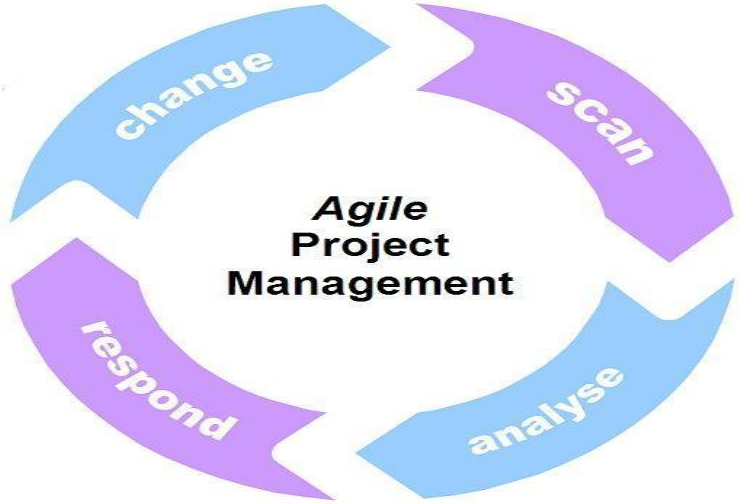
ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΙ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΔΙΑΧΕΙΡΙΣΗ ΕΥΕΛΙΚΤΩΝ ΕΡΓΩΝ



Της φοιτήτριας
Λημνιού Αικατερίνη
Αρ. Μητρώου: 02/2021

Επιβλέπων καθηγητής
Σφέτσος Παναγιώτης

Θεσσαλονίκη 2011

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία εκπονήθηκε στο Τμήμα Πληροφορικής του Αλεξάνδρειου Τεχνολογικού Εκπαιδευτικού Ιδρύματος Θεσσαλονίκης.

Το αντικείμενο της παρούσας πτυχιακής εργασίας είναι η Σχεδίαση και η Διαχείριση Ευέλικτων Έργων. Αναλύει και συγκρίνει ευέλικτες μεθοδολογίες και γίνεται μια εκτεταμένη έρευνα στην υιοθέτηση τους από διάφορες εταιρείες.

Υπεύθυνος και επιβλέπων αυτής της πτυχιακής ήταν ο κ.Π.Σφέτσος στον οποίο οφείλω ιδιαίτερες ευχαριστίες για την ευκαιρία που μου έδωσε να εργαστώ με ένα τόσο ενδιαφέρον θέμα καθώς και για την καθοδήγηση του.

Τέλος, θα ήθελα να ευχαριστήσω θερμά την οικογένεια μου για την συμπαράσταση τους σε κάθε μου βήμα όλα τα χρόνια της φοίτησης μου καθώς και την Τραχαλάκη Παναγιώτα για την πολύτιμη βοήθεια της.

ΠΕΡΙΛΗΨΗ

Σήμερα, η διαχείριση έργων λογισμικού γίνεται ολοένα και πιο σημαντική δεδομένου ότι ένα έργο χρειάζεται ένα οργανωμένο σχέδιο που πρέπει να ακολουθήσει. Οι ευέλικτοι μέθοδοι είναι μια καινοτόμος προσέγγιση που έρχεται από την επιστήμη της πληροφορικής, κατά κυριολεξία από το γνωστικό αντικείμενο της τεχνολογίας λογισμικού. Το δελεαστικό μέρος των ευέλικτων μεθόδων συνίσταται στα ακόλουθα γεγονότα. Πρώτον, η όλη διαδικασία μπορεί να ανταποκριθεί δυναμικά στις νέες συνθήκες, με άλλα λόγια, η διαδικασία της διαχείρισης είναι ευέλικτη. Οι ευέλικτες μέθοδοι χρησιμοποιούνται στον τομέα της μηχανικής λογισμικού τα τελευταία χρόνια, όπου το Μανιφέστο της ευέλικτης ανάπτυξης λογισμικού ήρθε περίπου το 2001. Παρά την δυναμικότητα τους δεν υπήρξαν γνωστές έξω από την κοινότητα της πληροφορικής.

Abstract

Nowadays, software project management is becoming more and more important since a project needs an organized plan to follow through. The agile methods are innovative approach coming in from the information science, strictly speaking from the subject field of software engineering. Principal inducements of these methods consist in following facts. Firstly, the whole process of the project management is accelerated. Secondly, the process can dynamically respond to new conditions, in other works, the process of the project management is flexible. The agile methods are in use in the field of software engineering as far as back as the Manifesto for Agile Software Development came into the existence by about 2001. Notwithstanding on their great potentiality, there are not well known informatics community.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1.ΕΙΣΑΓΩΓΗ

Μια από τις ταχέως εξελισσόμενες βιομηχανίες είναι αυτή της ανάπτυξης λογισμικού(software development). Τα προϊόντα που παράγονται πρέπει να ανταποκρίνονται συνεχώς και γρήγορα στις αλλαγές αλλά ταυτόχρονα να διατηρούν την υψηλή ποιότητα τους. Εν τω μεταξύ, παρά την πτώση του Internet, οι οικονομικές επιχειρήσεις εξακολουθούν να αλλάζουν με ταχείς ρυθμούς. Μ αυτό τον τρόπο τα καταστήματα τεχνολογίας πληροφοριών (information technology-IT) καθίστανται σε μια συνεχή μάχη συνύπαρξης τους με αυτό το ρυθμό αλλαγής. Αυτές οι αλλαγές έχουν οδηγήσει σε ένα αυξανόμενο ενδιαφέρον για τη μεθοδολογία ανάπτυξης(development methodology) ευέλικτου λογισμικού(agile software).

Οι ευέλικτες μεθοδολογίες(agile methodologies) όπως ο ακραίος προγραμματισμός(extreme programming-XP), ο Scrum και η ανάπτυξη με βάση τα χαρακτηριστικά(Feature Driven Development- FDD)προσπαθούν να μειώσουν το κόστος της αλλαγής μέσα από τη διαδικασία ανάπτυξης του λογισμικού. Παραδείγματος χάριν, ο ακραίος προγραμματισμός χρησιμοποιεί το ραγδαίο επαναληπτικό σχεδιασμό(iterative design) και την ανάπτυξη κυκλικής διάταξης(development encircling) έτσι ώστε να εξωθήσει την ανταλλαγή όλων των παραγόντων και τη μεταφορά δεδομένων ανωτέρας αξίας όσο το δυνατόν συντομότερα. Επιπλέον, ο συνεχής συστηματικός έλεγχος που αποτελεί μέρος του ακραίου προγραμματισμού διασφαλίζει την υψηλή ποιότητα μέσω της ανίχνευσης ελαττωμάτων και της αναλυτικότητας της.

Παρά την πρώιμη επιτυχία με τις ευέλικτες μεθοδολογίες, υπάρχει ένας αριθμός παραγόντων ο οποίος εμποδίζει την εκτεταμένη αποδοχή τους. Οι οπαδοί της ευέλικτης μεθοδολογίας συχνά αντιμετωπίζουν δυσκολίες στην απόκτηση υποστήριξης ως προς την ανάπτυξη εφαρμογών(application development). Αυτές οι μεθοδολογίες απαιτούν την ύπαρξη επικεφαλών και χρηστών αντιστοίχως για να μπορέσουν να αλλάξουν τον τρόπο εργασίας και σκέψης τους. Για παράδειγμα, οι πρακτικές του ακραίου προγραμματισμού όπως ο σχεδιασμός του πρώτου ελέγχου, η συνεχής ολοκλήρωση καθώς και η παρουσία πελάτη μπορούν να φαντάζουν ως τρομακτικές αλλαγές ως προς την υλοποίηση τους. Εκτός αυτού, αυτές οι μεθοδολογίες τείνουν να

είναι αναπτυξιακές και φαίνονται να αποτάσσουν το ρόλο της διαχείρισης για την επιβεβαίωση της επιτυχίας.

Μια ισχυρή διαχείριση είναι απολύτως κρίσιμη ως προς την επιτυχημένη υιοθέτηση και εφαρμογή των ευέλικτων μεθοδολογιών. Αλλά υπάρχει μια έλλειψη συσχέτισης ανάμεσα στις μεθοδολογίες και τα εργαλεία του παραδοσιακού χειρισμού προγραμμάτων καθώς και νεώτερων ευέλικτων μεθοδολογιών. Επιπλέον αυτή η έλλειψη συσχετισμού είναι ενδεικτική ενός βαθύτερου προβλήματος –διαφορές σε βασικές υποθέσεις σχετικά με την αλλαγή, τον έλεγχο, τη σειρά, την οργάνωση καθώς και τους ανθρώπους και την γενική προσέγγιση ως προς την επίλυση προβλημάτων.

Η παραδοσιακή θεωρία διαχείρισης αξιώνει το εξής:

- Χρειάζονται δογματικές διαδικασίες για να ρυθμίσουν την αλλαγή
- Οι ιεραρχικές και οργανωτικές δομές αποτελούν τα μέσα για την καθιέρωση της διαρρύθμισης.
- Τα αυξημένα αποτελέσματα ελέγχου στην αυξημένη διάταξη των οργανισμών πρέπει να αποτελούνται από δογματικές και στατικές ιεραρχίες.
- Οι εργαζόμενοι είναι ανταλλάξιμα μέλη στον οργανωτικό μηχανισμό.
- Τα προβλήματα επιλύονται πρωταρχικά μέσω της υπεραπλούστευσης της ταξινόμησης και της κατανομής.
- Το ερευνητικό έργο και τα ρίσκα είναι επαρκώς προβλέψιμα έτσι ώστε να επιτευχθούν μέσω ενός σύνθετου ανερχόμενου σχεδιασμού.

Στα πλαίσια αυτής της πτυχιακής επιχειρείται η έρευνα πάνω στο σχεδιασμό και τη διαχείριση των Ευέλικτων Έργων Λογισμικού. Η έρευνα θα μελετήσει το θεωρητικό υπόβαθρο, θα κάνει μία ανασκόπηση της βιβλιογραφίας και των εξελίξεων στον τομέα και θα ασχοληθεί εκτεταμένα με την υλοποίηση μέσω των υπαρχόντων εργαλείων και ιδιαίτερα του πιο διαδεδομένου Version One.

Στο πρώτο μέρος θα παρουσιασθούν και θα αναλυθούν δύο από τις ποιο βασικές ευέλικτες μεθοδολογίες: ο ακραίος προγραμματισμός(extreme programming-XP)και ο Scrum θα καθοριστούν και θα ταξινομηθούν περιγράφοντας για καθεμίαν τις διαδικασίες(procedures), τους ρόλους(roles) και τις πρακτικές(practices) που τις χαρακτηρίζουν. Στην συνέχεια θα

παρουσιαστεί η έρευνα που έγινε για την εφαρμογή των ευέλικτων μεθοδολογιών στον δημόσιο και ιδιωτικό τομέα παγκοσμίως. Στο επόμενο μέρος θα παρουσιασθούν και θα αναλυθούν κάποια από τα ποιο γνωστά ευέλικτα εργαλεία(agile tools) που χρησιμοποιούν οι μεθοδολογίες ακραίος προγραμματισμός και Scrum. Σ' αυτό το σημείο και πριν ακόμα γίνει αναφορά στις ευέλικτες μεθοδολογίες θα παρουσιαστούν κάποιες παραδοσιακές μεθοδολογίες στις οποίες βασίζονται οι νέες μεθοδολογίες. Αυτές οι μεθοδολογίες είναι:

1.1 Γραμμικό μοντέλο(Linear Model-LM) ή μοντέλο "καταρράκτη"(waterfall model)

Το μοντέλο του καταρράκτη είναι μια διαδοχική διαδικασία ανάπτυξης, στην οποία η ανάπτυξη θεωρείται πως διατρέχει σταθερά προς τα κάτω(όπως σε έναν καταρράκτη). Αποτελείται από τις φάσεις της ανάλυσης των απαιτήσεων (requirements analysis), του σχεδίου (plan), της εφαρμογής (application), της επικύρωσης(validation), της ολοκλήρωσης (completion) και της συντήρησης (maintenance). Για να ακολουθήσει κάποιος το πρότυπο του καταρράκτη, προχωρά από τη μια φάση στην επόμενη κατά τρόπο καθαρά διαδοχικό. Για παράδειγμα, πρώτα ολοκληρώνονται οι προδιαγραφές των απαιτήσεων οι οποίες δεν τίθενται σε περαιτέρω διαπραγματεύσεις. Όταν οι απαιτήσεις ολοκληρωθούν πλήρως, προχωράμε με την σχεδίαση του συστήματος. Το εν λόγω λογισμικό σχεδιάζεται και ένα προσχέδιο του δίνεται στους προγραμματιστές για να το ακολουθούν – αυτό το προσχέδιο πρέπει να είναι ένα σχέδιο για την εφαρμογή των απαιτήσεων που έχουν ήδη παγιωθεί. Όταν το σχέδιο ολοκληρωθεί πλήρως, μια εφαρμογή του σχεδίου γίνεται από τους προγραμματιστές. Προς τα τελευταία στάδια της φάσης εφαρμογής, τα χωριστά τμήματα λογισμικού συνδυάζονται για να ολοκληρωθεί η νέα λειτουργία. Το πρότυπο του καταρράκτη υποστηρίζει ότι κάποιος μπορεί να κινηθεί προς μια φάση μόνο όταν ολοκληρώνεται η προηγούμενη φάση της. Σύμφωνα με τις βασικές αρχές του μοντέλου κάθε έργο διαιρείται σε διαδοχικές φάσεις, με κάποια επικάλυψη και κάποιες αναθεωρήσεις να είναι αποδεκτές μεταξύ των φάσεων. Έμφαση δίνεται στον προγραμματισμό, στα χρονικά προγράμματα, τις προβλεπόμενες ημερομηνίες, τους προϋπολογισμούς και την δημιουργία ενός ολόκληρου συστήματος

συγχρόνως. Ο αυστηρός έλεγχος διατηρείται κατά την διάρκεια της ζωής του προγράμματος μέσω της χρήσης εκτενούς γραπτής τεκμηρίωσης καθώς επίσης και μέσω των επίσημων αναθεωρήσεων που έχουν την έγκριση του πελάτη.

1.2 Το μοντέλο δημιουργίας πρωτοτύπου(Prototyping) ή επαναληπτικό μοντέλο(iterative model)

Το μοντέλο δημιουργίας πρωτοτύπων είναι μια τεχνική ανάπτυξης λογισμικού κατά την οποία δημιουργούνται ελλιπείς εκδόσεις του συνολικού προγράμματος που θα αναπτυχθεί. Ένα πρωτότυπο μιμείται τα χαρακτηριστικά ή μερικές πτυχές των χαρακτηριστικών γνωρισμάτων του ενδεχόμενου προγράμματος και μπορεί να είναι απολύτως διαφορετικό από την τελική εφαρμογή. Ο πραγματικός σκοπός ενός πρωτότυπου είναι να επιτρέψει στους χρήστες του λογισμικού να αξιολογήσουν τις προτάσεις των υπεύθυνων για την ανάπτυξη. Αυτό πραγματοποιείται με το να δοκιμάζουν στην πράξη ένα προσχέδιο του προϊόντος. Η διαμόρφωση πρωτοτύπου μπορεί επίσης να χρησιμοποιηθεί από τους τελικούς χρήστες για να περιγράψουν τις απαιτήσεις που οι υπεύθυνοι για την ανάπτυξη δεν έχουν εξετάσει. Έτσι «ο έλεγχος του πρωτοτύπου» μπορεί να είναι ένας βασικός παράγοντας στην εμπορική σχέση μεταξύ των προμηθειών της λύσης και των πελατών τους.

Στο μοντέλο της δημιουργίας πρωτοτύπου μετά την αρχική διεύρυνση ξεκινά ένας κύκλος προσδιορισμού των προδιαγραφών(cycle definition of standards), σχεδιασμού του συστήματος(system design), κωδικοποίησης (coding) και δοκιμών(test) κατά τον οποίο δημιουργείται το πρωτότυπο. Μετά την δημιουργία και την αξιολόγηση του πρωτοτύπου υλοποιείται το λογισμικό. Στην τελευταία φάση μένει μόνο η συντήρηση του λογισμικού. Για πολλούς η δημιουργία πρωτοτύπου δεν είναι μια αυτόνομη, πλήρης μεθοδολογία ανάπτυξης, αλλά μάλλον μια προσέγγιση στο χειρισμό επιλεγμένων τμημάτων μιας μεγαλύτερης, παραδοσιακότερης μεθοδολογίας ανάπτυξης όπως η επαυξητική(incremental), ή σπειροειδής(spiral). Στην ανάπτυξη λογισμικού με αυτή την μέθοδο γίνονται προσπάθειες να μειωθεί ο εγγενής κίνδυνος ενός έργου με το σπάσιμο του σε μικρότερα τμήματα και την παροχή μεγαλύτερης προσαρμογής στην αλλαγή κατά τη διάρκεια της διαδικασίας ανάπτυξης.

Επιπλέον ο χρήστης συμμετέχει σε όλη τη διαδικασία ανάπτυξης, γεγονός που αυξάνει την πιθανότητα της αποδοχής της τελικής εφαρμογής. Τα μικρής κλίμακας πρωτότυπα του συστήματος αναπτύσσονται μετά από μια επαναληπτική διαδικασία τροποποίησης έως ότου εξελίσσεται το πρωτότυπο για να καλύψει τις απαιτήσεις των χρηστών. Ενώ τα περισσότερα πρωτότυπα αναπτύσσονται με την προσδοκία ότι θα απορριφθούν, είναι δυνατό σε μερικές περιπτώσεις να εξελιχθεί το πρωτότυπο στο ίδιο το σύστημα εργασίας. Η μέθοδος της δημιουργίας πρωτοτύπων θεωρεί βασική την κατανόηση του θεμελιώδους επιχειρηματικού προβλήματος για να αποφευχθεί η λύση του λανθασμένου προβλήματος.

1.3 Ο επαυξητικός(Incremental)συνδυασμός γραμμικού και επαναληπτικού τύπου μοντέλου

Διάφορες μεθοδολογίες είναι αποδεκτές για το συνδυασμό των γραμμικών και επαναληπτικών μεθοδολογιών ανάπτυξης συστημάτων. Ο αρχικός στόχος κάθε μιας είναι να μειώσει τον έμφυτο κίνδυνο του προγράμματος με το σπάσιμο ενός προγράμματος σε μικρότερα τμήματα. Επίσης παρέχει μεγαλύτερη ευκολία αλλαγής κατά τη διάρκεια της διαδικασίας ανάπτυξης.

Οι βασικές αρχές της επαυξητικής ανάπτυξης είναι:

- μια σειρά μίνι-καταρρακτών ολοκληρώνονται για ένα μικρό μέρος του συστήματος πριν προχωρήσει η ομάδα στην επόμενη επαύξηση,
- οι γενικές απαιτήσεις καθορίζονται έτσι ώστε να οδηγήσουν στην ανάπτυξη μίνι-καταρρακτών των μεμονωμένων αυξήσεων του συστήματος ή
- η αρχική σύλληψη του λογισμικού, η ανάλυση των απαιτήσεων και ο κεντρικός σχεδιασμός του πυρήνα της αρχιτεκτονικής και του συστήματος καθορίζεται χρησιμοποιώντας την προσέγγιση καταρράκτη.

1.4 Το σπειροειδής μοντέλο(Spiral):γραμμικός και επαναληπτικός τύπος πλαισίου μοντέλου

Το σπειροειδές πρότυπο καθορίστηκε από το Barry Boehm το 1988 στο άρθρο ‘Ένα σπειροειδές πρότυπο της ανάπτυξης και βελτίωσης λογισμικού’. Αυτό το πρότυπο δεν ήταν το πρώτο που ανέφερε την επαναληπτική ανάπτυξη, αλλά ήταν το πρώτο πρότυπο που εξηγεί γιατί η επανάληψη κάνει τη διαφορά.

Όπως αρχικά προέβλεπε, οι επαναλήψεις ήταν τυπικά της τάξης των 6 μηνών ή 2 ετών σε διάρκεια. Κάθε φάση αρχίζει με έναν στόχο σχεδίου και τελειώνει με τον πελάτη να αναθεωρεί την πρόοδο. Η ανάλυση και η εφαρμογή γίνονται σε κάθε φάση του προγράμματος, με στόχο το τέλος του προγράμματος.

Οι βασικές αρχές του μοντέλου είναι

-η εστίαση στην αξιολόγηση του κινδύνου και στην ελαχιστοποίηση του προγράμματος. Αυτό γίνεται με το σπάσιμο ενός προγράμματος σε μικρότερα τμήματα και την παροχή περισσότερης ευκολίας αλλαγής κατά τη διάρκεια της διαδικασίας ανάπτυξης. Καθώς επίσης και την παροχή της ευκαιρίας να αξιολογηθούν οι κίνδυνοι και να γίνει η εκτίμηση της συνέχειας προγράμματος σε όλο τον κύκλο ζωής(life cycle) του προϊόντος.

-«Κάθε κύκλος περιλαμβάνει μια πρόοδο μέσω της ίδιας ακολουθίας βημάτων, για κάθε τμήμα του προϊόντος και για κάθε ένα από τα επίπεδα επεξεργασίας του από ένα γενικό έγγραφο της επιθυμητής λειτουργίας έως την κωδικοποίηση κάθε μεμονωμένου προγράμματος.»(Boehm 1988)

-Κάθε ταξίδι πάνω στην σπείρα διαπερνά τέσσερα βασικά τεταρτημόρια:

1.καθορισμός των στόχων, εναλλακτικές λύσεις και περιορισμοί της επανάληψης

2.αξιολογήση των εναλλακτικών λύσεων, προσδιορισμός και επίλυση των κινδύνων

3.αναπτύξη και έλεγχος των προϊόντων της επανάληψης και

4.σχεδιασμός της επόμενης επανάληψης.

-«Αρχίστε κάθε κύκλο με τον προσδιορισμό των συμμετεχόντων στο πρόγραμμα και των όρων επιτυχίας και τελειώστε κάθε κύκλο με την αναθεώρηση και την δέσμευση των συμμετεχόντων για τον επόμενο κύκλο(Boehm, 1988).

1.5 Γρήγορη ανάπτυξη εφαρμογής(Rapid Application Development RAD): Επαναληπτικός τύπος μοντέλου

Η γρήγορη ανάπτυξη εφαρμογής είναι μια μεθοδολογία ανάπτυξης λογισμικού, η οποία περιλαμβάνει την επαυξητική ανάπτυξη και την κατασκευή πρωτοτύπων. Η γρήγορη ανάπτυξη εφαρμογής είναι ένας όρος που χρησιμοποιήθηκε αρχικά για να περιγράψει μια διαδικασία ανάπτυξης λογισμικού που εισάγε κατά τη διάρκεια της δεκαετίας του ’80 στην IBM ο James Martin και την τυποποίησε τελικά με την έκδοση ενός βιβλίου το 1991.

Οι βασικές αρχές της μεθόδου όπως περιγράφονται από τον Martin(1991) είναι:

-Ο βασικός στόχος είναι η γρήγορη ανάπτυξη και παράδοση ενός υψηλής ποιότητας συστήματος με σχετικά χαμηλό κόστος.

-Προσπάθεια μείωσης του εγγενής κίνδυνου του προγράμματος με το σπάσιμο του σε μικρότερα τμήματα και την παροχή μεγαλύτερης ευκολίας για αλλαγή κατά τη διάρκεια της διαδικασίας ανάπτυξης.

-Στόχος παραγωγής υψηλής ποιότητας συστημάτων γρήγορα, πρώτιστα μέσω της χρήσης της επαναληπτικής διαμόρφωσης πρωτοτύπου(σε οποιοδήποτε στάδιο ανάπτυξης) της ενεργού συμμετοχής των χρηστών και των αυτοματοποιημένων εργαλείων ανάπτυξης.

-Έμφαση δίνεται στην πραγματοποίηση των επιχειρησιακών αναγκών, ενώ η τεχνολογική τελειότητα είναι μικρότερης σπουδαιότητας.

-Ο έλεγχος δίνει προτεραιότητα στην ανάπτυξη και τον καθορισμό των προθεσμιών παράδοσης ή των χρονικών περιθωρίων. Εάν το πρόγραμμα αρχίζει να ολισθαίνει, η έμφαση που δίνεται είναι στη μείωση των απαιτήσεων για να χωρέσουν στα χρονικά περιθώρια και όχι στην μετακίνηση της προθεσμίας.

- Γενικά περιλαμβάνει την από κοινού ανάπτυξη εφαρμογών, όπου οι χρήστες περικλείονται έντονα στο σχεδιασμό συστημάτων.

-Η ενεργός συμμετοχή χρηστών είναι επιτακτική.

-Παράγει επαναληπτικά το λογισμικό, σε αντιδιαστολή με την χρήση ενός πρωτότυπου που δεν χρησιμοποιείται

-Παράγει την απαραίτητη τεκμηρίωση για να διευκολύνει τη μελλοντική ανάπτυξη και τη συντήρηση.

-Οι τυποποιημένες τεχνικές ανάλυσης και σχεδιασμού συστημάτων μπορούν να εναρμονιστούν σε αυτό το μοντέλο.

Η παραδοσιακή διαχείριση έργου(B.Hass, 2007) περιλαμβάνει πολύ πειθαρχημένο και σκόπιμο σχεδιασμό και μεθόδους ελέγχου. Με την προσέγγιση αυτή, διαφορετικές φάσεις του κύκλου ζωής του έργου είναι εύκολα αναγνωρίσιμες. Οι εργασίες ολοκληρώνονται η μία μετά την άλλη σε μια τακτική ακολουθία, απαιτώντας ένα σημαντικό μέρος του έργου που πρέπει να προγραμματιστεί. Η παραδοσιακή διαχείριση έργου προϋποθέτει ότι τα γεγονότα που επηρεάζουν το έργο είναι προβλέψιμα και τα εργαλεία για τις δραστηριότητες είναι πλήρως κατανοητά. Επιπλέον, με την παραδοσιακή διαχείριση του έργου όταν μία φάση ολοκληρωθεί δεν πρέπει να επανεξεταστεί. Τα πλεονεκτήματα της προσέγγισης αυτής είναι ότι καθορίζει τα βήματα για την ανάπτυξη και τονίζει τη σημασία των απαιτήσεων. Σήμερα οι επιχειρηματικές διαδικασίες είναι πιο περίπλοκες και διασυνδεδεμένες από ποτέ. Επιπλέον, απορρίπτουν τις παραδοσιακές οργανωτικές δομές και περιλαμβάνουν πολύπλοκες κοινωνίες των συμμαχιών με στρατηγικούς προμηθευτές, εξωτερικούς πωλητές, τα δίκτυα των πελατών, την εταιρική σχέση και τους ανταγωνιστές. Μέσα από αυτές τις συμμαχίες, οι οργανισμοί είναι σε θέση να αντιμετωπίσουν τις πιέσεις της περιόδου πρωτοφανών αλλαγών, τον παγκόσμιο ανταγωνισμό, την συμπίεση του χρόνου διοχέτευσης στην αγορά, τις ταχέως μεταβαλλόμενες τεχνολογίες και την αυξημένη πολυπλοκότητα σε κάθε μορφή. Για χρόνια οι οικονομολόγοι προειδοποιούν ότι η επιτυχία σε μια παγκόσμια αγορά εξαρτάται από την ικανότητα να παράγει μικρές ποσότητες των προϊόντων που προσαρμόζονται σε ένα αυστηρό χρονοδιάγραμμα για να ικανοποιήσει την αυξανόμενη ζήτηση στις αναδυόμενες αγορές. Η βελτίωση αυτών των επιδόσεων είναι ο στόχος για κάθε οργανισμό. Γι αυτό το λόγο η παραδοσιακή διαχείριση του έργου είναι μάλλον αναποτελεσματική και είναι ώρα να εξεταστούν άλλες μέθοδοι για τον σχεδιασμό και την υλοποίηση των έργων.

Μια εναλλακτική προσέγγιση, η Ευέλικτη Διαχείριση Έργων(agile project management) εμφανίζετε στον κλάδο(B.Hass,2007). Η Ευέλικτη Διαχείριση Έργων είναι μια ιδιαίτερα επαναληπτική και σταδιακή διαδικασία, όπου οι προγραμματιστές και οι ενδιαφερόμενοι φορείς του έργου συνεργάζονται

ενεργά για την κατανόηση του χώρου, για να προσδιορίσουν τι πρέπει να κατασκευαστεί και να δώσουν προτεραιότητα στην λειτουργικότητα. Η ευέλικτη προσέγγιση αποτελείται από γρήγορο επαναληπτικό σχεδιασμό και κύκλους ανάπτυξης, επιτρέποντας μια ομάδα έργου να αξιολογεί διαρκώς την εξέλιξη του προϊόντος και να λαμβάνει άμεση ανατροφοδότηση από τους χρήστες. Η ομάδα μαθαίνει και βελτιώνει το προϊόν, καθώς και τις μεθόδους εργασίας τους από κάθε διαδοχικό κύκλο. Μετά από ένα βελτιωμένο σχεδιασμό τον καθορισμό των απαιτήσεων και τη φάση του σχεδιασμού η λύση είναι να ολοκληρωθεί το έργο. Οι επαναλήψεις στον πιο λεπτομερή προγραμματισμό, οι προδιαγραφές, ο σχεδιασμός και η δοκιμή πραγματοποιούνται σε κύματα. Η προσέγγιση αυτή επιτρέπει την άμεση τροποποίηση του προϊόντος με τις απαιτήσεις να έρχονται σε θέα. Η ευέλικτη διαχείριση έργου πραγματοποιείται σε συνεργασία με μια μικρή συστεγαζόμενη ομάδα. Αυτή η ομάδα συνήθως αποτελείται από δύο προγραμματιστές που γράφουν κώδικα σε ζευγάρια(για τον ποιοτικό έλεγχο), τον πελάτη/τελικό χρήστη, τον αναλυτή της επιχείρησης και τον διαχειριστή του έργου. Υπάρχουν επίσης εκτελεστικά στελέχη(στελέχη επιχειρήσεων και διευθυντές του καταστήματος ανάπτυξης), οι οποίοι ενδιαφέρονται για τους προϋπολογισμούς και την απόδοση των επενδύσεων και τους ανθρώπινους πόρους. Καθένα από αυτούς τους ανθρώπινους πόρους έχει την δική του συμμετοχή σε ένα ευέλικτο έργο. Το έργο ολοκληρώνεται μέσα από μια σειρά συνεδριάσεων όπου η ομάδα γράφει τον κώδικα στην συνέχεια τον δοκιμάζει και επαναλαμβάνει την διαδικασία. Υπάρχει ελάχιστη τεκμηρίωση, καθώς η ομάδα στηρίζεται σχεδόν αποκλειστικά σε άτυπη εσωτερική επικοινωνία. Η ομάδα της ευέλικτης διαχείρισης έργου προσδιορίζει και δίνει προτεραιότητα στα χαρακτηριστικά βάση της αξίας των επιχειρήσεων. Αυτή η προσέγγιση λειτουργεί εάν η λύση μπορεί να παραδοθεί σταδιακά στον πελάτη. Αν αυτό δεν είναι δυνατό, τα χαρακτηριστικά εξακολουθούν να ανακατασκευάζονται και στη συνέχεια εντάσσονται στην πρώτη έκδοση του συστήματος. Οι προγραμματιστές πρέπει να είναι πρόθυμοι να δουλέψουν σε ομάδες, να είναι σε θέση να χειρίζονται τις συνεχείς αλλαγές και να είναι δημιουργικοί ώστε να λύσουν τα προβλήματα. Οι δοκιμαστές(tester) πρέπει να συνεργάζονται στενά με τους προγραμματιστές. Οι δοκιμαστές εστιάζουν στο σύστημα και τις λειτουργικές δοκιμές. Ίσως χρειαστεί να είναι πιο ικανοί ως προγραμματιστές

για την αυτοματοποίηση του συστήματος και τις λειτουργικές δοκιμές και την ενσωμάτωση με το αυτοματοποιημένο πλαίσιο δοκιμών. Αυτό μπορεί να αντιπροσωπεύει ένα διαφορετικό σύνολο δεξιοτήτων. Υπάρχουν δύο βασικοί ρόλοι για τον επικεφαλής του έργου στην ανάπτυξη λογισμικού, ο διαχειριστής του έργου και η επικεφαλής ομάδα. Καθένας έχει διαφορετικό σύνολο προκλήσεων. Οι ομάδες περιλαμβάνουν έμπειρο προσωπικό με μεγάλες ευθύνες. Η ομάδα με την προθυμία της επιτρέπει στα μέλη της να παίρνουν πρωτοβουλίες. Οι διαχειριστές του έργου είναι υπεύθυνοι για την παρακολούθηση της προόδου και την λήψη επιχειρηματικών αποφάσεων. Ο ρόλος του πελάτη στην ευέλικτη διαχείριση έργου είναι σημαντικός. Στην παραδοσιακή διαχείριση ο πελάτης συνήθως συμμετέχει στην αρχή και στο τέλος του έργου. Στην ευέλικτη διαχείριση ο πελάτης συμμετέχει πολύ πιο συχνά και με μεγαλύτερη επιρροή. Η ευέλικτη διαχείριση στηρίζεται σε μεγάλο βαθμό στην συνεργασία και την επικοινωνία. Ένας ισχυρογνώμων προγραμματιστής καθώς και ένας πελάτης που δεν ασχολείται με την ομάδα μπορούν να καταστρέψουν την συνεργατική φύση της ομάδας. Οι ευέλικτες διαδικασίες ενθαρρύνουν συχνά τις αρχές που αλλάζουν δραματικά την διαδικασία. Βασικό στην διαδικασία ανάπτυξης είναι το ενδιαφέρον για την ανατροφοδότηση, την μινιμαλιστική ανάπτυξη, την αξιολόγηση κώδικα και την συνεχή ολοκλήρωση. Η ανατροφοδότηση είναι μια διαδικασία λήψης του κώδικα και βελτίωσης του χωρίς να χάνει την λειτουργικότητα του. Ο κώδικας μπορεί να βελτιωθεί για την αναγνωσιμότητα, την συντηρησιμότητα και την απόδοση του. Στην ανατροφοδότηση ο κώδικας πρέπει να περάσει από όλες τις δοκιμές και να προσαρμοστεί με όλες τις καθορισμένες συμβάσεις και μετά να ξαναγραφτεί. Η συνεχής ολοκλήρωση είναι μια διαδικασία κατά την οποία το σύστημα δοκιμάζεται συχνά. Οι προγραμματιστές πρέπει να γράψουν ένα ολοκληρωμένο σύνολο δοκιμών που πρέπει να χρησιμοποιήσουν. Έπειτα πρέπει να αφιερώσουν λίγο χρόνο για να τις ενσωματώσουν και να τις δοκιμάσουν στον κώδικα. Οι ευέλικτοι μέθοδοι εφαρμόζονται πλέον σε έργα όπου οι απαιτήσεις είναι ασαφής, δεδομένου ότι επιδιώκουν να φιλοξενήσουν τις εύκολες αλλαγές. Το χρονικό διάστημα του έργου είναι ένα σημαντικό χαρακτηριστικό που επηρεάζει την αποτελεσματικότητα των ευέλικτων μεθόδων. Τα προϊόντα θα χρειαστούν πολύ χρόνο για να αναπτυχθούν και οι κίνδυνοι θα πρέπει να μετριάσουν την ευέλικτη μέθοδο που θα

χρησιμοποιηθεί. Τα μεγάλα σε εξέλιξη έργα αποτελούν επίσης πρόκληση, δεδομένου ότι τείνουν να είναι μεγάλα στη φύση τους με μεγάλο αριθμό χαρακτηριστικών και δυνατοτήτων. Αυτό μπορεί να οδηγήσει σε δυσκολίες στην ιεράρχηση των εργασιών. Μια άλλη σημαντική πτυχή της μακράς διάρκειας ζωής έργων είναι ότι τείνουν να έχουν μεγάλες διάρκειες συντήρησης.

Το 2001 εξέχοντες προγραμματιστές λογισμικού συγκλήθηκαν για να καταλήξουν σε ομοφωνία σχετικά με το πώς η βιομηχανία ανάπτυξης λογισμικού θα μπορούσε να παράγει καλύτερα αποτελέσματα. Η συνάντηση αυτή είχε ως αποτέλεσμα το Μανιφέστο για την Ευέλικτη Ανάπτυξη Λογισμικού το οποίο ορίζει ότι η “ύψιστη προτεραιότητα είναι η ικανοποίηση του πελάτη με την έγκαιρη και συνεχή προβολή των πολύτιμων λογισμικών”.

