάφορα στοιχεία ανατροφοδότησης μπορούν να συγκεντρωθούν για την ποιότητα, την προβλεπτικότητα, την παραγωγικότητα της παράδοσης, την υποστήριξη, την καταλληλότητα των νέων χαρακτηριστικών, κλπ.

ΕΡΕΥΝΑ ΙΚΑΝΟΠΟΙΗΣΗΣ ΕΡΓΑΖΟΜΕΝΩΝ
(Employee Satisfaction Survey)

Αποτελεί την μέτρηση της ποιοτικής και ποσοτικής εσωτερικής ανατροφοδότησης από τους υπαλλήλους με τακτική παρακολούθηση (ανά τρίμηνο) και την παρακολούθηση των αποτελεσμάτων στην πάροδο του χρόνου. Αυτό μπορεί να μετρήσει την αποτελεσματικότητα των ρόλων, την ποιότητα της εργασίας / προσωπικής ζωής, την ομαδική εργασία, τον ορισμό των προϊόντων, την τήρηση της διαδικασίας, την ανατροφοδότηση, την αξία ευτυχίας, κλπ. Ένα ευτυχισμένο εργατικό δυναμικό τείνει να αυξήσει την παραγωγικότητα του, και αυτό οφείλεται στους ανθρώπους, που βρίσκονται στο χώρο εργασίας και να συμβάλει στην αποτελεσματικότητα της ομάδας.

ΕΠΙΛΟΓΟΣ

Είναι φανερό πως τα αποτελέσματα τα οποία προκύπτουν από την εφαρμογή των μετρικών ποικίλουν και μπορεί να χρησιμοποιηθούν με πολλούς διαφορετικούς τρόπους. Από τα εξαγόμενα αυτά αποτελέσματα θα μπορέσει επίσης, να κατανοήσει κάποιος το ποσοστό απόδοσης αλλά και σωστής συνεργασίας / επικοινωνίας των εργαζομένων του. Όσο το ποσοστό της καλής επικοινωνίας και συνεργασίας αυξάνεται μεταξύ των προγραμματιστών τόσο πιο ποιοτικός είναι ο κώδικας που παράγουν και τόσο πιο γρήγορα θα ολοκληρώσουν το έργο που τους έχει ανατεθεί. Άρα, ο πελάτης θα είναι πιο ικανοποιημένος και η επιχείρηση και το προϊόν θα πάρουν παραπάνω αξία.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Η μηχανική λογισμικού βρίσκεται ακόμη σε πρώιμο στάδιο, έτσι όσο μεγαλώνει τόσο πιο πολλά θα προσφέρει στους μηχανικούς λογισμικού. Σίγουρα θα είναι πολλοί αυτοί που θα ισχυριστούν πως το επάγγελμα βρίσκεται ακόμα περίοδο κρίσης, αλλά πολλοί περισσότεροι θα είναι αυτοί, που θα βρίσκουν νέους τρόπους, για να μπορέσει να ξεπεράσει τις πιθανές αδυναμίες, που θα παρουσιάζονται.

Οι μετρικές λογισμικού θα είναι πάντοτε αυτές, που θα μπορούν να προσφέρουν πιο απτά αποτελέσματα για το έργο αλλά και το στάδιο στο οποίο αυτό βρίσκεται. Οι μεθοδολογίες από την άλλη θα προτείνουν τους βέλτιστους τρόπους για την ανάπτυξη ενός προγράμματος, και πάντοτε ανάλογα με τις απαιτήσεις του έργου θα υπάρχει και η κατάλληλη μεθοδολογία σχεδιασμού για το συγκεκριμένο έργο.

Είναι προφανές πως η SCRUM πλέον αποτελεί την πιο αποδοτική μέθοδο ανάπτυξης λογισμικού, μιας και η ανάπτυξη του προϊόντος γίνεται τμηματικά και ο πελάτης έχει την δυνατότητα να βλέπει την ανάπτυξη του έργου και να εκφράζει το ποσοστό ικανοποίησης του ως προς το αναμενόμενο αποτέλεσμα. Σε συνδυασμό με τις μετρικές λογισμικού, η SCRUM προβάλει τα πλεονεκτήματα από την χρήση των καλών / ευέλικτων πρακτικών. Ίσως θα έπρεπε όλες οι μεγάλες εταιρείες παραγωγής λογισμικού να αρχίσουν να χρησιμοποιούν την SCRUM, για να αναπτύσσουν πιο σωστά προϊόντα και σε πιο σύντομο χρονικό διάστημα.

ΛΕΞΙΚΟ ΟΡΩΝ

Ομάδα SCRUM – SCRUM Team

Την αποτελούν ο ιδιοκτήτης του προϊόντος, ο SCRUM Master και η ομάδα ανάπτυξης.

Ιδιοκτήτης Προϊόντος – Product Owner

Πρόκειται για το πρόσωπο το οποίο και είναι υπεύθυνο για την διατήρηση του ανεκτέλεστου προϊόντος των ενδιαφερόμενων μελών, και το οποίο είναι υπεύθυνο για την διασφάλιση της αξίας του έργου, που θα υλοποιήσει η ομάδα ανάπτυξης.

SCRUM Master

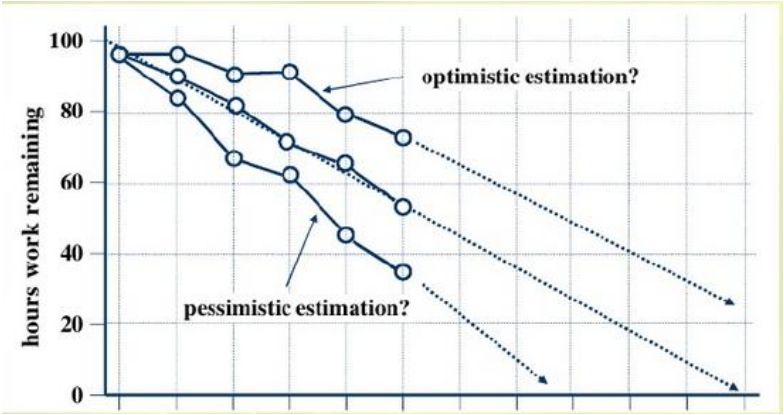
Είναι το άτομο που είναι υπεύθυνο για όλη την διαδικασία της μεθοδολογίας, και το οποίο φροντίζει, ώστε να χρησιμοποιηθεί σωστά και τα οφέλη τα οποία θα έχει να μεγιστοποιηθούν.

Ομάδα Ανάπτυξης – Development Team_____

Είναι μια ομάδα ανθρώπων η οποία είναι υπεύθυνη για την παροχή εκτελέσιμων προσαυξήσεων του προϊόντος με το τέλος κάθε επαναληπτικής φάσης.

Καθοδικό Διάγραμμα Φάσης – Sprint burn down chart

Στο καθοδικό διάγραμμα παρουσιάζεται η ημερήσια πρόοδος υλοποίησης του ανεκτέλεστου προϊόντος.



Εικόνα r Καθοδικό διάγραμμα φάσης.

Υπόλοιπο ανεκτέλεστο προϊόν – Product backlog

Περιλαμβάνει μία λίστα προτεραιότητας υψηλού επιπέδου απαιτήσεων.

