AGILE TESTING ΚΑΙ ΕΡΓΑΛΕΙΑ

Εργασία των:

***Αθανασίου Λυδία**, Lydia_athanasiou@hotmail.gr*
Α.Μ. 3170003

***Κωστούλας Ορέστης**, p3140102@aueb.gr*
Α.Μ. 3140102

Υπεύθυνος Καθηγητής: ***Μαλεύρης Νικόλαος***
Αθήνα, Μάιος, 2020

Πίνακας Περιεχομένων

1η ενότητα: Θεωρητική προσέγγιση του θέματος.

Η ανάπτυξη της τεχνολογίας έχει προσφέρει τεράστιες δυνατότητες σε όλους μας. Ανεξαιρέτως φύλου και ηλικίας, όλοι οι άνθρωποι επωφελούνται από τα πλεονεκτήματά της και, κατ' επέκταση, του λογισμικού που είναι διαθέσιμο στη ζωή τους. Με τη πάροδο του χρόνου, έχουμε οδηγηθεί από την ύπαρξη ελάχιστων υπολογιστών που αποσκοπούσαν στην εκτέλεση συγκεκριμένων πράξεων και στην επίλυση συγκεκριμένων προβλημάτων, στην ύπαρξη πολλών υπολογιστών στο σπίτι μας για την εξυπηρέτηση διαφορετικών αναγκών. Αυτό, φυσικά, είναι συνέπεια τόσο της βελτίωσης των υπολογιστών διαχρονικά, όσο και της ανάπτυξης ειδών λογισμικού για την εξυπηρέτηση της κάθε ανάγκης. Πλέον, ακόμα και μικρές επιχειρήσεις όπως και ιδιώτες χρειάζονται προσωποποιημένο λογισμικό για τις ανάγκες τους. Συνήθως, η δημιουργία αυτού του λογισμικού ανατίθεται σε ομάδες ειδικών. Υπήρχαν και υπάρχουν πολλές προσεγγίσεις για τον τρόπο ανάπτυξης αυτού του λογισμικού και μία από αυτές που πλέον έχει σχεδόν επικρατήσει υπέρ όλων των άλλων είναι η Agile.
Παρακάτω θα εξετάσουμε: Τι είναι η Agile, σε τι βασίζεται, τι προσφέρει και γιατί να την επιλέξουμε. Επιπλέον θα αναλυθούν τα δημοφιλέστερα Agile εργαλεία που χρησιμοποιούνται.

1.1 Τι είναι η Agile

Η Agile είναι μια ομάδα μεθόδων ανάπτυξης λογισμικού του οποίου οι απαιτήσεις και οι λύσεις εξελίσσονται μέσω της συνεχούς συνεργασίας μεταξύ αυτοοργανωμένων, διατμηματικών ομάδων εργασίας. Προωθεί τον προσαρμοστικό σχεδιασμό, την εξελικτική ανάπτυξη, την έγκαιρη παράδοση, τη συνεχή βελτίωση και ενθαρρύνει την ταχεία και ευέλικτη ανταπόκριση στις αλλαγές. [1,10,26]
Πραγματεύεται, ουσιαστικά, τον γρήγορο και σταδιακό σχεδιασμό, την παραγωγή και παράδοση ενός υπό-προϊόντος το οποίο μονίμως παρακολουθείται. Αυτό παραδίδεται το συντομότερο δυνατόν στον πελάτη με στόχο την ανάδρασή του. Έπειτα, το προϊόν προσαρμόζεται στις νέες απαιτήσεις που δημιουργήθηκαν. Αυτή η διαδικασία συνεχίζεται έως ότου καλυφθούν όλες οι απαιτήσεις του πελάτη και παραδοθεί το τελικό προϊόν.

1.2. Ιστορία

Αν και οι μηχανικοί λογισμικού ανέπτυσσαν και δοκίμαζαν τρόπους ανάπτυξης λογισμικού που είχαν χαρακτηριστικά της Agile από τη δεκαετία του ‘60, η Agile, στη μορφή που γνωρίζουμε σήμερα, καθιερώθηκε το 2001. Σε ένα χιονοδρομικό κέντρο στη Γιούτα (Utah) συνεδρίασαν 17 προγραμματιστές οι οποίοι καθιέρωσαν τον όρο “Agile” δημοσιεύοντας και το “*Manifesto for Agile Software Developmen*t” με το οποίο καθιερώθηκαν οι Agile τεχνικές. [26]

1.3. Το Agile μανιφέστο

Στο Agile μανιφέστο αναγράφεται [14] :

“Ανακαλύπτουμε καλύτερους τρόπους ανάπτυξης λογισμικού στην πράξη και βοηθάμε τους άλλους να κάνουν το ίδιο.
Αυτή η δραστηριότητα μας έχει οδηγήσει στο να αξιολογούμε:

- *Τα άτομα και τις αλληλεπιδράσεις πάνω από τις διαδικασίες και τα εργαλεία*
- *Το λογισμικό που λειτουργεί πάνω από την εκτενή τεκμηρίωση*
- *Τη συνεργασία με τον πελάτη πάνω από τις συμβατικές διαπραγματεύσεις*
- *Την ανταπόκριση στην αλλαγή πάνω από την τήρηση ενός προδιαγεγραμμένου σχεδίου*

Με άλλα λόγια, παρόλο που είναι αξιόλογες οι δηλώσεις στα δεξιά, προσδίδουμε μεγαλύτερη αξία στις δηλώσεις στα αριστερά ”

1.4. Οι 12 Αρχές της Agile

Μαζί με το Agile μανιφέστο δημοσιεύτηκαν και οι παρακάτω 12 αρχές που θα πρέπει να ακολουθούνται κατά τους κύκλους ανάπτυξης λογισμικού [**14**]:

1. Πρώτη προτεραιότητα είναι η ικανοποίηση του πελάτη μέσω της έγκαιρης και συνεχούς παράδοσης χρήσιμου λογισμικού.

2. Οι αλλαγές στις απαιτήσεις είναι ευπρόσδεκτες, ακόμα και σε προχωρημένα στάδια της ανάπτυξης. Οι ευέλικτες διαδικασίες δαμάζουν τις αλλαγές με στόχο την ενίσχυση του ανταγωνιστικού πλεονεκτήματος του πελάτη.

3. Παραδίδουμε συχνά λογισμικό που λειτουργεί, σε διαστήματα μερικών εβδομάδων ή μηνών, με προτίμηση στη συντομότερη χρονική κλίμακα.

4. Οι προγραμματιστές και οι ειδικοί της αγοράς πρέπει να συνεργάζονται καθημερινά καθ' όλη τη διάρκεια του έργου.

5. Θεμελιώνουμε τα έργα γύρω από άτομα με πάθος και ενδιαφέρον. Διαμορφώνουμε το κατάλληλο περιβάλλον, τους παρέχουμε την αναγκαία υποστήριξη και εμπιστευόμαστε την ικανότητά τους να φέρουν σε πέρας την αποστολή τους.

6. Η πιο αποδοτική και αποτελεσματική μέθοδος για τη μετάδοση πληροφορίας προς και εντός της ομάδας ανάπτυξης λογισμικού είναι η συνομιλία πρόσωπο με πρόσωπο.

7. Το λογισμικό που λειτουργεί είναι το κύριο μέτρο προόδου.

8. Οι ευέλικτες διαδικασίες προάγουν την αειφόρο ανάπτυξη. Οι χορηγοί, η ομάδα ανάπτυξης λογισμικού και οι χρήστες θα πρέπει να είναι σε θέση να διατηρούν ένα σταθερό ρυθμό επ' αόριστον.

9. Η διαρκής έμφαση στην τεχνική αρτιότητα και στην εύρυθμη σχεδίαση ενισχύουν την ευελιξία.

10. Η απλότητα, δηλαδή η τέχνη της μεγιστοποίησης του όγκου της δουλειάς που δεν χρειάζεται να γίνει, είναι ουσιώδης.

11. Οι καλύτερες αρχιτεκτονικές, απαιτήσεις και σχέδια προκύπτουν από ομάδες που οργανώνονται μόνες τους.

12. Σε τακτά χρονικά διαστήματα, η ομάδα συλλογίζεται για το πως θα γίνει πιο αποτελεσματική, ρυθμίζοντας και προσαρμόζοντας τη συμπεριφορά της αναλόγως.

1.5. Πλεονεκτήματα

Όπως και κάθε άλλος τρόπος ανάπτυξης λογισμικού, η Agile έχει τα θετικά και τα αρνητικά της.

Ας δούμε πρώτα τα θετικά της [1,8,13,25]:

- Είναι εύκολο να ανταποκριθεί το λογισμικό σε αλλαγές.

Ένα από τα μεγαλύτερα πλεονεκτήματα της Agile σε σχέση με άλλους τρόπους ανάπτυξης λογισμικού είναι η προσαρμοστικότητα που έχει στις απαιτήσεις του λογισμικού. Οι Agile μέθοδοι δεν μας αναγκάζουν να ακολουθούμε ένα αυστηρά οροθετημένο πλάνο το οποίο δεν αφήνει περιθώρια για αλλαγές. Δυναμικά, επιλέγονται τα πιο σημαντικά ζητήματα στα οποία πρέπει να επικεντρωθούν οι προγραμματιστές. Η διαδικασία ανάπτυξης λογισμικού αποτελείται από μεταπηδήματα από το πιο σημαντικό πράγμα στο αμέσως επόμενο κ.ο.κ. Έτσι, επιτυγχάνεται η καλύτερη ανταπόκριση του λογισμικού στις απαιτήσεις που ανακαλύπτονται στην πορεία ανάπτυξής του. Επιπλέον, λόγω αυτού του κύκλου ανάπτυξης λογισμικού, οι πελάτες παίρνουν και εκείνοι στα χέρια τους πιο γρήγορα το λογισμικό που χρειάζονται περισσότερο, χωρίς να χρειάζεται να περιμένουν ένα πλήρως ολοκληρωμένο παραδοτέο.