Τα μανιφέστο για την ευέλικτη ανάπτυξη λογισμικού δηλώνει:

«Ανακαλύπτουμε καλύτερους τρόπους ανάπτυξης λογισμικού με το να αναπτύσσουμε λογισμικό αλλά και με το να βοηθήσουμε άλλους που κάνουν το ίδιο. Μέσω αυτής της εργασίας έχουμε καταλήξει στο να εκτιμούμε περισσότερο:

- **Τα άτομα και τις αλληλεπιδράσεις** από τις διαδικασίες και τα εργαλεία
- **Το εν λειτουργία λογισμικό** από την περιεκτική τεκμηρίωση του
- **Την συνεργασία με τους πελάτες** από την διαπραγμάτευση συμβάσεων συμβολαίων
- **Την ανταπόκριση στην αλλαγή** από την εκτέλεση ενός σχεδίου

Αυτό που φαίνεται είναι ότι ενώ υπάρχει αξία στα στοιχεία στο δεξί μέρος των παραπάνω προτάσεων εκτιμούνται τα στοιχεία στο αριστερό περισσότερο.»(Beck et al., 2003)

Αρχικά, το ευέλικτο μοντέλο υπογραμμίζει την αλληλεγγύη μεταξύ των υπεύθυνων για την ανάπτυξη λογισμικού και το ανθρώπινο ρόλο που απεικονίζουν οι συμβάσεις σε αντιδιαστολή με τις θεσμοποιημένες διαδικασίες και τα εργαλεία ανάπτυξης. Στις υπάρχουσες ευέλικτες πρακτικές, αυτό φανερώνει τις στενές σχέσεις των ομάδων, στο στενό εργασιακό περιβάλλον στις ρυθμίσεις και τις άλλες διαδικασίες που προωθούν το ομαδικό πνεύμα.

Δεύτερον, το μανιφέστο υποστηρίζει πως ο πρωταρχικός στόχος της ομάδας είναι να παράγει συνεχώς δοκιμασμένο λειτουργικό λογισμικό. Οι νέες εκδόσεις του λογισμικού παράγονται ανά τακτά διαστήματα σε μερικές προσεγγίσεις ακόμα και ανά μία ώρα ή καθημερινά, αλλά η πιο συνήθης πρακτική είναι διμηνιαία ή μηνιαία. Οι υπεύθυνη για την ανάπτυξη ωθούνται για να κρατήσουν τον κώδικα απλό εύκολα αναγνώσιμο και όσο πιο τεχνικά προηγμένο γίνεται, ελαττώνοντας κατά συνέπεια το φορτίο της τεκμηρίωσης σε ένα κατάλληλο επίπεδο.

Τρίτον, η σχέση και η συνεργασία μεταξύ των υπευθύνων για την ανάπτυξη και των πελατών είναι προτιμότερη σε σύγκριση με τις ακριβείς συμβάσεις, αν και η σημασία μιας καλώς σχεδιασμένης σύμβασης αυξάνεται αναλογικά με το μέγεθος του λογισμικού. Η ίδια η διαδικασία διαπραγμάτευσης πρέπει να εξεταστεί ως μέσο επίτευξης και διατήρησης μιας βιώσιμης σχέσης. Από επιχειρηματικής άποψης, η ανάπτυξη λογισμικού εστιάζει στο να παραδίδεται αξία αμέσως στο έργο από την στιγμή της έναρξης του, μειώνοντας κατά συνέπεια τους κινδύνους μη-συμμόρφωσης με τη σύμβαση.

Τέταρτον, η ομάδα ανάπτυξης που περιλαμβάνει και τους υπεύθυνους για την ανάπτυξη λογισμικού και τους αντιπρόσωπους των πελατών, πρέπει να είναι καλά πληροφορημένοι, ικανοί και εξουσιοδοτημένοι να εξετάζουν τις πιθανές ανάγκες που προκύπτουν κατά τη διάρκεια του κύκλου ζωής της διαδικασίας ανάπτυξης(Lifecycle Process Development) και να την αναδιαμορφώνουν αντίστοιχα. Αυτό σημαίνει ότι οι συμμετέχοντας πρέπει να είναι έτοιμοι να κάνουν τις αλλαγές και ότι υπάρχουσες συμβάσεις διαμορφώνονται με τέτοια εργαλεία ώστε να μπορούν να υποστηρίζουν και να επιτρέπουν τις αλλαγές.

Στη συνέχεια παρουσιάζονται οι αρχές του ευέλικτου μανιφέστο όπως αυτές διατυπώθηκαν από τους δημιουργούς του.

«Ακολουθούμε αυτές τις αρχές:

1. Η πιο υψηλή προτεραιότητα μας είναι να ικανοποιήσουμε τον πελάτη μέσω της γρήγορης και συνεχούς παράδοσης του «πολύτιμου» λογισμικού.
2. Καλωσορίζουμε τις μεταβαλλόμενες απαιτήσεις/προδιαγραφές ακόμη και αν αυτές φτάσουν αργά. Οι ευέλικτες διαδικασίες θα τιθασεύουν την αλλαγή δημιουργώντας ανταγωνιστικό πλεονέκτημα για τον πελάτη.

3. Παραδώστε λειτουργικό λογισμικό συχνά μέσα σε μια με δυο εβδομάδες ή σε ένα με δύο μήνες, με προτίμηση στο πιο σύντομο χρονοδιάγραμμα.
4. Οι επιχειρησιακοί υπάλληλοι και οι προγραμματιστές πρέπει να λειτουργούν μαζί καθημερινά.
5. Δομήστε τα έργα γύρω από τα άτομα με ισχυρή υποκίνηση. Δώστε τους το περιβάλλον και την υποστήριξη που αυτοί χρειάζονται και εμπιστευθείτε τους πως θα εκπληρώσουν την δουλειά.
6. Η αποδοτικότερη και αποτελεσματική μέθοδος για να μεταβιβάζονται οι πληροφορίες προς την ομάδα αλλά και μέσα σε αυτή είναι οι κατ' ιδίαν συνομιλίες.
7. Το εν λειτουργία λογισμικό είναι το πρωταρχικό μέτρο της προόδου.
8. Οι ευέλικτες διαδικασίες προωθούν τη βιώσιμη ανάπτυξη. Οι χορηγοί, οι υπεύθυνοι για την ανάπτυξη και οι χρήστες πρέπει να είναι σε θέση να διατηρούν έναν σταθερό ρυθμό επ' αόριστο.
9. Η συνεχής προσοχή στην τεχνική τελειότητα και ο καλός σχεδιασμός ενισχύει την ευελιξία.
10. Η απλότητα –η τέχνη της μεγιστοποίησης του ποσού της εργασίας που δεν γίνεται – είναι θεμελιώδης χαρακτηριστικό της ευελιξίας.
11. Οι καλύτερες αρχιτεκτονικές, απαιτήσεις/προδιαγραφές και σχέδια προκύπτουν από αυτορυθμιζόμενες ομάδες.
12. Σε τακτά χρονικά διαστήματα, η ομάδα εξετάζει το πώς μπορεί να γίνει αποτελεσματικότερη, συντονίζει έπειτα και ρυθμίζει τη συμπεριφορά της αναλόγως.»(Beck et al., 2003)

Σ' αυτό το σημείο θα γίνει μία επισκόπηση των ευέλικτων μεθόδων:

1.6 Ακραίος Προγραμματισμός(Extreme Programming-XP)

Ο ακραίος προγραμματισμός(Honda el at,. 2008) είναι ίσως η πιο γνωστή και η πιο διαδεδομένη από τις ευέλικτες μεθόδους. Ονομάστηκε έτσι από τον Beck γιατί αυτή η μέθοδος δημιουργήθηκε χρησιμοποιώντας καλές και αναγνωρισμένες πρακτικές όπως η επαναλαμβανόμενη ανάπτυξη και η ανάμειξη του πελάτη σε υπερβολικό(extreme)βαθμό.

Στην μέθοδο του ακραίου προγραμματισμού, όλες οι απαιτήσεις εκφράζονται ως σενάρια, τα οποία υλοποιούνται άμεσα ως μια σειρά εργασιών. Οι

προγραμματιστές εργάζονται ανά δύο και δοκιμάζουν κάθε εργασία πριν γράψουν τον κώδικα. Όλες οι δοκιμές πρέπει να ολοκληρωθούν με επιτυχία πριν ολοκληρωθεί ο κώδικας τους στο σύστημα.

1.7 Scrum

Scrum είναι μια ευέλικτη μεθοδολογία ανάπτυξης (Honda et al, 2005) που αναπτύχθηκε από τον Jeff Sutherland και επισημοποιήθηκε από τον Ken Schwaber.

Οι ρόλοι που εμπλέκονται σε αυτή τη διαδικασία είναι ο ιδιοκτήτης του προϊόντος(product owner), ο κύριος του Scrum(Scrum master) και η ομάδα Scrum(Scrum team). Ο ιδιοκτήτης του προϊόντος έχει την ευθύνη για την διατήρηση της ορθής επιχειρηματικής προοπτικής. Ο Scrum Master συνεργάζεται με τον ιδιοκτήτη προϊόντος και διευκολύνει την Scrum team. Η Scrum team θα πρέπει να περιλαμβάνει επτά(συν/πλην δύο) μέλη.Ο Scrum Master και η Scrum team έχουν διαφορετικά καθήκοντα να φέρουν εις πέρας. Ο Scrum Master δεν είναι αρχηγός της ομάδας ο ρόλος του είναι να αποκρίνει τα εμπόδια που σχετίζονται με την ικανότητα της ομάδας.

1.8 Οι μεθοδολογίες κρυστάλλου (Crystal methodologies)

Η οικογένεια Crystal(Abrahamsson et al., 2003) μεθοδολογιών περιλαμβάνει μια σειρά διαφορετικών μεθόδων από τις οποίες επιλέγει την πλέον κατάλληλη για κάθε μεμονωμένο έργο. Εκτός από τις μεθόδους η προσέγγιση Crystal περιλαμβάνει επίσης αρχές για την προσαρμογή αυτών των μεθόδων για να ταιριάζει διαφορετικές περιστάσεις των διάφορων έργων. Κάθε μέλος της οικογένειας Crystal σημειώνεται με ένα χρώμα με το οποίο σημειώνεται η «βαρύτητα» της μεθόδου. Η Crystal προτείνει την επιλογή του κατάλληλου χρώματος μεθόδου για ένα σχέδιο με βάση το μέγεθος του και την κρισιμότητα του. Τα μεγαλύτερα έργα είναι πιθανόν να ζητήσουν περισσότερο συντονισμό και βαρύτερες μεθόδους από τα μικρότερα. Η Crystal μέθοδος είναι ανοιχτή σε όλες τις αναπτυξιακές πρακτικές, τα εργαλεία ή την εργασία προϊόντων επιτρέποντας έτσι την ενσωμάτωση, για παράδειγμα ακραίου προγραμματισμού και SCRUM πρακτικές.

1.9 Ανάπτυξη με βάση τα χαρακτηριστικά (Feature-driven development-FDD)

Η ανάπτυξη με βάση τα χαρακτηριστικά(Abrahamsson et al., 2003) είναι μια προσανατολισμένη διαδικασία μεθόδου ανάπτυξης λογισμικού για την ανάπτυξη επιχειρηματικών κρίσιμων συστημάτων. Η προσέγγιση του FDD επικεντρώνεται στο σχεδιασμό και την δημιουργία φάσεων. Η προσέγγιση του FDD ενσαρκώνει επαναληπτική ανάπτυξη με τις πρακτικές που πιστεύεται ότι είναι αποτελεσματικές στη βιομηχανία. Το συγκεκριμένο μείγμα αυτών των συστατικών καθιστά τις FDD διαδικασίες μοναδικές για κάθε περίπτωση. Τονίζει τα θέματα ποιότητας σε όλη τη διαδικασία και περιλαμβάνει συχνά και απτά τις παραδόσεις μαζί με την ακριβή παρακολούθηση της προόδου του έργου.

1.10 Μέθοδος ανάπτυξης δυναμικών συστημάτων(Dynamic Systems Development Method)

Η μέθοδος αυτή αναπτύχθηκε στην Βρετανία(Coram and Bohner, 2005) στα μέσα της δεκαετίας του 90. Η φιλοσοφία της μεθόδου είναι ότι τίποτα δεν φτιάχνεται καλά από την πρώτη φόρα. Οι εννέα αρχές της μεθόδου είναι οι ακόλουθες:1) Συνεχή συμμετοχή του χρήστη, 2)Δυνατές ομάδες οι οποίες αναλαμβάνουν πρωτοβουλίες και παίρνουν αποφάσεις, 3)Εστίαση στη συχνή μεταφορά προϊόντων, 4)Επαναληπτική ανάπτυξη ώστε να επιβεβαιωθεί η σύγκλιση στις επιχειρησιακές λύσεις, 5)Αναστρέψιμες αλλαγές κατά τη διάρκεια ανάπτυξης, 6)Απαιτήσεις υψηλού επιπέδου, 7)Ολοκληρωμένος έλεγχος κατά τον κύκλο ζωής, 8)Συνεργασία μεταξύ όλων των εμπλεκόμενων.

1.11 Εναρμονισμένη ανάπτυξη λογισμικού(Adaptive Software Development-ASD)

Η εναρμονισμένη ανάπτυξη λογισμικού αναπτύχθηκε και δημοσιεύτηκε από τον James A. High smith III,(2000). Η ASD εστιάζει κυρίως στα προβλήματα κατά την ανάπτυξη των σύνθετων και μεγάλων συστημάτων. Η μέθοδος αυτή ενθαρρύνει την επαυξητική, επαναληπτική ανάπτυξη με συνεχείς διαμόρφωση πρωτοτύπου.

2. ΕΥΕΛΙΚΤΕΣ ΜΕΘΟΔΟΛΟΓΙΕΣ ΑΝΑΠΤΥΞΗΣ ΛΟΓΙΣΜΙΚΟΥ

2.1 Εισαγωγή

Ευέλικτος(agile) –δηλώνει την ιδιότητα του ευκίνητου, την ετοιμότητα για την κίνηση, την σβελτάδα, την δραστηριότητα, την επιδεξιότητα σε κίνηση(Abrahamsson et al, 2003). Οι μέθοδοι ανάπτυξης λογισμικού προσπαθούν να προσφέρουν και πάλι μια απάντηση στην πρόθυμη επιχειρηματική κοινότητα ζητώντας μικρότερο βάρος, μαζί με ταχύτερη διαδικασία ανάπτυξης λογισμικού. Οι κύριες πτυχές των ευέλικτων μεθόδων είναι η απλότητα και η ταχύτητα. Αυτό που κάνει μια μέθοδος ανάπτυξης λογισμικού ευέλικτη είναι όταν αυτή είναι:

- **Επαυξητική**(incremental). Μικρές εκδόσεις λογισμικού, με γρήγορους κύκλους.
- **Συνεταιριστική**(cooperative). Πελάτης και υπεύθυνοι για την ανάπτυξη που λειτουργούν συνεχώς μαζί με στενή επικοινωνία.
- **Απλή**(simple). Η ίδια η μέθοδος είναι εύκολο να μαθευτεί και να τροποποιηθεί
- **Προσαρμοστική**(adaptive). Ικανή να κάνει τις τελευταίες αλλαγές της στιγμής.

Παρακάτω παρουσιάζονται οι πιο διαδομένες ευέλικτες μεθοδολογίες –οι διαδικασίες, οι ρόλοι και οι πρακτικές τους.

2.2 ΑΚΡΑΙΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ (Extreme programming-XP)

Οι δημιουργοί του ακραίου προγραμματισμού αποσκοπούν στην ανάπτυξη μιας μεθοδολογίας κατάλληλη για αντικειμενοστραφή σχέδια(object-oriented design) που χρησιμοποιεί ομάδες των δώδεκα ή λιγότερων προγραμματιστών σε κάθε θέση.

Ο ακραίος προγραμματισμός είναι μια μεθοδολογία που προορίζεται να βελτιώσει την ποιότητα του λογισμικού και την ανταπόκριση του στις μεταβαλλόμενες απαιτήσεις πελατών. Ως τύπος ευέλικτης ανάπτυξης λογισμικού, υποστηρίζει τις συχνές εκδόσεις και τους σύντομους κύκλους ανάπτυξης. Ο σκοπός είναι να βελτιώσει την παραγωγικότητα και να εισαγάγει

εκείνα τα σημεία ελέγχου στα οποία οι νέες απαιτήσεις των πελατών μπορούν να υιοθετηθούν.

Η μεθοδολογία του ακραίου προγραμματισμού(Laurie, 2007) βασίζεται σε πέντε βασικές αξίες: την επικοινωνία, την απλότητα, την ανατροφοδότηση, το θάρρος και τον σεβασμό.

- Η επικοινωνία(communication). Η αξία της επικοινωνίας βασίζεται στην παρατήρηση. Οι περισσότερες δυσκολίες του έργου προκύπτουν επειδή κάποιος έπρεπε να μιλήσεις με κάποιον άλλον για να διευκρινίσει μια ερώτηση, να συνεργαστεί ή να λάβει βοήθεια. Προβλήματα με τα σχέδια μπορούν πάντα να αναχθούν σε κάποιον που δεν μιλάει σε κάποιον άλλο για κάτι σημαντικό.
- Απλότητα(simplicity). Σχεδιάζει το πιο απλό προϊόν που ανταποκρίνεται στις ανάγκες του πελάτη. Μια σημαντική πτυχή της αξίας είναι ότι συνεχώς απλοποιεί και βελτιώνει τον κώδικα με ανατροφοδότηση. Όταν ο κώδικας είναι απλός είναι περισσότερο κατανοητός, ενώ έχει λιγότερα λάθη.
- Ανατροφοδότηση(feedback). Η ομάδα αποκτά ανατροφοδότηση της ανάπτυξης από τους πελάτες στο τέλος κάθε επανάληψης και των εξωτερικών απελευθερώσεων. Αυτό βοηθά στην συνειδητοποίηση προβλημάτων πριν την ολοκλήρωση του έργου.
- Θάρρος(courage). Το θάρρος είναι ένας από τους πιο σημαντικούς παράγοντες. Οι προγραμματιστές δεν πρέπει να χάνουν το θάρρος τους σε κάθε αποτυχία και πρέπει να ξεκινούν από την αρχή.
- Σεβασμός(respect). Τα μέλη της ομάδας πρέπει να φροντίσουν ο ένας τον άλλον και για το έργο.

Άλλα στοιχεία του ακραίου προγραμματισμού είναι: ο προγραμματισμός ανά ζεύγη(pair programming), το παιχνίδι του σχεδιασμού(planning game), έλεγχοι πριν την κωδικοποίηση (test-first-design),απλός σχεδιασμός(simple design), μικρές εκδόσεις(small releases), πρότυπα κωδικοποίησης(coding standards), ανακατασκευή κώδικα(refactoring), διαρκείς ενσωματώσεις στον κώδικα(continuous integration), συλλογική ιδιοκτησία του κώδικα(collective code ownership), διαρκής παρουσία πελάτη(on-site customer), υποφερτός ρυθμός εργασίας (sustainable pace), συνολική εικόνα του συστήματος(system metaphor).

Η μεθοδολογία παίρνει το όνομα της από την ιδέα ότι τα ευεργετικά στοιχεία των παραδοσιακών πρακτικών ανάπτυξης λογισμικού λαμβάνονται στα «ακραία» επίπεδα, βασισμένο στη θεωρία ότι εάν κάτι είναι καλό, τότε πολύ περισσότερο από αυτό είναι ακόμη καλύτερο. Ακόμη δεν υποστηρίζει τα εξοντωτικά προγράμματα εργασιών, αλλά αντί αυτού την εργασία σε έναν σταθερό ρυθμό.

Ο ακραίος προγραμματισμός δημιουργήθηκε από τον Kent Beck κατά τη διάρκεια της εργασίας του στο πρόγραμμα μισθοδοσίας της Chrysler Comprehensive Compensation System. Ο Beck έγινε ο υπεύθυνος του προγράμματος τον Μάρτιο του 1996 και άρχισε να βελτιώνει τη μέθοδο ανάπτυξης. Μετά από διάφορες επιτυχείς δοκιμές στην πράξη (Anderson et al 1998), η μεθοδολογία του ακραίου προγραμματισμού καταγράφηκε πρώτη φορά ως θεωρία με τις βασικές αρχές της και τις χρησιμοποιούμενες πρακτικές στο βιβλίο «Extreme Programming Explained» από τον Beck(Beck and Anders,2005). Η Chrysler ακύρωσε το πρόγραμμα τον Φεβρουάριο του 2000, αφότου εξαγοράστηκε από Daimler- Benz.

Αν και ο ακραίος προγραμματισμός είναι κάτι σχετικά νέο, πολλές από τις πρακτικές του προϋπήρχαν, εξάλλου η μεθοδολογία ωθεί τις «καλύτερες πρακτικές» στα ακραία. Παραδείγματος χάριν, η πρακτική του σχεδιασμού των δοκιμών πριν γίνει η υλοποίηση του προγράμματος αυτού καθ’ αυτού χρησιμοποιήθηκε από το πρόγραμμα της NASA Mercury, στις αρχές της δεκαετίας του ’60. Η ανατροφοδότηση του κώδικα, η διαμόρφωση του σε τμήματα, η σχεδίαση από κάτω προς τα επάνω και ο επαυξητικός σχεδιασμός περιγράφηκαν πρώτη φορά από τον Leo Brodie στο βιβλίο του που δημοσιεύθηκε το 1984(Brodie, 1984).

2.2.1 Έγγραφα και εκθέματα

Σε γενικές γραμμές, (Laurie, 2007) ο ακραίος προγραμματισμός στηρίζεται στην «τεκμηρίωση» μέσω της προφορικής επικοινωνίας από τον ίδιο τον κώδικα και τη σιωπηρή μεταφορά γνώσεων και όχι από γραπτά έγγραφα και αντικείμενα. Ωστόσο, ενώ η προφορική επικοινωνία μπορεί να χρησιμοποιηθεί από τις μικρές ομάδες, δεν είναι μια συνιστώμενη διαδικασία για μεγάλα συστήματα υψηλού κινδύνου. Στις περιπτώσεις αυτές προκύπτουν τα ακόλουθα: τα «εργαλεία μπορεί να χρειαστεί να είναι πιο επίσημα στη

διαχείριση, να καταγράφονται και να επανέρχονται τακτικά ως μέρος μιας πιο «επίσημης» και ανιχνεύσιμης διαδικασίας ακραίου προγραμματισμού.

- **Κάρτες ιστοριών χρήστη,** είναι κάρτες ευρετηρίου που περιέχουν συνοπτικές απαιτήσεις των περιγραφών. Οι χρήστες καρτών ιστορίας σκοπίμως δεν αποτελούν πλήρη αναφορά απαιτήσεων, αλλά είναι αντίθετα μια δέσμευση για περαιτέρω συζήτηση μεταξύ του κυρίου του έργου και του πελάτη. Κατά την συνομιλία, τα δύο μέρη θα έρθουν σε προφορική συνεννόηση στο τι είναι αναγκαίο για τις απαιτήσεις που πρέπει να πληρούνται. Η προτεραιότητα του πελάτη και η εκτίμηση του προγραμματιστή των πόρων προστίθενται στην κάρτα. Η εκτίμηση των πόρων για μια ιστορία που ο χρήστης δεν πρέπει να υπερβαίνει τη διάρκεια επανάληψης.
- **Η λίστα εργασιών,** είναι μια λίστα εργασιών (κατά το ήμισυ έως τρείς ημέρες στη διάρκεια) για τις ιστορίες του χρήστη που πρέπει να ολοκληρωθούν για την επανάληψη. Οι προγραμματιστές κάνουν εθελοντικά τα καθήκοντα αντί να τα αναθέτουν κάπου αλλού.
- **Οι κάρτες CRC (προαιρετικό),** είναι κάρτες κατά τις οποίες καταγράφονται οι ευθύνες και οι συνεργάτες των τάξεων που μπορούν να τις χρησιμοποιήσουν ως βάση για το σχεδιασμό. Οι κατηγορίες, οι ευθύνες και οι συνεργάτες επισημάνθηκαν κατά τη διάρκεια του σχεδιασμού ιδεών που παίζουν ρόλο –τη συμμετοχή πολλών προγραμματιστών.
- **Δοκιμές αποδοχής πελατών,** είναι υπό μορφή κειμένου γίνονται περιγραφές και αυτοματοποιημένες περιπτώσεις δοκιμών που αναπτύσσονται από τον πελάτη. Η ομάδα ανάπτυξης δείχνει την ολοκλήρωση από μια ιστορία των χρηστών και την επικύρωση των απαιτήσεων των πελατών με το πέρασμα αυτών των περιπτώσεων δοκιμών.
- **Ορατοί πίνακες γραφημάτων,** για την προώθηση της επικοινωνίας και της ευθύνης, η πρόοδος γραφημάτων είναι συνήθως τοποθετημένες σε χώρο ομαδικής εργασίας. Αυτά τα διαγράμματα προόδου συχνά περιλαμβάνουν πόσες ιστορίες έχουν ολοκληρωθεί και πόσες περιπτώσεις αποδοχής παίρνουν.

2.2.2 Ρόλοι

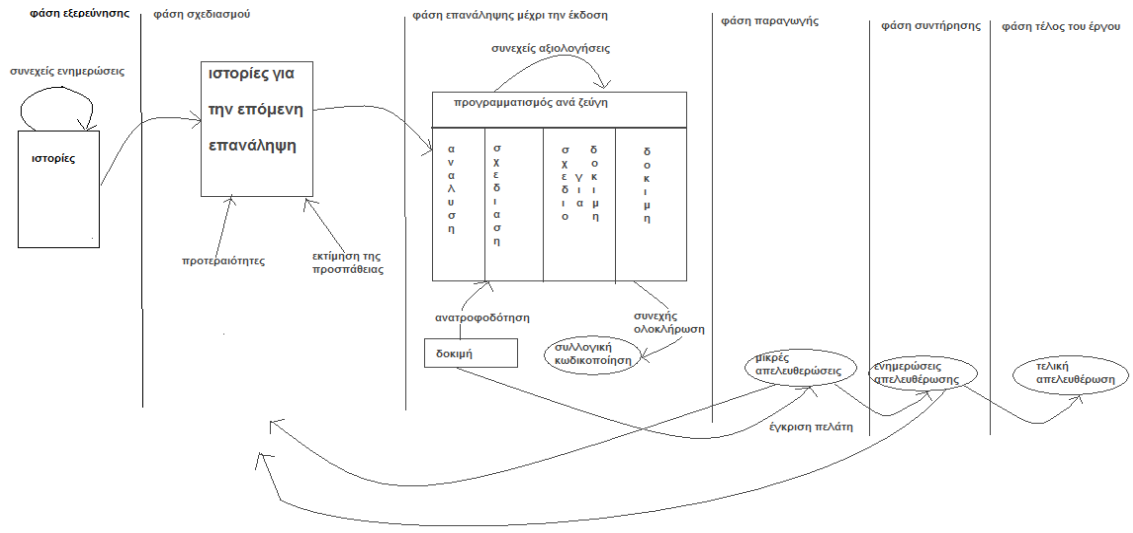
- **Διευθυντής**(Director). Ο διευθυντής κατέχει την ομάδα και τα προβλήματα της και λαμβάνει τις αποφάσεις. Προκειμένου να είναι σε θέση να κάνει αυτό, επικοινωνεί με την ομάδα προγράμματος για να καθορίσει την τρέχουσα κατάσταση και για να διακρίνει οποιεσδήποτε δυσκολίες ή ανεπάρκειες στη διαδικασία.
- **Εκπαιδευτής**(Instructor). Ο εκπαιδευτής διδάσκει τα μέλη της ομάδας σχετικά με τη διαδικασία ακραίου προγραμματισμού. Όπου είναι απαραίτητο παρεμβαίνει στην περίπτωση των ζητημάτων και παρακολουθεί κατά πόσο ακολουθείται η διαδικασία ακραίου προγραμματισμού. Ο εκπαιδευτής είναι αρμόδιος για τη διαδικασία συνολικά. Μια συνολική κατανόηση του ακραίου προγραμματισμού είναι σημαντική από αυτόν τον ρόλο, για να επιτρέψει σε αυτόν να καθοδηγήσει και τα άλλα μέλη των ομάδων στη διαδικασία. Ο εκπαιδευτής είναι συνήθως προγραμματιστής και όχι διαχειριστής.
- **Ιχνηλάτης**(Tracker). Ο ιχνηλάτης δίνει την ανατροφοδότηση στον ακραίο προγραμματισμό. Παρακολουθεί τις εκτιμήσεις που γίνονται από την ομάδα(π.χ. εκτιμήσεις προσπάθειας) και δίνει την ανατροφοδότηση για το πόσο ακριβείς ήταν προκειμένου να βελτιωθούν οι μελλοντικές εκτιμήσεις. Παρακολουθεί επίσης την πρόοδο κάθε επανάληψης και αξιολογεί εάν ο στόχος είναι εφικτός μέσα στους δεδομένους περιορισμούς των πόρων και χρόνου ή εάν οποιεσδήποτε αλλαγές απαιτούνται στη διαδικασία. Ο ιχνηλάτης είναι προγραμματιστής, δεν είναι διαχειριστής ούτε πελάτης.
- **Προγραμματιστής**(Programmer). Οι προγραμματιστές γράφουν τις δοκιμές και διατηρούν τον κώδικα του προγράμματος όσο το δυνατόν πιο απλούστερο και καθορισμένο. Ο πρώτος παράγοντας που καθιστά τον ακραίο προγραμματισμό επιτυχημένο είναι η επικοινωνία και ο συντονισμός μεταξύ των προγραμματιστών και των άλλων μελών της ομάδας.
- **Ελεγκτής**(Controller). Ο ελεγκτής βοηθά τους πελάτες να γράφουν και να αναπτύσσουν τις δοκιμές (το άτομο αυτό μπορεί επίσης να

αποτελέσει προγραμματιστής). Τρέχουν δοκιμές συχνά, ανακοινώνουν τα αποτελέσματα της δοκιμής και συντηρούν τα εργαλεία δοκιμών.

- **Πελάτης**(Customer). Ο πελάτης γράφει την ιστορία και τις δοκιμές έγκρισης, διαλέγει ιστορίες για μια απελευθέρωση και μια επανάληψη. Μια κοινή παρανόηση είναι ότι ο ρόλος του πελάτη πρέπει να διαδραματίζεται από ένα άτομο από τον οργανισμό πελάτη. Αντίθετα, μια ομάδα πελατών μπορεί να συμμετέχει ή ένας εκπρόσωπος του πελάτη μπορεί να επιλεγεί από το εσωτερικό του οργανισμού για την ανάπτυξη (αλλά και οι εξωτερικοί στην ομάδα ανάπτυξης). Ο πελάτης θέτει τις προτεραιότητες εφαρμογής των απαιτήσεων.
- **Σύμβουλος**(Consultant). Ο σύμβουλος είναι ένα εξωτερικό μέλος που κατέχει τις συγκεκριμένες τεχνικές γνώσεις που απαιτούνται. Ο σύμβουλος καθοδηγεί την ομάδα στην επίλυση των συγκεκριμένων προβλημάτων τους(Laurie, 2007).

2.2.3 Διαδικασία

Ο κύκλος ζωής του ακραίου προγραμματισμού(Pekka et al,. 2003) αποτελείτε από πέντε φάσεις: της εξερεύνησης (Exploration), του προγραμματισμού (Programming), της επανάληψης της έκδοσης (Iterations to release), της παραγωγής (Product ionizing), της συντήρησης (Maintenance) και το τέλος του έργου(end of the project).



Σχήμα 2.1 Διαδικασία ακραίου προγραμματισμού

Παρακάτω περιγράφονται αυτές οι φάσεις σύμφωνα με τον Beck (Beck and Andres 2005).

Στην **φάση εξερεύνησης(exploration phase)**, οι πελάτες καταγράφουν σε κάρτες «ιστορίας» (story cards) αυτό που επιθυμούν να περιληφθεί στην πρώτη έκδοση. Κάθε κάρτα ιστορίας περιγράφει ένα χαρακτηριστικό γνώρισμα που προστίθεται στο πρόγραμμα. Συγχρόνως η ομάδα του προγράμματος εξοικειώνεται με τα εργαλεία, την τεχνολογία και τις πρακτικές που θα χρησιμοποιούν στο πρόγραμμα. Η τεχνολογία που χρησιμοποιείται θα εξεταστεί και οι αρχιτεκτονικές δυνατότητες του συστήματος διερευνούνται με την δημιουργία ενός πρωτοτύπου του συστήματος. Η φάση εξερεύνησης διαρκεί από μερικές εβδομάδες έως μερικούς μήνες, κάτι που εξαρτάται κατά ένα μεγάλο μέρος από το πόσο εξοικειωμένοι είναι με την τεχνολογία οι προγραμματιστές.

Στην **φάση του σχεδιασμού(planning phase),** τίθενται οι προτεραιότητες μεταξύ των ιστοριών και γίνεται μια συμφωνία για το περιεχόμενο της πρώτης μικρής έκδοσης. Οι προγραμματιστές υπολογίζουν αρχικά πόση προσπάθεια απαιτείται από κάθε ιστορία και έπειτα συμφωνείται το πρόγραμμα σχετικά με αυτές. Η χρονική έκταση του προγράμματος της πρώτης έκδοσης κανονικά δεν υπερβαίνει τους δύο μήνες. Η ίδια η φάση προγραμματισμού διαρκεί μερικές ημέρες.

Η **φάση των επαναλήψεων μέχρι την έκδοση(iteration to release phase),** περιλαμβάνει διάφορες επαναλήψεις του συστήματος πριν από την πρώτη έκδοση. Το πρόγραμμα που έχει συμφωνηθεί στο στάδιο προγραμματισμού χωρίζεται σε διάφορες επαναλήψεις που κάθε μια μπορεί να διαρκέσει από μια έως τέσσερις εβδομάδες. Η πρώτη επανάληψη δημιουργεί ένα σύστημα, με την αρχιτεκτονική που θα έχει ολόκληρο το σύστημα. Αυτό επιτυγχάνεται με την επιλογή των κατάλληλων ιστοριών. Ο πελάτης είναι αυτός που αποφασίζει τις ιστορίες που επιλέγονται να ενσωματωθούν σε κάθε επανάληψη. Οι λειτουργικές δοκιμές που δημιουργούνται από τον πελάτη τρέχουν στο τέλος κάθε επανάληψης. Στο τέλος της τελευταίας επανάληψης το σύστημα είναι έτοιμο για την παραγωγή.

Η **φάση παραγωγής(product phase),** απαιτεί επιπλέον δοκιμές και έλεγχο της απόδοσης του συστήματος προτού το σύστημα παραδοθεί στον πελάτη. Σε αυτήν την φάση, νέες αλλαγές μπορούν ακόμα να βρεθούν και πρέπει να

ληφθεί η απόφαση εάν θα συμπεριληφθούν στην τρέχουσα έκδοση. Κατά τη διάρκεια αυτής της απόφασης, οι επαναλήψεις μπορεί να πρέπει να επιταχυνθούν από τις τρείς έως μια εβδομάδα. Οι ιδέες και οι προτάσεις που προκύπτουν, τεκμηριώνονται για να είναι δυνατή η εφαρμογή τους κατά τη διάρκεια άλλων σταδίων, π.χ., της φάσης συντήρησης.

Αφότου δοθεί η πρώτη έκδοση στους πελάτες το XP πρόγραμμα πρέπει και να κρατήσει το σύστημα στην παραγωγή εν λειτουργία και να παράγει νέες επαυξήσεις. Προκειμένου να γίνει αυτό, η **φάση συντήρησης(maintenance phase),** απαιτεί μια επιπλέον προσπάθεια για τις εργασίες υποστήριξης των πελατών. Κατά συνέπεια, η ταχύτητα της ανάπτυξης μπορεί να επιβραδυνθεί από την στιγμή που το σύστημα πάει στην παραγωγή. Η φάση συντήρησης μπορεί να απαιτήσει την εισαγωγή νέων ανθρώπων στην ομάδα και την αλλαγή της δομής της ομάδας.

Η **φάση ολοκλήρωσης του έργου(end of the project),** έρχεται όταν ο πελάτης δεν έχει πλέον οποιεσδήποτε ιστορίες που χρειάζονται εφαρμογή. Αυτό απαιτεί ότι το σύστημα ικανοποιεί τις ανάγκες των πελατών από όλες τις απόψεις(π.χ., σχετικά με την απόδοση και την αξιοπιστία). Αυτή είναι η στιγμή στη διαδικασία του ακραίου προγραμματισμού που γράφεται τελικά η απαραίτητη τεκμηρίωση του συστήματος καθώς καμία άλλη αλλαγή στην αρχιτεκτονική, στο σχέδιο ή στον κώδικα δεν γίνεται. Η λήξη του έργου μπορεί επίσης να εμφανίζεται εάν το σύστημα δεν αποδίδει το επιθυμητό αποτέλεσμα ή εάν αρχίζει να γίνεται πάρα πολύ ακριβό για περαιτέρω ανάπτυξη.