Φάση ανεκτέλεστου προϊόντος – Sprint backlog

Λίστα με τα καθήκοντα του υπόλοιπου ανεκτέλεστου προϊόντος το οποίο και θα πρέπει να ολοκληρωθεί κατά την διάρκεια μίας επαναληπτικής φάσης.

Επαναληπτική φάση - Sprint

Ένα χρονικό διάστημα από μία έως και τέσσερις εβδομάδες συνήθως, κατά την διάρκεια της οποίας η ομάδα ανάπτυξης δεσμεύεται για την ανάπτυξη ενός συνόλου απαιτήσεων της λίστας με το ανεκτέλεστο προϊόν.

Πίνακας b Πίνακας φάσεων ανεκτέλεστου προϊόντος.

Tasks	Mon	Tues	Wed	Thur	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	4	
Test the middle tier	8	16	16	11	8
Write online help	12				
Write the helper class	8	8	8	8	8
Add error logging			8	4	

(Χρήστης) Ιστορία - (User) Story

Ένα χαρακτηριστικό που προστίθεται στην λίστα με το ανεκτέλεστο προϊόν, συνήθως αναφέρεται και ως ιστορία και έχει μια συγκεκριμένη δομή. Η δομή της είναι της μορφής «Ως <τύπος χρήστη> θέλω να <τύπος δραστηριότητας> έτσι ώστε <τύπος αποτελέσματος>». Ο λόγος για τον οποίο έχει αυτή την συγκεκριμένη δομή, είναι για να μπορεί η ομάδα αναγνωρίσει το αποτέλεσμα, που επιθυμούν να παίρνουν οι χρήστες, την ενέργεια, που θέλουν να εκτελείται καθώς είναι ένας τρόπος γραφείς που μπορεί ο καθένας να καταλάβει. Για παράδειγμα : «Ως χρήστης της εφαρμογής wiki θέλω μία εργαλειοθήκη, έτσι ώστε να μπορώ να εφαρμόσω εύκολα μία μορφοποίηση σε κάποια γραμματοσειρά». Μια ιστορία είναι μια ανεξάρτητη, διαπραγματεύσιμη, πολύτιμη , μικρή, εκτελέσιμη απαίτηση («INVEST»). Παρά το γεγονός ότι είναι ανεξάρτητη, δηλαδή δεν έχουν άμεσες εξαρτήσεις με άλλες απαιτήσεις, οι ιστορίες/απαιτήσεις μπορούν να ομαδοποιηθούν σε «έπη», και αυτό όταν εκπροσωπείται από έναν κατευθυντήριο χαρτί ή από το πιο κάτω/προηγούμενο ανεκτέλεστο προϊόν.

Πίνακας c Πίνακας ιστοριών χρήστη / περιπτώσεων χρήσης

ID	User Stories	Story Points
1.1	Members should be able to sign in to the website so that they can make use of the My Account Area. *note: login via username/password. Lock account after 5 unsuccessfull attempts. Notify user before 5th attempt and send email to the concerned email address.*	3
1.2	Members should be able to reset their password if they forget it, so that our involvement to reset is low. *note: send a link to reset password to the concerned email address after asking for username/email address. Have link for forget password on all sign in boxes.*	2
1.3	Members should be able to change their password so that their account is secure *Note: user must validate old password and confirm new password twice*	2
1.4	Members should be able to change their email address *Note: again must validate password*	1
1.5	Members should see a link for Help on Sign In/ Forget Password pages	1

Θέμα – Theme

Πρόκειται για ένα στόχο ανώτατου επιπέδου που μπορεί να καλύψει τα έργα και τα προϊόντα. Τα θέματα μπορούν να χωριστούν σε επιμέρους, τα οποία το πιο πιθανό να αφορούν το συγκεκριμένο προϊόν. Τα θέματα μπορούν επίσης να χρησιμοποιηθούν και σε επίπεδο προγράμματος και σχεδίου για να οδηγήσουν την στρατηγική σε μια ευθύγραμμη και σαφή κατεύθυνση επικοινωνίας.

Έπη - Epic

Είναι μία ομάδα συσχετιζόμενων ιστοριών , που χρησιμοποιούνται ως οδηγός του προϊόντος αλλά και του υπολοίπου του ανεκτέλεστου τμήματος , τα οποία δεν έχουν αναλυθεί αρκετά, ώστε να διασπαστούν σε ιστορίες. Ο διαχωρισμός αυτός πρέπει να γίνεται πριν από κάθε επαναληπτική διαδικασία, για να μειωθεί το ποσοστό της αβεβαιότητας για τα στοιχεία που θα ανατηχθούν. Τα έπη χρησιμοποιούνται και σε επίπεδο έργου αλλά και σε επίπεδο προγράμματος.

Συλλέκτης - Spike

Κάποια χρονικά διαστήματα χρησιμοποιούνται για την έρευνα ιδεών ή/και για την δημιουργία ενός απλού πρωτοτύπου. Ένας τέτοιος συλλέκτης μπορεί να προγραμματιστεί είτε μεταξύ τον επαναληπτικών διαδικασιών ή όταν οι ομάδες είναι μεγάλες, μπορεί να προγραμματιστεί πριν την παράδοση πολλών στόχων. Συλλέκτες μπορεί να εισάγονται και πριν την παράδοση μεγάλων επών ή και ιστοριών/απαιτήσεων χρηστών, προκειμένου να εξασφαλιστεί ο προϋπολογισμός,

και να επιταθούν οι γνώσεις ή/και να παράγουν νέες έννοιες. Η διάρκεια και ο στόχος ενός συλλέκτη πρέπει να συμφωνηθούν από τον ιδιοκτήτη του προϊόντος και την ομάδα παράδοσης του προϊόντος πριν την έναρξη της υλοποίησης του. Σε αντίθεση με τις δεσμεύσεις μια επαναληπτικής ανάπτυξης, ένας συλλέκτης άλλοτε μπορεί και άλλοτε ‘όχι να παράγει απτό, εκτελέσιμο αρχείο. Για παράδειγμα, ο στόχος ενός συλλέκτη θα μπορούσε να είναι να φτάσει με επιτυχία σε μία απόφαση για την πορεία της δράσης. Ο συλλέκτης είναι πιο πάνω από τον χρόνο, και δεν είναι απαραίτητα ο στόχος, που πρέπει να παραδοθεί.

Σφαίρα ιχνηλάτησης – Tracer Bullet

Είναι ένας συλλέκτης για την τρέχουσα αρχιτεκτονική, το τρέχον σύνολο τεχνολογίας και βέλτιστων πρακτικών, που οδηγεί σε πιο ποιοτικό κώδικα παραγωγής. Θα μπορούσε να είναι απλά μια πολύ στενή εφαρμογή λειτουργικότητας, αλλά δεν είναι ορθό να πετάμε κώδικα. Αποτελεί μέρος της ποιότητας παραγωγής αλλά και του υπόλοιπου επαναλήψεων και στηρίζεται σε αυτόν τον κώδικα.