- Ανταποκρίνεται στις “ρεαλιστικά” υπάρχουσες γνώσεις των ομάδων ανάπτυξης λογισμικού

Κατά την ανάπτυξη λογισμικού με ένα σύνολο μεθοδολογιών που παρέχουν ευελιξία στα στάδια ανάπτυξης, υπάρχει χώρος για λάθη και για απόκτηση νέων γνώσεων από τους εργαζόμενους. Με ένα αυστηρά ορθολογικό σύστημα, πρέπει να έχουν ληφθεί πολύ σοβαρά υπόψη οι γνώσεις των προγραμματιστών που θα αναπτύξουν το λογισμικό. Ο εκτιμώμενος χρόνος παράδοσης του λογισμικού θα βασιστεί στις γνώσεις και στον αριθμό των εργαζομένων και θα υπάρχουν ελάχιστα περιθώρια για λάθη. Οι ευέλικτοι τρόποι ανάπτυξης λογισμικού αποδέχονται μία πιο ρεαλιστική κατάσταση, κατά την οποία, στην πορεία της ανάπτυξης του λογισμικού, θα ανακαλύπτονται νέες λύσεις, νέα προβλήματα και νέες γνώσεις στις οποίες θα προσαρμόζονται οι προγραμματιστές.

- Λιγότερη ανούσια δουλειά.

Σε ένα σύστημα όπου χρειάζεται να περιγραφεί και να αναλυθεί εξονυχιστικά εξ αρχής ολόκληρο το λογισμικό, το ενδεχόμενο να μην περιγραφεί κάτι σωστά και να χρειαστούν μη προβλεπόμενες αλλαγές είναι σχεδόν σίγουρο. Αυτή η εκτεταμένη ανάλυση του λογισμικού δεν έχει θέση στους Agile τρόπους. Φυσικά, υπάρχει ένα πλάνο και ένας στόχος που πρέπει να επιτευχθεί αλλά οι λεπτομέρειες καθορίζονται στην πορεία με βάση το τι θα προκύψει. Έτσι, επιτυγχάνεται ίσως και ένα καλύτερο αποτέλεσμα, με τη χρήση λιγότερων πόρων πριν την εκκίνηση δημιουργίας του λογισμικού.

- Περισσότερη και πιο ουσιώδης ανάδραση.

Κατά την ανάπτυξη ενός λογισμικού, του οποίου τα χαρακτηριστικά και τα κριτήρια που πρέπει να πληροί εμφανίζονται κατά την ανάπτυξή του, κατέχει πολύ σημαντικό ρόλο η σωστή αξιολόγησή του. Αυτό γίνεται με συντομότερους κύκλους αξιολόγησης οι οποίοι καθορίζουν τι πρέπει να γίνει έως τον επόμενο. Καταλαβαίνουμε, πως η αξία τους είναι πολύ μεγάλη καθώς εξετάζουν όχι μόνο τυχόν λάθη που έχουν γίνει, αλλά καθορίζουν και πώς θα εξελιχθεί το λογισμικού με βάση το τι έχει γίνει έως τώρα.

- Φτάνει πιο γρήγορα το λογισμικό στον πελάτη

Αφού οι κύκλοι ανάπτυξης δεν είναι κλειστοί, αλλά βασίζονται στην ανάδραση που αναφέρθηκε και παραπάνω, μέρος της διαδικασίας ανάπτυξης λογισμικού είναι πλέον και ο πελάτης του προϊόντος. Η ανάγκη για την αξιολόγηση του λογισμικού από τον πελάτη, όχι μόνο για το τι είναι πιο σημαντικό αλλά και για τι πρέπει να αλλάξει, να καταργηθεί η να προστεθεί, καθιστούν τη συνεχόμενη παράδοση εκδόσεων λογισμικού απαραίτητη. Αυτό έχει και ως αποτέλεσμα την πιο γρήγορη, χρονικά, χρήση - έστω και μίας πιο βασικής έκδοσης λογισμικού - από τον πελάτη.