2.2.4 Πρακτικές

Η αρχική έκδοση του λογισμικού της μεθοδολογίας του ακραίου προγραμματισμού που δημοσιεύθηκε το 2000 είχε 12 κεντρικούς προγραμματιστές τεχνικών πρακτικών. Οι τεχνικές αυτές αλληλεπιδρούν αντίβαρο και αλληλοενισχύονται. Ωστόσο σε μια έρευνα της διαχείρισης έργου από διευθυντικά στελέχη, προγραμματιστές και αντιπρόεδροι των μηχανικών για 21 έργα λογισμικού διαπιστώθηκε ότι καμιά εταιρεία δεν υιοθέτησε ακραίο προγραμματισμό σε μια καθαρή μορφή όπου οι 12 πρακτικές χρησιμοποιήθηκαν χωρίς προσαρμογή. Το 2005, ο ακραίος προγραμματισμός άλλαξε ώστε να συμπεριλάβει 13 πρωτογενείς πρακτικές και 11 απόρροια πρακτικών. Οι πρωτογενείς πρακτικές χρησιμοποιήθηκαν με

στόχο να είναι χρήσιμες και ανεξάρτητες μεταξύ τους με τις άλλες πρακτικές που χρησιμοποιούνται αν και η αλληλεπίδραση μεταξύ των πρακτικών μπορούν να διευρύνουν τα αποτελέσματα. Το επιστέγασμα πρακτικών είναι πιθανόν να είναι δύσκολο χωρίς πρώτα να υπάρχει ένα βασικό σύνολο των πρωτογενών πρακτικών. Παρακάτω περιγράφονται συνοπτικά οι 13 πρωτογενείς τεχνικές πρακτικές του ακραίου προγραμματισμού:

Κάθονται μαζί, όλη η ομάδα αναπτύσσεται σε ένα ανοικτό χώρο.

Όλη η ομάδα, χρησιμοποιεί μια πολλαπλή λειτουργική ομάδα όλων εκείνων που απαιτούνται για την επιτυχία του προϊόντος.

Ενημερωτικός χώρος εργασίας, χώρος με ορατά γραφήματα τοίχου γύρω από το χώρο εργασίας, έτσι ώστε τα μέλη της ομάδας (ή άλλοι ενδιαφέροντες παρατηρητές) να μπορούν να πάρουν μια γενική ιδέα για το πώς συνεχίζεται το έργο.

Ενεργός εργασία, οι ομάδες ακραίου προγραμματισμού δεν λειτουργούν με υπερβολικές υπερωρίες για πολλές ώρες. Το κίνητρο πίσω από αυτήν την πρακτική, είναι να διατηρήσουν τον κώδικα υψηλής ποιότητας (κουρασμένοι προγραμματιστές έχουν περισσότερα ελαττώματα) και του ευτυχή προγραμματισμού (για την μείωση των εργαζόμενων κύκλων εργασιών).

Προγραμματισμός ανά ζεύγη, αναφέρεται στην πρακτική κατά την οποία δύο προγραμματιστές εργάζονται σε έναν υπολογιστή, η συνεργασία στο ίδιο σχέδιο, αλγόριθμος, κώδικας, δοκιμή.

Ιστορίες, η ομάδα γράφει σύντομες δηλώσεις του πελάτη στην λειτουργικότητα που επιθυμείτε στο προϊόν. Οι προγραμματιστές εκτιμούν την ιστορία, ο πελάτης δίνει προτεραιότητα στην ιστορία.

Εβδομαδιαίος κύκλος, στην αρχή κάθε εβδομάδας η συνεδρίαση διεξάγει την παρακολούθηση της προόδου μέχρι εκείνη την στιγμή. Έχει πάρει μια εβδομάδα από τις ιστορίες για την υλοποίηση αυτής της εβδομάδας (με βάση τις εκτιμήσεις για την ανάπτυξη και τη δική τους προτεραιότητα), καθώς έχει σπάσει τις ιστορίες και στα καθήκοντα που πρέπει να ολοκληρωθούν αυτή την εβδομάδα. Μέχρι το τέλος της εβδομάδας, οι υποθέσεις δοκιμής αποδοχής για τις επιλεγμένες ιστορίες θα πρέπει να τρέχουν για επίδειξη στον πελάτη και να οδηγούν στον επόμενο εβδομαδιαίο κύκλο.

Τριμηνιαίο κύκλο, όλη η ομάδα πρέπει να διαλέξει ένα θέμα ή θέματα της ιστορίας για ένα τρίμηνο που να αξίζει. Τα θέματα θα βοηθήσουν την ομάδα

να προβληματιστεί σχετικά με τη γενικότερη εικόνα. Στο τέλος του τριμήνου, εκδίδεται η επιχειρηματική αξία.

Χαλαρότητα, σε κάθε επανάληψη, το σχέδιο κάποιας χαμηλότερης προτεραιότητας σε καθήκοντα που μπορεί να εγκαταλειφθεί.

Δέκα λεπτά με την κατασκευή, τη δομή του έργου και των συναφών δοκιμών του τέτοιες ώστε το όλο σύστημα να μπορεί να κατασκευαστεί και όλες οι δοκιμές να μπορούν να εκτελεστούν σε δέκα λεπτά, έτσι ώστε όταν το σύστημα κατασκευαστεί οι δοκιμές θα τρέξουν πολλές φορές.

Πρώτη δοκιμή προγραμματισμού, όλες οι ιστορίες έχουν τουλάχιστον μια δοκιμή αποδοχής, κατά προτίμηση αυτοματοποιημένη. Όταν η δοκιμή αποδοχής (ες)για μια ιστορία περνά από τον χρήστη, η ιστορία αυτή θεωρείται πλήρες. Επιπλέον, οι αυτοματοποιημένες δοκιμές γίνονται εναλλάξ και είναι σταδιακά γραμμένες σε λεπτό ανά λεπτό.

Συνεχής ολοκλήρωση, οι προγραμματιστές ελέγχουν την ολοκλήρωση του βασικού κώδικα και τις συναφείς δοκιμές του αρκετές φορές την ημέρα. Ο κώδικας μπορεί μόνο να ελεγχθεί εάν όλοι οι συναφείς δοκιμές της μονάδας περάσουν από όλους τους ελέγχους.

Στοιχειώδη σχεδιασμού, αντί να αναπτύξει προληπτικά τον λεπτομερή σχεδιασμό του πριν από την εφαρμογή, επενδύει στο σχεδιασμό του συστήματος κάθε μέρα στο φώς της εμπειρίας του παρελθόντος. Η βιωσιμότητα και η σύνεση προβλεπτικού σχεδιασμού έχει αλλάξει δραματικά σε ασταθές περιβάλλον των επιχειρήσεων μας. Η ανατροφοδότηση για την βελτίωση του σχεδιασμού του ήδη γραμμένου κώδικα είναι απαραίτητη. Οι ομάδες με ισχυρές δοκιμές μονάδων μπορούν με σιγουριά να πειραματιστούν με ανατροφοδότηση επειδή το δίχτυ ασφαλείας είναι στην θέση του.

Παρακάτω περιγράφονται συνοπτικά οι 11 τεχνικές πρακτικές του ακραίου προγραμματισμού:

Πραγματική συμμετοχή του πελάτη, ο πελάτης είναι διαθέσιμος να διευκρινίσει τις απαιτήσεις των ερωτημάτων και έχει εξουσία να λαμβάνει αποφάσεις σχετικά με τις απαιτήσεις και τις προτεραιότητες. Επιπλέον, ο πελάτης γράφει τις δοκιμές αποδοχής.

Στοιχειώδη ανάπτυξη, σταδιακή ανάπτυξη λειτουργικότητας σε ένα ζωντανό περιβάλλον για να μειωθεί ο κίνδυνος μιας μεγάλης ανάπτυξης.

Η ομάδα της συνέχειας, διατηρεί μαζί τις ομάδες.

Συρρίκνωση ομάδας, σαν ομάδα μεγαλώνει με την ιδιότητα (λόγω εμπειρίας), να τηρούν το φόρτο εργασίας σταθερό αλλά σταδιακή μείωση του μεγέθους της ομάδας.

Ανάλυση των αρχικών αιτιών, να εξετάσει την αιτία ενός ελαττώματος που ανακαλύφθηκε από γραπτή δοκιμασία αποδοχής (ες) και μονάδες δοκιμών για να αποκαλύψει το ελάττωμα. Στην συνέχεια, εξετάζει τους λόγους που δημιουργήθηκαν τα ελαττώματα, αλλά δεν εμπίπτει στην διαδικασία της ανάπτυξης.

Κοινόχρηστος κώδικας, ο κώδικας και οι συναφείς δοκιμές του ελέγχονται μια φορά στη βάση του κώδικα, ο κώδικας μπορεί να τροποποιηθεί από οποιοδήποτε μέλος της ομάδας. Αυτή η συλλογική ιδιοκτησία κώδικα παρέχετε, σε κάθε μέλος της ομάδας με το συναίσθημα που του ανήκει το σύνολο της βάσης του κώδικα και εμποδίζει τα σημεία συμφόρησης που θα μπορούν να έχουν προκληθεί αν ο 'ιδιοκτήτης' ενός στοιχείου, δεν είναι διαθέσιμος να κάνει τις απαραίτητες αλλαγές.

Κώδικας και δοκιμές, να διατηρήσει μόνο τον κώδικα και τις δοκιμές, όπως μόνιμα εκθέματα. Στηρίζονται στην κοινωνική του μηχανισμού για να κρατάει ζωντανή την σπουδαία ιστορία του έργου.

Καθημερινές εγκαταστάσεις, βάλε τον νέο κώδικα στην παραγωγή κάθε βράδυ.

Διαπραγμάτευση των συμβάσεων του πεδίου εφαρμογής, να καθορίσει το χρόνο, το κόστος και την απαιτούμενη ποιότητα του έργου, αλλά έκκληση για μια συνεχή διαπραγμάτευση του πεδίου εφαρμογής του σχεδίου.

Πληρωμή ανά χρήση, να χρεώνει τον χρήστη κάθε φορά που χρησιμοποιεί το σύστημα για την απόκτηση της ανατροφοδότησης από τους τρόπους χρήσης τους.

Αν και δεν είναι μια από τις επίσημες πρακτικές του ακραίου προγραμματισμού, ουσιαστικά όλες οι ομάδες του ακραίου προγραμματισμού έχουν σύντομες καθημερινές συναντήσεις. Σε αυτές τις συναντήσεις, η ομάδα βρίσκεται σε ένα κύκλο. Με τη σειρά του, κάθε μέλος της ομάδας λέει στην ομάδα:

- Ποιος επιτεύχθηκε την προηγούμενη ημέρα
- Τι σχεδιάζει να κάνει σήμερα
- Τα εμπόδια ή δυσκολίες που βιώνει

Συχνά τα ζεύγη προγραμματιστών σχηματίζονται δυναμικά κατά την διάρκεια της καθημερινής συνάντηση όπου συζητούνται τα καθήκοντα για την ημέρα και οι προγραμματιστές που είναι καλύτερα εξοπλισμένοι για να χειριστούν το έργο μαζί.

Ένας «θαρραλέος» διαχειριστής ακραίου προγραμματισμού θα τηρεί αρχείο των εν λόγω συναντήσεων προκειμένου να μετατραπούν σε ποσοτικά μέτρα της πορείας του έργου. Το βάρος του εν λόγου ποσοτικού προσδιορισμού μπορεί να χαλαρώσει και αυτοματοποιημένα μέσω των κατάλληλων εργαλείων.

2.3 SCRUM

Το Scrum είναι ένα επαναληπτικό επαυξητικό μοντέλο για την επίτευξη σύνθετων εργασιών(όπως η ανάπτυξη νέων προϊόντων) που χρησιμοποιείται συνήθως με την ευέλικτη ανάπτυξη λογισμικού. Αν και το Scrum προοριζόταν αρχικά για την διαχείριση των έργων ανάπτυξης λογισμικού, μπορεί να χρησιμοποιηθεί για να διοικηθούν και οι ομάδες συντήρησης λογισμικού, ή ως γενική διοικητική προσέγγιση έργου. Στην μεθοδολογία Scrum συνήθως δημιουργούνται Scrum ομάδες έργων που συστεγάζονται. Ωστόσο, έχουν υπάρξει Scrum ομάδες που εργάζονται γεωγραφικά κατανεμημένες. Σύμφωνα με αυτές τις ομάδες τα μέλη τους συμμετέχουν στην καθημερινή συνάντηση μέσω του μεγαφώνου. Οι ομάδες Scrum είναι αυτό- κατευθυνόμενες και αυτό-οργανωτικές. Η ομάδα αναλαμβάνει την υποχρέωση να καθορίσει τον στόχο για την επανάληψη και να έχει την εξουσία, την αυτονομία και την ευθύνη ώστε να αποφασίσει τον καλύτερο τρόπο για να τα αντιμετωπίσει. Το 1986 ο Hirotaka Takeuchi και ο Ikugiro Nonaka περιέγραψαν μια νέα ολιστική προσέγγιση η οποία αυξάνει την ταχύτητα και η ευελιξία στην ανάπτυξη ενός νέου εμπορικού προϊόντος (Takeuchi and Nonaka 1986). Αυτοί σύγκριναν αυτήν την νέα ολιστική προσέγγιση, στην οποία οι φάσεις επικαλύπτονται έντονα και ολόκληρη η διαδικασία εκτελείται από μια διαγώνιο. Οι τότε μελέτες

περιπτώσεων είχαν γίνει σε αυτοκινητοβιομηχανίες, σε βιομηχανίες φωτογραφικών μηχανών, υπολογιστών και εκτυπωτών.

Η προσέγγιση Scrum έχει αναπτυχθεί για τη διαχείριση της διαδικασίας ανάπτυξης ενός συστήματος. Είναι μια εμπειρική προσέγγιση που εφαρμόζει την θεωρία του ελέγχου μιας βιομηχανίας στην ανάπτυξη συστημάτων και καταλήγει να είναι μια προσέγγιση που επανεισάγει τις ιδέες της ευελιξίας, της προσαρμοστικότητας και της παραγωγικότητας(Schwaber and Beedle, 2008). Δεν καθορίζει συγκεκριμένες τεχνικές ανάπτυξης λογισμικού για τη φάση εφαρμογής. Το Scrum επικεντρώνεται στον τρόπο με τον οποίο τα μέλη της ομάδας πρέπει να λειτουργήσουν προκειμένου να παραχθεί ένα ευπροσάρμοστο σύστημα σε ένα συνεχώς μεταβαλλόμενο περιβάλλον. Το Scrum βοηθά να βελτιώσει τις υπάρχουσες πρακτικές(π.χ. πρακτικές δοκιμής του λειτουργικού) σε έναν οργανισμό. Αυτό επιτυγχάνεται επειδή περιλαμβάνει τις συχνές διοικητικές δραστηριότητες που στοχεύουν συνεχώς να προσδιορίσουν οποιεσδήποτε ανεπάρκειες ή εμπόδια στη διαδικασία ανάπτυξης καθώς επίσης και τις καλύτερες πρακτικές που χρησιμοποιούνται ήδη.

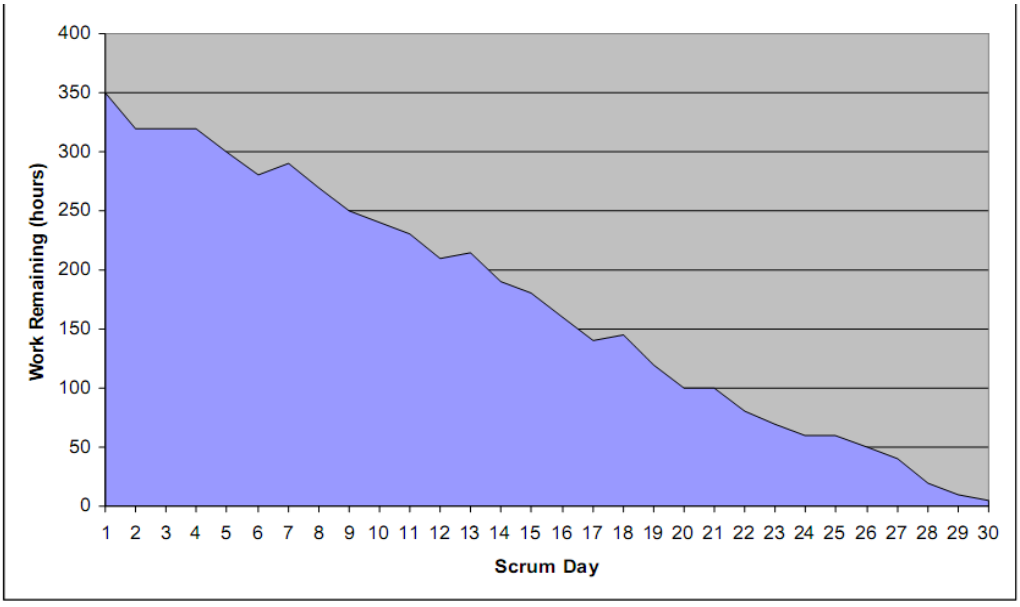
2.3.1 Έγγραφα και εκθέματα

Υπάρχουν τρία βασικά αντικείμενα(Laurie, 2007) που παράγονται από την ομάδα του Scrum: το ανεκτέλεστο προϊόν(product backlog), οι ανεκτέλεστες ορμές(sprints backlog) και τα γραφήματα ορμών(chart sprints). Όλα αυτά είναι απόλυτα προσβάσιμα και εκ προθέσεως ορατά από την ομάδα του Scrum.

- **Ανεκτέλεστο προϊόν**(product backlog), είναι μια εξελισσόμενη ουρά των επιχειρηματικών και τεχνικών λειτουργιών που πρέπει να αναπτυχθούν σε ένα σύστημα και τα ελαττώματα που πρέπει να καθορίζονται κατά την ελευθέρωση. Για κάθε απαίτηση, το ανεκτέλεστο προϊόν περιέχει ένα μοναδικό αναγνωριστικό για την απαίτηση με την κατηγορία(χαρακτηριστικό, βελτίωση, ελάττωμα), την κατάσταση, την προτεραιότητα καθώς και την εκτίμηση για την δυνατότητα.
- **Ανεκτέλεστη ορμή**(backlog spring), είναι ένας κατάλογος όλων των επιχειρήσεων και τα χαρακτηριστικά γνωρίσματα της τεχνολογίας, αξεσουάρ καθώς και ελαττώματα που έχουν προγραμματιστεί για την

τρέχουσα επανάληψη. Η ανεκτέλεστη ορμή διατηρείται επίσης σε ένα υπολογιστικό φύλλο. Οι απαιτήσεις αναλύονται σε εργασίες. Για κάθε έργο υποθέσεων, το λογιστικό φύλλο περιλαμβάνει μια σύντομη περιγραφή καθηκόντων, τα οποία εκκινεί η αποστολή που είναι: ο ιδιοκτήτης του έργου, το καθεστώς και ο αριθμός των ωρών που απομένουν για να ολοκληρωθεί το έργο. Η ανεκτέλεστη ορμή ενημερώνεται κάθε μέρα από έναν ημερήσιο ιχνηλάτη ο οποίος επισκέπτεται τα μέλη της ομάδας για να αποκτήσει τις τελευταίες εκτιμήσεις που απομένουν για να ολοκληρωθεί το έργο. Οι εκτιμήσεις μπορούν να αυξηθούν όταν το μέλος της ομάδας αντιλαμβάνεται ότι το έργο έχει υποτιμηθεί.

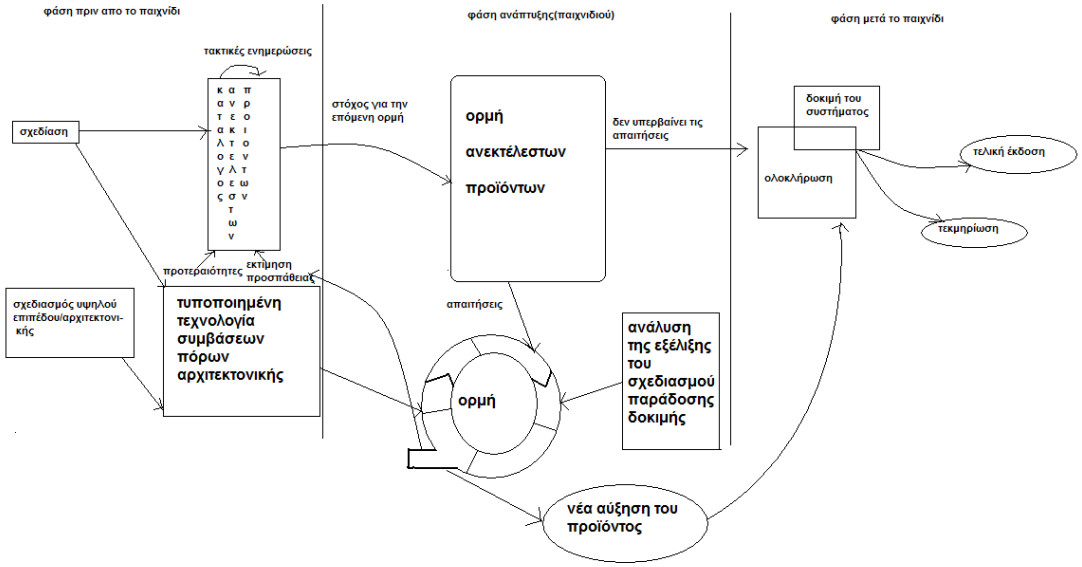
- **Γράφημα ορμών**(chart sprints). Τα γραφήματα ορμών απεικονίζουν τις ώρες που απομένουν για να ολοκληρωθεί το γράφημα των καθηκόντων της ορμής και κατά κύριο λόγο εμφανίζεται για την ομάδα. Το σχήμα 2.3 είναι ένα γράφημα ορμών



Σχήμα 2.2 Γράφημα ορμών

2.3.2 Διαδικασία

Μια επισκόπηση της διαδικασίας Scrum προβλέπεται στο σχήμα 2.2 κάθε ένα από τα στοιχεία της διαδικασίας, θα συζητηθούν παρακάτω:



Σχήμα 2.3 Διαδικασία Scrum

Η διαδικασία Scrum περιλαμβάνει τρεις φάσεις: πριν το παιχνίδι (Pre game), την ανάπτυξη (το παιχνίδι)(Development) και μετά το παιχνίδι (Post game). Παρακάτω, αναλύονται οι φάσεις του Scrum σύμφωνα με τους Schwaber και Beedle (Schwaber and Beedle, 2008).

Η **φάση πριν το παιχνίδι(pre game),** περιλαμβάνει δύο υπό-φάσεις:

- Τον προγραμματισμό (Planning) και
- Τον σχεδιασμό υψηλού επιπέδου/ την αρχιτεκτονική(Architecture/ High level design)

Ο προγραμματισμός περιλαμβάνει τον καθορισμό του συστήματος που θα αναπτυχθεί.

Ένας **κατάλογος ανεκτέλεστών προϊόντων** (Product Backlog list) της παραγγελίας δημιουργείται περιέχοντας όλες τις απαιτήσεις που είναι γνωστές την παρούσα περίοδο. Οι απαιτήσεις μπορεί να προέλθουν από τον πελάτη, τις πωλήσεις και το τμήμα μάρκετινγκ, την υποστήριξη πελατών ή τους ίδιους τους υπεύθυνους για την ανάπτυξη λογισμικού. Οι απαιτήσεις κατατάσσονται ανάλογα με την προτεραιότητα τους και υπολογίζεται η προσπάθεια που απαιτείται για την εφαρμογή τους. Ο κατάλογος ανεκτέλεστων προϊόντων ενημερώνεται συνεχώς με νέα και πιο λεπτομερή στοιχεία, καθώς επίσης και με ακριβέστερες εκτιμήσεις για την αναγκαία προσπάθεια και με νέες προτεραιότητες. Ο προγραμματισμός περιλαμβάνει επίσης τον καθορισμό της ομάδας προγράμματος, των εργαλείων και των άλλων πόρων, την αξιολόγηση του κινδύνου, τις ανάγκες εκπαίδευσης και την διοικητική έγκριση. Σε κάθε επανάληψη, ο ενημερωμένος κατάλογος ανεκτέλεστης παραγγελίας αναθεωρείται από την ομάδα Scrum ώστε να αποκτηθεί η δέσμευση τους για την επόμενη επανάληψη.

Στη **φάση αρχιτεκτονικής και του σχεδιασμού υψηλού επιπέδου** η αρχιτεκτονική προγραμματίζεται βασισμένη στα τρέχοντα στοιχεία που βρίσκονται στον κατάλογο ανεκτέλεστων εργασιών. Σε περίπτωση βελτίωσης ενός υπάρχοντος συστήματος, οι αλλαγές που απαιτούνται για την εφαρμογή των στοιχείων του καταλόγου προσδιορίζονται μαζί με τα προβλήματα που μπορούν να προκαλέσουν. Μια συνεδρίαση για την εφαρμογή μπορεί να λάβει αποφάσεις. Επιπλέον, προετοιμάζονται τα προκαταρκτικά σχέδια για το περιεχόμενο τις επικείμενης νέας έκδοσης.

Η **φάση ανάπτυξης(development phase)** (αποκαλούμενη και ως η φάση του παιχνιδιού) είναι το ευέλικτο μέρος της προσέγγισης Scrum. Αυτή η φάση αντιμετωπίζεται ως «μαύρο κουτί» όπου αναμένονται όλα τα προβλεπόμενα. Οι διαφορετικές περιβαλλοντικές και τεχνικές μεταβλητές(όπως το χρονικό πλαίσιο, η ποιότητα, οι απαιτήσεις, οι πόροι, οι τεχνολογίες και τα εργαλεία εφαρμογής και ακόμη και οι μέθοδοι ανάπτυξης) που προσδιορίζονται στο Scrum και που μπορεί να αλλάξει κατά τη διάρκεια της διαδικασίας, παρατηρούνται και ελέγχονται μέσω των διάφορων πρακτικών του Scrum κατά τη διάρκεια των ορμών (Sprints) της φάσης ανάπτυξης. Αντί να λάβει

υπόψη αυτά τα θέματα μόνο στην αρχή της ανάπτυξης του έργου, το Scrum στοχεύει στο να ελέγχει τα παραπάνω, για να είναι σε θέση συνεχώς να προσαρμόζεται στις αλλαγές.
Στη φάση ανάπτυξης το σύστημα αναπτύσσεται σε ορμές (Sprints) δηλαδή σε επαναληπτικούς κύκλους όπου η λειτουργία αναπτύσσεται ή εμπλουτίζεται για να παραγάγει μικρά νέα κομμάτια (προσαυξήσεις). Κάθε ορμή περιλαμβάνει τις παραδοσιακές φάσεις ανάπτυξης λογισμικού: τις προδιαγραφές, την ανάλυση, το σχέδιο, την εξέλιξη και την παράδοση. Η αρχιτεκτονική και το σχέδιο του συστήματος εξελίσσονται κατά τη διάρκεια του sprint. Ένα sprint προγραμματίζεται για να διαρκέσει από μια εβδομάδα έως έναν μήνα. Μπορεί να υπάρξουν, παραδείγματος χάριν, τρία έως οκτώ sprint σε μια διαδικασία ανάπτυξης προτού το σύστημα να είναι έτοιμο για τη διανομή. Επίσης μπορεί να υπάρξουν περισσότερες από μια ομάδες που δουλεύουν παράλληλα για την ολοκλήρωση της ίδιας προσαύξησης στο σύστημα.

Η **φάση μετά το παιχνίδι(post game),** περιέχει την ολοκλήρωση της νέας έκδοσης. Σε αυτή τη φάση εισερχόμαστε όταν γίνει μια συμφωνία ότι οι μεταβλητές του περιβάλλοντος όπως οι απαιτήσεις έχουν ολοκληρωθεί. Σε αυτή την περίπτωση δεν μπορούν να βρεθούν άλλα στοιχεία και ζητήματα ούτε κάποια νέα να εισαχθούν. Το σύστημα είναι τώρα έτοιμο για την έκδοση του και η προετοιμασία για αυτό γίνεται κατά τη διάρκεια της φάσης «μετά το παιχνίδι», συμπεριλαμβανομένων των εργασιών όπως η προσαρμογή της προσαύξησης στο υπάρχον σύστημα, η δοκιμή του συστήματος (system testing) και η τεκμηρίωση(documentation).

2.3.3 Ρόλοι

- **Ο ιδιοκτήτης του προϊόντος(owner of the product).** Είναι το πρόσωπο που είναι υπεύθυνο για την δημιουργία και την ιεράρχηση του ανεκτέλεστου προϊόντος, επιλέγοντας αυτό που θα συμπεριληφθεί στην επόμενη επανάληψη /sprint και επανεξετάζει το σύστημα (με άλλα ενδιαφέροντα μέρη) στο τέλος της ορμής.
- **Ο «κύριος» του scrum(scrum master)**. Ξέρει και ενισχύει την επανάληψη των προϊόντων, τους στόχους, τις αξίες Scrum και τις

πρακτικές, διεξάγει την καθημερινή συνάντηση (συνέλευση Scrum) και την επίδειξη επαναλήψεων (sprint review) ακούει την πρόοδο, ελέγχει τα κωλύματα (μπλοκ) και παρέχει πόρους. Ο Scrum master είναι επίσης ένας προγραμματιστής και συμμετέχει στην ανάπτυξη του προϊόντος(product development) (δεν είναι μόνο διαχειριστής).

- **Προγραμματιστής**. Ο προγραμματιστής είναι μέλος της ομάδας του Scrum. Η ομάδα Scrum έχει δεσμευτεί για την επίτευξη μιας επιτυχημένης ορμής και έχει πλήρη εξουσία να κάνει οτιδήποτε χρειάζεται για να επιτευχθεί ο στόχος. Το μέγεθος της ομάδας Scrum είναι επτά άτομα, συν ή πλην δύο.
- **Η ομάδα Scrum**. Η ομάδα Scrum είναι η ομάδα του έργου που έχει την δικαιοδοσία για να αποφασίσει σχετικά με τις απαιτήσεις του έργου και να οργανωθεί αυτόνομα προκειμένου να επιτευχθούν οι στόχοι κάθε ορμής. Η ομάδα του Scrum εμπλέκεται, παραδείγματος χάριν, στην εκτίμηση προσπάθειας που δημιουργεί τον κατάλογο της κάθε ορμής, αναθεωρεί τον κατάλογο και προτείνει τα εμπόδια που πρέπει να αφαιρεθούν από το έργο.
- **Ο πελάτης.** Ο πελάτης συμμετέχει στις εργασίες που είναι σχετικές με τα στοιχεία ανεκτέλεστης παραγγελίας προϊόντων για το σύστημα που αναπτύσσεται ή ενισχύεται.
- **Η διοίκηση.** Η διοίκηση είναι υπεύθυνη για την λήψη της τελικής απόφασης καθώς και για τα πρότυπα και τις συμβάσεις που ακολουθούνται στο πρόγραμμα. Η διοίκηση συμμετέχει επίσης στο καθορισμό των στόχων και των απαιτήσεων. Παραδείγματος χάριν, η διοίκηση εμπλέκεται στην επιλογή του ιδιοκτήτη προϊόντων, τη μέτρηση της προόδου και τη μείωση της ανεκτέλεστης παραγγελίας με τον Scrum master(Laurie, 2007).

2.3.4 Πρακτικές

Το Scrum δεν απαιτεί ή δεν παρέχει οποιεσδήποτε συγκεκριμένες μεθόδους /πρακτικές ανάπτυξης λογισμικού. Αντ' αυτού, απαιτεί ορισμένες διοικητικές πρακτικές και εργαλεία στις διάφορες φάσεις του Scrum για να αποφευχθεί το

χάος που προκαλείται από τη αβεβαιότητα και την πολυπλοκότητα (Schwaber, K. 1995).
Παρακάτω δίνεται, η περιγραφή των πρακτικών του Scrum που είναι βασισμένη στους Schwaber και Beedle (Schwaber and Beedle, 2008).

Ο κατάλογος της ανεκτέλεστης παραγγελίας του προϊόντος (Product Backlog), καθορίζει όλα αυτά που απαιτούνται στο τελικό προϊόν βασισμένο στην τρέχουσα γνώση. Κατά συνέπεια, ο κατάλογος της ανεκτέλεστης παραγγελίας καθορίζει την εργασία που γίνεται στο πρόγραμμα. Περιλαμβάνει συνεχώς ενημερωμένο και δομημένο ανά σημαντικότητα κατάλογο επιχειρήσεων και τεχνικών απαιτήσεων για το σύστημα που χτίζεται ή ενισχύεται. Τα στοιχεία του καταλόγου μπορεί να περιλαμβάνουν τα χαρακτηριστικά γνωρίσματα, τις λειτουργίες, τις διορθώσεις, τις ατέλειες, τις ζητούμενες επαυξήσεις και τις τεχνολογικές βελτιώσεις. Επίσης περιλαμβάνει θέματα τα οποία απαιτούν λύση προτού να μπορέσουν να γίνουν άλλα στοιχεία του καταλόγου. Πολλοί μπορούν να συμμετέχουν στην παραγωγή των στοιχείων του καταλόγου, όπως ο πελάτης, η ομάδα προγράμματος, το μάρκετινγκ και οι πωλήσεις, η διοίκηση και η υποστήριξη πελατών. Αυτή η πρακτική περιλαμβάνει όχι μόνο την δημιουργία του καταλόγου, αλλά και τον συνεχή έλεγχο του με την προσθήκη, αφαίρεση, αναθεώρηση και επαναπροσδιορισμό της βαρύτητας των εργασιών στον κατάλογο. Ο ιδιοκτήτης των προϊόντων είναι αρμόδιος για τη διατήρηση του καταλόγου ανεκτέλεστης παραγγελίας προϊόντων.

Εκτίμηση προσπάθειας (Effort estimation), είναι μια επαναληπτική διαδικασία, στην οποία οι εκτιμήσεις των στοιχείων του καταλόγου της ανεκτέλεστης παραγγελίας στρέφονται σε ένα ακριβέστερο επίπεδο όταν περισσότερες πληροφορίες είναι διαθέσιμες. Ο ιδιοκτήτης των προϊόντων μαζί με την ομάδα του Scrum είναι αρμόδιοι για την εκτίμηση της προσπάθειας.

Συνεδρίαση προγραμματισμού της ορμής (Sprint Planning meeting), είναι μια συνεδρίαση με δύο φάσεις που οργανώνεται από τον Scrum master. Οι πελάτες, οι χρήστες, η διοίκηση, ο ιδιοκτήτης των προϊόντων και η ομάδα Scrum συμμετέχουν στην πρώτη φάση της συνεδρίασης για να αποφασίσουν

για τους στόχους και τη λειτουργία της επόμενης ορμής. Στη δεύτερη φάση της συνεδρίασης συμμετέχει μόνο ο Scrum master και εστιάζει στον τρόπο με τον οποίο η επαύξηση των προϊόντων θα πραγματοποιηθεί κατά τη διάρκεια της ορμής.

Ο κατάλογος ανεκτέλεστων εργασιών της ορμής(Sprint Backlog), είναι η αφετηρία για κάθε ορμή. Είναι ένας κατάλογος στοιχείων που επιλέγονται για να παραχθούν στην επόμενη ορμή. Τα στοιχεία επιλέγονται από την ομάδα Scrum μαζί με τον Scrum master και τον ιδιοκτήτη των προϊόντων στην συνεδρίαση προγραμματισμού της ορμής, βάσει των πλέον σημαντικών στοιχείων και των στόχων που τίθενται για την ορμή. Αντίθετα από τον κατάλογο της ανεκτέλεστης παραγγελίας του προϊόντος ο κατάλογος ανεκτέλεστων εργασιών της ορμής είναι σταθερός έως ότου ολοκληρωθεί η ορμή (δηλ. 30 ημέρες). Όταν όλα τα στοιχεία στον κατάλογο ανεκτέλεστων εργασιών της ορμής ολοκληρωθούν, παραδίδεται μια νέα έκδοση του συστήματος.

Καθημερινή συνεδρίαση του Scrum(Daily Scrum meeting), οργανώνεται για να παρακολουθείτε η πρόοδος της ομάδας Scrum συνεχώς. Επίσης χρησιμεύει ως συνεδρίαση προγραμματισμού: τι έχει γίνει από την τελευταία συνεδρίαση και τι είναι να γίνει πριν από την επόμενη. Επίσης τα προβλήματα και άλλα μεταβλητά θέματα συζητούνται και ελέγχονται σε αυτήν την σύντομη (περίπου 15 λεπτά) συνεδρίαση που πραγματοποιείται σε καθημερινή βάση. Οποιεσδήποτε ανεπάρκειες ή εμπόδια στις πρακτικές της διαδικασίας ή του τρόπου ανάπτυξης των συστημάτων διερευνούνται, προσδιορίζονται και αφαιρούνται για να βελτιώσουν τη διαδικασία. Ο Scrum master διευθύνει αυτές τις συνεδριάσεις. Εκτός από την ομάδα Scrum μπορεί επίσης να συμμετέχει και η διοίκηση στην συνεδρίαση.

Συνεδρίαση της αναθεώρησης ορμής (Sprint Review meeting). Την τελευταία ημέρα της ορμής, η ομάδα Scrum και ο Scrum master παρουσιάζουν τα αποτελέσματα (δηλ. την επαύξηση του προϊόντος) της ορμής στη διοίκηση, στους πελάτες, στους χρήστες και στον ιδιοκτήτη των προϊόντων σε μια άτυπη συνεδρίαση. Οι συμμετέχοντες αξιολογούν την

επαύξηση στο προϊόν και λαμβάνουν την απόφαση για τις ακόλουθες δραστηριότητες. Η συνεδρίαση της αναθεώρησης μπορεί να αναδείξει νέα στοιχεία ανεκτέλεστης παραγγελίας και να αλλάξει ακόμη και την κατεύθυνση δημιουργίας του συστήματος.