Σημεία κλίμακας/Προσπάθειας/Σημεία Ιστορίας-Απαιτήσεων – Point Scale/Effort/Story Points

Σχετίζονται με ένα αφηρημένο σημείο του συστήματος, το οποίο και χρησιμοποιείται για να συζητηθούν οι δυσκολίες των απαιτήσεων, χωρίς να χρειάζεται να γίνεται ανάθεση στον πραγματικό χρόνο. Η κλίμακα η οποία και χρησιμοποιείται πιο συχνά είναι μια ολοκληρωμένη ακολουθία Fibonacci (1,2,3,5,8,13,20,40,100), αν και μερικές ομάδες χρησιμοποιούν γραμμική κλίμακα (1,2,3,4,..), οι αρμοδιότητες των δύο (1,2,4,8,…) και μεγέθη ρούχων (XS, S, M, L, XL, XXL,…).

Εργασίες - Tasks

Μια εργασία προστίθεται στην αρχή μια επαναληπτικής διαδικασίας και κατανέμεται σε ώρες. Κάθε εργασία δεν θα πρέπει να ξεπερνάει σε όριο τις 12 ώρες, ωστόσο είναι κοινό για τις ομάδες να επιμένουν ότι η ολοκλήρωση ενός έργου δεν θα πρέπει να υπερβαίνει την μία μέρα.

Ορισμό του Γίνεται – Definition of Done (DoD)

Είναι μια έξοδος κριτηρίων για να καθοριστεί, αν το υπόλοιπο του ανεκτέλεστου προϊόντος είναι πλήρες. Σε πολλές περιπτώσεις το DoD απαιτεί όλες οι αναδρομικές επαναλήψεις να είναι επιτυχείς.

Εικόνα s Λίστα με ορισμό του Γίνεται.

Ταχύτητα - Velocity

Είναι η συνολική προσπάθεια μια ομάδα να είναι ικανή σε μία επαναληπτική δομή. Ο αριθμός της ταχύτητας προέρχεται από την προσθήκη όλων των σημείων των ιστοριών/απαιτήσεων της τελευταίας επαναληπτικής δομής. Αύτη η τιμή της ταχύτητας αποτελεί και την κατευθυντήρια γραμμή για την ομάδα, ώστε να κατανοήσει τους τρόπους με τους οποίους μπορούν να υλοποιήσουν πολλές ιστορίες/απαιτήσεις στην ίδια επαναληπτική φάση.

Εμπόδιο - Impediment

Μπορεί να είναι οτιδήποτε εμποδίζει έναν μέλος της ομάδας από την εκτέλεση της εργασίας όσο το δυνατόν αποτελεσματικότερα.

Sashimi

Είναι μια έκθεση που περιγράφει ότι κάτι «γίνεται/εκτελείταιι». Ο ορισμός του «γίνεται/εκτελείται» μπορεί να διαφέρει από την μια ομάδα της SCRUM στην άλλη, αλλά θα πρέπει να είναι συνεπείς μέσα σε μία ομάδα.

Ανώμαλος Τερματισμός – Abnormal Termination

Ο ιδιοκτήτης του προϊόντος μπορεί να ακυρώσει μια φάση της ανάπτυξη του έργου, αν το κρίνει απαραίτητο. Μπορεί να το πετύχει με την είσοδο μια ομάδας SCRUM, ενός SCRUM Master ή ενός Manager. Για παράδειγμα, η διοίκηση

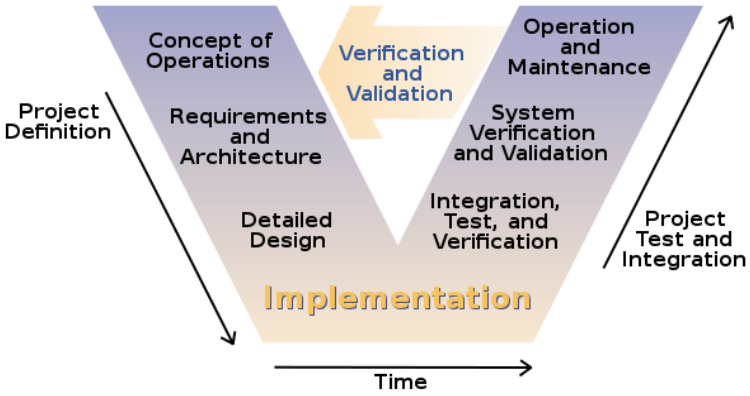
μπορεί να θέλει να ακυρώσει μια φάση, αν οι εξωτερικές περιστάσεις αναιρούν την αξία της φάσης. Αν μια φάση τερματιστεί ασυνήθιστα, τότε το επόμενο βήμα είναι να προβεί σε μια νέα συνάντηση για τον σχεδιασμό μίας φάσης, όπου ο λόγος είναι η τελική αξιολόγηση.

Σχεδιασμός Πόκερ- Planning Poker

Στην συνάντηση σχεδιασμού μίας φάσης, η ομάδα κάνει προσπάθειες εκτίμησης για το υπόλοιπου του ανεκτέλεστου προϊόντος. Ο ιδιοκτήτης του προϊόντος χειρίζεται αυτές τις εκτιμήσεις, έτσι ώστε να έχει την «εξουσία» να δώσει προτεραιότητα στα αποτελεσματικά τμήματα του ανεκτέλεστου προϊόντος , και να προβλέψει κάποιο αποτέλεσμα με βάση την ταχύτητα της ομάδας.

Μοντέλο V - V-model

Το V-μοντέλο αντιπροσωπεύει μια διαδικασία ανάπτυξης λογισμικού (εφαρμόζεται επίσης για την ανάπτυξη υλικού), το οποίο μπορεί να θεωρηθεί ως επέκταση του μοντέλου καταρράκτη. Αντί της προς τα κάτω κίνησης με γραμμικό τρόπο, τα βήματα της διαδικασίας είναι λυγισμένα προς τα πάνω μετά την φάση κωδικοποίησης, για να σχηματίσει το τυπικό σχήμα V. Το V-Model καταδεικνύει τις σχέσεις μεταξύ κάθε φάση του κύκλου ζωής της ανάπτυξης και των συναφών φάσεων των δοκιμών. Οι οριζόντιοι και κατακόρυφοι αξόνες αναπαριστούν το χρόνο ή την πληρότητα του έργου (από αριστερά προς τα δεξιά) και το επίπεδο αφαίρεσης (αδρότερων κόκκους ανώτατη αφαίρεσης), αντιστοίχως.



Εικόνα t Το μοντέλο-V της προόδου ανάπτυξης ενός συστήματος.

ΠΑΡΑΠΟΜΠΕΣ

[1] Αίθουσα Η/Υ : επιλέγω τον όρο αυτό, για να αποδώσω τον αγγλικό όρο «machine room», όπου εννοείται το δωμάτιο που ήταν εγκατεστημένος ο υπολογιστής που χρησιμοποιούσαν για την εργασία τους.

[2] Ομάδα χρηστών SHARE : πρόκειται για μια εθελοντική ομάδα χρηστών της IBM. Ιδρύθηκε το 1955 στο Λος Άντζελες

[3] Η έκθεση της διάσκεψης υπό την αιγίδα της επιστημονικής Επιτροπής του NATO το 1968 στο Garmisch της Γερμανίας. http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF

[4] Η έκθεση της διάσκεψης υπό την αιγίδα της επιστημονικής Επιτροπής του NATO το 1969 στη Ρώμη της Ιταλίας. http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF.