2.4 ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΤΩΝ ΜΕΘΟΔΩΝ

Υπάρχουν τρία κύρια πλεονεκτήματα από τις ευέλικτες τεχνικές: η μείωση του κινδύνου, η βελτίωση του ελέγχου και η βελτίωση των επικοινωνιών. Κάθε ένα από αυτά τα πλεονεκτήματα παρουσιάζονται παρακάτω σύμφωνα με την μελέτη του K.Aguanno(2004).

2.4.1 Μείωση του κινδύνου

Η μείωση του κινδύνου είναι ίσως η σημαντικότερη από τα τρία πλεονεκτήματα των ευέλικτων μεθοδολογιών. Τα πλεονεκτήματα από το βελτιωμένο έλεγχο και τις βελτιωμένες επικοινωνίες οδηγούν κατά ένα μεγάλο μέρος σε μειωμένους κινδύνους και έτσι πρέπει να δώσουμε πρόσθετη έμφαση στη συζήτηση της χρήσης των ευέλικτων τεχνικών για τη μείωση του κινδύνου. Τρείς από τους σημαντικότερους κινδύνους που υπάρχουν είναι:
Η αποφυγή δημιουργίας λάθους προγράμματος
Η αποφυγή της δημιουργίας του σωστού προγράμματος με κακή ποιότητα
Η αποφυγή των συνεχών αναθεωρήσεων του σχεδίου

2.4.2 Βελτίωση του ελέγχου

Εκτός από τη μείωση του κινδύνου, οι ευέλικτες μέθοδοι επιτρέπουν στο διευθυντή προγράμματος να βελτιώσει τον έλεγχο του προγράμματος. Αυτό μπορεί να ηχεί λίγο αντιφατικό, αλλά στην πράξη ισχύει: η χρησιμοποίηση μιας λιγότερο άκαμπτης, δομημένης προσέγγισης που επιτρέπει στον διευθυντή για να ελέγξει καλύτερα το πρόγραμμα σε καταστάσεις που θα υπάρχουν υψηλά επίπεδα αλλαγής στο πρόγραμμα. Στην πραγματικότητα, οι ευέλικτοι μέθοδοι επιτρέπουν στον διευθυντή να ανταποκριθεί καλύτερα και να προσαρμοστεί στις μεταβαλλόμενες απαιτήσεις με έναν τρόπο που δεν οδηγεί σε συνεχείς αναθεωρήσεις του σχεδίου, σε αντιπαραθετικές διαπραγματεύσεις συμβάσεων και σε δυσαρεστημένους συμμετέχοντες.

2.4.3 Βελτίωση των επικοινωνιών

Τέλος, οι ευέλικτοι μέθοδοι ενθαρρύνουν την καλύτερη επικοινωνία μεταξύ των μελών ομάδων από τις παραδοσιακές μεθόδους ανάπτυξης προϊόντων. Οι επικοινωνίες είναι βασικές σε οποιοδήποτε πρόγραμμα και όλοι οι διευθυντές προγράμματος υπάρχουν για να διευκολύνουν τις επικοινωνίες είτε αυτό είναι να επικοινωνούν με τη κατάσταση του προγράμματος, είτε να συζητούν τις λύσεις στα προβλήματα, να δηλώσουν τις αλληλεξαρτήσεις ή να εξασφαλίσουν ότι ο καθένας μοιράζεται το ίδιο όραμα και την ίδια στρατηγική για το πρόγραμμα. Ένα καλά σχεδιασμένο πρόγραμμα επικοινωνιών μεγιστοποιεί τη ροή πληροφοριών και γνώσης.

2.5 ΣΥΓΚΡΙΤΙΚΗ ΑΞΙΟΛΟΓΗΣΗ ΤΟΥ ΑΚΡΑΙΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΚΑΙ SCRUM

Οι ευέλικτες μεθοδολογίες ανάπτυξης λογισμικού,(Qumer and Henderson-Sellers,2006) όπως ο ακραίος προγραμματισμός και ο Scrum απαιτούν να παραδώσουν μια γρήγορη λύση λογισμικού σε έναν πελάτη με ταχέως μεταβαλλόμενες απαιτήσεις. Παρά τις εν λόγω απαιτήσεις, οι επιχειρήσεις συχνά εξακολουθούν να είναι απρόθυμες στο να υιοθετήσουν και να εισαγάγουν ένα ευέλικτο ύφος της ανάπτυξης στις οργανώσεις τους. Το επίκεντρο της έρευνας είναι η σύγκριση και η ανάλυση των δύο ευέλικτων μεθόδων ανάπτυξης λογισμικού (XP και SCRUM) χρησιμοποιώντας τα 4-Διαστατικά αναλυτικά εργαλεία (4DAT) με βάση τις τέσσερις διαστάσεις :

1η διάσταση(πεδίο εφαρμογής),

2η διάσταση(ευελιξία),

3η διάσταση (ευέλικτες αξίες) και

4η διάσταση (διαδικασία λογισμικού).

Το 4-DAT αναπτύχθηκε για τους ερευνητές και τους επαγγελματίες με σκοπό την ανάλυση και τη σύγκριση των ευέλικτων μεθόδων.

2.5.1 Πεδίο εφαρμογής του ακραίου προγραμματισμού και scrum

Το πεδίο εφαρμογής των δύο μεθόδων ακραίου προγραμματισμού και Scrum αναλύεται ποσοτικά. Ο ακραίος προγραμματισμός και ο Scrum είναι κατάλληλοι για μικρού και μεσαίου μεγέθους έργα και μπορούν να επεκταθούν και σε μεγαλύτερα έργα. Ο ακραίος προγραμματισμός και Scrum χρησιμοποιούν μια επαναληπτική και αυξανόμενη οριοθετημένη προσέγγιση για την ανάπτυξη προϊόντων λογισμικού, με το μέγεθος της ομάδας να είναι κάτω των 10 ατόμων. Ο κώδικας στον ακραίος προγραμματισμός είναι καθαρός και απλός. Το περιβάλλον τεχνολογίας απαιτεί γρήγορη αναδιοργάνωση. Το φυσικό περιβάλλον απαιτεί συστέγαση των ομάδων ώστε η επικοινωνία να είναι εφικτή ανά πάσα στιγμή. Η επιχειρηματική κουλτούρα απαιτεί συνεργασία και συλλογικότητα. Ο Scrum παραμένει ανενεργός για τα θέματα αυτά καθώς είναι περισσότερο μια μεθοδολογία διοίκησης του προγράμματος.

2.5.2 Βαθμός ευελιξίας σε ακραίο προγραμματισμό και Scrum

Ο βαθμός ευελιξίας (DA) μετράται με βάση τις πέντε μεταβλητές (χαρακτηριστικά): την ευελιξία (FY), την ταχύτητα (SD), την επιείκεια (LS), την μάθηση (LG) και την δυνατότητα ανταπόκρισης (RS), που μπορεί να υπάρχουν σε κάποιο ειδικό επίπεδο ή στην φάση του κύκλου ζωής είτε ως αποτέλεσμα των πρακτικών που χρησιμοποιούνται στις φάσεις του ακραίου προγραμματισμού και Scrum. Η φάση σχεδιασμού στον ακραίο προγραμματισμό δεν είναι σταθερή και μπορεί να αλλάξει καθώς προχωράει η ανάπτυξη λογισμικού έτσι ώστε να φανεί ότι πρόκειται για μία ευέλικτη φάση. Ο σχεδιασμός γίνεται γρήγορα για κάθε θέση και είναι δρομολογημένος, έτσι μπορεί να χαρακτηριστεί ως ταχείς. Το σχέδιο μπορεί να αλλάξει ανά πάσα στιγμή, με αποτέλεσμα ένα επιπλέον κόστος για τον καθορισμό αυτό γι'αυτό το λόγο δεν είναι επιεικές. Ωστόσο, το σχέδιο μπορεί να θεωρηθεί αναδυόμενο στον ακραίο προγραμματισμό, με τη μάθηση από τις προηγούμενες εκδόσεις. Τέλος, το σχέδιο ανταποκρίνεται σε κάθε ζήτηση. Ο βαθμός ευελιξίας στον ακραίο προγραμματισμό είναι υψηλός σε όλες τις κατηγορίες εκτός από την επιείκεια, παρόμοια έλλειψη παρατηρείται και στο Scrum ο οποίος έχει ακόμα χαμηλότερες τιμές στο χαρακτηριστικό της επιείκειας.

2.5.3 Ευέλικτες αξίες σε ακραίο προγραμματισμό και Scrum

Η σύγκριση στο πλαίσιο των ευέλικτων αξιών είναι ποιοτική. Ο ακραίος προγραμματισμός και ο Scrum προσφέρουν υποστήριξη για όλες τις ευέλικτες αξίες. Ο ακραίος προγραμματισμός δεν προσφέρει υποστήριξη στο να κρατά την διαδικασία ευέλικτη ή στο να κρατάει το κόστος της διαδικασίας αποτελεσματικό. Αυτό το τελευταίο χαρακτηριστικό δεν είναι ορατό και στο Scrum.

2.5.4 Διαδικασία λογισμικού σε ακραίο προγραμματισμό και Scrum

Σε αυτό το πλαίσιο της αξιολόγησης εξετάζονται(Fernandes and Almeida) οι πρακτικές των δύο μεθόδων για την υποστήριξη των διαδικασιών λογισμικού. Όπως και με τις ευέλικτες τιμές, η σύγκριση είναι καθαρά ποιοτική και ενημερωτική (περιγραφική και όχι περιοριστική). Οι πρακτικές και των δύο μεθόδων προσφέρονται για την ανάπτυξη και την διεργασία του έργου, αλλά δεν λένε τίποτα για τη διαχείριση της διάρθρωσης ή της διαδικασίας διαχείρισης. Η διαδικασία ανάπτυξης στον ακραίο προγραμματισμό έχει απλή σχεδίαση, οι προγραμματιστές εργάζονται ανά ζεύγη, γίνονται συνεχής δομικές και τέλος ο πελάτης μπορεί να εκφράσει την γνώμη του. Στο Scrum η διαδικασία ανάπτυξης πραγματοποιείται από την ομάδα του Scrum και ο επαναληπτικός κύκλος αναθεωρείται συνέχεια.

2.6 Επίλογος

Σε αυτό το κεφάλαιο έχουν αναλυθεί και συγκριθεί δύο ευέλικτες μεθοδολογίες ανάπτυξης λογισμικού(XP και SCRUM). Η ανάλυση των μεθόδων πραγματοποιήθηκε με μια ποιοτική όσο και με μια ποσοτική προσέγγιση για την αξιολόγηση τους τόσο σε επίπεδο φάσης όσο και σε επίπεδο πρακτικής. Ο ακραίος προγραμματισμός μπορούμε να πούμε ότι έχει πιο ευέλικτες φάσεις αλλά λιγότερο ευέλικτες πρακτικές από το Scrum . Στο κεφάλαιο 3 θα δούμε πώς ο ακραίος προγραμματισμός και ο Scrum εφαρμόζονται στην πράξη.

3. ΕΦΑΡΜΟΓΗ ΤΩΝ ΕΥΕΛΙΚΤΩΝ ΜΕΘΟΔΩΝ XP ΚΑΙ SCRUM

3.1 Εισαγωγή

Τα τελευταία χρόνια το ενδιαφέρον που υπάρχει για την χρησιμοποίηση ευέλικτων μεθόδων ανάπτυξης λογισμικού όπως και οι συζητήσεις έχουν αυξηθεί εντυπωσιακά-όπως έχουν αυξηθεί και τα στοιχεία για την αποτελεσματικότητα τους σε ορισμένα περιβάλλοντα και σε συγκεκριμένους τύπους προγραμμάτων. Ωστόσο παραμένει ακόμη ασαφές σε ποιά ακριβώς περιβάλλοντα και πλαίσια μπορεί να λειτουργήσουν οι ευέλικτες μέθοδοι(J. Bowers et al, 2002).

Αν και οι περισσότερες εταιρείες γνωρίζουν τις μελέτες που δείχνουν ότι οι ευέλικτες μεθοδολογίες πράγματι λειτουργούν, παραμένουν μη πεπεισμένες ότι αυτές οι πρακτικές θα λειτουργήσουν για τις ίδιες. Για παράδειγμα, θα πρέπει να αξιολογήσουν εάν οι πρακτικές των ευέλικτων μεθόδων θα αυξήσουν την παραγωγικότητα και θα μειώσουν τον κύκλο ζωής, διατηρώντας το τρέχον επίπεδο ποιότητας και συντηρησιμότητας. Επιπλέον, πρέπει να εξετάσουν εάν θα μπορούσαν να χρησιμοποιήσουν τις ευέλικτες πρακτικές για να αναπτυχθούν μεγάλα, σύνθετα και κρίσιμα για την ασφάλεια συστήματα που θα μπορούσαν να διατηρούνται για δεκαετίες. Επίσης, πρέπει να αξιολογήσουν εάν θα μπορούσαν να χρησιμοποιήσουν τις πρακτικές για να μειώσουν την ανάπτυξη των προϊόντων. Στην έρευνα που έχει γίνει για την χρησιμοποίηση του ακραίου προγραμματισμού και Scrum σε διάφορες οργανώσεις οι υπεύθυνοι των οργανώσεων χρησιμοποίησαν και προσάρμοσαν τον ακραίο προγραμματισμό και Scrum με κάποιο τρόπο, είτε επέλεξαν κάποιες πρακτικές των μεθόδων και τις ενσωμάτωσαν στις κανονικές διαδικασίες τους.

Οι οργανώσεις πρέπει να εξετάσουν διάφορες πτυχές των αποτελεσμάτων εισαγωγής των ευέλικτων μεθόδων. Από επιχειρηματική άποψη είναι ουσιαστικό, να παραδώσουν το υψηλής ποιότητας λογισμικό εγκαίρως και στα πλαίσια του εκτιμώμενου κόστους. Όταν εφαρμόζονται ευέλικτες πρακτικές το να γίνεται αντιληπτό ποιες πτυχές της διαδικασίας έγιναν πιο ευέλικτες είναι εξίσου σημαντικό. Άλλες πτυχές που πρέπει να εξεταστούν περιλαμβάνουν το πόσο δύσκολη θα είναι η εισαγωγή και η στήριξη των ευέλικτων πρακτικών, οι επιθυμητές και ανεπιθύμητες παρενέργειες τους και η ικανοποίηση των υπαλλήλων με τη χρησιμοποίηση των μεθόδων.

3.2 Εμπειρικές μελέτες

Η Fst είναι μία εταιρεία(Murru et al, 2003) που προσφέρει υπηρεσίες διαδικτύου και χρησιμοποιούσε για την διαχείριση του έργου την Rational Unified Process (RUP) μια επαναληπτική διαδικασία ανάπτυξης λογισμικού. Από τον Φεβρουάριο του 2001 μελέτησε τον ακραίο προγραμματισμό ως βελτίωση στις τρέχουσες πρακτικές της, ελπίζοντας ότι θα βελτιώσει την χαμηλή ποιότητα λογισμικού. Χρησιμοποίησε τον ακραίο προγραμματισμό για την σχεδίαση και την διαχείριση έργων και τον είδε ως μια φιλοσοφία διαχείρισης του κινδύνου. Το έργο διήρκησε περίπου τρείς μήνες και απασχόλησε τρία άτομα. Ο διευθυντής έρευνας διαδραμάτισε τον ρόλο του πελάτη. Η ομάδα ανάπτυξης δεν είχε μεγάλη εμπειρία. Η ομάδα δούλεψε σε ένα ανοικτό χώρο εργασίας. Οι προγραμματιστές ισχυρίζονται ότι ο ακραίος προγραμματισμός είναι πολύ χρήσιμος για την ανάθεση των κατάλληλων προτεραιοτήτων στα διάφορα στάδια της διαδικασίας ανάπτυξης λογισμικού. Η εμπειρία δείχνει ότι ο ακραίος προγραμματισμός δεν είναι «μαγεία» αλλά μπορεί να λειτουργήσει καλά. Αν μια ομάδα δεν έχει εμπειρία προγραμματισμού τα αποτελέσματα χρησιμοποιώντας ακραίο προγραμματισμό θα είναι στην καλύτερη περίπτωση οριακά καλύτερα από αυτό που θα προέκυπτε από οποιοδήποτε άλλη μεθοδολογία.

Δύο μελέτες περιπτώσεων πραγματοποιήθηκαν σε εταιρείες ανάπτυξης λογισμικού στις Φιλιππίνες(Sison and Yang, 2007) χρησιμοποιώντας ακραίο προγραμματισμό και Scrum για την παραγωγή λογισμικού. Η πρώτη περίπτωση αφορά μια μικρή εταιρεία λογισμικού που ανέλαβε την ανάπτυξη λογισμικού μιας κορυφαίας αυτοκινητοβιομηχανίας χρησιμοποιώντας ακραίο προγραμματισμό. Οι πέντε πρακτικές ακραίου προγραμματισμού (μικρές εκδόσεις, μεταφορά, απλή σχεδίαση, εβδομαδιαίος κύκλος και τα πρότυπα της κωδικοποίησης) εφαρμόστηκαν ακριβώς όπως είναι χωρίς καμία παραλλαγή. Ωστόσο οι υπόλοιπες πρακτικές (η πραγματική συμμετοχή πελάτη, προγραμματισμός ανά ζεύγη, συλλογική ιδιοκτησία, δοκιμή με γνώμονα την ανάπτυξη, ανατροφοδότηση και συνεχείς ενσωμάτωση) δεν τέθηκαν σε εφαρμογή έτσι ακριβώς όπως παρουσιάζονται αλλά προσαρμόστηκαν με κάποιες παραλλαγές στις διαδικασίες της εταιρείας. Η πραγματική συμμετοχή του πελάτη δεν ήταν εφικτή γιατί υπάρχουν δύο

περιοχές: η ΗΠΑ(για την ανάλυση και σχεδιασμό) και οι Φιλιππίνες(για την ανάπτυξη). Ως εκ τούτου το παιχνίδι σχεδιασμού έπρεπε να διεξαχθεί μέσω ηλεκτρονικού ταχυδρομείου. Υπήρξαν πάντως πολλές επισκέψεις που έγιναν από εκπροσώπους πελατών και από την εταιρεία αντίστροφα. Ο προγραμματισμός ανά ζεύγη εκτελέστηκε κατά την έναρξη του πρώτου έργου με τους έμπειρους προγραμματιστές σε συνδυασμό με τους νέους προγραμματιστές. Η ένωση αυτή ήταν δυναμική γιατί οι νέοι προγραμματιστές έμαθαν πολλά από τους παλαιότερους. Επιπλέον, αναπτύχθηκε συντροφικότητα μεταξύ των μελών της ομάδας ανάπτυξης. Η συλλογική ιδιοκτησία του κώδικα ασκήθηκε κατά την έναρξη του πρώτου έργου. Ωστόσο, ήταν σχεδόν απούσα προς το τέλος του έργου. Η ανατροφοδότηση γινόταν όταν αυτό ήταν δυνατόν. Η συνεχής ολοκλήρωση δεν ασκήθηκε όπως περιγράφεται κατά την οποία ο κώδικας είναι ενσωματωμένος και δοκιμάζεται μετά από μερικές ώρες ή μετά από μία ημέρα της ανάπτυξης. Αντ΄ αυτού η ένταξη πραγματοποιήθηκε όταν οι μονάδες είχαν ολοκληρωθεί. Τα αποτελέσματα της εμπειρίας χρησιμοποιώντας ακραίο προγραμματισμό έδειξαν ότι είναι αποτελεσματικός για την γρήγορη παραγωγή λογισμικού που ικανοποιεί τις απαιτήσεις των πελατών. Βρέθηκε επίσης ότι οι επιχειρήσεις οδηγούνται σε μεγαλύτερη μάθηση(μέσω της σιωπηρής μετάδοσης της γνώσης) και την μεγαλύτερη ομαδική εργασία(λόγω των συνεχών υγιών αλληλεπιδράσεων).

Η περίπτωση 2 περιλαμβάνει ένα παιχνίδι ανάπτυξης που ιδρύθηκε το 2001. Τον Αύγουστο του 2006 στη μέση της προ-παραγωγής του νέου παιχνιδιού η ομάδα ανάπτυξης αποφάσισε να στρέψει την διαδικασία ανάπτυξης από το μοντέλο του καταρράκτη σε Scrum. Η αλλαγή σε Scrum είχε το κίνητρο κατά κύριο λόγο την ανάγκη για μείωση των υπερωριών και την μείωση των επισκευών. Η μετάβαση σε Scrum έλαβε χώρα γρήγορα. Ουσιαστικά, το προσωπικό ενημερώθηκε από τον επικεφαλή του έργου για την μεθοδολογία. Διαμορφώθηκαν τρεις ομάδες με 3-4 μέλη η κάθε μία. Η πρώτη ομάδα εργάστηκε σχετικά με τα επίπεδα παιχνιδιού, η επόμενη εργάστηκε για τους χαρακτήρες και η τελευταία ασχολήθηκε με την μηχανή του παιχνιδιού. Κάθε ομάδα είχε μία σχεδιάστρια, έναν ή δύο 3D σχεδιαστές και μια κοινή προγραμματίστρια. Η ομάδα της μηχανής του παιχνιδιού αποτελούνταν μόνο από προγραμματιστές. Από τις πρακτικές του Scrum το προσωπικό φάνηκε

να εκτιμάει γρήγορα το sprint συγκεκριμένα μικρής διάρκειας(από ένα μήνα). Οι μικρές διατμηματικές ομάδες έχουν τον τελευταίο λόγο για τους δικούς τους στόχους sprint όσον αφορά τις εκκρεμότητες και τις συνεδριάσεις επανεξέτασης sprint. Η σύντομη διάρκεια των ορμών (σε σύγκριση με την προηγούμενη κατασκευή που ήταν περίπου 2-3 μήνες) και οι σύντομες συναντήσεις εξέτασης στο τέλος ενός αγώνα ταχύτητας επέτρεπε τις ομάδες και τους διαχειριστές να ανακαλύπτουν νωρίς τις αδυναμίες στο σχεδιασμό του παιχνιδιού υπό ανάπτυξη. Οι σύντομες καθημερινές συναντήσεις Scrum ήταν η άλλη πρακτική που εκτίμησε το προσωπικό. Αυτές οι ημερήσιες συνεδριάσεις επέτρεπαν τον Scrum master να παρακολουθεί την πρόοδο της ομάδας του πιο συχνά από ότι πριν και επίσης επέτρεψαν στο προσωπικό να ανταλλάσουν μεταξύ τους απόψεις για οτιδήποτε θέμα ή πρόβλημα παρουσιαζόταν ώστε να λάβουν γρήγορες απαντήσεις. Οι συναντήσεις αύξησαν επίσης την συλλογική ομάδα και την συντροφικότητα. Στο τέλος του έργου το προσωπικό αντιλήφθηκε ότι η παραγωγικότητα είχε όντως αυξηθεί λόγω των λιγότερων επαναλήψεων που με τις σειρά τους οφείλονται στις μικρές ορμές και τις συνεδριάσεις απολογισμού. Ένα θετικό αποτέλεσμα από την χρήση Scrum ήταν η βελτίωση της ομαδικής εργασίας.

Από το πιλοτικό πρόγραμμα της ABB που χρησιμοποίησε ακραίο προγραμματισμό προέκυψε ότι το προϊόν ικανοποιούσε τις αρχικές προδιαγραφές του και το πρόγραμμα είχε μόνο ελαφρώς υπερέβη τον απαραίτητο χρόνο παράδοσης(A. Domingo et al, 2004). Ο κώδικας είχε υψηλή ποιότητα χάριν και στον προγραμματισμό ανά ζεύγη που απέτρεψε το σύνθετο σχεδιασμό και τις αυτοματοποιημένες δοκιμές οι οποίες απέτρεψαν την εισαγωγή λαθών. Ο προγραμματισμός ανά ζεύγη βοήθησε να διαδοθούν οι πληροφορίες και η γνώση σε όλη την ομάδα και οι καθημερινές συνεδριάσεις βελτίωσαν την πειθαρχεία στην εργασία. Η συνεχής ολοκλήρωση βοήθησε στην βελτίωση της ποιότητας των αλλαγών με την γρήγορη αποκάλυψη των προβλημάτων ολοκλήρωσης. Έδειξε επίσης ότι όταν συνδυάζεται με τις μικρές απελευθερώσεις εγγυάται τη σταθερή διαθεσιμότητα ενός εκτελέσιμου συστήματος. Κατά συνέπεια, η ομάδα θα μπορούσε πάντα να παραδώσει το εν λειτουργία λογισμικό όποτε χρειαστεί. (A. Domingo et al, 2004). Η ABB κατέληξε στο συμπέρασμα ότι οποιαδήποτε οργάνωση που αναπτύσσει τα μικρά στοιχεία συστημάτων με μικρές ομάδες

θα μπορούσε να δοκιμάσει τον ακραίο προγραμματισμό. Τίποτα βέβαια δεν αποτρέπει τις μεγάλες οργανώσεις από το να δοκιμάσουν τον ακραίο προγραμματισμό σε μικρή κλίμακα.

Η εμπειρία στην Daimler-Chrysler έδειξε ότι η χρησιμοποίηση ακραίου προγραμματισμού συνδυασμένος με συνεχείς δοκιμές και άλλες κλασσικές τεχνικές διασφάλισης της ποιότητας παρήγαγε υψηλής ποιότητας λογισμικό σε όλα τα προγράμματα της. (A. Domingo et al, 2004).

Η ομάδα ανάπτυξης της Daimler-Chrysler ανέφερε ότι χρησιμοποιώντας ακραίο προγραμματισμό περιέκοψε τις δαπάνες επιτυγχάνοντας υψηλά επίπεδα ποιότητας λογισμικού και την ικανοποίηση των πελατών. Επίσης θεώρησε το πρόγραμμα μια επιτυχία που οφείλεται στο μεγαλύτερο μέρος του στην υιοθέτηση των πρακτικών του ακραίου προγραμματισμού. Η επικοινωνία μεταξύ των μελών προγράμματος και του πελάτη βελτιώθηκε ουσιαστικά χάριν σε μια παραλλαγή στην πρακτική της «ολοκλήρωσης ομάδας» και την πρακτική «παιχνιδιών προγραμματισμού»(A. Domingo et al, 2004). Η Daimler-Chrysler συνειδητοποίησε ότι η μείωση του χρόνου για την εισαγωγή του προϊόντος στην αγορά έχει γίνει όλο και πιο δύσκολη. Επίσης αναγνωρίζει ότι οι ώριμες διαδικασίες απαιτούν τις δραστηριότητες όπως η ποιότητα διαχείρισης, η τεκμηρίωση και η μέτρηση.

Στα πιλοτικά προγράμματα της Motorola τα στοιχεία δείχνουν ότι τα προγράμματα πέτυχαν ποιοτικά επίπεδα συγκρίσιμα με καλύτερα από άλλες διαδικασίες αν και ακολούθησαν όλα διαφορετικές διαδικασίες(J.Broewrs et al, 2002). Παρουσίασαν σχετικά συνεπή αποτελέσματα βασισμένα στα συλλεχθέντα στοιχεία και στις ποιοτικές και ποσοτικές πτυχές της διαδικασίας. Η εμπειρία στην Motorola έδειξε ότι ο ακραίος προγραμματισμός δεν είναι μια διαδικασία ανάπτυξης λογισμικού που ταιριάζει ομοιότυπα στα πάντα. Ένα πρόγραμμα της Motorola αντιμετώπισε κινδύνους όταν η αναδόμηση του κώδικα δημιούργησε σημαντικές ατέλειες προγράμματος. Η ομάδα ανάπτυξης της Motorola πείσθηκε από την επιτυχία των πιλοτικών προγραμμάτων ότι είναι δυνατόν να χρησιμοποιηθεί ακραίος προγραμματισμός για την ανάπτυξη μεγάλων, σύνθετων και κρίσιμων για την ασφάλεια συστήματα με μακροχρόνιους κύκλους ζωής.

Η εμπειρία στη Nokia έδειξε ότι η εισαγωγή του ακραίου προγραμματισμού χωρίς εκτενή προσαρμογή σε μια μεγάλη οργάνωση είναι γενικά

απραγματοποίητη. Ο τρόπος προσαρμογής των πρακτικών του ακραίου προγραμματισμού και η υποστήριξη της επικοινωνίας των ομάδων ήταν ένα σημαντικό μάθημα στην Nokia. Η ομάδα της Nokia θεώρησε την ανατροφοδότηση του κώδικα έναν παράγοντα στην επιτυχία ενός προγράμματος επειδή οι αρχιτεκτονικές που παρέμειναν στο πρόγραμμα, αναδομούσαν συνεχώς την αρχιτεκτονική και κατάφεραν την επιβίωση και την εξέλιξη της αρχιτεκτονικής (J. Vanhanen et al, 2003). Η Nokia σημειώνει ότι οι μικρές ομάδες ανάπτυξης λογισμικού είναι παραγωγικότερες από τις μεγάλες. Κατά συνέπεια, προσπαθούν να εφαρμόσουν την πλέον κατάλληλη μεθοδολογία ανάπτυξης για κάθε έργο και θεωρούν τις ευέλικτες μεθοδολογίες ως ένα άλλο εργαλείο στην εργαλειοθήκη βελτίωσης της διαδικασίας λογισμικού. Η εταιρεία αποφάσισε ότι ο ακραίος προγραμματισμός λειτουργεί καλύτερα για μικρά, ανεξάρτητα προγράμματα στα οποία όλοι οι συμμετέχοντες βρίσκονται στον ίδιο χώρο και ότι η χρησιμοποίηση των επιλεγμένων ευέλικτων τεχνικών θα γίνονται όλο και πιο συχνά.

Σε μια έρευνα που έγινε σε μια παγκόσμια εταιρεία(Cristal et al, 2008) που υδρεύει στην Αμερική και Ασία ο Scrum έχει γίνει αποδεκτός από την εν λόγω εταιρεία διότι περιλαμβάνει τη διαχείριση του έργου ως μέρος από τις πρακτικές. Ο Scrum έχει χρησιμοποιηθεί με στόχο την απλούστευση του ελέγχου του έργου μέσα από απλές διαδικασίες. Τα πρώτα αποτελέσματα ήταν πολύ θετικά και τα μελλοντικά σχέδια περιλαμβάνουν τη χρήση του Scrum με πιο συστηματικό τρόπο μέσω της θεσμοθέτησης των ευέλικτων πρακτικών σε παγκόσμιο επίπεδο εντός της εταιρείας.

Το Φθινόπωρο του 2001 το πανεπιστήμιο της Λιουμπλιάνας στην Σλοβενία(Mahnic and Drnovscek, 2005) δημιούργησε ένα πιλοτικό πρόγραμμα για την ανάπτυξη λογισμικού για το πληροφοριακό σύστημα του πανεπιστημίου χρησιμοποιώντας Scrum. Τα μέλη της ομάδας βρήκαν το Scrum πολύ χρήσιμο. Η χρήση του Scrum βελτίωσε την επικοινωνία μεταξύ τους και μεγιστοποιήθηκε η συνεργασία.

Επίσης, αύξησε τα κίνητρα τους και την ευθύνη για την επιτυχία του έργου. Από την άλλη πλευρά, τους έδωσε την ελευθερία να εκμεταλλευτούν το μέγιστο ταλέντο και την γνώση τους κατά τη διάρκεια κάθε ορμής. Ήταν σε θέση να οργανώνουν την εργασία τους από μόνοι τους εξετάζοντας τις

προτιμήσεις τους και τις ειδικές γνώσεις. Στο τέλος του έργου αισθάνθηκαν καλά για την δουλειά τους, τις εισφορές τους και πίστευαν ότι είχαν κάνει ότι καλύτερο μπορούσαν.

Μια ακόμη έρευνα έγινε για τρείς εταιρείες στην Νορβηγία(Brekkan and Mathisen 2010), οι οποίες ανέπτυξαν το δικό τους λογισμικό και χρησιμοποίησαν Scrum. Με βάση την έρευνα διαπιστώθηκε ότι τα διευθυντικά στελέχη ήταν ενθουσιασμένα με τον scrum. Όλες οι εταιρείες ανάμεναν να αυξήσουν την παραγωγικότητα, όταν άρχισαν να χρησιμοποιούν το Scrum, αλλά στην έρευνα διαπιστώθηκε ότι όλες οι εταιρείες διαμαρτυρήθηκαν ότι δεν είχαν επιτύχει την παραγωγικότητα. Από την έρευνα προέκυψε ότι η εφαρμογή του Scrum που χρησιμοποιήθηκε από τις τρείς εταιρείες ήταν μια μέθοδο που δεν αναλύθηκε διεξοδικά. Δοκιμάστηκε στις εταιρείες πριν από την καθιέρωση και την εφαρμογή, αν και αναφέρεται ότι η νέα μέθοδος θα πρέπει να αξιολογείται πριν την εφαρμογή για να διαπιστωθεί αν ταιριάζει στην εταιρεία. Αυτό που είναι σαφές, ωστόσο είναι ότι η παραγωγικότητα και η αποτελεσματικότητα, όταν χρησιμοποιείται ο Scrum πρέπει να διερευνηθούν περαιτέρω.

Στο πανεπιστήμιο του Maryland University College(UMUC) το φθινόπωρο του 2008 προσαρμόστηκε μια σειρά από ευέλικτες μεθόδους για την πραγματοποίηση κάποιου μεταπτυχιακού μαθήματος(Rico and Sayani, 2009). Τρείς ομάδες των πέντε φοιτητών κλήθηκαν να χρησιμοποιήσουν ευέλικτες μεθόδους για την κατασκευή αντίστοιχης ιστοσελίδας ηλεκτρονικού εμπορίου(e-commerce site). Με λίγη εκπαίδευση σε ευέλικτες μορφές μεθόδων, σε εικονικές ομάδες και εργαλεία συνεργασίας καθεμία από τις τρείς ομάδες ολοκλήρωσε πλήρως από μια ιστοσελίδα ηλεκτρονικού εμπορίου σε λίγο περισσότερο από 13 εβδομάδες. Οι ομάδες έχτισαν μια τέλεια ισορροπία συνεργασίας με τον πελάτη, με την χρήση των ευέλικτων μεθόδων και την τεχνική ικανότητα του προγραμματισμού με αποτέλεσμα να υπάρχει μια ικανή παραγωγικότητα και καλή ποιότητα της ιστοσελίδας. Η πρώτη ομάδα ονομάστηκε Awesomesauce και είχε την μεγαλύτερη προβολή σε ευέλικτες μεθόδους καθώς και τους καλύτερους προγραμματιστές υπολογιστών. Επέλεξαν τα χρησιμοποιήσουν Scrum ως ευέλικτο τρόπο και το εργαλείο versionone για την διαχείριση του έργου. Η ομάδα όπως και οι υπόλοιπες διαδόθηκαν σε τέσσερις χρονικές ζώνες. Ο πελάτης άρχισε τις επαφές με την

ομάδα και τους παρείχε μια ευρεία θεώρηση του έργου καθώς και μια λίστα με τις αρχικές ιστορίες χρηστών. Θα αποσυνθέσει τις ιστορίες χρηστών σε εργασίες ανάπτυξης και την ανεκτέλεστη ορμή χαμηλού επιπέδου σε τεχνικές δραστηριότητες. Με άλλα λόγια, οι ιστορίες χρηστών φάνηκε να εξαφανίζονται και ο καθορισμός προτεραιοτήτων ήταν ασαφής. Η πρώτη τους επανάληψη ήταν η πιο ανώριμη και από τις τρείς ομάδες. Ωστόσο, ολοκληρώθηκε σχεδόν η ιστοσελίδα ηλεκτρονικού εμπορίου με το τέλος της δεύτερης επανάληψης. Η ομαδική εργασία είναι ένα από τα πλεονεκτήματα της ομάδας. Οι περισσότεροι προγραμματιστές είχαν μεγάλο ταλέντο και επιδεξιότητα. Όλοι στην ομάδα είχαν εμπειρία στις ευέλικτες μεθόδους. Η ομάδα ολοκλήρωσε τις περισσότερες ιστορίες χρηστών. Ένα από τα μειονεκτήματα της ομάδας ήταν ότι απέρριπτε τις αρχικές ιστορίες χρηστών. Η επαφή με τον πελάτη ερχόταν πάντα τελευταία. Η ομάδα χρησιμοποίησε λιγότερες ευέλικτες πρακτικές.

Η δεύτερη ομάδα ονομάστηκε Kestrel είχε κάποια έκθεση σε ευέλικτες μεθόδους και είχε τους δεύτερους πιο ικανούς προγραμματιστές. Η δεύτερη ομάδα χρησιμοποίησε επίσης Scrum. Ο πελάτης άρχισε τις επαφές με την ομάδα του με την οποία μοιράζεται το όραμα του για την ιστοσελίδα και έδωσε μια λίστα με τις ιστορίες χρηστών. Ο Scrum master ασχολείται συστηματικά με την καθοδήγηση κατά μήκος της διαδρομής για την τελική επιτυχία του έργου. Όλες οι ομάδες ενθάρρυναν την επικοινωνία με τον πελάτη για οποιαδήποτε απορία σχετικά με τις ιστορίες χρηστών, πράγμα και το οποίο έπραξαν. Ο επικεφαλής προγραμματιστής ανέπτυξε πλαίσια για την επικύρωση των ιστοριών χρήστη πριν την εφαρμογή. Ο πελάτης ήταν σε επαφή για να εγκρίνει αυτά τα πλαίσια πριν από το τέλος των επαναλήψεων. Η πρώτη ομάδα συχνά έκανε αλλαγές της στιγμής κατά την διάρκεια της εκτέλεσης αντί να χρησιμοποιήσει τα πλαίσια. Οι ομάδες παροτρύνονται επίσης να επικοινωνούν με τον Scrum master με ερωτήσεις σχετικά με τις ευέλικτες μεθόδους, πράγμα το οποίο και έπρατταν. Οι δύο πρώτες ομάδες χρησιμοποίησαν επίσης intranet του πανεπιστημίου για συνεργασία. Η πρώτη επανάληψη στην ομάδα Kestrel ήταν λίγο βασική, αλλά όπως και στην πρώτη ομάδα πήραν τον ρυθμό για να εφαρμόσουν ένα πλήρως λειτουργικό ιστοχώρο ηλεκτρονικού εμπορίου μέχρι το τέλος της τρίτης επανάληψης. Η ομάδα kestrel δεν είχε κατανοήσει τις ευέλικτες μεθόδους. Κατανόησαν όμως από την αρχή τις ιστορίες χρηστών και πρότειναν νέες. Τα αποτελέσματα

μπορεί να ήταν μέτρια αλλά υπήρχε ικανοποιητική συνεργασία. Το μειονέκτημα της ομάδας είναι ότι συνεχίζει να χρησιμοποιεί παραδοσιακές πρακτικές. Η ομάδα στηρίζεται μόνο σε ένα με δύο προγραμματιστές. Τέλος απαιτεί καλύτερη αισθητική της ιστοσελίδας.

Η τρίτη ομάδα ονομάστηκε Yellowstone είχε την μικρότερη έκθεση σε ευέλικτες μεθόδους και τους λιγότερο ικανούς προγραμματιστές. Η τρίτη ομάδα είχε την μεγαλύτερη εμπειρία στην βιομηχανία. Ωστόσο, ένα λιγότερο πεπειραμένο μέλος τους ανέλαβε την ευθύνη για να οδηγήσει την ομάδα. Αυτή η ομάδα χρησιμοποίηση όπως και η δεύτερη ένα Wiki για την ευέλικτη διαχείριση του έργου. Αντίθετα με τις άλλες ομάδες ήταν η πρώτη που επικοινώνησε με τον πελάτη και καθιέρωση στενή και συνεχή επαφή. Δεν είναι ασυνήθιστο για τις ακαδημαϊκές ομάδες να αγωνίζονται με την έννοια της συνεργασίας και αυτή η ομάδα ήταν σίγουρα μία από αυτές. Η ομάδα αυτή εφάρμοσε την πρώτη επιχειρησιακή ιστοσελίδα χρησιμοποιώντας διάφορες συνεργατικές τεχνολογίες. Δημιούργησε επίσης ισχυρή πελατειακή σχέση. Το κύριο μειονέκτημα της ομάδας είναι ότι αγωνίστηκε με ατομικισμό. Απαίτησε πολλές συναντήσεις με τον πελάτη και συχνά υπήρχε απόκλιση από τις ιστορίες του χρήστη. Η ταχύτητα του προγραμματισμού ήταν αρκετά αργή σε σχέση με τις υπόλοιπες ομάδες. Τέλος απαιτεί καλύτερη αισθητική της ιστοσελίδας.

Κάθε μια από τις ομάδες έδειξε ένα πλήρως λειτουργικό ιστοχώρο ηλεκτρονικού εμπορίου. Οι ομάδες είχαν ενθαρρύνει την εκτέλεση των επαναλήψεων για οποιοδήποτε ζήτημα πριν από την δοκιμή αποδοχής. Όλες οι ιστορίες χρηστών ήταν σε σειρά προτεραιότητας πριν, μετά και κατά την διάρκεια των επαναλήψεων. Οι ομάδες με την υψηλότερη ταχύτητα προγραμματισμού είχαν το μικρότερο ποσοστό αλληλεπίδρασης με τους πελάτες. Αυτές οι ομάδες χρησιμοποίησαν επίσης λιγότερες ευέλικτες πρακτικές. Οι ομάδες με την χαμηλότερη ταχύτητα προγραμματισμού είχαν το υψηλότερο ποσοστό αλληλεπίδρασης με τους πελάτες, αλλά χρησιμοποίησαν ευέλικτες πρακτικές όπως και εικονικές συνεδριάσεις προγραμματισμού ανά ζεύγη. Οι ομάδες που παραπονέθηκαν σχετικά με την ομαδική εργασία που απαιτείται περισσότερη εποπτεία είχαν χαμηλότερη ταχύτητα. Οι ομάδες παρουσίασαν θεαματική βελτίωση μετά από κάθε επανάληψη, λόγω της ανατροφοδότησης που επισημάνθηκε. Ένας πελάτης ο οποίος ήταν

ευχαριστημένος με τις ευέλικτες μεθόδους ήταν ένας κρίσιμος παράγοντας για την επιτυχία.

Το καλοκαίρι του 2001 πραγματοποιήθηκε μια έρευνα για τα πιλοτικά έργα που χρησιμοποιούν ακραίο προγραμματισμό(Rumpe and Schroder, 2002). Η έρευνα βασίζεται σε 45 ερωτηματολόγια. Στην έρευνα παίρνουν μέρος διάφορες εταιρείες από όλες τις ηπείρους. Οι έρευνα επικεντρώθηκε σε ερωτήσεις που αφορούσαν την εταιρεία, το XP-project και τα μελλοντικά σχέδια της εταιρείας με βάση τον ακραίο προγραμματισμό. Τα αποτελέσματα έδειξαν ότι τα περισσότερα έργα αξιολογήθηκαν με επιτυχία. Το 100% των προγραμματιστών ζήτησαν να ξανά χρησιμοποιήσουν ακραίο προγραμματισμό κατά την επόμενη φάση του έργου, αν θεωρηθεί σκόπιμο. Η συχνή απουσία του πελάτη αναγνωρίστηκε ως μεγάλος κίνδυνος για το έργο. Τα προβλήματα προέρχονταν συχνά από «εμπόδια μνήμης»:η διαχείριση ήταν διστακτική, η φιλοσοφία της εταιρεία δεν επέτρεπε την τοποθέτηση του πελάτη, οι προγραμματιστές αρνιόταν τον προγραμματισμό ανά ζεύγη. Τα πιο χρήσιμα σημεία του ακραίου προγραμματισμού θεωρήθηκαν ο κώδικας κοινής ιδιοκτησίας, ο έλεγχος και η συνεχής ολοκλήρωση. Οι πιο σημαντικοί παράγοντες που έχουν αναφερθεί είναι οι δοκιμές και ο προγραμματισμός ανά ζεύγη. Στο ένα τρίτο των ερωτηθέντων ομάδων τα μέλη τους ήταν έμπειρα σχετικά με την τεχνολογία λογισμικού. Στο 42% των ομάδων η εμπειρία ήταν μικτή(ειδικοί και νεοεισερχόμενοι). Περισσότερες από τις μισές ομάδες δεν είχαν εξωτερικό σύμβουλο ως μέλος της ομάδας σχεδίου μόνο το 21% είχε ένα σύμβουλο. Η διάρκεια των έργων ήταν κατανεμημένη εξίσου με 6 μήνες, ένα χρόνο και μέχρι τρία χρόνια. Το μέγεθος των ομάδων ξεκινούσε από πέντε μέχρι σαράντα άτομα. Το μεγαλύτερο ποσοστό των ομάδων ήταν μέχρι δέκα άτομα. Το πεδίο εφαρμογής ήταν μικτό. Το 73% των συστημάτων που είχαν αναπτυχθεί ήταν εντελώς νέα, ενώ στα υπόλοιπα προστέθηκαν νέες λειτουργίες ή ήταν έργα συντήρησης. Οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν ήταν η Java, C++, Smalltalk, Lisp με ποιο διαδεδομένη την Java. Διαπιστώθηκε ότι ο ακραίος προγραμματισμός είναι πιο αποδοτικός όταν χρησιμοποιεί γλώσσες προγραμματισμού υψηλού επιπέδου. Οι λόγοι που οδήγησαν στην χρησιμοποίηση ακραίου προγραμματισμού ήταν η απογοήτευση από άλλες μεθόδους, το προσωπικό ενδιαφέρον, η καλή εμπειρία από άλλα έργα και η επιθυμία του πελάτη για ακραίο

προγραμματισμό. Η εμπειρία έδειξε ότι όλοι ήταν ευχαριστημένοι με την χρήση ακραίου προγραμματισμού και ήθελαν να υποστηρίξουν ενεργά τον ακραίο προγραμματισμό στο μέλλον. Η δομή της εταιρείας δείχνει επίσης, ότι ο ακραίος προγραμματισμός δεν περιορίζεται στην νέα οικονομία και στον κόσμο του internet, αλλά είναι ελκυστικός σε όλες τις καινοτόμες επιχειρήσεις.
Μια έρευνα έγινε σε διάφορες οργανώσεις στην Σρι Λάνκα για την ευέλικτη διαχείριση έργων πληροφορικής και ποιοι παράγοντες την επηρεάζουν. Η έρευνα έχει διεξαχθεί με ερωτηματολόγια(Jayawardena and Ekanayake, 2010). Οι ερωτηθέντες αποτελούνται από 45 IT διαχειριστές έργου που εργάζονται σε 19 διαφορετικές οργανώσεις στην Σρι Λάνκα. Το 53% των οργανώσεων χρησιμοποιούσαν μέχρι τότε παραδοσιακές μεθόδους για την διαχείριση έργων. Το 13% χρησιμοποιούσε τόσο παραδοσιακές όσο και ευέλικτες μεθόδους και το 16% χρησιμοποιούσε μόνο την ευέλικτη διαχείριση έργου. Δύο κορυφαίες εταιρείες πληροφορικής δήλωσαν ότι το αίτημα των πελατών ήταν ο βασικός παράγοντας που αποφάσισαν να επιλέξουν την ευέλικτη διαχείριση έργου. Εκτός από την απαίτηση του πελάτη οι εταιρείες που χρησιμοποιούσαν και την παραδοσιακή και την ευέλικτη προσέγγιση έπρεπε να αποφασίσουν ποιο μεθοδολογία θα χρησιμοποιήσει για ένα συγκεκριμένο έργο με βάση τις απαιτήσεις του έργου. Κατά την διάρκεια του αρχικού σταδίου εκτέλεσης μια εταιρεία διαπίστωσε ότι τα μέλη της ομάδας δυσκολεύτηκαν να υιοθετήσουν την ευέλικτη προσέγγιση. Ωστόσο, αυτό επιλύθηκε την στιγμή που η ομάδα απέκτησε εμπειρία. Μια άλλη εταιρεία που εργάζεται σε διαφορετικές χρονικές ζώνες δυσκολεύτηκε να διεξάγει τις καθημερινές συναντήσεις. Ωστόσο αυτό ξεπεράστηκε με την αποστολή ενός μηχανικού στον χώρο του πελάτη που βρίσκεται εκτός συνόρων. Μια εταιρεία ισχυρίστηκε ότι με την ευέλικτη διαχείριση έργου είναι πρόκληση το προϊόν να φθάσει σε κάποιο επίπεδο ωριμότητας. Η εταιρεία δήλωσε επίσης ότι οι πτυχές της διαχείρισης της γνώσεις, όπως οι αλγόριθμοι, η διαχείριση των ρυθμίσεων, ο έλεγχος της έκδοσης μπορούν επίσης να είναι πρόκληση για την ευέλικτη διαχείριση έργου. Οι εταιρείες που ασκούν επιτυχώς την ευέλικτη διαχείριση στην Σρι Λάνκα προτείνουν την κατάλληλη εκπαίδευση σε όλα τα επίπεδα της εταιρείας που μπορεί να οδηγήσουν στην επιτυχή προσαρμογή της ευέλικτης διαχείρισης. Είναι απαραίτητο για την ομάδα να γνωρίζει γιατί πρέπει να μάθει να κάνει πράγματα και να μην πηγαίνει στα τυφλά. Οι

εταιρείες πιστεύουν ότι για να έχουν καλύτερη απόδοση στις ευέλικτες μεθοδολογίες είναι αναγκαίο να υπάρχει ένας ειδικός στον τομέα που να έχει καλή αντίληψη για το θέμα. Ωστόσο, από την εμπειρία στις ευέλικτες μεθόδους η Σρι Λάνκα δεν έχει ικανοποιητικές εγκαταστάσεις για την απαιτούμενη εκπαίδευση. Μια εταιρεία αποκάλυψε ότι με την ευέλικτη διαχείριση έργου ήταν σε θέση να παρέχει πολλές ιδέες μέσα σε σύντομο χρονικό διάστημα. Επίσης όλες οι εταιρείες έχουν εντοπίσει μείωση του κόστους στις αρχικές φάσεις του έργου. Μια εταιρεία ανέφερε ότι η μεθοδολογία δεν πρέπει να γίνει μια θρησκεία της οργάνωσης αλλά ένα εργαλείο για την παράδοση επιτυχών έργων. Η ευέλικτη διαχείριση έργων δεν θα λύσει όλα τα προβλήματα στην διαχείριση έργων και δεν φαίνεται να είναι κατάλληλη για όλους τους τύπους έργων και ομάδων. Ωστόσο, οι οργανώσεις πρέπει να εκτιμήσουν προσεκτικά την ετοιμότητα τους πριν από την μετάβαση στο δρόμο για την ευέλικτη διαχείριση έργου.

Πολλαπλές μελέτες περιπτώσεων(Paasivaara et al, 2009) έγιναν για τρία παγκοσμία έργα ανάπτυξης λογισμικού που χρησιμοποίησαν Scrum και ήταν γεωγραφικά κατανεμημένα. Η συλλογή στοιχείων πραγματοποιήθηκε με συνεντεύξεις. Η έρευνα έγινε σε μεγάλες φιλανδικές εταιρείες. Η χρήση των καθημερινών συναντήσεων ήταν σαφώς η πιο σημαντική πρακτική που χρησιμοποιήθηκε από όλες τις περιπτώσεις έργων. Οι συναντήσεις διαρκούσαν περίπου 15 λεπτά. Για τις καθημερινές συναντήσεις των έργων EnergySoftware και PrintCo χρησιμοποιήθηκαν τηλεφωνικές διασκέψεις και web κάμερες γιατί οι ομάδες ήταν διανεμημένες. Ο Scrum master συμμετείχε στην αρχή καθημερινά στις συναντήσεις Scrum αλλά αργότερα συνήθιζε inter-voice συναντήσεις. Στο PrintCo οι συναντήσεις ήταν συχνές και μικρές. Στο EnergySoftware οι συναντήσεις κρατούσαν μόνο λίγα λεπτά. Ο Scrum master ενθάρρυνε πάντα τα μέλη της ομάδας να αναφέρουν πιθανά προβλήματα που αντιμετωπίζουν. Αυτό τους ωφέλησε στο να επικοινωνούν μεταξύ τους και να λύνουν όσα εμπόδια εμφανίζονται. Οι καθημερινές συναντήσεις προβλέπονται επίσης ως ένας καλός τρόπος για όλους για να πάρουν μια γενική εικόνα της κατάστασης του έργου. Στο PrintCo αναφέρθηκε κάποιο πρόβλημα στην επικοινωνία. Τα άτομα της ομάδας δεν μπορούσαν να προσδιορίσουν σωστά τα προβλήματα που υπήρχαν. Το EnergySoftware ήταν το μόνο έργο με κατανεμημένες ομάδες έτσι ήταν και το μόνο που διοργάνωνε εβδομαδιαίες

συναντήσεις. Ένα μέλος από κάθε ομάδα συμμετείχε στην συνεδρίαση. Το επέλεγε η ομάδα και κάθε φορά ήταν διαφορετικό. Εκτός από τους εκπροσώπους της κάθε ομάδας συμμετείχαν και όλοι οι Scrum masters. Οι εβδομαδιαίες συναντήσεις θεωρήθηκαν πολύ χρήσιμες μεταξύ των ομάδων και αποκαλύφθηκαν τα πιθανά προβλήματα από νωρίς. Επίσης άνοιξαν διαύλους επικοινωνίας μεταξύ των ομάδων και ενθαρρύνθηκε η άτυπη επικοινωνία μεταξύ τους. Το μήκος του κάθε sprint στο Scrum διαρκεί συνήθως τέσσερις εβδομάδες. Στις περιπτώσεις των έργων της συγκεκριμένης έρευνας τα sprint διήρκησαν δύο ή τέσσερις εβδομάδες. Στο PrintCo η κανονική διάρκεια επαναλήψεων ήταν δύο εβδομάδες, η οποία θεωρήθηκε ότι ήταν πολύ καλή σε αυτό το πολύ μικρό έργο. Στο EnergySoftware που είναι ένα μεγαλύτερο έργο οι επαναλήψεις διήρκησαν τέσσερις εβδομάδες. Αυτό σημαίνει ότι όλες οι ορμές ξεκινούσαν και τελείωναν την ίδια στιγμή. Η διακύμανση των ημερομηνιών λήξης και αρχής έγινε στο μέγιστο μια-δύο μέρες. Η ομάδα συντήρησης ήταν η μόνη εξαίρεση σ αυτόν τον κύκλο sprint τεσσάρων εβδομάδων. Ο λόγος ήταν ότι καθόριζε τους πελάτες κάθε δύο εβδομάδες. Επίσης στο EnergySoftware σε όλες τις συνεντεύξεις τα άτομα ήταν ικανοποιημένα από το τρέχον μήκος της ορμής. Στο PulpCo το μήκος της ορμής για την ομάδα που βρισκόταν στην χώρα ήταν τεσσάρων εβδομάδων, ενώ για την ομάδα που ήταν εκτός της Φιλανδίας ήταν δύο εβδομάδες. Η ορμή στο PulpCo μερικές φορές επιμηκύνθηκε λόγω των διαφορετικών χρονικών στιγμών που είχαν τα μέλη της ομάδες στις δύο χώρες. Με την παράταση της ορμής οι ομάδες θα μπορούσαν να παραμείνουν συγχρονισμένες. Λόγω του μικρού μήκους της ορμής οι προθεσμίες και στόχοι ήταν αρκετά σαφείς για όλα τα μέλη της ομάδας για το τι πρέπει να γίνει στην επόμενη ορμή. Εκτός από τις κανονικέ ορμές, υπήρχε μια περίπτωση στο PulpCo, η ορμή απελευθέρωσης, που πραγματοποιούταν δύο έως τρείς φορές τον χρόνο. Ο σκοπός του sprint απελευθέρωσης ήταν να ολοκληρώσουν όλα τα στοιχεία και τα προχωρήσουν στην δοκιμή του συστήματος. Το πλεονέκτημα του sprint απελευθέρωσης είναι ότι ο διαχειριστής δίνει μεγαλύτερη προσοχή στο προϊόν και δίνει περισσότερη ανατροφοδότηση. Πριν από την έναρξη κάθε ορμής η σύσκεψη σχεδιασμού ορμής λάμβανε χώρα. Στο PrintCo και EnergySoftware η σύσκεψη σχεδιασμού ορμής είχε κατανεμηθεί δεδομένου ότι οι ομάδες ήταν και αυτές

κατανεμημένες. Στο PulpCo οι συναντήσεις ήταν ξεχωριστές για τις δύο ομάδες. Για όλα τα έργα οι συναντήσεις των ομάδων που ήταν σε απομακρυσμένη περιοχή έγιναν με τηλεδιάσκεψη. Οι συναντήσεις σχεδιασμού του sprint θεωρήθηκαν πολύ χρήσιμες δεδομένου ότι παρέχεται η δυνατότητα προβολής του έργου και στις δύο περιοχές και έδωσαν στα μέλη της ομάδας την δυνατότητα να συμμετέχουν στον σχεδιασμό του. Ωστόσο, οι συναντήσεις κάποιες φορές ήταν κουραστικές λόγω της κακής ποιότητας φωνητικής κλήσης. Επιπλέον ήταν δύσκολο κάθε φορά να καταλάβουν ποιος μιλάει όταν δεν έβλεπαν το πρόσωπο. Επίσης, τα θέματα που έπρεπε να συζητηθούν ήταν πολλές φορές δύσκολο να εξηγηθούν. Οι ανεκτέλεστες ορμές είναι κατάλογοι των αντικειμένων που πρόκειται να αναπτυχθούν. Στην συνάντηση σχεδιασμού ορμής, ένας ιδιοκτήτης προϊόντος με την ομάδα επιλέγει τα στοιχεία ανεκτέλεστου προϊόντος που θα αναπτυχθούν στην επόμενη ανεκτέλεστη ορμή. Οι περιπτώσεις των τριών έργων χρησιμοποιούν διαφορετικά εργαλεία για την διαχείριση των ανεκτέλεστων ορμών. Το EnergySoftware χρησιμοποιεί το εργαλείο Jira. Όλη η ομάδα ήταν ικανοποιημένη από το εργαλείο. Το PrintCo χρησιμοποίησε ένα wiki για τις ανεκτέλεστες ορμές αλλά έψαξε για ένα καλύτερο εργαλείο γιατί θεώρησε ότι το wiki δεν ήταν πραγματικά σχεδιασμένο για τον σκοπό αυτό. Το PulpCo το Team Foundation Server. Μόνο το EnergySoftware ήταν ευχαριστημένο με το εργαλείο που είχε επιλέξει. Τα άλλα δύο έργα είχαν προβλήματα με τα εργαλεία και τις ευθύνες των διάφορων ρόλων. Όλα τα έργα αντιμετώπισαν προβλήματα κατά την έναρξη χρήσης Scrum. Ήταν μία νέα μέθοδος για όλους και έτσι η εκπαίδευση ήταν αναγκαία. Με το Scrum αναμενόταν ανοιχτή επικοινωνία, η οποία ήταν δύσκολη στην αρχή ειδικά στις υπεράκτιες ομάδες. Για παράδειγμα, στο EnergySoftware οι καθημερινές συναντήσεις Scrum ήταν πολύ μικρές στην αρχή και κυρίως για τους ανθρώπους στην Ασία που τους ήταν δύσκολο να αναφέρουν τα εμπόδια. Ωστόσο, η κατάσταση βελτιώθηκε πολύ σε όλα τα έργα όταν οι συμμετέχοντες έμαθαν να ακολουθούν τις Scrum πρακτικές και είδαν τα οφέλη της συχνής και ανοικτής επικοινωνίας. Παρά το γεγονός ότι όλα τα έργα αντιμετώπιζαν προβλήματα στην αρχή του Scrum η συλλογική εμπειρία ήταν πολύ θετική. Όλα τα έργα ήταν πολύ ευχαριστημένα με την απόφαση να αρχίσουν να χρησιμοποιούν αυτή την ευέλικτη μέθοδο. Οι

ευέλικτες πρακτικές θεωρήθηκαν ιδιαίτερα κατάλληλες για κατανεμημένα έργα, ειδικά επειδή προωθούν συχνή και ανοιχτή επικοινωνία.

3.3 Επίλογος

Οι ευέλικτες μεθοδολογίες ανάπτυξης λογισμικού απασχόλησαν ένα σημαντικό μέρος της βιβλιογραφίας και πολλές συζητήσεις. Ωστόσο, η ακαδημαϊκή έρευνα για το θέμα εξακολουθεί να είναι ανεπαρκείς όπου οι περισσότερες δημοσιεύσεις γράφονται από επαγγελματίες. Ωστόσο, πολλοί οργανισμοί εξετάζουν ακόμη το ενδεχόμενο να χρησιμοποιήσουν ευέλικτες πρακτικές όπου κάνουν τον τρόπο από την εκτέλεση και παροχή λογισμικού πιο ευέλικτο. Σκοπός αυτού του κεφαλαίου είναι να δείξει με βάση την έρευνα πως οι ευέλικτες μεθοδολογίες XP και SCRUM έχουν χρησιμοποιηθεί στην διαχείριση πραγματικών έργων, τι αποτελέσματα είχαν και με ποιόν τρόπο βοήθησαν τις εταιρίες στην δημιουργία ενός υγιούς λογισμικού. Όλες οι οργανώσεις έμαθαν την απόλυτη ανάγκη να προσαρμόζουν τον ακραίο προγραμματισμό και Scrum στις ιδιαίτερες απαιτήσεις τους. Οι περισσότερες εταιρείες που χρησιμοποίησαν ακραίο προγραμματισμό και Scrum ανάφεραν ότι η χρησιμοποίηση ευέλικτων μεθόδων συνδυασμένες με συνεχείς δοκιμές παρήγαγαν υψηλής ποιότητας λογισμικό. Τέλος η επικοινωνία των ομάδων ήταν ένα σημαντικό «μάθημα» για αυτούς που τις χρησιμοποιούσαν.

4. ΕΥΕΛΙΚΤΑ ΕΡΓΑΛΕΙΑ ΔΙΑΧΕΙΡΙΣΗΣ ΕΡΓΟΥ

4.1 Εισαγωγή

Υπάρχει ένας μεγάλος αριθμός εργαλείων διαχείρισης έργων με σκοπό να στηρίξει τις ευέλικτες μεθοδολογίες, όπως το Scrum και το XP συμπεριλαμβανομένων αρκετών ανοικτού πηγαίου κώδικα. Η αγορά για τα ευέλικτα εργαλεία διαχείρισης έργου είναι πλέον ώριμη και κορεσμένη με δεκάδες προσφορές τόσο από μικρούς όσο και μεγάλους προμηθευτές. Κορυφαία στην παγκόσμια εμπορική αγορά είναι το VersionOne, Rally, Thoughtworks Mingle και Danube ScrumWorks. Λίγα εργαλεία ανοικτού κώδικα υπήρχαν εδώ και καιρό και πολλά αναδύθηκαν πιο πρόσφατα. Σε σύγκριση με του ανοιχτού κώδικα, τα πρώτα εμπορικά εργαλεία πρόσφεραν καλύτερες δυνατότητες για μεγάλους οργανισμούς και μεγάλα προϊόντα και έργων. Τείνουν επίσης να προσφέρουν περισσότερη ενοποίηση με εφαρμογή τρίτων κατασκευαστών.

4.2 IceScrum

Έννοιες

Τα προϊόντα (ονομάζονται επίσης σχέδια σε ορισμένα σημεία) είναι το υψηλότερο επίπεδο κατασκευής και κάθε ανάπτυξη μπορεί να έχει πολλά προϊόντα. Κάθε προϊόν έχει ένα ανεκτέλεστο και ένα χάρτης πορείας. Ένα ανεκτέλεστο περιέχει χαρακτηριστικά, ιστορίες χρηστών, ελαττώματα και τεχνικές ιστορίες. Ο χάρτης πορείας περιέχει πολλαπλές εκδόσεις, καθεμία από τις οποίες έχει ένα ενιαίο σχέδια διαχείρισης των εκδόσεων. Ένα σχέδιο διαχείρισης των εκδόσεων αποτελείται από πολλαπλές ορμές. Κάθε ορμή περιέχει ιστορίες, που με τις σειρά τους περιέχουν τα καθήκοντα και τις δοκιμές αποδοχής. Εμπόδια μπορούν να ακολουθήσουν για κάθε προϊόν.

Το IceScrum επιτρέπει μόνο μια ενιαία θέση και μια ενιαία ορμή να δραστηριοποιούνται σε ένα χρόνο(για ένα συγκεκριμένο προϊόν), καθιστώντας το ακατάλληλο για τις μεγαλύτερες επιχειρήσεις που χρειάζονται πολλαπλές ταυτόχρονες ορμές με πολλαπλές ομάδες τρέχουν παράλληλα για ένα μόνο προϊόν. Οι χρήστες του IceScrum μπορούν να έχουν οποιοδήποτε από τους ρόλους του Scrum καθώς και προσαρμοσμένους ρόλους που

μπορούν να δημιουργήσουν. Δεν επιτρέπει στους χρήστες να ομαδοποιούνται σε ομάδες.

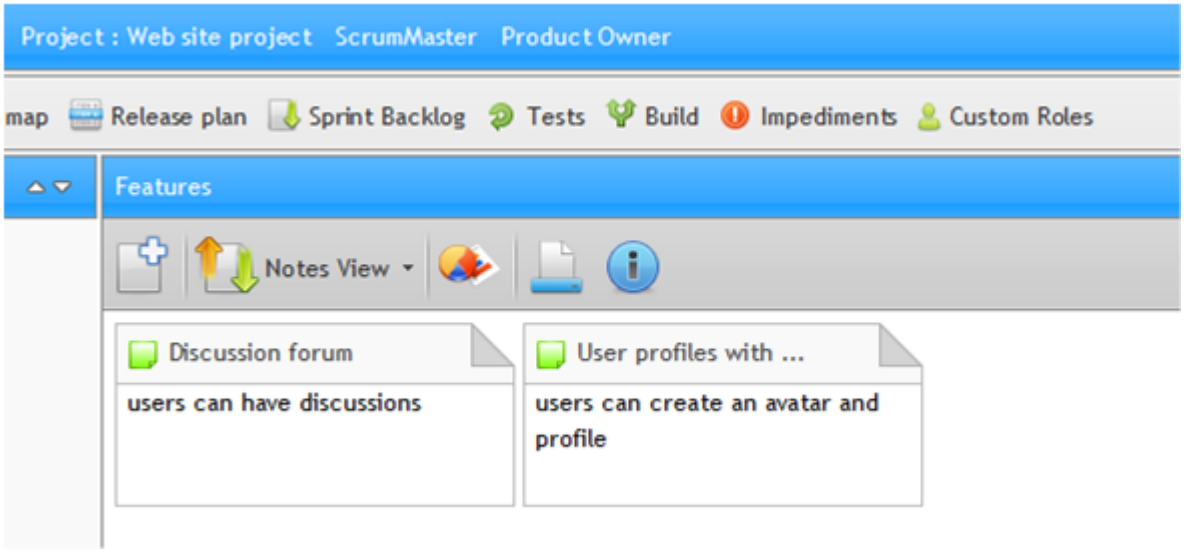
Πλεονεκτήματα:

- Πλούσιο σύνολο λειτουργιών
- Η ανεκτέλεστη ορμή μοιάζει με φυσική εργασία αρκετά αποτελεσματική.
- Οι δοκιμές αποδοχής μπορούν να καταγράφονται για κάθε ιστορία.
- Περιλαμβάνει τον προγραμματισμό χαρακτηριστικό του πόκερ.
- Υποστηρίζει απόλυτη ιεράρχηση των ιστοριών με drag and drop και εκτιμήσεις των ιστοριών.

Μειονεκτήματα:

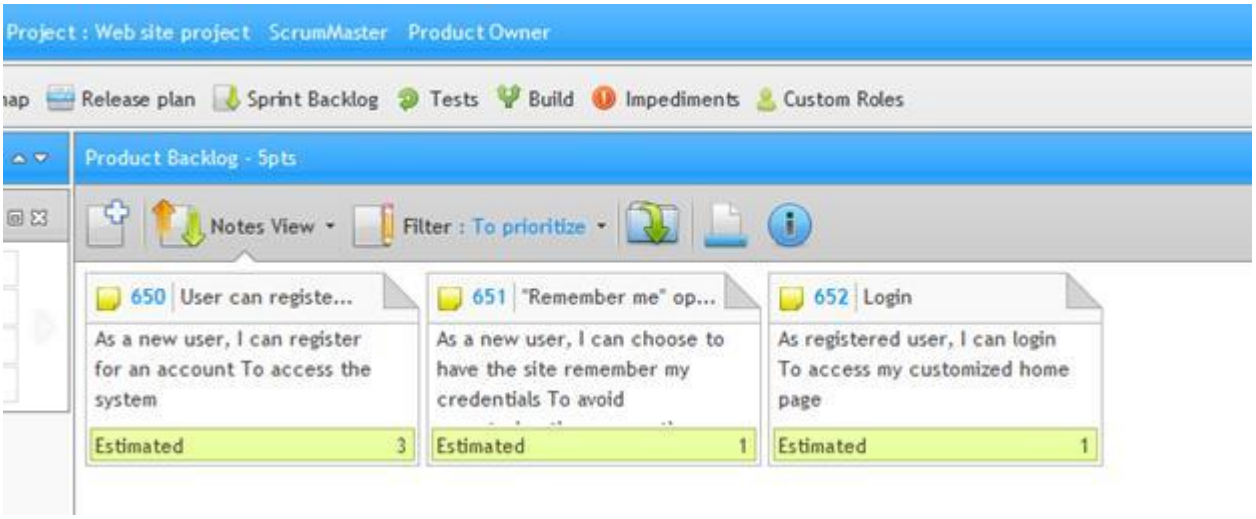
- Το σχεδιάγραμμα των καρτών ιστορίας καθιστά δύσκολη τη κατάταξη ενός μεγάλου αριθμού εκκρεμών υποθέσεων.
- Μερικά χαρακτηριστικά δεν είναι διαισθητικά. Η μεταφορά και η απόθεση λειτουργεί σε ορισμένες περιοχές αλλά όχι σε όλες.
- Δεν είναι κατάλληλο για μεγάλα έργα με πολλαπλές ομάδες που εργάζονται σε ένα συγκεκριμένο προϊόν
- Μια μόνο έκδοση και μία ενιαία ορμή είναι ενεργή σε ένα χρόνο.

Τα χαρακτηριστικά είναι υψηλού επίπεδου απαιτήσεις.

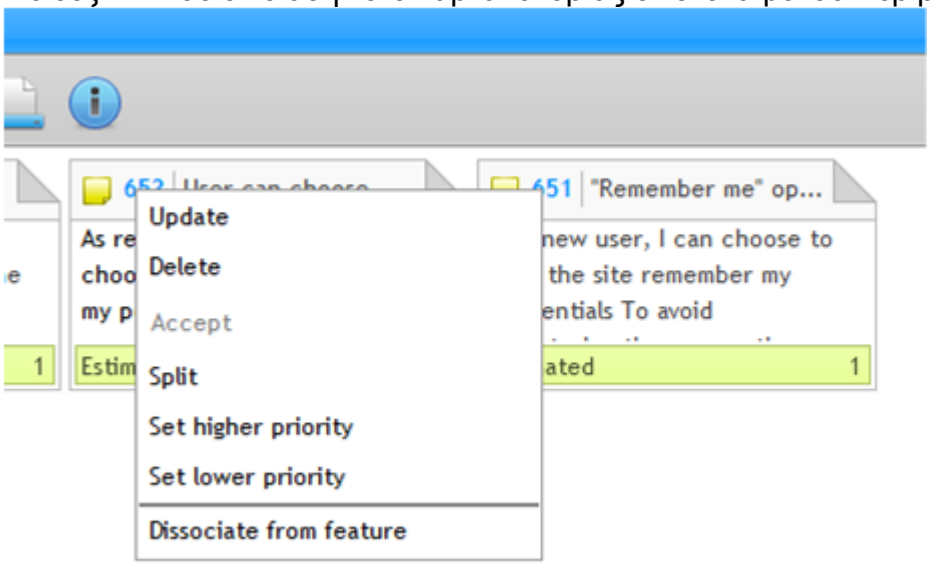


Δημιουργώντας μια ιστορία χρηστών σε ανεκτέλεστο υπόλοιπο προϊόν του IceScrum.

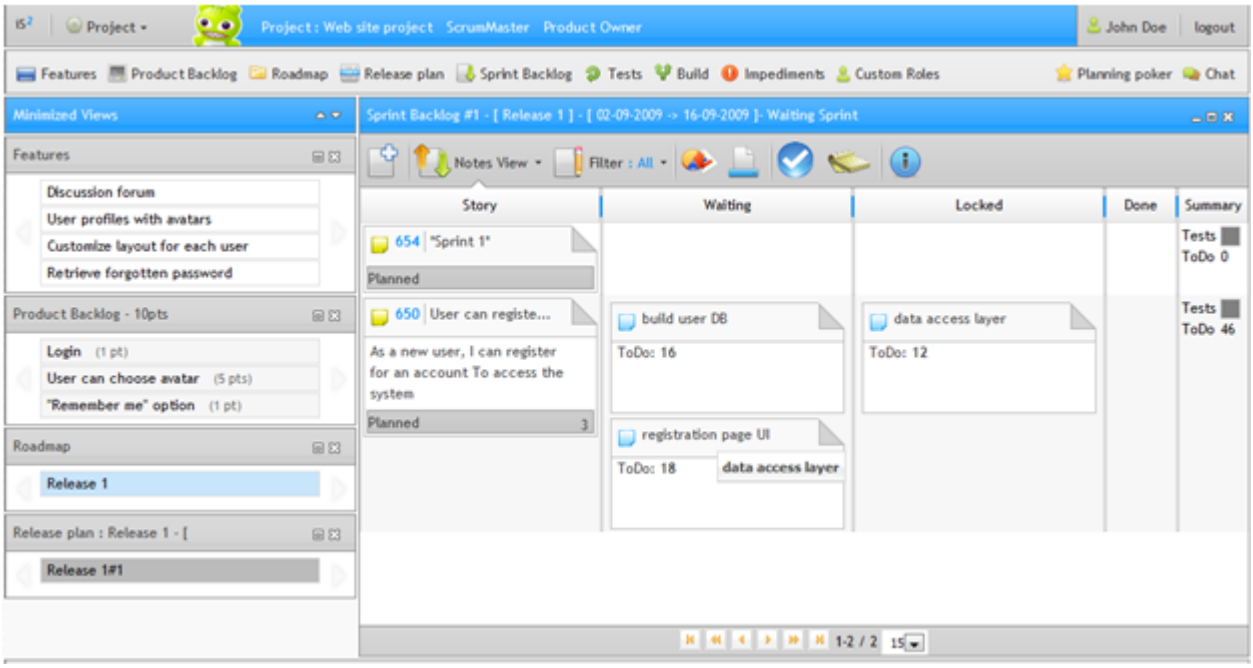
Ανεκτέλεστο προϊόν. Κάθε κάρτα ιστορίας μπορεί με drag and drop να αλλάξει κατάταξη. Τα χαρακτηριστικά συνδέονται με μια ιστορία σύροντας το χαρακτηριστικό από την αριστερή πλευρά της σελίδας πάνω στη επιθυμητή ιστορία.