[5] The Mythical Man-Month : βιβλίο σχετικό με την μηχανική λογισμικού και κύριο θέμα ενασχόλησης «πως η προσθήκη ανθρώπινου δυναμικού σε ένα ήδη «καθυστερημένο» έργο θα συμβάλει στην περαιτέρω καθυστέρηση του» και αυτό είναι γνωστό ως ο νόμος του Brooks.

[6] Τυπικές μέθοδοι είναι ένα ιδιαίτερο είδος μαθηματικών με βάση τις τεχνικές προδιαγραφές για την ανάπτυξη και τον έλεγχο του λογισμικού και του υλικού των συστημάτων. Η χρήση των τυπικών μεθόδων για το λογισμικό και τον σχεδιασμό του υλικού υπαγορεύεται από την προσδοκία ότι, όπως και σε άλλους κλάδους της μηχανικής, την εκτέλεση κατάλληλων μαθηματικών αναλύσεων που μπορούν να συμβάλουν στην αξιοπιστία και την ευρωστία του σχεδίου. Οι τυπικές μέθοδοι μπορούν να περιγραφούν καλύτερα ως μια εφαρμογή μιας μεγάλης ποικιλίας από βασικά θεωρητικά στοιχεία της επιστήμης των υπολογιστών.

[7] Object Oriented Programming – OOP : αποτελεί παράδειγμα προγραμματισμού με χρήση «αντικειμένων» - συνήθως μια κλάση – που αποτελείται από πεδία δεδομένων και μεθόδους σε συνδυασμό με τις αλληλεπιδράσεις τους, για να σχεδιάσουν τις εφαρμογές και τα προγράμματα ενός Η/Υ . Οι τεχνικές προγραμματισμού μπορεί να περιλαμβάνουν χαρακτηριστικά όπως ο πολυμορφισμός, η αφαίρεση δεδομένων, η ενθυλάκωση, η κληρονομικότητα. Οι σύγχρονες γλώσσες προγραμματισμού υποστηρίζουν πολλές μορφές του OOP .

[8] Εργαλεία CASE – Computer Aided Software Engineering : επιστημονική εφαρμογή σειράς εργαλείων και μεθόδων για ένα σύστημα λογισμικού το οποίο έχει στόχο να οδηγήσει σε υψηλής ποιότητας, χωρίς ελαττώματα, διατηρήσιμο προϊόν. Αναφέρεται και στις μεθόδους για την ανάπτυξη συστημάτων πληροφοριών μαζί με τα αυτοματοποιημένα εργαλεία, που μπορούν να χρησιμοποιηθούν στην διαδικασία ανάπτυξης λογισμικού.

[9] ADA είναι ένα δομημένο, στατικά δακτυλογραφημένο, επιτακτικής ανάγκης ευρέως φάσματος, και OOP υψηλού επιπέδου γλώσσα προγραμματισμού, που ξεκίνησε από την Pascal και άλλες γλώσσες προγραμματισμού.

[10] Τεκμηρίωση : είναι ένας όρος με πολλές σημασίες, οι πιο κοινές από τις οποίες είναι τα εξής:

- Ένα σύνολο από έγγραφα που παρουσιάζονται στο χαρτί, ή σε απευθείας σύνδεση, ή σε ψηφιακό ή αναλογικό μέσο, όπως κασέτα ή CD.
- Αποτελεί τη διαδικασία της καταγραφής της γνώσης, όπως σε επιστημονικά άρθρα.
- Επιπλέον, είναι μια διαδικασία για την παροχή αποδεικτικών στοιχείων.
- Η συγγραφή της τεκμηρίωσης ενός προϊόντος, όπως η τεκμηρίωση του λογισμικού.
- Ένα συνώνυμο του όρου είναι και ο όρος έγγραφο.
- Ένα άλλο συνώνυμο του όρου είναι βιβλιογραφία.
- Αποτελεί ένα πεδίο της μελέτης και ένα επάγγελμα που ιδρύθηκε από τον Paul Otlet (1868-1944) και τον Henri La Fontaine (1854-1945), η οποία ονομάζεται επίσης επιστήμη τεκμηρίωση. Οι επαγγελματίες που έχουν εκπαιδευτεί στον τομέα αυτό ονομάζονται τεκμηριωτές. Αυτό το πεδίο άλλαξε το όνομά της σε επιστήμη πληροφορίας το 1968, αλλά ορισμένες χρήσεις του όρου τεκμηρίωσης εξακολουθούν να υφίστανται και έχουν γίνει προσπάθειες να επαναφέρει τον όρο τεκμηρίωση ως ένα πεδίο μελέτης.

[11] Προτυποποίηση είναι μια διαδικασία της ανάπτυξης και της εφαρμογής τεχνικών προτύπων. Οι στόχοι της προτυποποίησης μπορεί να βοηθήσει με την ανεξαρτησία των μεμονωμένων προμηθευτών, τη

συμβατότητα, λειτουργικότητα, την ασφάλεια, την επαναστατικότητα ή την ποιότητα .

[12] Capability Maturity Model (CMM) αποτελεί το σήμα κατατεθέν της υπηρεσίας του Carnegie Mellon University, (CMU) είναι ένα μοντέλο ανάπτυξης, που δημιουργήθηκε μετά από την μελέτη των στοιχείων, που συλλέγονται από τους οργανισμούς που έχουν συναφθεί με το Υπ.Άμυνας των ΗΠΑ, ο οποίος χρηματοδότησε την έρευνα. Αυτό το μοντέλο έγινε με βάση τις έρευνες του Carnegie Mellon University, που δημιούργησε το Software Engineering Institute (SEI). Ο όρος Maturity σχετίζεται με τον βαθμό της διατύπωσης και βελτιστοποίησης των διαδικασιών, από ad hoc πρακτικές, να ορίζει τα επίσημα βήματα, για να διαχειρίζεται τα αποτελέσματα των μετρήσεων, την ενεργή βελτιστοποίηση των διαδικασιών. Όταν το μοντέλο εφαρμόζεται στο λογισμικό ανάπτυξης ενός υπάρχοντος οργανισμού ή στις διαδικασίες του, επιτρέπει μια πιο αποτελεσματική προσέγγιση για την βελτίωση του. Τελικά, έγινε σαφές ότι το μοντέλο θα μπορούσε να εφαρμοστεί και σε άλλες διεργασίες. Αυτό οδήγησε σε μια πιο γενική έννοια που εφαρμόζεται στην επιχείρηση.

[13] No Silver Bullet είναι ένα άρθρο που έχει συζητηθεί ευρέως και ασχολείται με την μηχανική λογισμικού, το οποίο είναι γραμμένο από τον Fred Brooks το 1986 . Ο Brooks υποστηρίζει ότι «δεν υπάρχει ενιαία ανάπτυξη, είτε τεχνική ή τεχνολογία διαχείρισης, η οποία από μόνη της. υπόσχεται ακόμη μία τάξη μεγέθους [δεκαπλάσια] βελτίωση μέσα σε μια δεκαετία στην παραγωγικότητα, στην αξιοπιστία, στην απλότητα. "Δηλώνει, επίσης, ότι «δεν μπορούμε να περιμένουμε ποτέ να δούμε δύο φορές τα κέρδη κάθε δύο χρόνια" στην ανάπτυξη λογισμικού, όπως υπάρχει στην ανάπτυξη του υλικού.