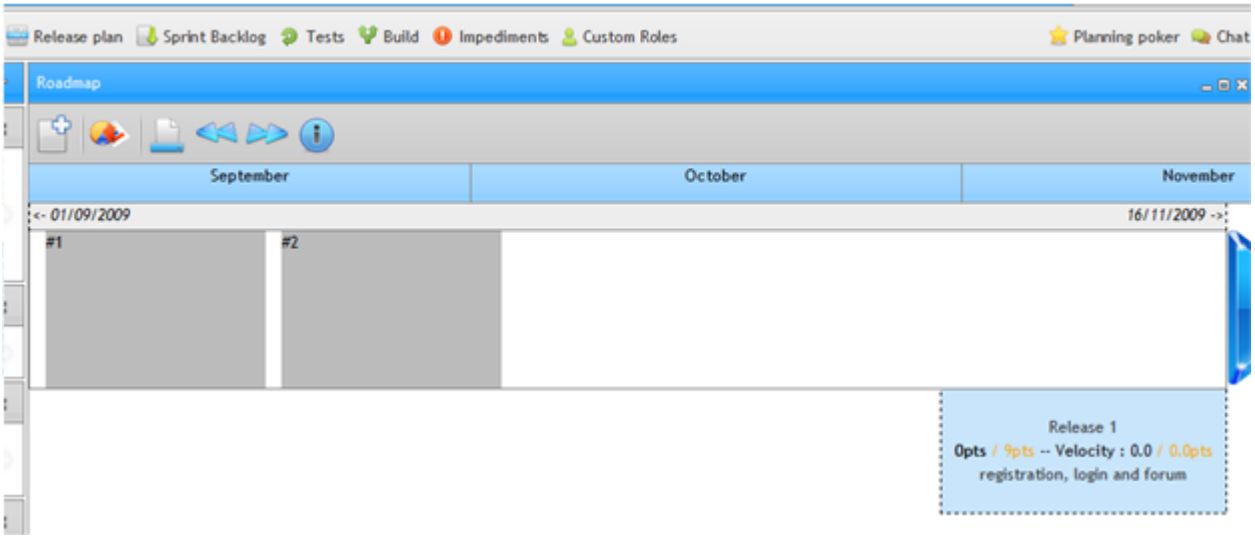
Με δεξί κλικ σε οποιαδήποτε κάρτα ιστορίας δίνει ένα μενού περιβάλλοντος.



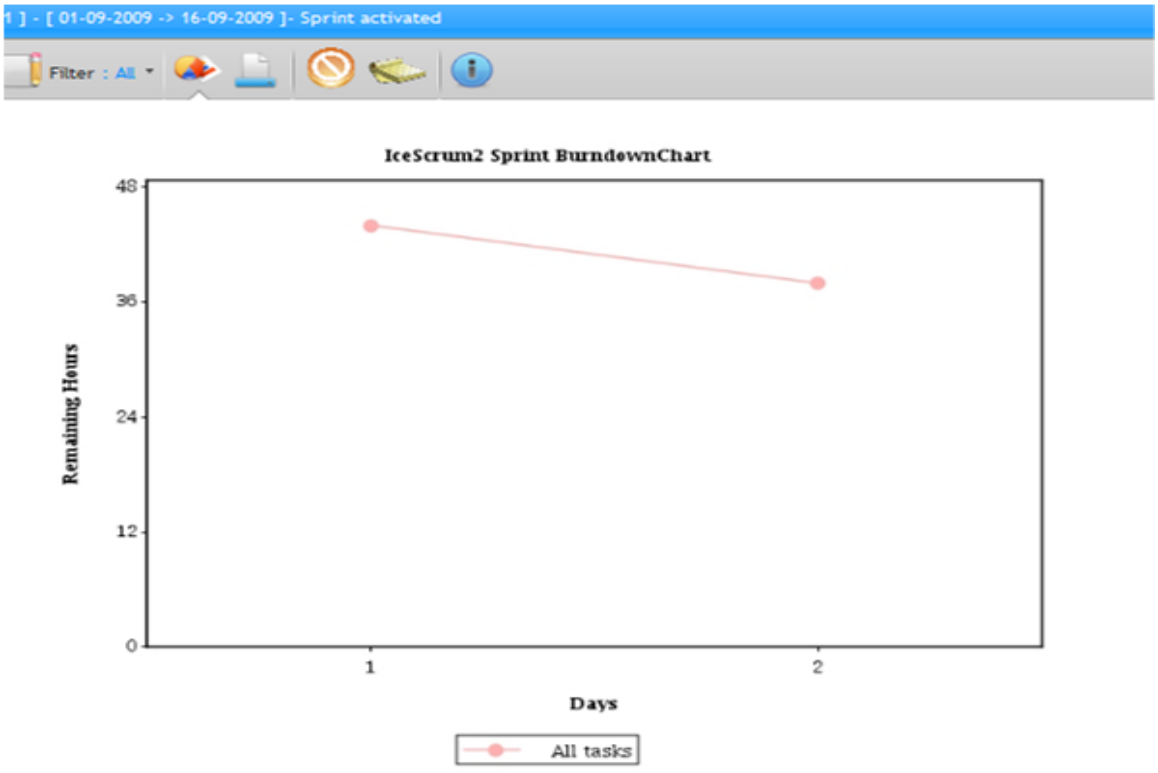
Ανεκτέλεστο προϊόν. Για να προσθέσουμε μία ιστορία από το ανεκτέλεστο υπόλοιπο προϊόν στην ορμή σέρνουμε την ιστορία από την αριστερή πλευρά της σελίδας και κολλάμε αυτή την ιστορία στην περιοχή της ορμής.



Ο χάρτης πορείας δείχνει το χρονοδιάγραμμα των εκδόσεων και των ορμών κατά την απελευθέρωση.



Το γράφημα ορμών είναι ελλιπείς , δείχνει μόνο τις πραγματικές ημέρες που έχουν περάσει μέχρι στιγμής στον οριζόντιο άξονα και όχι το πλήρες φάσμα της ορμής. Στην παρακάτω οθόνη η ορμή είναι 14 ημερών αλλά το διάγραμμα δείχνει μόνο τις πρώτες 2 ημέρες.

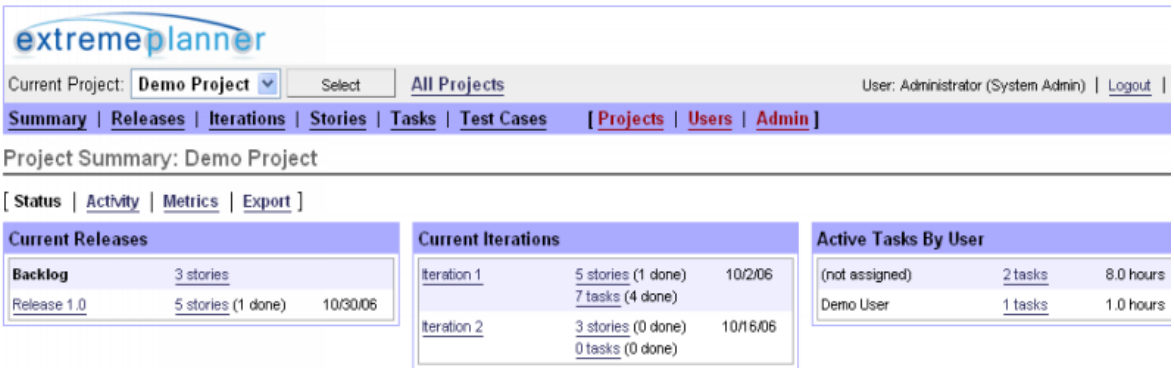


4.3 ExtremePlanner

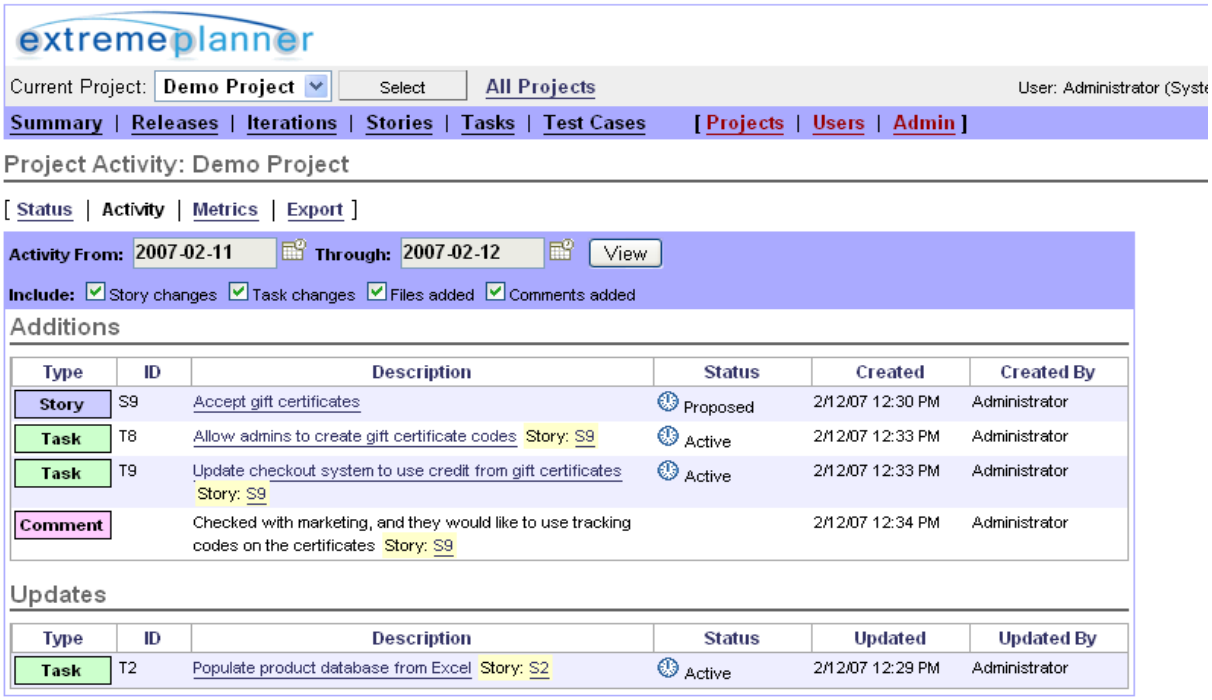
Το ExtremePlanner είναι μια μηχανή αναζήτησης βασισμένη στην Ευέλικτη Διαχείριση Έργου για τις σύγχρονες ομάδες ανάπτυξης λογισμικού που εργάζονται σε διαφορετικές τοποθεσίες. Σε αντίθεση με τα παραδοσιακά εργαλεία διαχείρισης έργων που επικεντρώνονται κυρίως στις λίστες εργασιών, το ExtremePlanner επικεντρώνεται στο σχεδιασμό και την παρακολούθηση της προόδου των χαρακτηριστικών που έχουν την πραγματική επιχειρηματική αξία στους πελάτες.

- Είναι σχεδιασμένο ειδικά για την υποστήριξη ευέλικτων μεθόδων, συμπεριλαμβανομένων Scrum και XP.
- Υποστηρίζει όλη την ομάδα: διαχειριστές έργων, προγραμματιστές, τεχνικοί υποστήριξης και ενδιαφερόμενους.
- Εκτίμηση και σχεδιασμός έκδοσης λογισμικού με ευκολία drag and drop.
- Διαχείριση των χαρακτηριστικών, των ελαττωμάτων, των βασικών δοκιμών και των καθηκόντων της ανάπτυξης σε ένα μέρος.
- Ολοκληρωμένα θέματα παρακολούθησης για την διαχείριση των αιτημάτων των πελατών από την αρχή μέχρι το τέλος.
- Παραμονή πάνω από τις τελευταίες αλλαγές με ειδοποιήσεις μέσω e-mail και εκθέσεων δραστηριότητας του έργου.

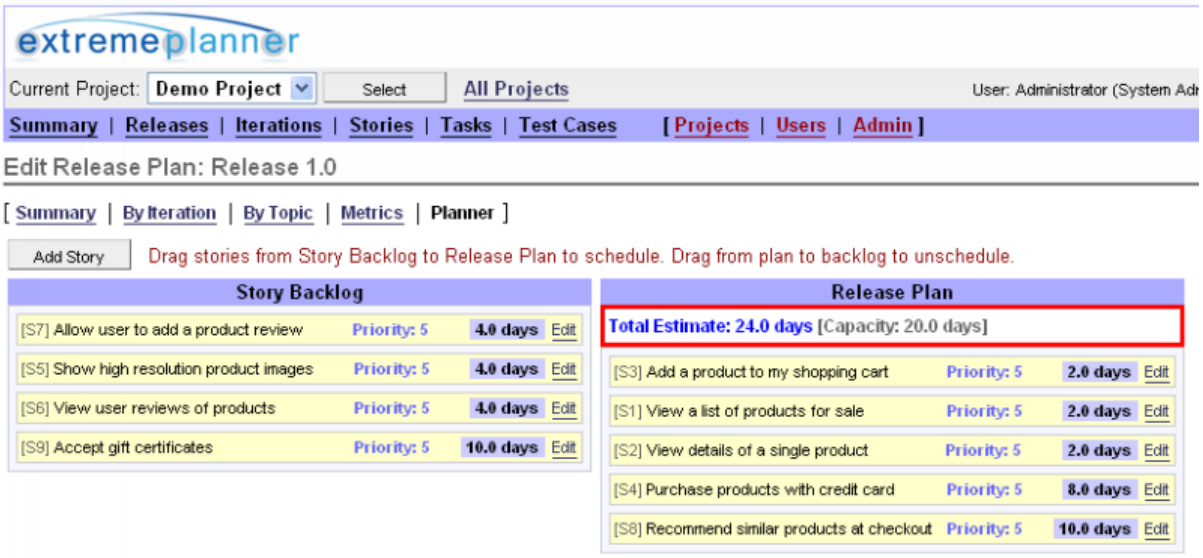
Στην πρώτη οθόνη βλέπουμε τις τρέχουσες εκδόσεις που έχουν προγραμματιστεί για το έργο, τις επαναλήψεις που είναι σε εξέλιξη και με το τι ασχολείται ο καθένας.



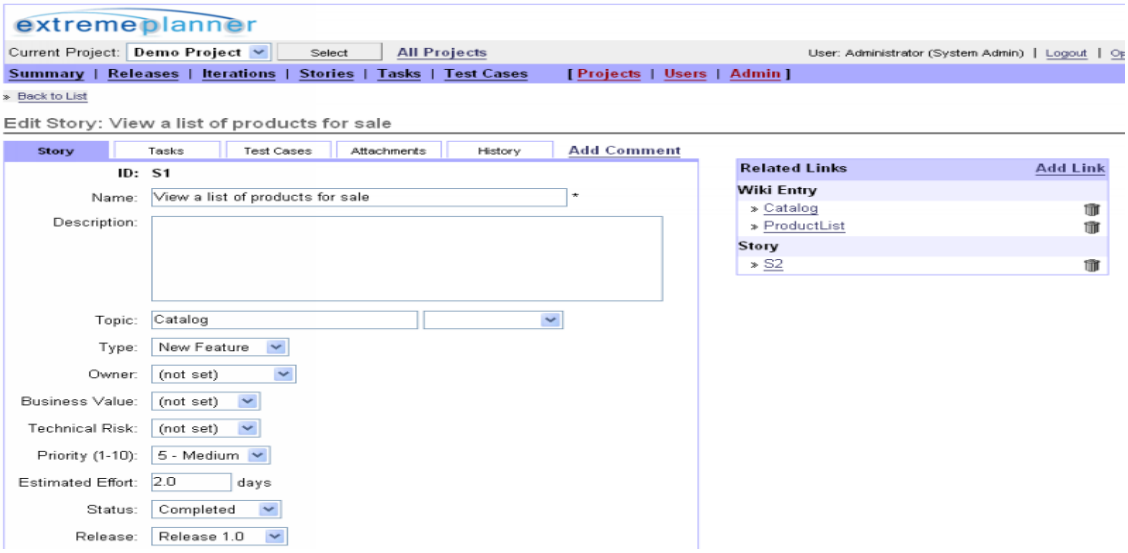
Στις αναφορές καταστάσεων και ειδοποιήσεων βλέπουμε τι νέο υπάρχει ή τι αλλάζει στο πρόγραμμα σε σχέση με χθες(η οποιαδήποτε ημερομηνία επιλέξουμε)



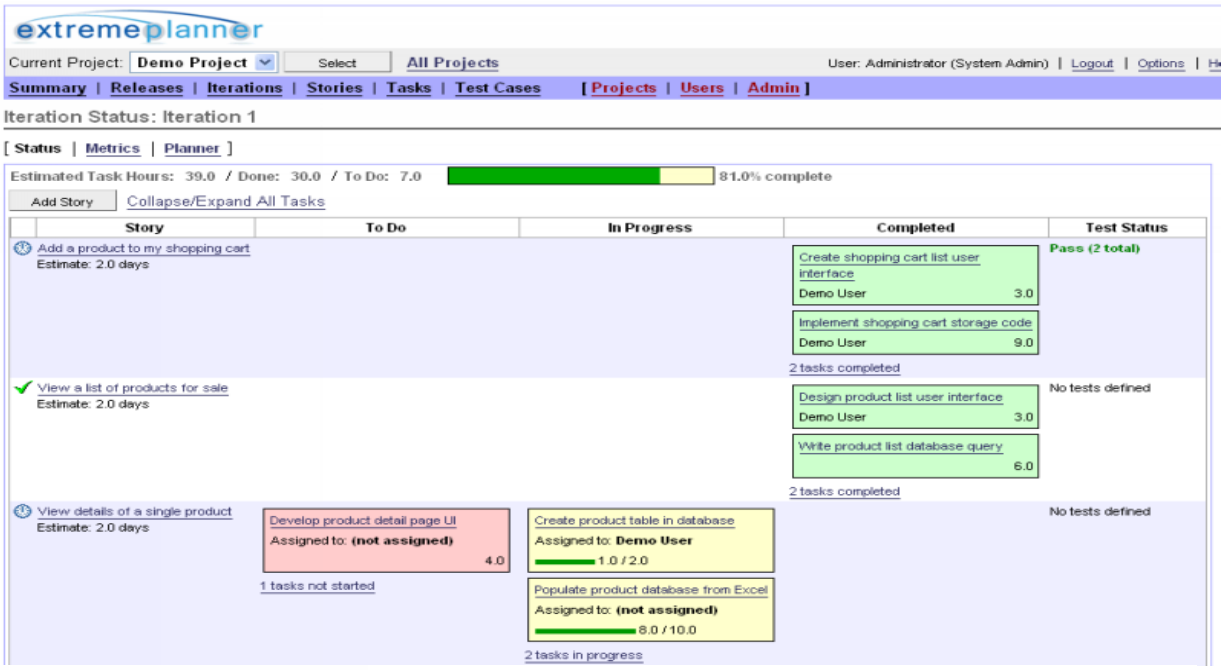
Στο σχέδιο διαχείρισης των εκδόσεων σύρετε τα χαρακτηριστικά σχεδιασμού της κυκλοφορίας. Το ExtremePlanner παρακολουθεί την συνολική προσπάθεια και την ικανότητα και μας προειδοποιεί αν ξεπερνάμε τα όρια μας.

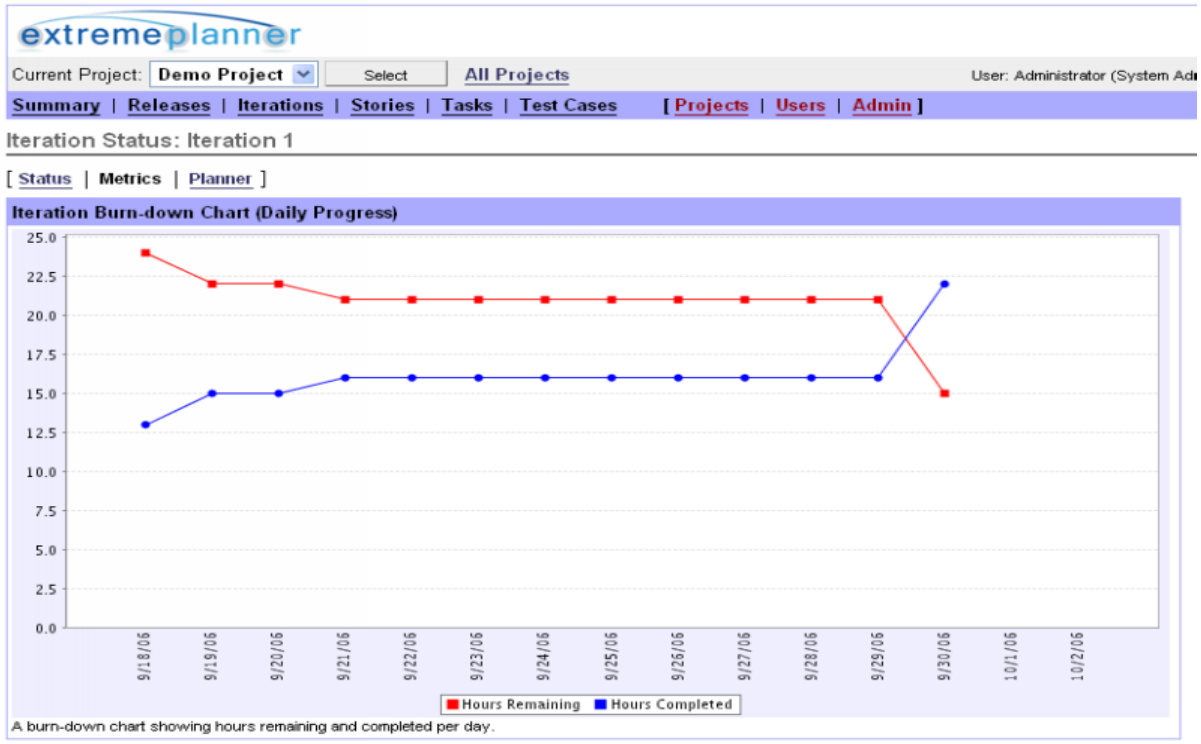


Διαχείριση των ιστοριών, θεμάτων, εργασιών και δοκιμών με περισσότερη ευκολία. Οργανώνουμε όλες τις πληροφορίες του έργου που χρειαζόμαστε, συμπεριλαμβανομένων τις ιστορίες καρτών, τα θέματα, τις εργασίες, τις περιπτώσεις δοκιμών, τα συνημμένα αρχεία και συνδέσμους σε εσωτερικούς ή εξωτερικούς πόρους. Από τις απαιτήσεις των πελατών και τις κλήσεις υποστήριξης σε μια τελική έκδοση λογισμικού.

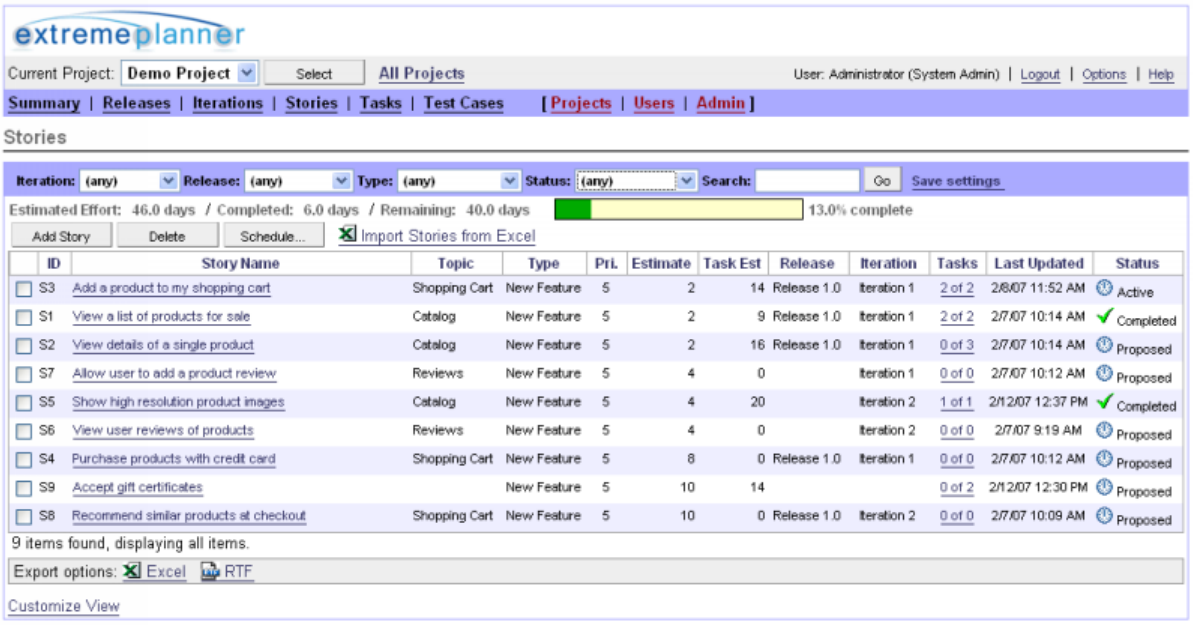


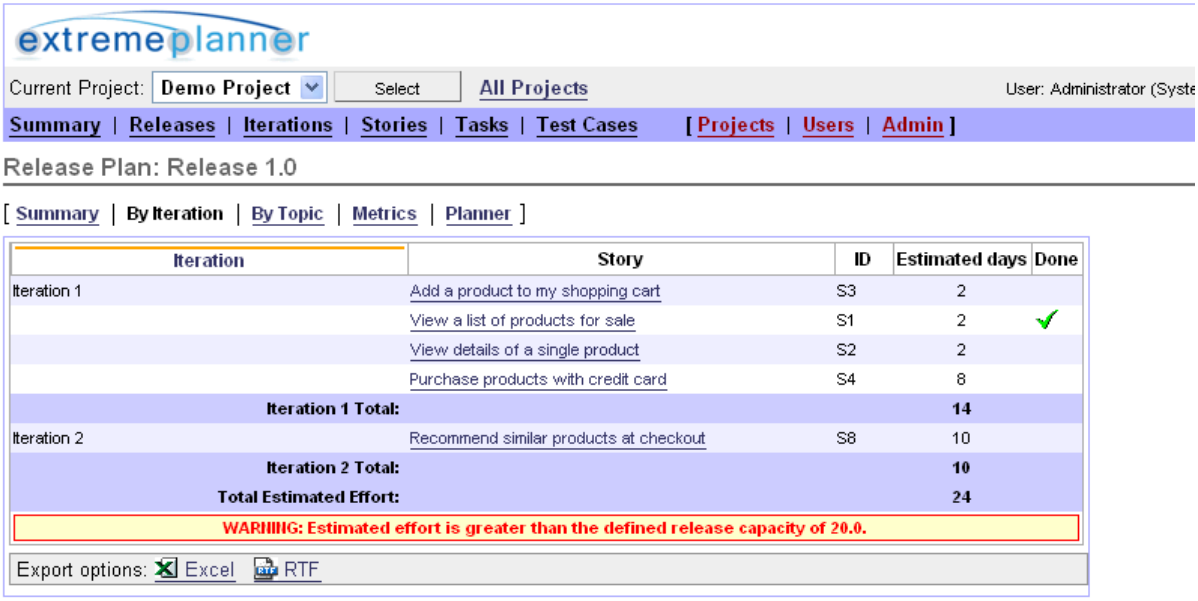
Η παρακολούθηση της προόδου στις καθημερινές συναντήσεις με τις απόψεις των καθηκόντων που βρίσκονται σε εξέλιξη ή δεν έχουν αρχίσει ακόμα ή έχουν ολοκληρωθεί. Η παρακολούθηση των δραστηριοτήτων έργου σε ολόκληρο τον κύκλο της ανάπτυξης με την απελευθέρωση και την επανάληψη φαίνεται στο παρακάτω διάγραμμα.



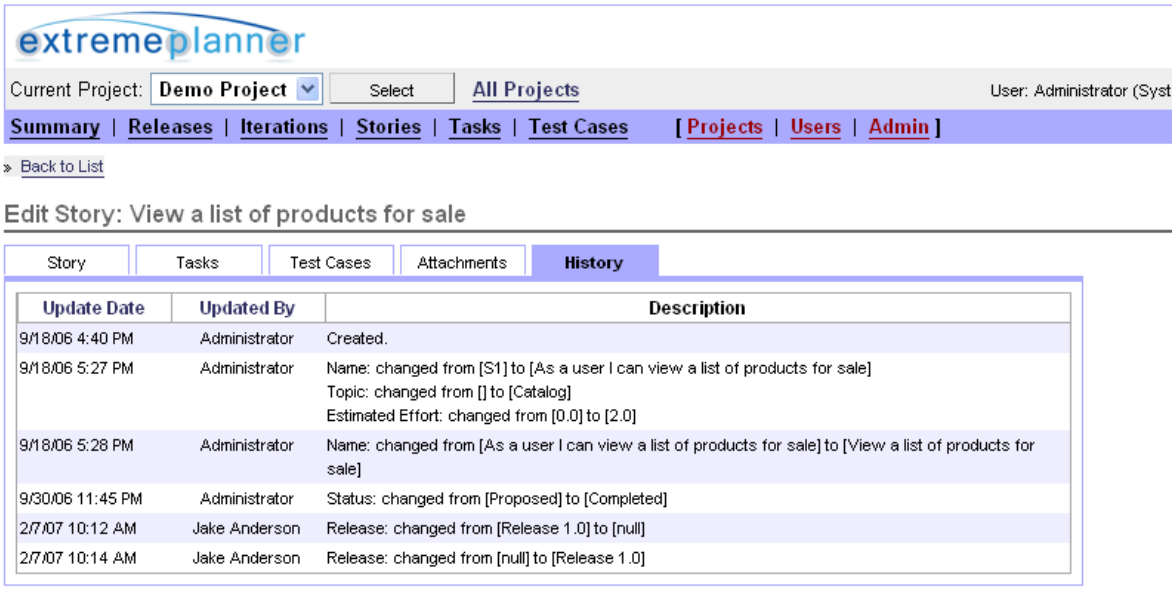


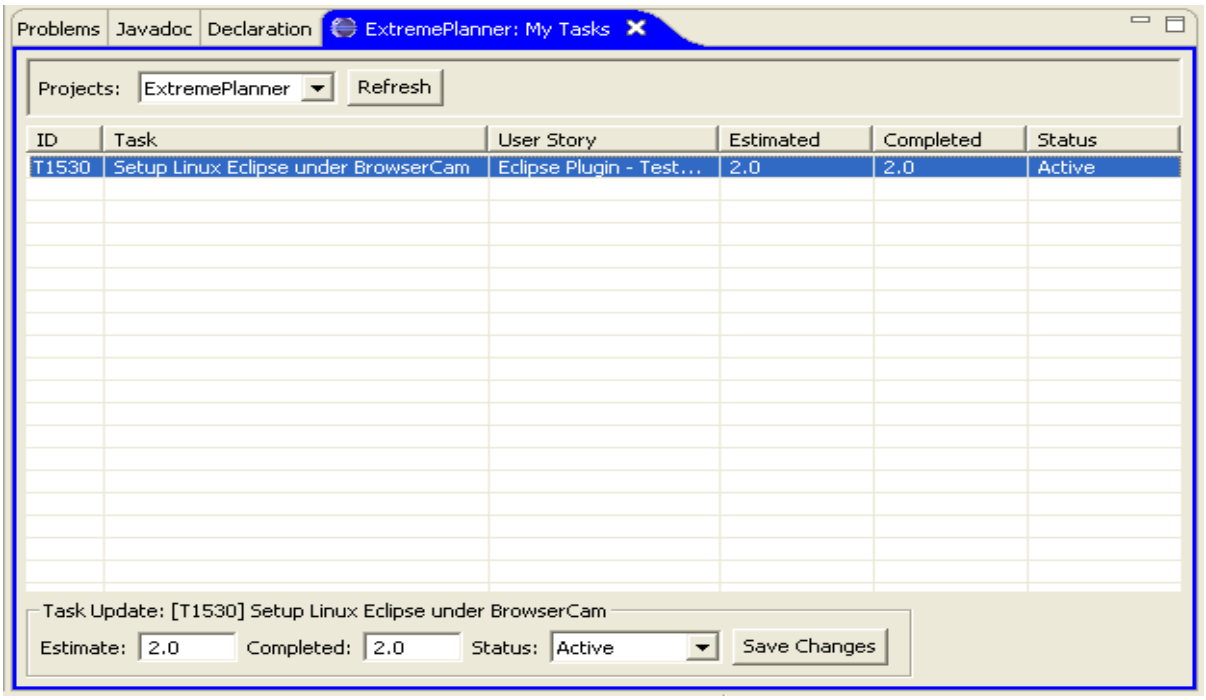
Στην έκθεση και εξαγωγή γίνετε ταξινόμηση, αναζήτηση και φιλτράρισμα της λίστας των ιστοριών του έργου, τις εργασίες ή τις περιπτώσεις δοκιμής για να πάρουμε περισσότερες πληροφορίες σχετικά με την στάση μας. Εξαγωγή των δεδομένων του έργου σε Excel, Word ή σε μορφή XML.


extremeplanner
Current Project: Demo Project Select All Projects
User: Administrator (System Admin) | Logout | Options | Help
Summary | Releases | Iterations | Stories | Tasks | Test Cases [Projects | Users | Admin]
Stories
Iteration: (any) Release: (any) Type: (any) Status: (any) Search: Go Save settings
Estimated Effort: 46.0 days / Completed: 6.0 days / Remaining: 40.0 days 13.0% complete
Add Story Delete Schedule... Import Stories from Excel
ID Story Name Topic Type Pri. Estimate Task Est Release Iteration Tasks Last Updated Status
S3 Add a product to my shopping cart Shopping Cart New Feature 5 2 14 Release 1.0 Iteration 1 2 of 2 2/8/07 11:52 AM Active
S1 View a list of products for sale Catalog New Feature 5 2 9 Release 1.0 Iteration 1 2 of 2 2/7/07 10:14 AM Completed
S2 View details of a single product Catalog New Feature 5 2 16 Release 1.0 Iteration 1 0 of 3 2/7/07 10:14 AM Proposed
S7 Allow user to add a product review Reviews New Feature 5 4 0 Iteration 1 0 of 0 2/7/07 10:12 AM Proposed
S5 Show high resolution product images Catalog New Feature 5 4 20 Iteration 2 1 of 1 2/12/07 12:37 PM Completed
S6 View user reviews of products Reviews New Feature 5 4 0 Iteration 2 0 of 0 2/7/07 9:19 AM Proposed
S4 Purchase products with credit card Shopping Cart New Feature 5 8 0 Release 1.0 Iteration 1 0 of 0 2/7/07 10:12 AM Proposed
S9 Accept gift certificates New Feature 5 10 14 0 of 2 2/12/07 12:30 PM Proposed
S8 Recommend similar products at checkout Shopping Cart New Feature 5 10 0 Release 1.0 Iteration 2 0 of 0 2/7/07 10:09 AM Proposed
9 items found, displaying all items.
Export options: Excel RTF
Customize View


extremeplanner
Current Project: Demo Project Select All Projects
User: Administrator (Syste
Summary | Releases | Iterations | Stories | Tasks | Test Cases [Projects | Users | Admin]
Release Plan: Release 1.0
[Summary | By Iteration | By Topic | Metrics | Planner]
Iteration Story ID Estimated days Done
Iteration 1 Add a product to my shopping cart S3 2
View a list of products for sale S1 2
View details of a single product S2 2
Purchase products with credit card S4 8
Iteration 1 Total: 14
Iteration 2 Recommend similar products at checkout S8 10
Iteration 2 Total: 10
Total Estimated Effort: 24
WARNING: Estimated effort is greater than the defined release capacity of 20.0.
Export options: Excel RTF

Έλεγχος της πρόσβασης και ιχνηλασιμότητα. Παρακολουθούμε οποιαδήποτε αλλαγή στο πρόγραμμα με πλήρη έλεγχο. Περιορίζουμε την πρόσβαση των χρηστών μόνο με τα προγράμματα που ασχολούνται.