[14] Search engine optimization – SEO είναι μια διαδικασία που επηρεάζει την προβολή μια ιστοσελίδας ή μια ιστοσελίδα έπειτα από μια αναζήτηση. Γενικά, όσο πιο πολύ εμφανίζεται μια ιστοσελίδα στην λίστα των αποτελεσμάτων μίας αναζήτησης τόσο πιο πολύ θα είναι και οι χρήστες τις συγκεκριμένης μηχανής αναζήτησης που θα την επισκεφτούν. Οι βελτιστοποιημένες τεχνικές αναζήτησης έχουν στόχο διάφορα είδη αναζήτησης, συμπεριλαμβανομένων και της αναζήτησης εικόνων, βίντεο, ακαδημαϊκή αναζήτηση, ειδησεογραφική αναζήτηση, τοπική αναζήτηση σε κάποιον Η/Υ και αρκετών άλλων ειδών.

[15] Extreme Programming (XP) είναι μια μεθοδολογία ανάπτυξης λογισμικού η οποία έχει στόχο την βελτίωση της ποιότητας του λογισμικού και την ανταπόκριση στις μεταβαλλόμενες απαιτήσεις των πελατών. Ως ένας τύπος ευέλικτης ανάπτυξης λογισμικού υποστηρίζει ότι συχνά «απελευθερώνεται» σε σύντομους κύκλους ανάπτυξης, το οποίο έχει στόχο να βελτιώσει την παραγωγικότητα και την εισαγωγή σημείων ελέγχου, όπου οι νέες απαιτήσεις των πελατών μπορούν να υιοθετηθούν.

[16] Κωδικός boilerplate ή boilerplate στον προγραμματισμό των Η/Υ είναι ένα από τα τμήματα του κώδικα που πρέπει να περιλαμβάνεται σε πολλά σημεία με μικρή ή ακόμη και καθόλου τροποποίηση. Πιο συχνά χρησιμοποιείται, όταν αναφερόμαστε σε γλώσσες που θεωρούνται αναλυτικές, ,δηλαδή, σε γλώσσες όπου ο προγραμματιστής πρέπει να γράψει πολύ κώδικα, για να κάνει ελάχιστες μόνο εργασίες. Η ανάγκη για αυτό το στερεότυπο μπορεί να μειωθεί μέσω μηχανισμών υψηλού επιπέδου, όπως το meta-programming (το οποίο είναι ένα στερεότυπο κείμενο που έχει γράψει αυτόματα ο υπολογιστής), σύμβαση για τη διαμόρφωση (η οποία παρέχει καλά προκαθορισμένες τιμές, μειώνοντας την ανάγκη για να καθοριστούν οι λεπτομέρειες του προγράμματος σε κάθε έργο) και το μοντέλο με γνώμονα την τεχνολογία (η οποία χρησιμοποιεί τα μοντέλα και τα το μοντέλο-σε-γεννήτριες κώδικα, εξαλείφοντας την ανάγκη για χειροκίνητη κώδικα boilerplate).

[17] TickIT είναι ένας τρόπος διαχείρισης του προγράμματος με σκοπό την πιστοποίηση για την ανάπτυξη λογισμικού, που κατά κύριο λόγο υποστηρίζεται από το Hv. Βασίλειο και την σουηδική βιομηχανία μέσω UKAS και SWEDAC αντίστοιχα.

[18] Το Software Engineering Body of Knowledge – SWEBOK (Σώμα Μηχανικών Λογισμικού της Γνώσης) είναι ένα προϊόν της Συντονιστικής Επιτροπής Τεχνολογίας Λογισμικού, που χρηματοδοτείται από την Κοινωνία της Πληροφορικής IEEE.

[19] Το SDM - software development methodology ή system development methodology είναι ένα πλαίσιο που χρησιμοποιείται για την κατασκευή, το σχεδιασμό, τον έλεγχο της διαδικασίας ανάπτυξης ενός συστήματος πληροφόρησης.

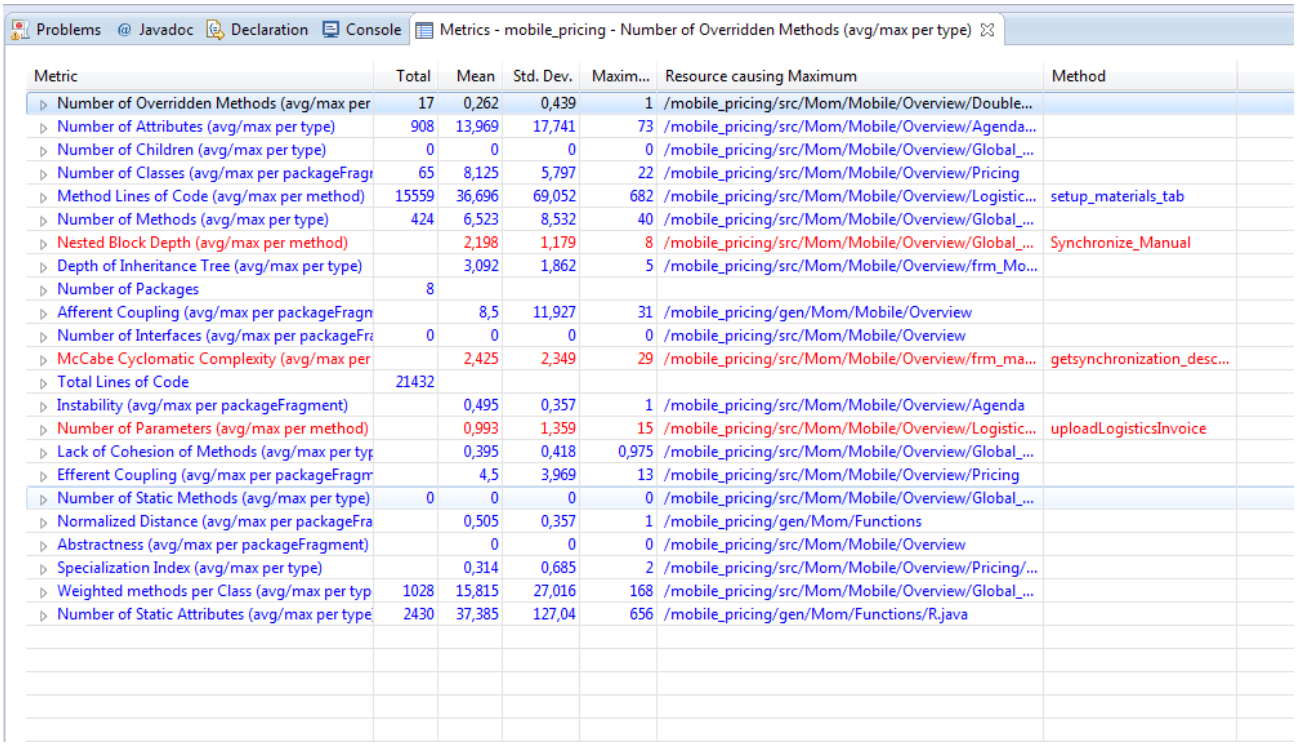
[20] Το OOPSLA (Object-Oriented Programming, Systems, Languages & Applications / Αντικειμενοστρεφής Προγραμματισμός, Συστήματα, Γλώσσες &

Εφαρμογές) είναι ένα ετήσιο συνέδριο έρευνας της ACM (Association for Computing Machinery διεθνή κοινότητα για την πληροφορική με βάση τις Ηνωμένες Πολιτείες). Το OOPSLA λαμβάνει χώρα κυρίως στις Ηνωμένες Πολιτείες, ενώ ένα παραπλήσιο του συνεδρίου OOPSLA, το ECOOP, πραγματοποιείται στην Ευρώπη.

ΠΑΡΑΡΤΗΜΑΤΑ

Για να γίνει πιο κατανοητή η χρήση των μετρικών λογισμικού αλλά και ο τρόπος με τον οποίο γίνεται η εξαγωγή των αποτελεσμάτων των μετρικών και η μορφή που αυτά έχουν στο πρόγραμμα Eclipse, εγκατέστησα την πρόσθετη εφαρμογή για τις μετρικές λογισμικού και τις εφάρμοσα σε μια εφαρμογή που ανέπτυξα στα πλαίσια της πρακτικής μου άσκησης.

Η εφαρμογή αποτελεί ένα πρόγραμμα φορητής τιμολόγησης και παρακάτω παρουσιάζονται τα εξαγόμενα αποτελέσματα της εφαρμογής των μετρικών στην συγκεκριμένη εφαρμογή.



Problems | Javadoc | Declaration | Console | Metrics - mobile_pricing - Number of Overridden Methods (avg/max per type)

Metric	Total	Mean	Std. Dev.	Maxim...	Resource causing Maximum	Method
Number of Overridden Methods (avg/max per	17	0,262	0,439	1	/mobile_pricing/src/Mom/Mobile/Overview/Double...	
Number of Attributes (avg/max per type)	908	13,969	17,741	73	/mobile_pricing/src/Mom/Mobile/Overview/Agenda...	
Number of Children (avg/max per type)	0	0	0	0	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	
Number of Classes (avg/max per packageFragr	65	8,125	5,797	22	/mobile_pricing/src/Mom/Mobile/Overview/Pricing	
Method Lines of Code (avg/max per method)	15559	36,696	69,052	682	/mobile_pricing/src/Mom/Mobile/Overview/Logistic...	setup_materials_tab
Number of Methods (avg/max per type)	424	6,523	8,532	40	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	
Nested Block Depth (avg/max per method)		2,198	1,179	8	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	Synchronize_Manual
Depth of Inheritance Tree (avg/max per type)		3,092	1,862	5	/mobile_pricing/src/Mom/Mobile/Overview/frm_Mo...	
Number of Packages	8					
Afferent Coupling (avg/max per packageFragn		8,5	11,927	31	/mobile_pricing/gen/Mom/Mobile/Overview	
Number of Interfaces (avg/max per packageFra	0	0	0	0	/mobile_pricing/src/Mom/Mobile/Overview	
McCabe Cyclomatic Complexity (avg/max per		2,425	2,349	29	/mobile_pricing/src/Mom/Mobile/Overview/frm_ma...	getsynchronization_desc...
Total Lines of Code	21432					
Instability (avg/max per packageFragment)		0,495	0,357	1	/mobile_pricing/src/Mom/Mobile/Overview/Agenda	
Number of Parameters (avg/max per method)		0,993	1,359	15	/mobile_pricing/src/Mom/Mobile/Overview/Logistic...	uploadLogisticsInvoice
Lack of Cohesion of Methods (avg/max per typ		0,395	0,418	0,975	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	
Efferent Coupling (avg/max per packageFragm		4,5	3,969	13	/mobile_pricing/src/Mom/Mobile/Overview/Pricing	
Number of Static Methods (avg/max per type)	0	0	0	0	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	
Normalized Distance (avg/max per packageFra		0,505	0,357	1	/mobile_pricing/gen/Mom/Functions	
Abstractness (avg/max per packageFragment)		0	0	0	/mobile_pricing/src/Mom/Mobile/Overview	
Specialization Index (avg/max per type)		0,314	0,685	2	/mobile_pricing/src/Mom/Mobile/Overview/Pricing/...	
Weighted methods per Class (avg/max per typ	1028	15,815	27,016	168	/mobile_pricing/src/Mom/Mobile/Overview/Global_...	
Number of Static Attributes (avg/max per type	2430	37,385	127,04	656	/mobile_pricing/gen/Mom/Functions/R.java	

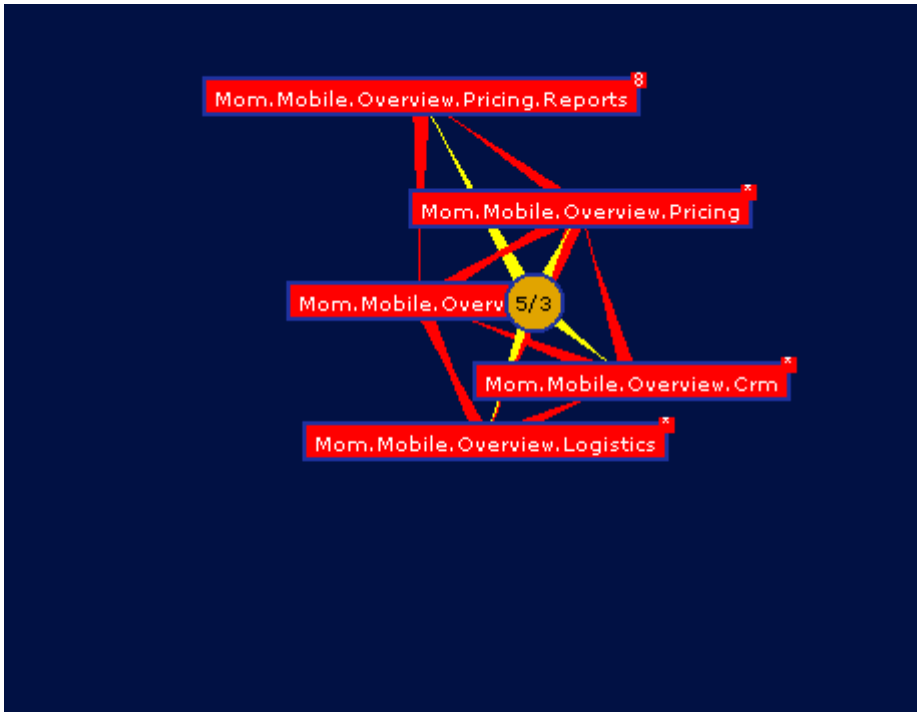
Εικόνα ιι Παρουσίαση όλων των μετρικών λογισμικού στον πρόγραμμα Eclipse