4.4 Agilefant

Έννοιες

Τα προϊόντα είναι το υψηλότερο επίπεδο κατασκευής και κάθε ανάπτυξη μπορεί να έχει πολλά προϊόντα. Κάθε προϊόν μπορεί να έχει ένα η περισσότερα έργα, τα οποία ουσιαστικά απελευθερώνει. Κάθε έργο μπορεί να έχει μία ή περισσότερες επαναλήψεις. Κάθε προϊόν, έργο και επανάληψη έχει το δικό του ανεκτέλεστο υπόλοιπο, το οποίο περιέχει ιστορίες. Οι ιστορίες μπορούν να μετακινηθούν σε οποιαδήποτε άλλη ανεκτέλεστη, για παράδειγμα, από το ανεκτέλεστο υπόλοιπο προϊόν σε ανεκτέλεστη επανάληψη. Οι ιστορίες μπορούν να αποτελούνται από μηδέν ή περισσότερα καθήκοντα. Τα έργα μπορούν να δώσουν προτεραιότητα στην προβολή χαρτοφύλακα.

Το Agilefant είναι ένα web-based εργαλείο γραμμένο σε java και το εργαλείο αυτό είναι υπό ενεργό ανάπτυξη. Είναι αρκετά πλούσιο σε δυνατότητες και δίνει τη δυνατότητα δημιουργίας προϊόντων, έργων, επαναλήψεων και ιστοριών χρήστη. Έχει χρονική καταδίωξη, γραφήματα και διαχειριστές.

Το εργαλείο αυτό υποστηρίζει πολλαπλές ταυτόχρονες επαναλήψεις οι οποίες επιτρέπουν στις μεγαλύτερες οργανώσεις να χρησιμοποιούν το εργαλείο αποτελεσματικά. Δουλεύει πολύ γρήγορα.

Το Agilefant υποστηρίζει πολλούς χρήστες οι οποίοι μπορούν να χωριστούν σε ομάδες. Ωστόσο, δεν υποστηρίζει κανένα ρόλο χρήστη, ο χρήστης είναι χρήστης ενώ διαπιστώθηκαν διαφορές στα δικαιώματα ή πρόσβαση σε λειτουργίες.

Τα χαρακτηριστικά που λείπουν είναι η απελευθέρωση και η επανάληψη του σχεδιασμού. Το εργαλείο δεν έχει την έννοια της έκδοσης σε όλα και αυτό είναι παράξενο δεδομένου ότι έχει τα προϊόντα. Δεν έχει υποβολή εκθέσεων και κακή προσαρμογή. Δεν υποστηρίζει μονάδες και σημεία για τον προγραμματισμό(αρκετά περίεργο για ευέλικτο εργαλείο). Το περιβάλλον του χρήστη έχει πολύ λίγες οθόνες και εμφανίζει υπερβολικό φόρτο(όπως η προεπιλεγμένη οθόνη του έργου περιέχει τις λεπτομέρειες, τα θέματα, τις ανεκτέλεστες παραγγελίες και τις επαναλήψεις με όλες τις λεπτομέρειες). Υπάρχουν ορισμένα ζητήματα χρήσης, αλλά σε γενικές γραμμές το εργαλείο είναι πολύ εύκολο να κατανοηθεί. Δεν μοιάζει με μια περίπλοκη λύση.

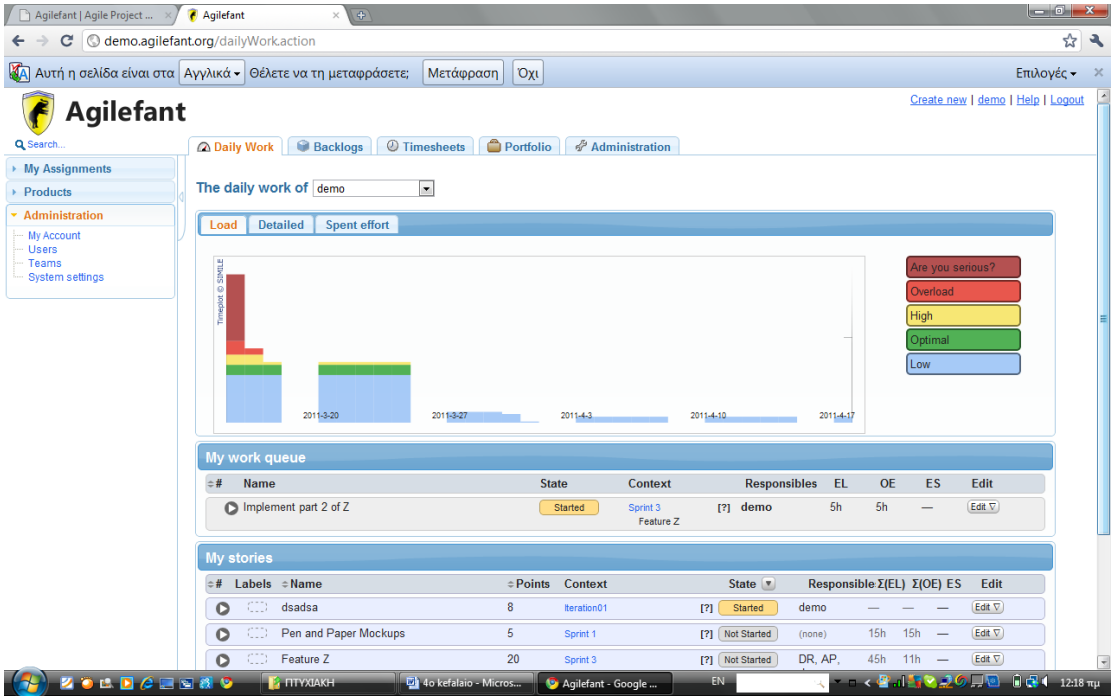
Το Agilefant χρησιμοποιείται για μικρά έργα, δεν είναι βολικό για μεγάλα έργα με παραπάνω από είκοσι άτομα.
Τα πλεονεκτήματα του Agilefant είναι:

- Προϊόντα υποστήριξης και όμορφα γραφήματα της πορείας του προϊόντος.
- Πολύ εύκολο στην χρήση
- Χρόνος παρακολούθησης
- Χορήγηση υποστήριξης
- Πλούσιο σύνολο λειτουργιών
- Κατάλληλο για μεγάλες εταιρείες και σχέδια

Τα μειονεκτήματα του Agilefant είναι:

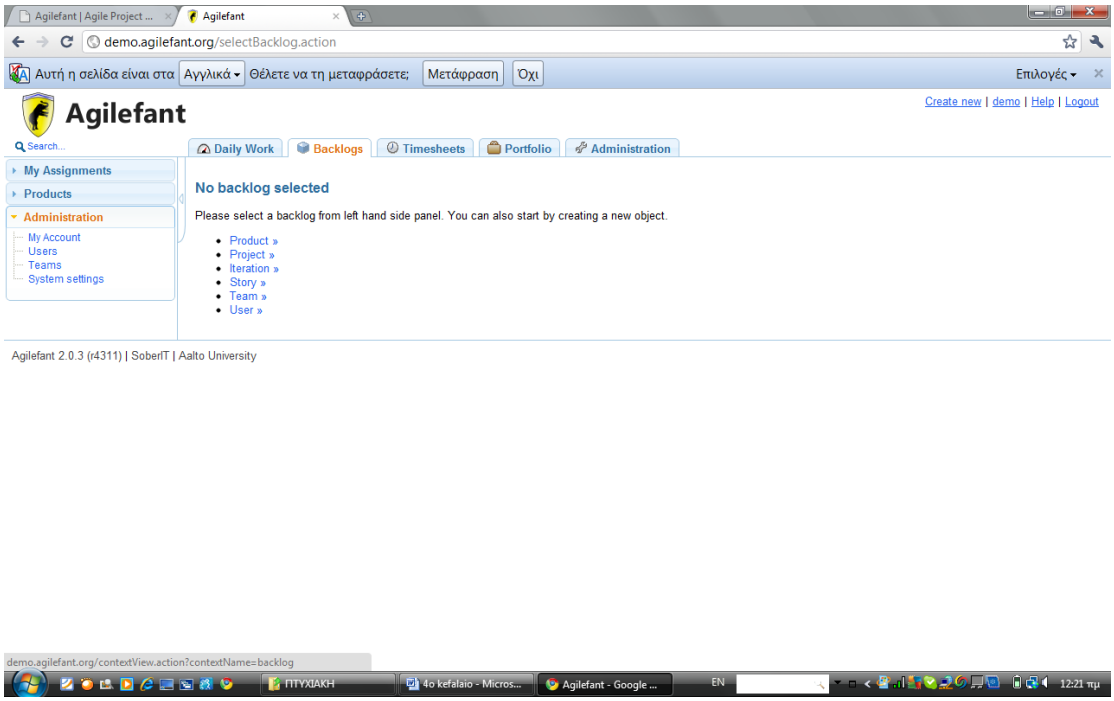
- Τα σημεία που λείπουν
- Έλλειψη αναφοράς
- Χωρίς εκδόσεις/επαναλήψεις προγραμματισμού
- Κακή δυνατότητα προσαρμογής
- Δεν απαιτείται καμία διαφοροποίηση μεταξύ των ρόλων των χρηστών

Καθημερινή εργασία
Η καθημερινή εργασία συγκεντρώνει όλες τις αναθέσεις ενός ενιαίου χρήστη από διαφορετικές επαναλήψεις σε μια ενιαία θέση. Παρέχει τη δυνατότητα να οργανώσει την εργασία από διαφορετικές επαναλήψεις με την εργασία ουράς. Κάθε χρήστης μπορεί να προσθέσει τα καθήκοντα της ουράς εργασίας και να ταξινομήσει τις εργασίες στην ουρά. Αυτό λειτουργεί ως προσωπική λίστα εργασιών, ενώ παράλληλα μπορεί κάποιος να δει τι κάνει ο χρήστης την συγκεκριμένη στιγμή. Εκτός από την εισαγωγή των ιστοριών και τα καθήκοντα η καθημερινή εργασία περιέχει μια γραφική παρουσίαση των χρηστών και τον επερχόμενο φόρτο εργασίας.



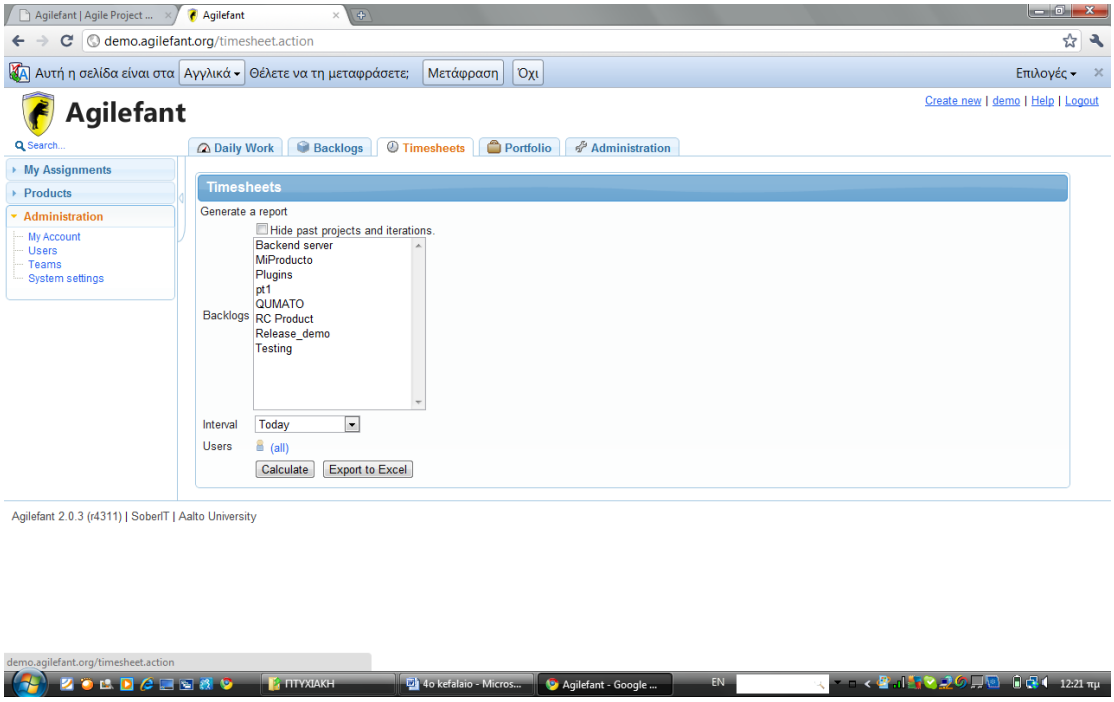
Ανεκτέλεστες

Τα προϊόντα είναι οι κορυφαίες ανεκτέλεστες υποθέσεις στο Agilefant και δεν έχουν σταθερό χρονικό διάστημα. Τα προϊόντα χωρίζονται περαιτέρω στην απελευθέρωση των έργων. Από το επίπεδο του προϊόντος είναι ορατή ολόκληρη η ιεραρχία της ιστορίας του προϊόντος.

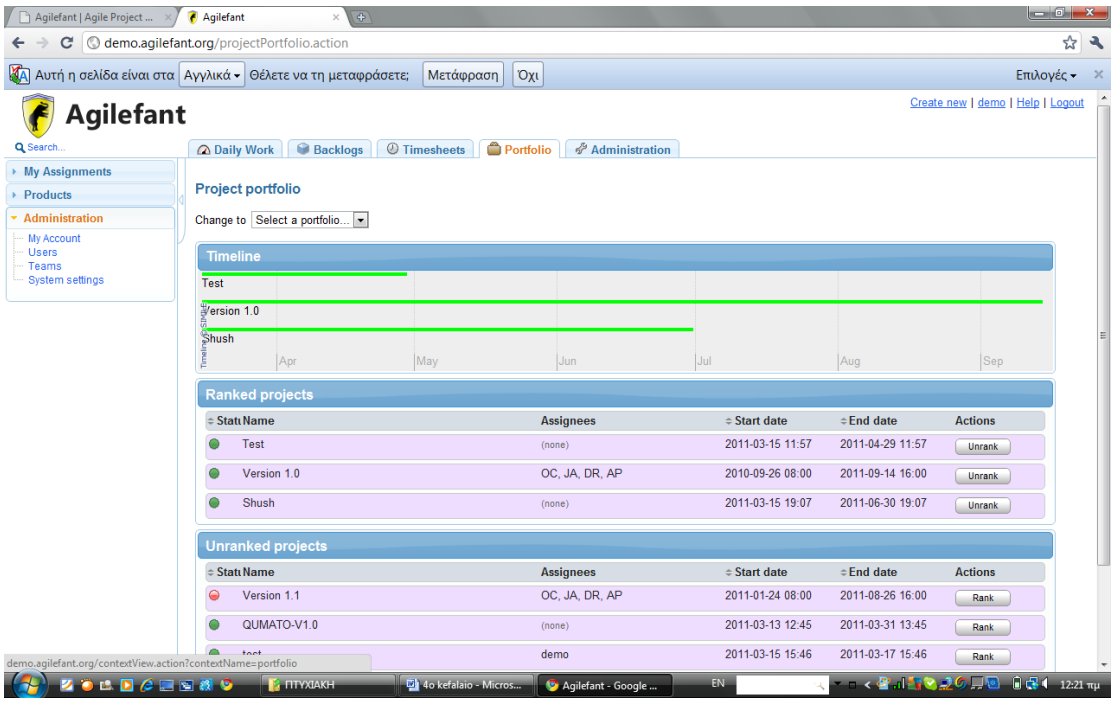


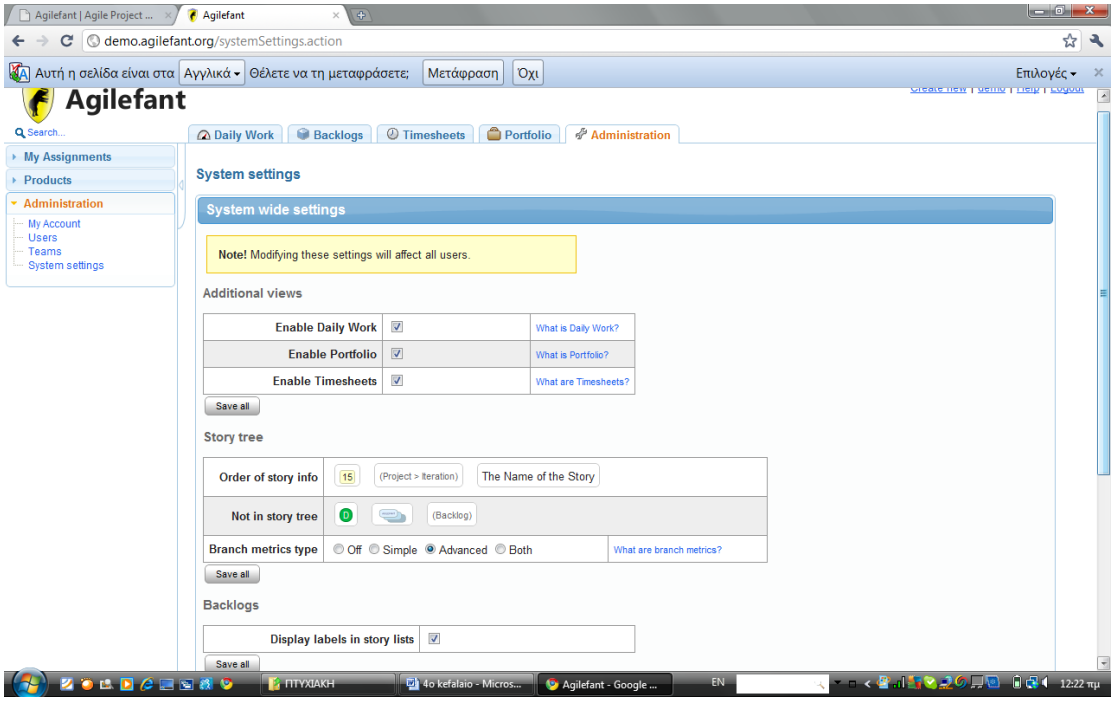
Κατανομή χρόνου

Το Agilefant έχει ένα ενσωματωμένο σύστημα παρακολούθησης χρόνου, το οποίο επιτρέπει την καταγραφή της ανεκτέλεστης προσπάθειας, τις ιστορίες και τα καθήκοντα.



Όλα τα εν εξελίξει και επερχόμενα έργα απελευθέρωσης συγκεντρώνονται σε ένα ενιαίο χώρο, επιτρέποντας έτσι την διαχείριση έργων χαρτοφυλακίου. Το έργο χαρτοφυλακίου μπορεί να εκφράζεται με την ιεράρχηση των έργων.





4.5 Agilo

Έννοιες

Κάθε Agilo εγκατάσταση υποστηρίζει ένα μόνο προϊόν/έργο-ένα ανεκτέλεστο υπόλοιπο. Κάθε ανάπτυξη μπορεί να έχει πολλαπλά στάδια(εκδόσεις). Το ανεκτέλεστο περιέχει απαιτήσεις, ιστορίες χρηστών και καθήκοντα.

Το εργαλείο Agilo είναι εργαλείο υπό ενεργό ανάπτυξη. Αν και έχει πολλά εντυπωσιακά χαρακτηριστικά όπως το χρονοδιάγραμμα, τα διαγράμματα, το ολοκληρωμένο wiki, η έννοια πίσω από το Agilo έχει πολλές ατέλειες. Δεν υπάρχουν έργα στο σύστημα, αυτό σημαίνει ότι αν υπάρχουν πολλά έργα θα πρέπει να εφαρμοστεί ένας εναλλακτικός τρόπος αντιμετώπισης. Η επανάληψη και ο τύπος σχεδιασμού είναι εξαιρετικά επαχθής, δεν είναι εύκολο να καταλάβουμε πώς να δημιουργήσουμε ένα σχέδιο επανάληψης.

Οι χρήστες μπορούν να χρησιμοποιήσουν τους ρόλους του Scrum. Οι χρήστες μπορούν επίσης να χωριστούν σε ομάδες. Επειδή το Agilo επιτρέπει ένα μόνο προϊόν ανά εγκατάσταση είναι κατάλληλο μόνο για μικρές ομάδες που εργάζονται για ένα μόνο προϊόν ή τις οργανώσεις που είναι πρόθυμες να εγκαταστήσουν μια ξεχωριστή εμφάνιση του εργαλείου για κάθε προϊόν/έργο.

Το Agilo είναι ένα καλό δωρεάν εργαλείο για την ευέλικτη ανάπτυξη.

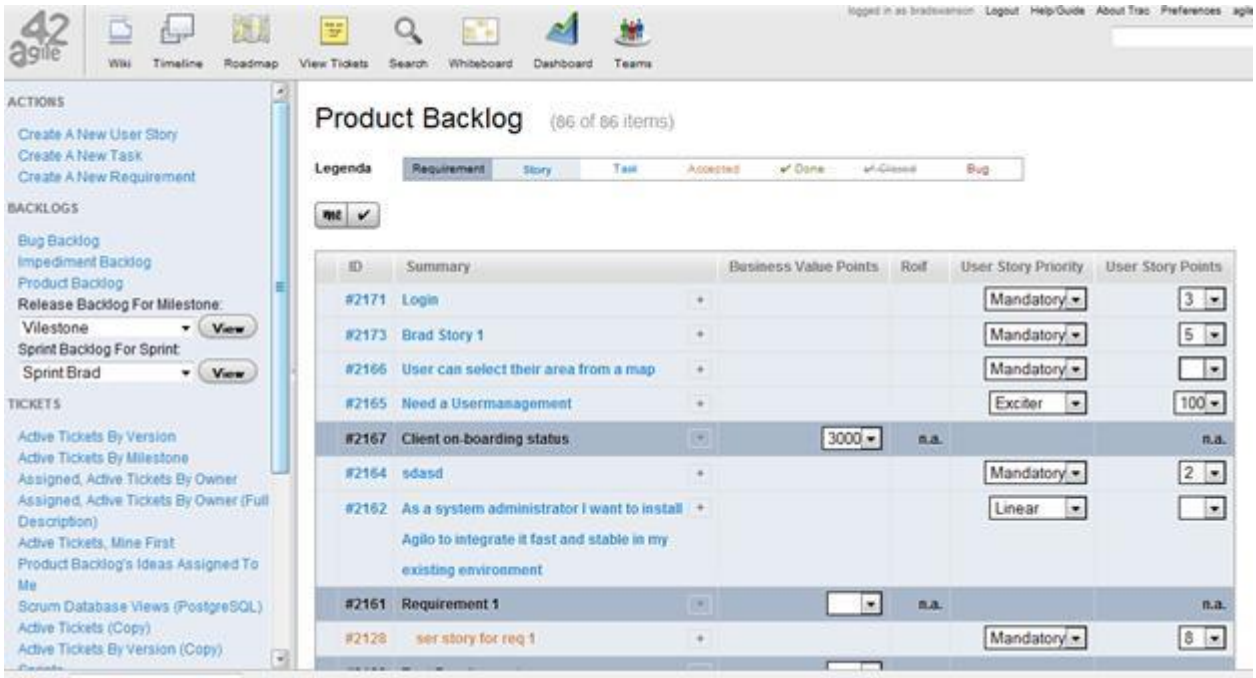
Τα πλεονεκτήματα του Agilo είναι:

- Ολοκληρωμένο wiki
- Καλά διαγράμματα
- Ιστορίες αλλαγής (χρονοδιάγραμμα)
- Αποθηκευμένα ερωτήματα αναζήτησης
- Η ιστορία μπορεί να χωριστεί σε καθήκοντα
- Υποστηρίζει ιστορίες, σφάλματα και εμπόδια

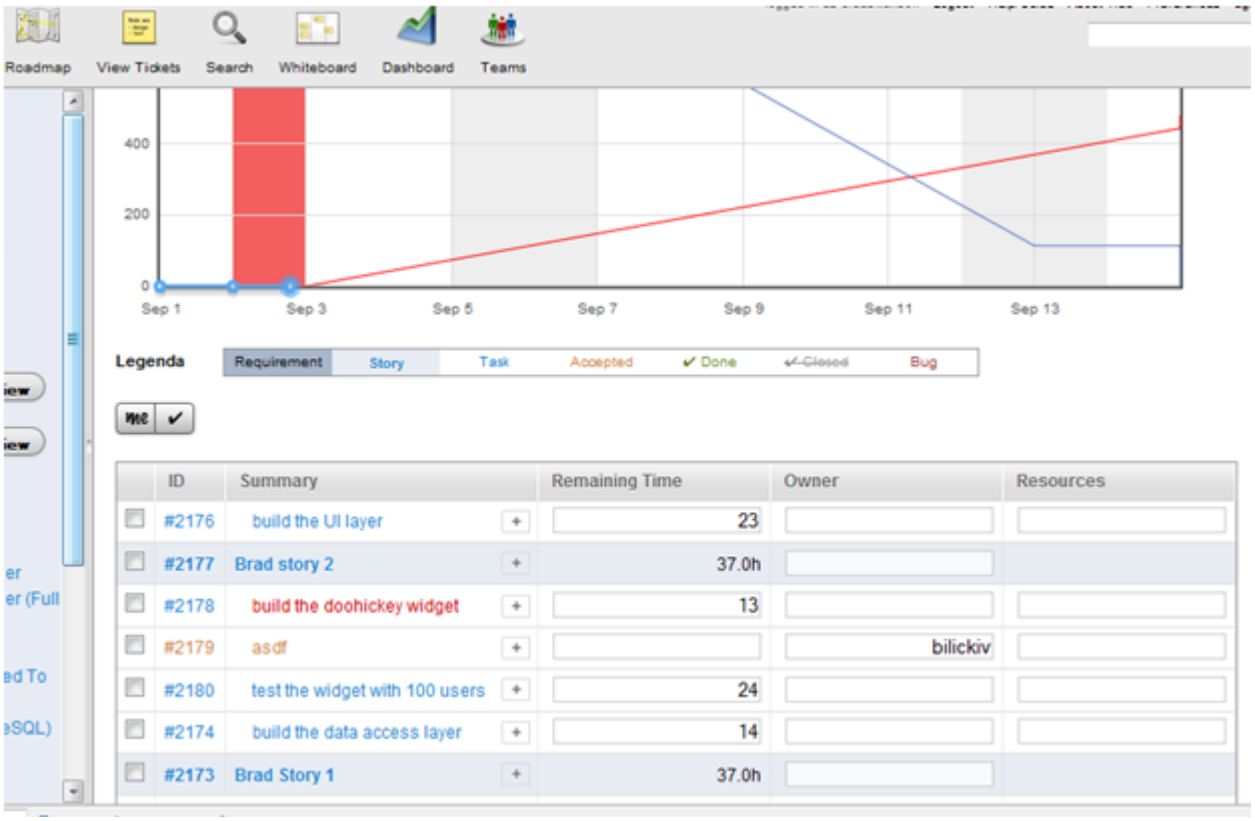
Τα μειονεκτήματα του Agilo είναι:

- Δεν είναι πολύ χρησιμοποιήσιμη διεπαφή χρήστη
- Χωρίς έργα
- Οι διαχειριστές δεν υπάρχουν στην διεπαφή χρήστη
- Όταν δημιουργείται μια εργασία, μπορεί να συνδέεται με μία ορμή αλλά όχι με μια ιστορία.
- Πολλές επιχειρήσεις απαιτούν πολλά κλικ για την ολοκλήρωση
- Υποστηρίζει μόνο ένα ανεκτέλεστο προϊόν ανά εγκατάσταση.

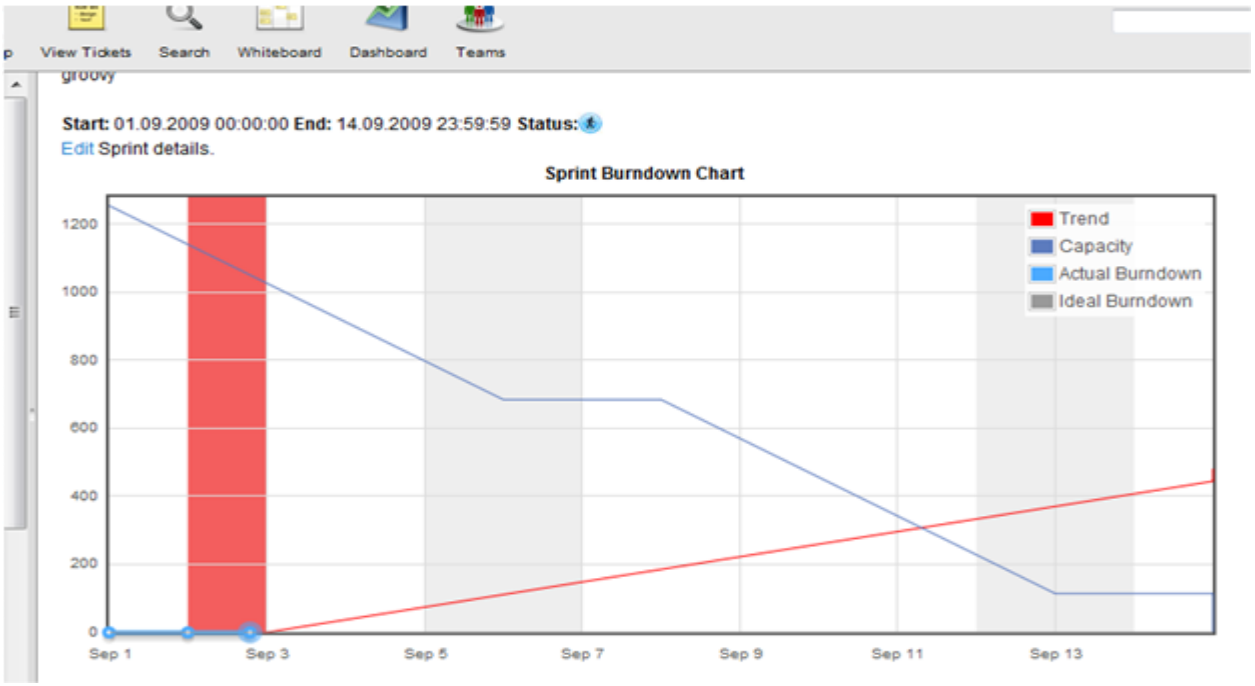
Ανεκτέλεστο προϊόν



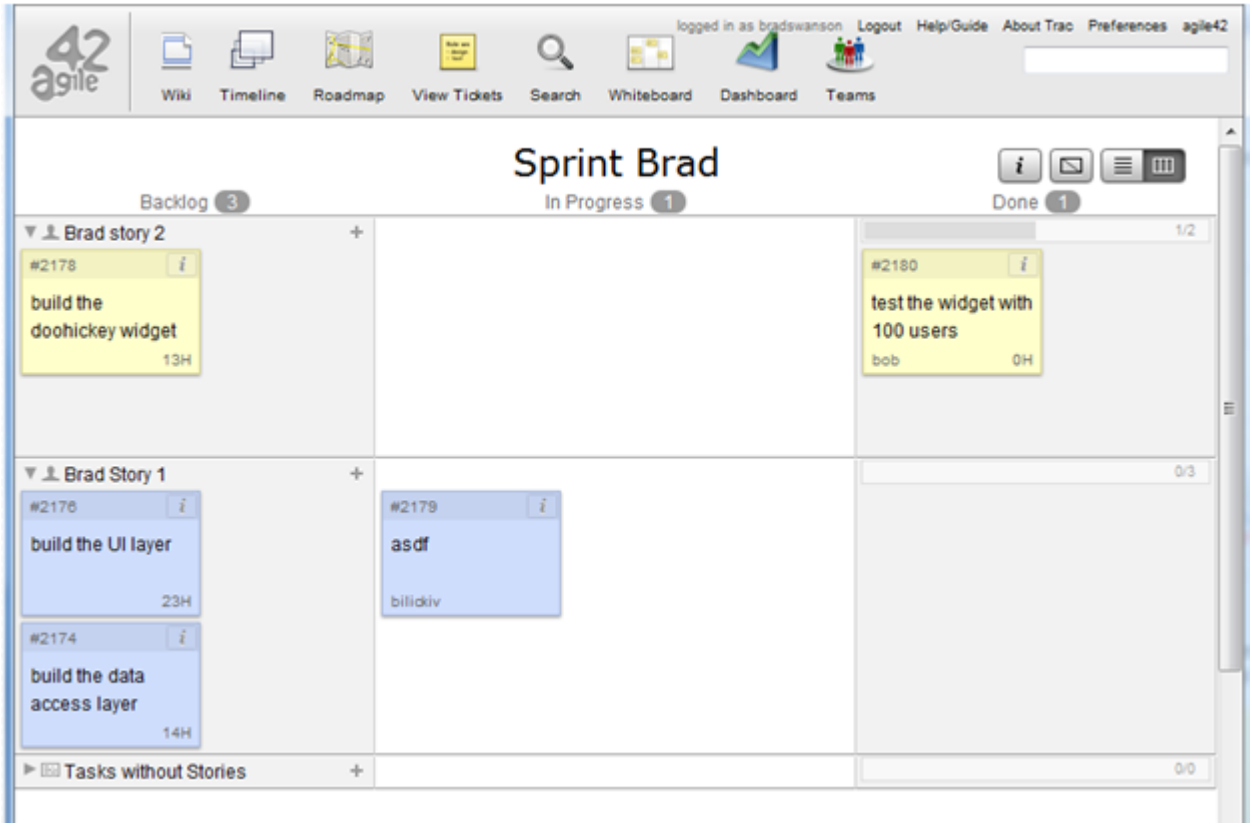
Ανεκτέλεστη ορμή. Η άποψη αυτή δημιουργεί σύγχυση, διότι τα καθήκοντα δεν εμφανίζονται κάτω από τις ιστορίες τους.



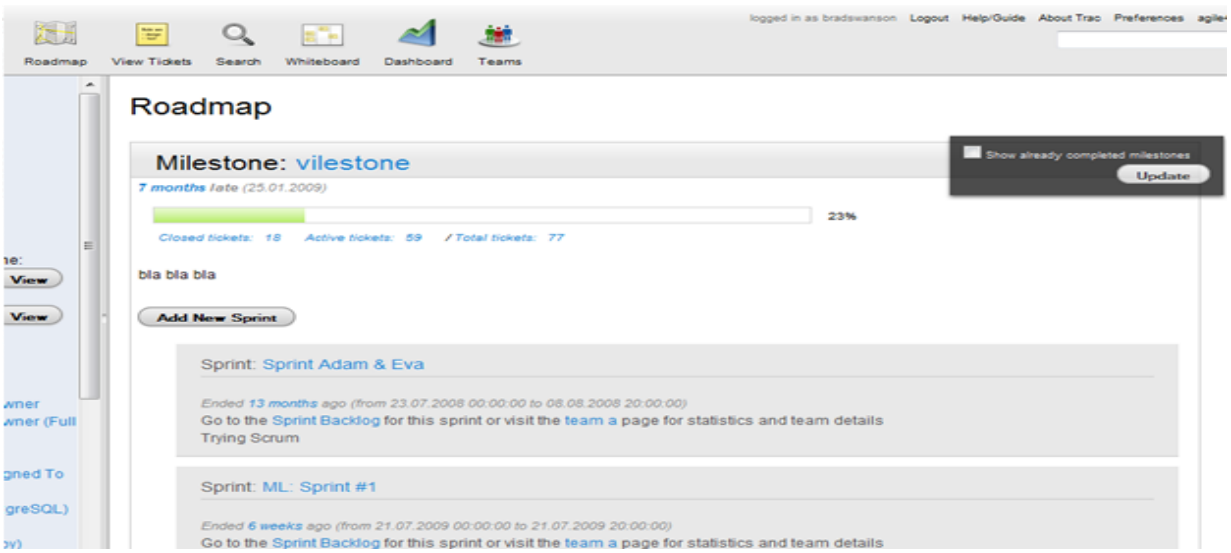
Διάγραμμα ορμής. Αυτό το διάγραμμα είναι λανθασμένο, δείχνοντας 1200 ώρες στον κάθετο άξονα, όταν μόνο 37 ώρες εργασίας έχουν προγραμματιστεί.