Total Lines of Code	21432
src	19002
Mom.Mobile.Overview.Pricing	6607
frm_pricing.java	1437
frm_loading.java	1094
frm_receipt.java	1069
customer_card.java	648
frm_receipts_list.java	451
material_card.java	450
frm_invoices_list.java	449
frm_loadings_list.java	408
customer_details.java	218
materials_details.java	178
Mobile_pricingActivity.java	95
frm_report_viewer.java	55
frm_reports.java	55
Mom.Mobile.Overview	4789
Global_Class.java	3493
frm_Mobile_Overview.java	377
settings.java	376
frm_manual_update.java	279
DoubleProgressDialog.java	221
about.java	43
Mom.Mobile.Overview.Logistics	3492
frm_logistics_invoice.java	1969
frm_logistics_add_colla.java	809
frm_logistics_destroy_colla.java	554
frm_logistics_main.java	160
Mom.Mobile.Overview.Agenda	2203
frm_agenda_activities.java	563
frm_agenda_contacts.java	500
frm_agenda_category.java	281
frm_agenda_situation.java	269
frm_agenda_recources.java	269
frm_agenda_group.java	261
frm_agenda_main.java	60
Mom.Mobile.Overview.Crm	1325
frm_crm_daily_program.java	1035
frm_crm_main.java	268
frm_crm_online_reports.java	22
Mom.Mobile.Overview.Pricing.Reports	586
Report_Receipts.java	202
Report_MaterialsOnTruck.java	195
Report_CustomersBalance.java	189

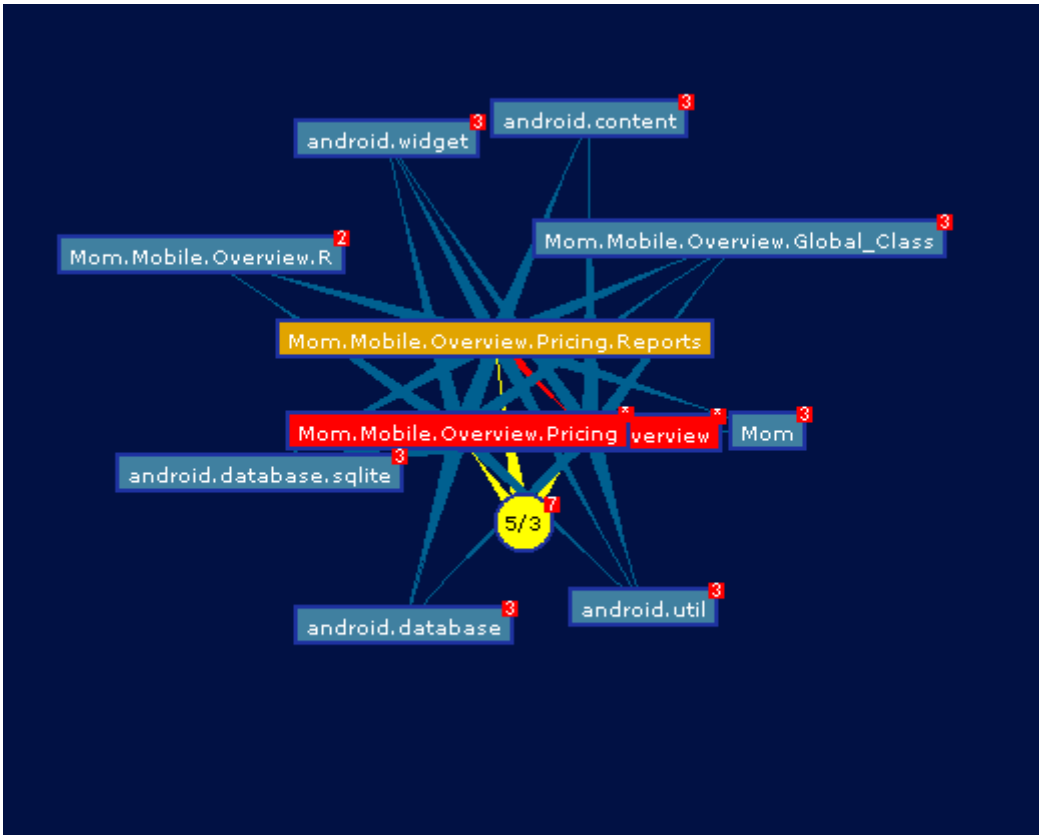
Εικόνα ν Συνολικές Γραμμές Κώδικα ανά κλάση του προγράμματος.

Method Lines of Code (avg/max per method)	15559	36,696	69,052	682	/n
src	15559	36,696	69,052	682	/n
Mom.Mobile.Overview.Logistics	3143	68,326	136,762	682	/n
frm_logistics_invoice.java	1838	87,524	162,361	682	/n
frm_logistics_add_colla.java	724	72,4	150,685	523	/n
frm_logistics_destroy_colla.java	466	46,6	74,131	266	/n
frm_logistics_main.java	115	23	19,411	56	/n
Mom.Mobile.Overview	3801	45,25	72,891	594	/n
Global_Class.java	2791	63,432	89,016	594	/n
settings.java	304	101,333	76,57	207	/n
frm_manual_update.java	240	60	48,286	142	/n
frm_Mobile_Overview.java	305	30,5	25,746	69	/n
DoubleProgressDialog.java	138	6,571	12,534	62	/n
about.java	23	11,5	6,5	18	/n
Mom.Mobile.Overview.Pricing	5324	27,729	45,808	390	/n
frm_pricing.java	1269	42,3	76,001	390	/n
frm_receipt.java	892	34,308	68,064	368	/n
frm_loading.java	921	30,7	42,996	247	/n
customer_details.java	163	27,167	38,186	111	/n
customer_card.java	528	31,059	24,216	96	/n
materials_details.java	129	25,8	32,695	90	/n
Mobile_pricingActivity.java	69	34,5	24,5	59	/n
frm_invoices_list.java	334	19,647	13,385	52	/n
frm_loadings_list.java	297	17,471	12,989	51	/n
material_card.java	334	19,647	13,311	51	/n
frm_receipts_list.java	334	18,556	12,518	51	/n
frm_reports.java	28	9,333	8,26	21	/n
frm_report_viewer.java	26	6,5	4,153	13	/n
Mom.Mobile.Overview.Crm	1156	60,842	85,613	369	/n
frm_crm_daily_program.java	937	78,083	100,708	369	/n
frm_crm_main.java	211	35,167	34,739	105	/n
frm_crm_online_reports.java	8	8	0	8	/n
Mom.Mobile.Overview.Agenda	1692	27,29	36,279	234	/n
frm_agenda_activities.java	452	41,091	65,392	234	/n
frm_agenda_contacts.java	428	47,556	45,204	166	/n
frm_agenda_category.java	198	19,8	12,991	43	/n
frm_agenda_main.java	45	22,5	15,5	38	/n
frm_agenda_situation.java	190	19	13,483	36	/n
frm_agenda_recources.java	189	18,9	13,597	36	/n
frm_agenda_group.java	190	19	13,653	36	/n
Mom.Mobile.Overview.Pricing.Reports	443	21,095	9,768	44	/n
Report_Receipts.java	154	22	9,856	44	/n
Report_MaterialsOnTruck.java	147	21	9,304	41	/n
Report_CustomersBalance.java	142	20,286	10,053	41	/n