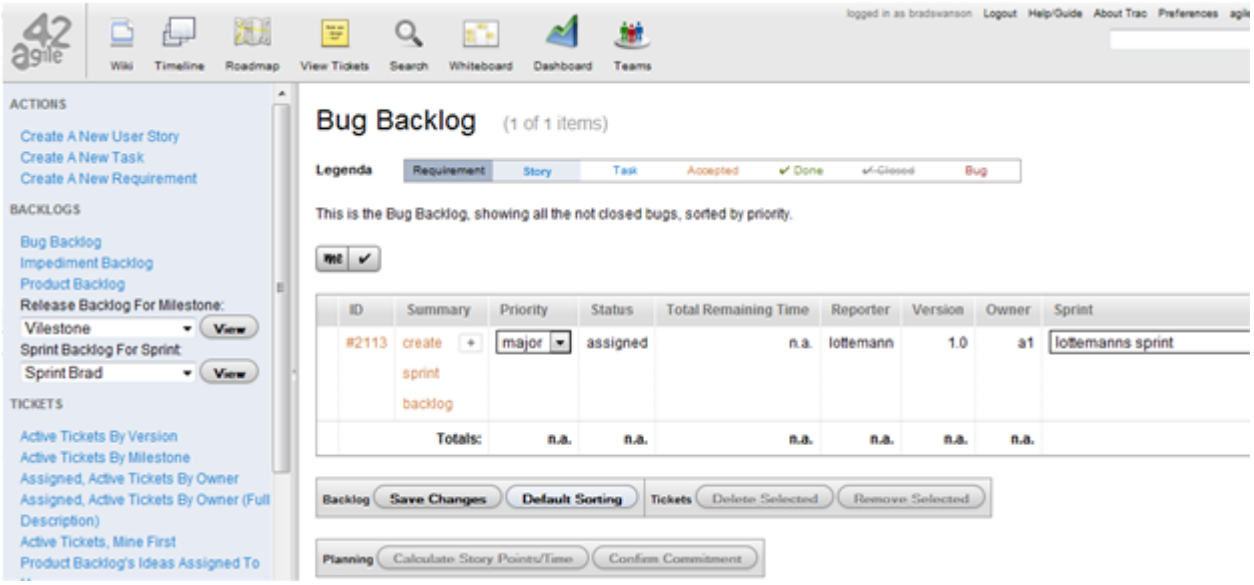
Λευκός πίνακας. Αυτό είναι ένα από τα καλύτερα χαρακτηριστικά του εργαλείου. Είναι αρκετά έξυπνο, με drag and drop ικανότητα. Πρόσθεσε ένα νέο καθήκον για μια ιστορία κάνοντας κλικ με σύμβολο «+».



Χάρτης πορείας. Αυτή η προβολή εμφανίζει όλα τα ορόσημα(εκδόσεις), καθώς και περίληψη κάθε ορμής σε κάθε ορόσημο. Με πολλαπλές ορμές ανά ορόσημο, πολλά scrolling απαιτούνται για να δούμε την μεγάλη εικόνα, γεγονός που καθιστά αυτό το χαρακτηριστικό μάλλον αναποτελεσματικό.



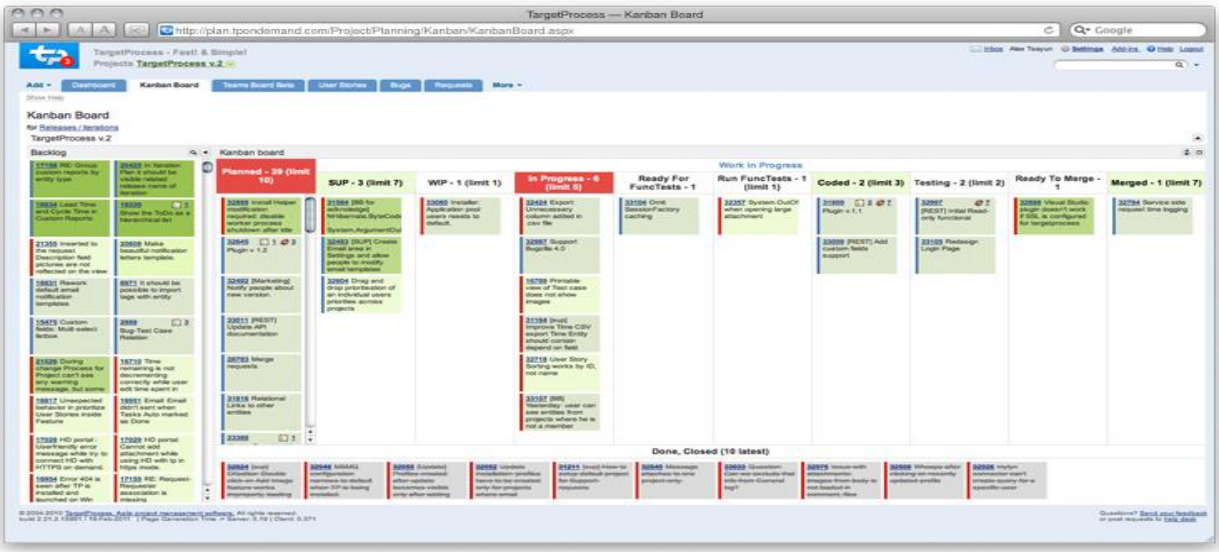
Ανεκτέλεστο σφάλμα. Τα σφάλματα παρακολουθούνται σε δικές τους ξεχωριστές καθυστερήσεις, το οποίο είναι ένα μειονέκτημα, δεδομένου ότι δεν μπορούν να έχουν προτεραιότητα σε σχέση με άλλα στοιχεία ανεκτέλεστων προϊόντων, όπως ιστορίες χρηστών.



4.6 TargetProcess

Το TargetProcess είναι ένα ευέλικτο εργαλείο διαχείρισης έργου. Έχει σχεδιαστεί για να λύσει τα προβλήματα διανομής ομάδας και υποστήριξης ευέλικτων διαδικασιών ανάπτυξης.

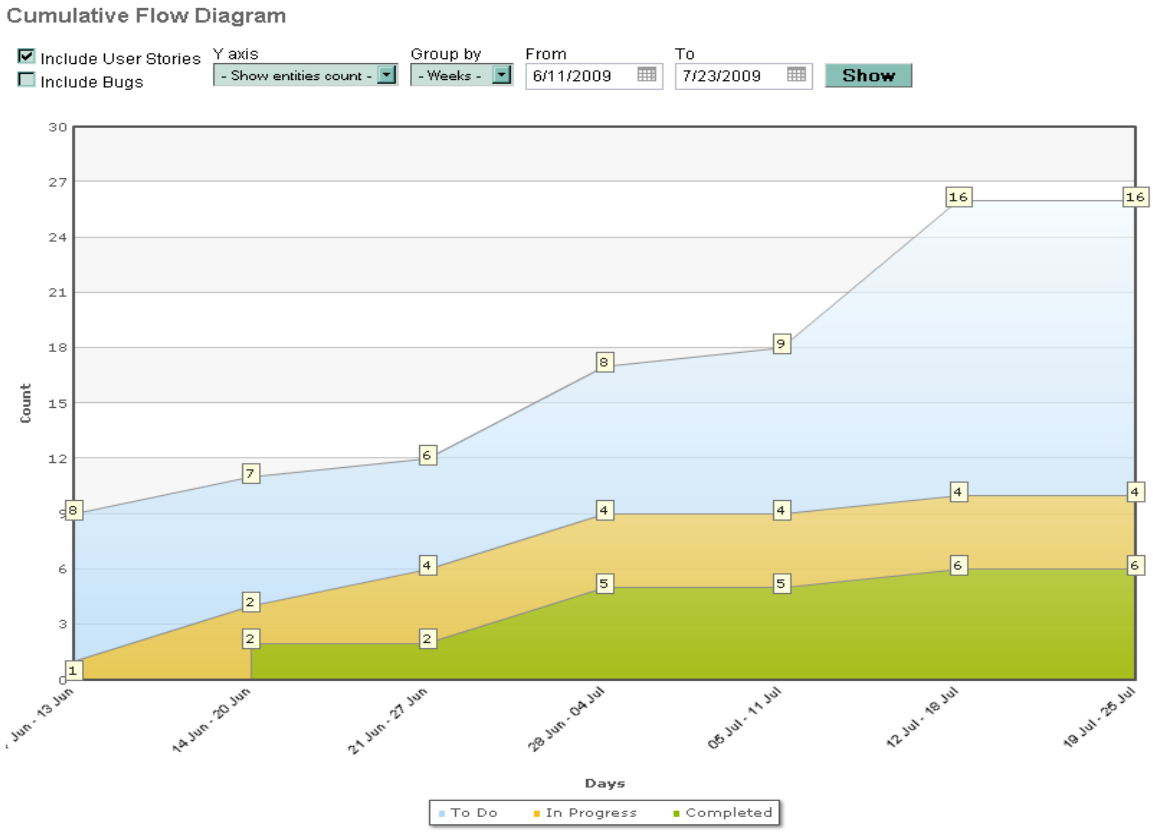
Το Kanban συμβούλιο οπτικοποιεί τη ροή των ιστοριών χρήστη και τα σφάλματα.



Μπορούμε να σχεδιάσουμε ιστορίες χρηστών από τις ανεκτέλεστες λίστες με drag and drop. Γίνεται αλλαγή μελών ιστοριών χρήστη με σφάλματα χρησιμοποιώντας drag and drop. Θέτονται όρια και υπογραμμίζουν τις στήλες αν το όριο υπέρβασης μπορεί να ρυθμίσει τους κίνοες.

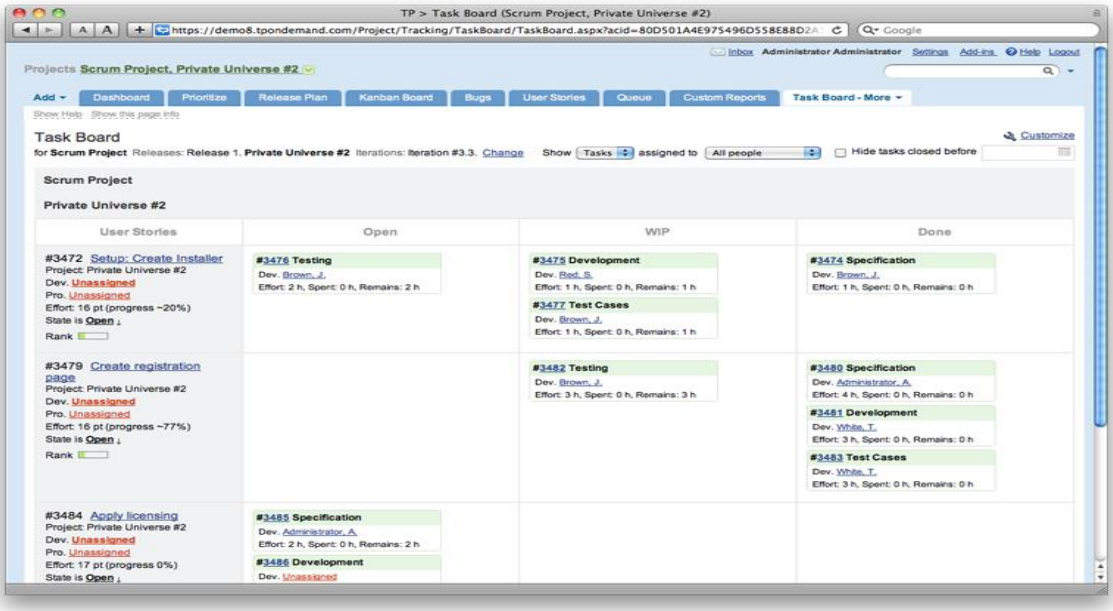
Συγκεντρωτικό διάγραμμα ροής.

Το συγκεντρωτικό διάγραμμα ροής (CFD) βοηθά να παρακολουθούμε τις ιστορίες χρηστών και τα σφάλματα από την πλευρά μας και παρουσιάζει το έργο σε εξέλιξη (WIP) και βοηθάει να βρούμε τα σημεία συμφόρησης.



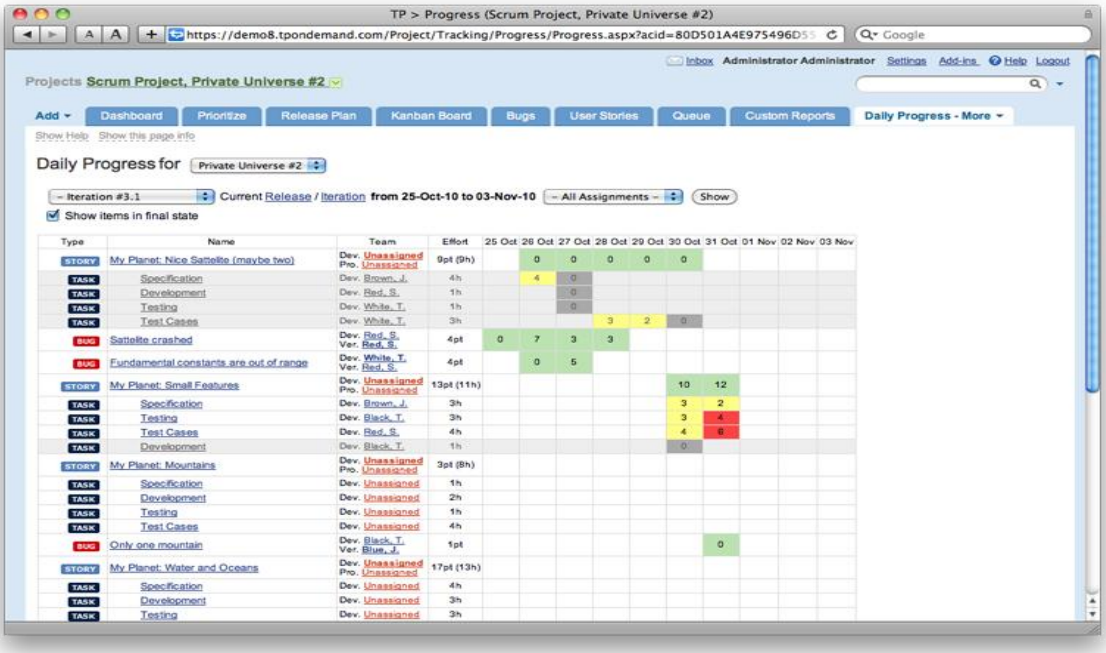
Εργασία συμβουλίου.

Η εργασία συμβουλίου δείχνει την τρέχουσα επισκόπηση επανάληψης και επιτρέπει την γρήγορη αλλαγή στο έργο και κάποιες άλλες δημοφιλής δράσεις.



Ημερήσια πρόοδος.

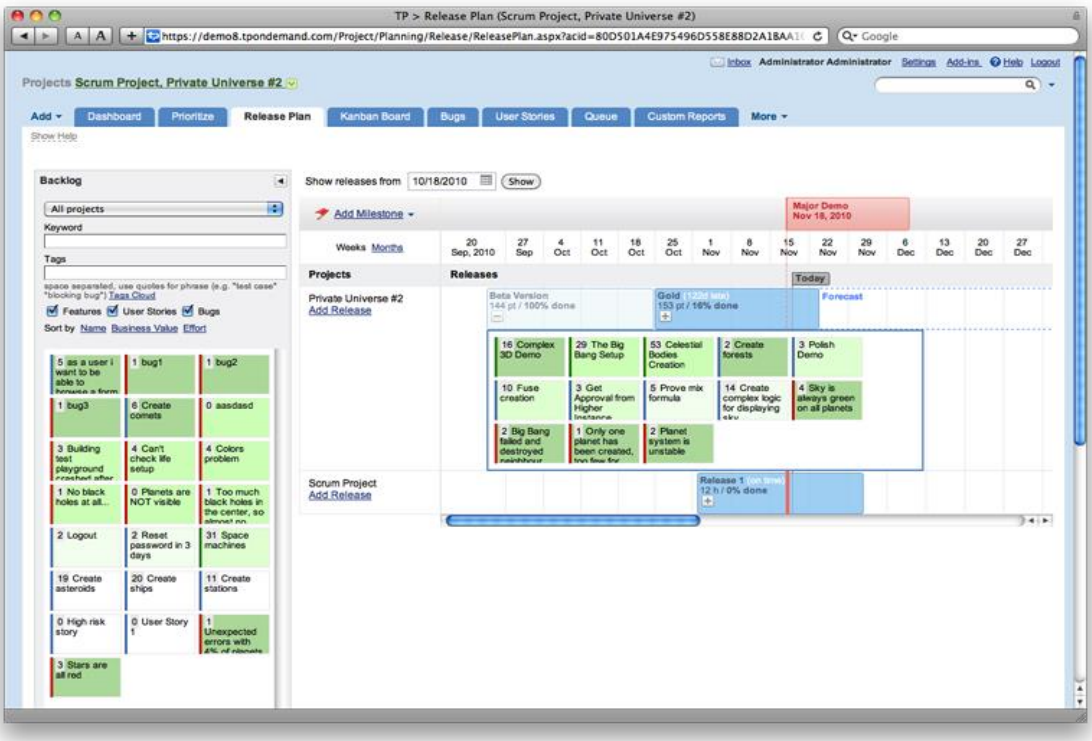
Ο πίνακας προόδου δείχνει πόση προσπάθεια εξακολουθεί να ολοκληρώσει κάθε ιστορία χρήστη σε επιλεγμένες ορμές.



Το TargetProcess υποστηρίζει τις εκδόσεις και τις επαναλήψεις σχεδιασμού μέσω των ευέλικτων drag and drop περιοχών.

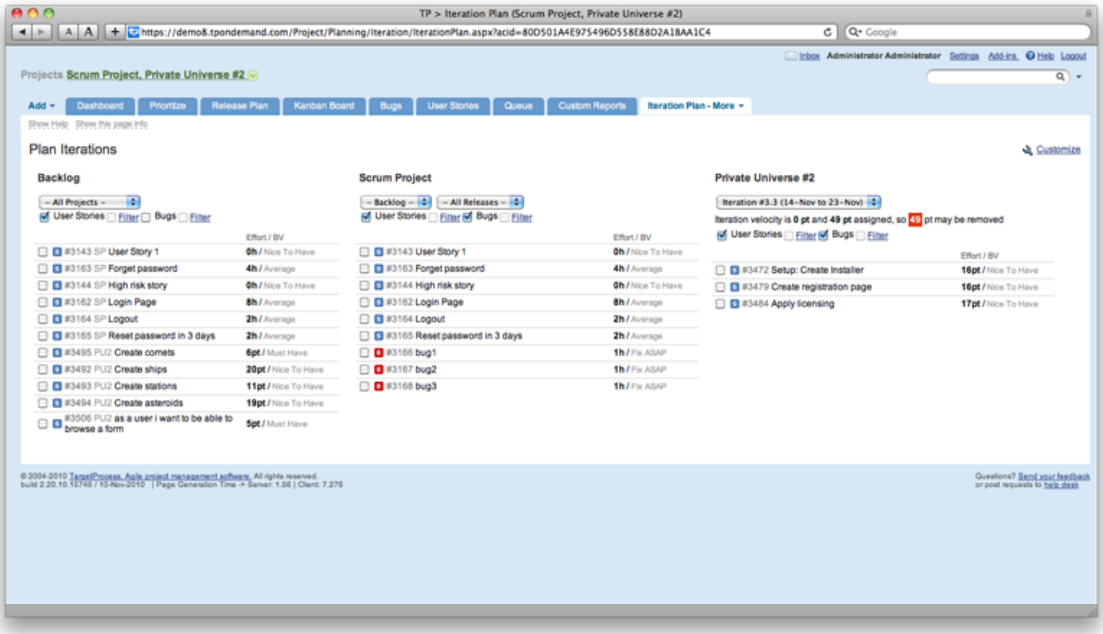
Έκδοση σχεδιασμού.

Σε όλο τον σχεδιασμό υπάρχουν: τα χαρακτηριστικά, οι ιστορίες χρηστών και τα σφάλματα στο ανεκτέλεστο υπόλοιπο, η χρωματική κωδικοποίηση των προτεραιοτήτων, το χρονοδιάγραμμα των εκδόσεων, η απελευθέρωση της πραγματικής ημερομηνίας λήξης. Η πρόβλεψη βασίζεται σε ιστορικά δεδομένα, ταξινόμησης και φιλτράρισμα.



Επανάληψη σχεδιασμού.

Οι ιστορίες σχεδίου και τα σφάλματα σε επαναλήψεις από διαφορετικά έργα σε μία μόνο οθόνη. Η επανάληψη χωρητικότητας είναι σαφώς ορατή.

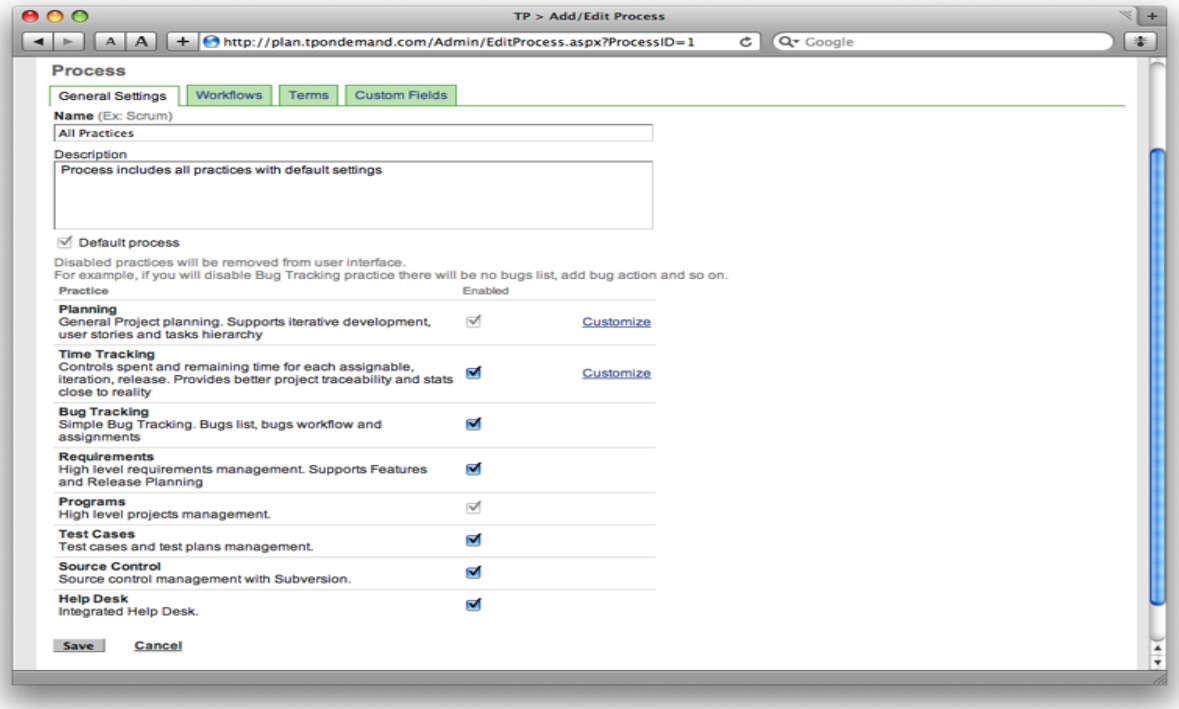


Προσαρμογή

Διαφορετικές εταιρείες ή ακόμη και διαφορετικά σχέδια στο πλαίσιο μιας εταιρείας συχνά χρησιμοποιούν διαφορετικές μεθόδους ανάπτυξης. Το TargetProcess είναι σχεδιασμένο με τη ‘μοναδική διαδικασία για το μοναδικό έργο’. Μπορούμε να δημιουργήσουμε μια μοναδική προσαρμοσμένη διαδικασία για κάθε ένα από τα έργα.

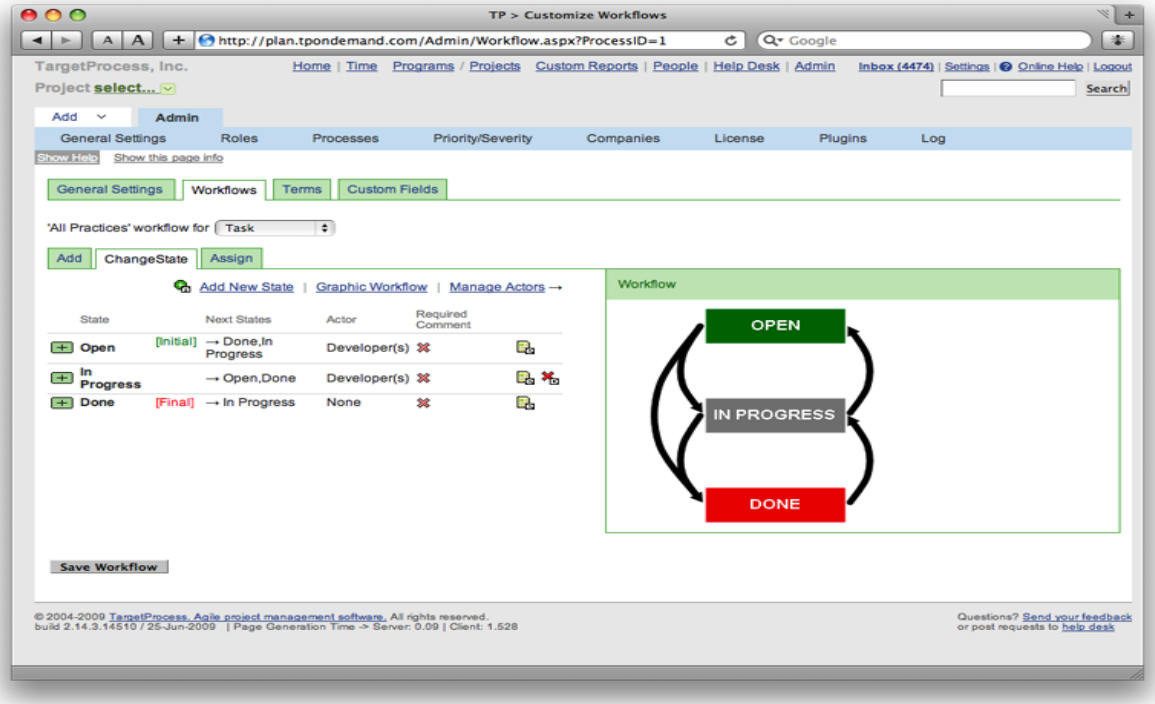
Ανάπτυξη προσαρμογής της διαδικασίας

Κάθε διαδικασία αποτελείται από τις πρακτικές όπως ο σχεδιασμός, η ώρα της παρακολούθησης, τα σφάλματα παρακολούθησης, κλπ.



Οι προσαρμόσιμες ροές εργασίας

Με το TargetProcess εύκολα μπορεί να προσαρμοστούν οι φορείς του κύκλου ζωής και τις εγκρίνει την υπάρχουσα διαδικασία ανάπτυξης.



4.7 Επίλογος

Το IceScrum είναι ένα πολύ ικανό εργαλείο με ένα πλούσιο σύνολο χαρακτηριστικών γνωρισμάτων και μερικές αδυναμίες. Παρά το γεγονός ότι υποστηρίζει πολλαπλά προϊόντα(έργα), είναι κατάλληλα μόνο για μικρά έργα με μία μόνο ομάδα που εργάζεται σε μια ορμή σε ένα χρόνο για κάθε προϊόν.

Το ExtremePlanner είναι ένα εργαλείο που επικεντρώνεται στον σχεδιασμό και την παρακολούθηση της προόδου που έχουν την πραγματική επιχειρηματική αξία στους πελάτες. Το ExtremePlanner υποστηρίζει όλη την ομάδα ανάπτυξης.

Το Agilefant είναι ένα πολύ ικανό εργαλείο με ένα πλούσιο σύνολο χαρακτηριστικών γνωρισμάτων και μερικές αδυναμίες. Είναι καταλληλότερο για μεγάλα έργα και μεγάλες οργανώσεις από οποιαδήποτε από άλλο εργαλείο, αλλά η έλλειψη ιεράρχησης των επιπέδων ιστορίας/απαίτηση είναι ένα σημαντικό μειονέκτημα για τα μεγάλα έργα.

Παρόλο που στο Agilo το διοικητικό συμβούλιο εργασία/προβολή είναι διαισθητικό και εξαιρετικό, άλλα χαρακτηριστικά είναι λιγότερο διαισθητικά και συχνά δεν είναι πολύ εύκολα στη χρήση. Ο περιορισμός ενός προϊόντος ανά εγκατάσταση είναι μεγάλο μειονέκτημα.

Το TargetProcess μπορεί να χρησιμοποιηθεί για όλους τους τύπους σχεδιασμού παρά το γεγονός ότι λειτουργεί καλύτερα αν χρησιμοποιεί ακραίο προγραμματισμό. Είναι κατάλληλο για μικρές και μεσαίες επιχειρήσεις.

5.ΣΥΜΠΕΡΑΣΜΑΤΑ

Οι ευέλικτες μεθοδολογίες ανάπτυξης λογισμικού είναι ένας σχετικά πρόσφατος τομέας ανάπτυξης λογισμικού. Ανάμεσα σε αυτές τις μεθοδολογίες συγκαταλέγονται ο «Ακραίος προγραμματισμός», το Scrum, οι «Μεθοδολογίες κρυστάλλου», η «Ανάπτυξη με βάση τα χαρακτηριστικά», η «Μέθοδος ανάπτυξης δυναμικής συστημάτων», η «Εναρμονισμένη ανάπτυξη λογισμικού».

Οι ευέλικτες μεθοδολογίες αναδύθηκαν μέσα από τον πειραματισμό των εταιρειών ώστε να καλύψουν κάποιες εταιρικές ανάγκες όταν αναζητούσαν εναλλακτικές λύσεις στις παραδοσιακές μεθοδολογίες ανάπτυξης λογισμικού, τις οποίες έβρισκαν πάρα πολύ δυσκίνητες, γραφειοκρατικές και άκαμπτες. Τα προβλήματα σχετικά με τις απαιτήσεις παρείχαν μια άλλη ανάγκη για την δημιουργία νέων μοντέλων ανάπτυξης λογισμικού καθώς επίσης και οι γρήγοροι ρυθμοί αλλαγών των απαιτήσεων αλλά και του επιχειρηματικού περιβάλλοντος.

Οι ευέλικτες μεθοδολογίες ήρθαν να καλύψουν όλες αυτές τις απαιτήσεις των εταιριών. Προσφέρουν την ευκαιρία να παραδίδει μια εταιρεία πιο γρήγορα το προϊόν της στους πελάτες της, να εκμεταλλεύεται της ευκαιρίες της αγοράς, να ανταποκρίνεται πιο γρήγορα στις αλλαγές του περιβάλλοντος και στις αλλαγές των απαιτήσεων των πελατών. Επιπλέον, οι ευέλικτες μεθοδολογίες επιτρέπουν στους εργαζόμενους να απομακρύνουν όλα τα περιττά τμήματα των διαδικασιών που δυσχεράνουν ή καθυστερούν την εργασία τους κάνοντας τους πιο παραγωγικούς αλλά ταυτόχρονα μειώνοντας και τα περιττά κόστη. Στα πλεονεκτήματα των ευέλικτων μεθοδολογιών είναι και η μείωση του κινδύνου, η βελτίωση του ελέγχου και των επικοινωνιών.

Όλες οι ευέλικτες μεθοδολογίες μοιράζονται ένα παρόμοιο σύνολο συνεργατικών αξιών και αρχών και εκτιμούν την ύπαρξη μιας μόλις ικανοποιητικά ορισμένης μεθοδολογίας. Εντούτοις, κάθε μέθοδος προσεγγίζει τα προβλήματα που αντιμετωπίζονται στην τεχνολογία λογισμικού από μια διαφορετική οπτική. Φυσικά κάθε μεθοδολογία δεν είναι κατάλληλη για οποιοδήποτε πρόγραμμα.

Οι εταιρείες που έχουν εφαρμόσει πιλοτικά προγράμματα ακολουθώντας τις ευέλικτες μεθόδους φαίνεται να έχουν αντιμετωπίσει αρκετά προβλήματα. Εντούτοις, δεν παραγνωρίζουν και τα ευεργετικά στοιχεία που φέρουν οι

μέθοδοι. Αυτά τα στοιχεία είναι που κάνουν τις περισσότερες να συνεχίζουν να χρησιμοποιούν στοιχεία των ευέλικτων μεθόδων σε όλα τους τα προγράμματα, αλλά και να συνεχίζουν να προσπαθούν να μετατρέψουν τις οργανωτικές τους δομές σε πιο συμβατές με τις ευέλικτες μεθοδολογίες.

6.ΕΠΙΛΟΓΟΣ

Τις τελευταίες δεκαετίες, ενώ οι δυνάμεις της αγοράς, οι απαιτήσεις των συστημάτων, η τεχνολογία των εφαρμογών και το προσωπικό των προγραμμάτων άλλαζαν σε ένα σταθερά αυξανόμενο ποσοστό, ένας διαφορετικός τρόπος ανάπτυξης, ο ευέλικτος παρουσίασε τα πλεονεκτήματα του πέρα από το παραδοσιακό. Αυτός ο ευέλικτος τρόπος της ανάπτυξης εξετάζει άμεσα στα προβλήματα αλλαγών.

Ο όρος ευέλικτη ανάπτυξη, είναι πλασμένος από μια ομάδα ανθρώπων πεπειραμένη στην ανάπτυξη του λογισμικού και έχει δύο ευδιάκριτες υποδηλώσεις. Η πρώτη είναι η ιδέα ότι ο κόσμος των επιχειρήσεων και της τεχνολογίας έχουν γίνει υψηλής ταχύτητας και αβέβαιος, απαιτώντας μια διαδικασία που ταυτόχρονα να δημιουργεί και να ανταποκρίνεται γρήγορα στην αλλαγή. Η πρώτη δήλωση υπονοεί την δεύτερη: μια ευέλικτη διαδικασία απαιτεί ανθρώπους και οργανώσεις με καλά αντανακλαστικά στις αλλαγές. Η ευέλικτη ανάπτυξη εστιάζει στα ταλέντα και τις δεξιότητες των ατόμων και διαμορφώνει τη διαδικασία γύρω από τους συγκεκριμένους ανθρώπους και τις ομάδες.

Κατά την διάρκεια αυτής της μελέτης παρουσιάστηκαν δύο από τις πιο σημαντικές ευέλικτες μεθοδολογίες ο «ακραίος προγραμματισμός» και το «SCRUM» μέσω των διαδικασιών, των ρόλων και των πρακτικών που διακατέχει η κάθε μία. Στην συνέχεια παρουσιάστηκαν κάποια παραδείγματα της εφαρμογής των ευέλικτων μεθόδων σε μεγάλες διεθνείς επιχειρήσεις. Τέλος, παρουσιάστηκαν πέντε ευέλικτα εργαλεία διαχείρισης έργου.

Δυστυχώς παραδείγματα εφαρμογής των ευέλικτων μεθόδων δεν υπάρχουν για την ελληνική πραγματικότητα. Οι επιχειρήσεις παραγωγής λογισμικού στην χώρα μας είναι συνήθως είτε μικρού μεγέθους είτε παραστήματα κάποιων πολυεθνικών

ΒΙΒΛΙΟΓΡΑΦΙΑ

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R., Melllor, S., Schwaber, K., Sutherland, J. and Thomas, D. *Manifesto for Agile Software Development*(22.3.2002).

Brekkan Elin and Eystein Mathisen.(2010) *Introducing Scrum in Companies in Norway: A Case Study*.InSITE

Browers J. et al., *Tailoring XP for Large System Mission Critical Software Development, Proc. 2nd XP Universe and 1st Agile Universe Conf. on Extreme Programming and Agile Methods*, Springer, 2002, pp.100-111.

Dinusha S. Jayawardena, Dr Lesly L. Ekanayake.(2010). *Adaptation Analysis of Agile Project Management for managing IT projects in Sri Lanka*. IEEE

Domingo A. et al, '*Agile Software Development in Large Organizations',* IEEE Computer, Dec 2004, pp. 26-34.

Dr. David F. Rico, Dr. Hasan H. Sayani.(2009). *Use of Agile Methods in Software Engineering Education*. IEEE computer society

Qumer Asif, Brian Henderson-Sellers.(2006) *Comparative evaluation of XP and scrum using the 4D Analytical tool(4-DAT).* EMCIS

Paasivaara Maria, Sandra Durasiewicz and Casper Lassenius.(2009). *Using Scrum in Distributed Agile Development: A Multiple Case Study*. IEEE computer society.

Highsmith, J (2002). *Agile software development ecosystems*. Boston, MA., Pearson Education.

Joao M. Fernandes and Mauro Almeida. *Classification and Comparison of Agile Methods*

Kathleen B. Hass, PMP. (2007) *The Blending of Traditional and Agile Project Management*. PMWorldToday

Laurie Williams.(2007) *A Survey of Agile Development Methodologies*
Orlando Murru, Robrto Deias, and Giampiero Mugheddu, Fst s.r.l(2003) *Assessing XP at a European Internet Company*. IEEE software.

Pekka Abrahamsson, Juhani Wastra, Mikko T. Siponen and Jussi Ronkainen(2003).*New Directions on Agile Methods A Comparative Analysis*. IEEE software

Pekka, Salo, Outi, Rankainen, Jussi & Wastra, Juhani. (2002) *Agile software development methods*, VVT Publications

Rumpe Bernhard, Astrid Schroder. *Quantitative Survey on Extreme Programming Projects.*

Sison Raymund and Theresa Yang.(2007). *Use of Agile Methods and Practices in the Philippines*. IEEE computer society

Vanhanen J, J. Jartti, and T. Kahkonen, *Practical Experiences of Agility in the Telecom Industry,* Proc. 4th Int' 1 Conf. Extreme Programming and Agile Processes in Software Eng., Springer, 2003, pp. 279-287.

Vijan Mahnic, Slavko Drnovscek. *Agile Software Project Management with Scrum*, Ljubljana, Slovenia.