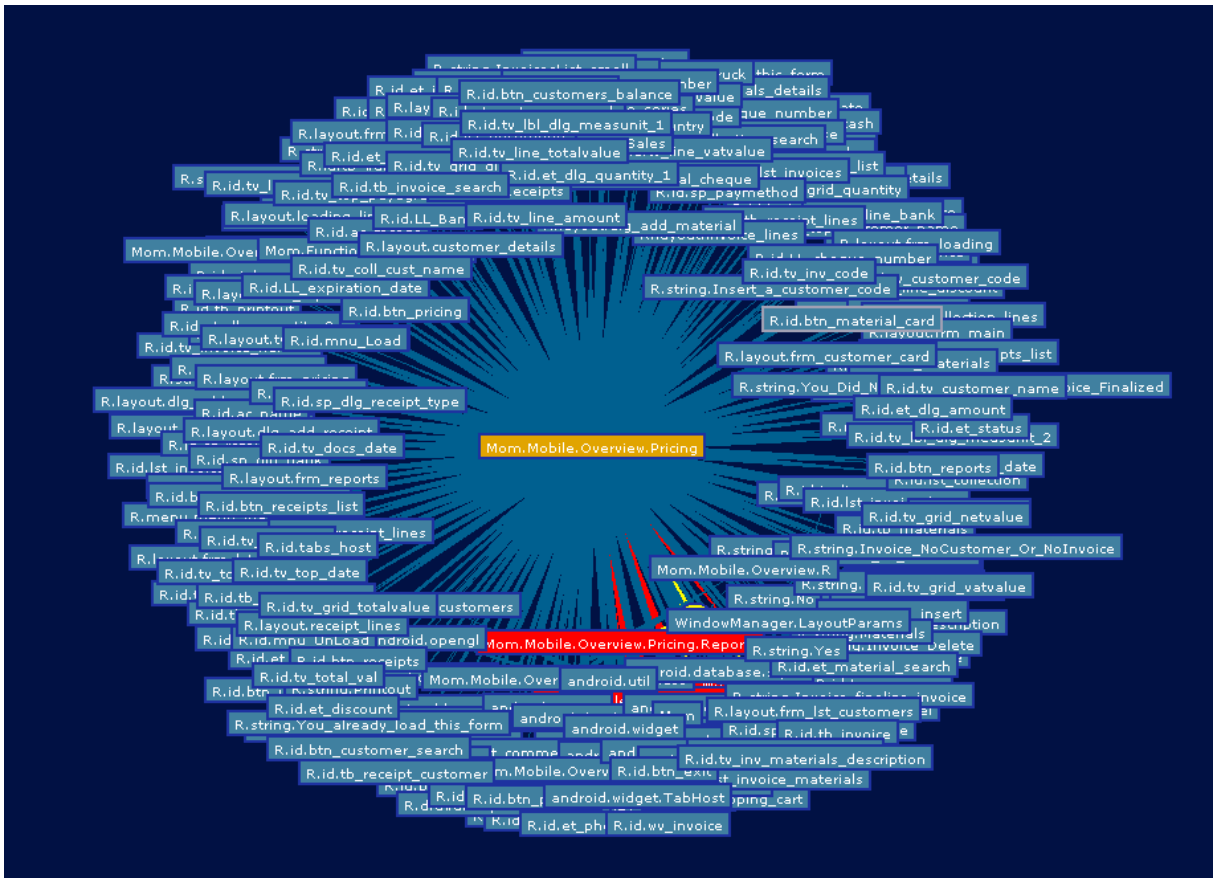
Εικόνα w Αποτελέσματα για το πλήθος γραμμών κώδικα (Μ.Ο. ανά μέθοδο)



Εικόνα x Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος.



Εικόνα y Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος και μιας συγκεκριμένης κλάσης.



Εικόνα 7 Γραφική αναπαράσταση των εξαρτήσεων μεταξύ των πακέτων του προγράμματος αλλά και όλων των στοιχείων / μεθόδων που περιλαμβάνονται σε όλες τις κλάσεις του προγράμματος λογισμικού.

ΒΙΒΛΙΟΓΡΑΦΙΑ - ΑΝΑΦΟΡΕΣ

Leslie J. Waguespack and William T. Schiano (2012), SCRUM Project Architecture and Thriving Systems Theory, 45th Hawaii International Conference on System Sciences.

Paul L. Bannerman, Emam Hossain and Ross Jeffery (2012), Scrum Practice Mitigation of Global Software Development Coordination Challenges: A Distinctive Advantage?, 45th Hawaii International Conference on System Sciences.

Ken H. Judy and Ilio Krumins-Beens (2008), Great Scrums Need Great Product Owners: Unbounded Collaboration and Collective Product Ownership, Proceedings of the 41st Hawaii International Conference on System Sciences.

Brent Barton and Evan Campbell (2007), Implementing a Professional Services Organization Using Type C Scrum, Proceedings of the 40th Hawaii International Conference on System Sciences.

Martin Blom (2010), Is Scrum and XP suitable for CSE Development?, International Conference on Computational Science, ICCS 2010.

Schaber Ken, Agile Project Management with SCRUM, Microsoft O'Reilly Media, Inc., 30 Νοε 2009.

Mike Beedle,Martine Devos,Yonat Sharon,Ken Schwaber and Jeff Sutherland, SCRUM: An extension pattern language for hyperproductive software development.

Pete Deemer, Gabrielle Benefield, Craig Larman and Bas Vodde, THE SCRUM PRIMER.

Jeff Sutherland (2001), Inventing and Reinventing SCRUM in Five Companies, CTO, PatientKeeper, Inc. 2001.

Viljan Mahnic, Slavko Drnovscek, Agile Software Project Management with Scrum.
Juyun Cho, Issues and challenges of agile software development with SCRUM, Issues in Information Systems VOL IX, No. 2, 2008.

Juyun Cho (2007), Distributed Scrum for Large-Scale and MissionCritical Projects, AMCIS 2007 Proceedings Americas Conference on Information Systems (AMCIS).

Σχετικοί Ιστότοποι

1. http://www.noop.nl/2008/07/the-definitive-list-of-software-development-methodologies.html/
2. http://en.wikipedia.org/wiki/Scrum_%28development%29
3. http://en.wikipedia.org/wiki/Software_engineering
4. http://en.wikipedia.org/wiki/History_of_software_engineering
5. http://en.wikipedia.org/wiki/Software_development_methodology
6. http://www.emilianosoldipmp.info/2012/09/scrum-metrics-when-to-use/
7. http://scrumorlando09.pbworks.com/w/page/15490672/Scrum%20Metrics%20and%20Myths
8. http://www.scrumshortcuts.com/blog/planning-metrics/scrum-metrics-and-reporting-measure-what-you-manage/
9. http://www.rapidscrum.com/Agile2010-ScrumMetrics.pdf
10. http://developer.rallydev.com/help/epic-progress-app
11. http://scruminc.com/tl_files/scrum_inc/documents/ScrumPapers.pdf