1.6. Μειονεκτήματα

Τα μειονεκτήματα της χρήσης των ευέλικτων μεθόδων ανάπτυξης λογισμικού είναι [8,11]:

- Έλλειψη κατανόησης της Agile.

Μεγάλο πρόβλημα των μη αυστηρά οροθετημένων μεθοδολογιών, ανεξαιρέτως του κλάδου επιστήμης τους, είναι η έλλειψη κατανόησης για το πως δουλεύουν. Η Agile δυστυχώς δεν είναι εξαίρεση σε αυτό - όταν μία ομάδα δεν έχει αυστηρές οδηγίες για το πώς θα λειτουργήσει, είναι εύκολο να χαθεί ή και να παραβιάσει μερικές αρχές χωρίς να το καταλάβει. Αυτό παρατηρείται ειδικά σε εταιρείες και ομάδες που θέλουν να είναι "Agile" αλλά δεν έχουν προσλάβει το απαραίτητο προσωπικό για να εκπαιδεύσει τα μέλη τους για το πώς να λειτουργήσουν, με αποτέλεσμα να γίνεται χαοτική η ανάπτυξη λογισμικού και, σπανίως, να αποδίδονται τα λάθη της ανάπτυξης λογισμικού στους Agile τρόπους αντί να εντοπίζονται τα πραγματικά προβλήματα.

- Κακές συμπεριφορές

Η ευελιξία που παρέχουν οι ευέλικτοι τρόποι ανάπτυξης λογισμικού σε συνδυασμό με μη εκπαιδευμένους προγραμματιστές μπορεί να φέρει δυσάρεστες συνέπειες στο εργασιακό περιβάλλον. Η βάπτιση χαοτικών μεθοδολογιών ως Agile είναι συχνό φαινόμενο. Επιπλέον, παρατηρείται η εστίαση μόνο στις αριστερά προτάσεις που παρουσιάζονται στο Agile μανιφέστο και παραλείπονται οι δεξιά. Αυτό οδηγεί στην έλλειψη πλάνου, σωστών εργαλείων, τεκμηρίωσης λογισμικού και πολλών άλλων που, μολονότι δεν θεωρούνται τα σημαντικότερα από την Agile, δεν σημαίνει πως η πλήρης έλλειψή τους δεν θα έχει καταστροφικές συνέπειες.

- Λιγότερο προβλέψιμη

Φυσικά, όπως αναφέρθηκε και παραπάνω, σε ένα αυστηρό πλάνο θα προκύψουν ζητήματα που μάλλον δεν θα έχουν προβλεφθεί, όμως παρόλα αυτά, αυτό το αυστηρό πλάνο θα είναι πιο προβλέψιμο από την ενδεχόμενη πορεία ενός λογισμικού που αναπτύσσεται με ευέλικτους τρόπους. Σε ένα τέτοιο λογισμικό πολύ συχνά θα παρατηρηθεί μείωση ή αύξηση των κύκλων ανάπτυξης τους, με αποτέλεσμα να μπορεί δυσκολότερα να διαχειριστεί τον χρόνο και τις μελλοντικές υποχρεώσεις.

1.7. Agile Testing

Βασικό συστατικό της αποτελεσματικότητας των ευέλικτων τρόπων ανάπτυξης λογισμικού είναι η κατανόηση πως ο έλεγχος του λογισμικού δεν βρίσκεται σε διαφορετικό κύκλο ανάπτυξης, αλλά είναι ενταγμένος σε όλες τις φάσεις δημιουργίας λογισμικού. Τα μέλη της ομάδας που έχουν ως ρόλο τον έλεγχο του λογισμικού έχουν καταλυτικό ρόλο στην ποιότητα του τελικού προϊόντος.
Η αλληλεπίδραση μεταξύ πελατών και προγραμματιστών είναι κρίσιμη. Οφείλουν να βρίσκονται σε συνεχή ενημέρωση τόσο για τις ανάγκες των πελατών όσο και για την κατάσταση και την πορεία του λογισμικού που αναπτύσσεται. Αφού ο προγραμματισμός, ο έλεγχος και η ανάδραση γίνονται επαυξητικά και διακριτικά, οι testers καλούνται να κρίνουν όχι μόνο τη ποιότητα του δημιουργηθέντος λογισμικού αλλά και κατά πόσο αποκρίνεται στις ανάγκες των πελατών.

Το Agile testing καθώς δεν γίνεται στο τέλος του έργου, αλλά καθ' όλη τη διάρκεια, έχει ως αποτέλεσμα όχι μόνο την ευκολότερη επίλυση των προβλημάτων που θα παρουσιαστούν, αλλά και τη μείωση του οικονομικού και χρονικού κόστους των προβλημάτων για την ομάδα ανάπτυξής τους.[12,18]

1.8. Το Agile μανιφέστο από την σκοπιά του testing

I) Τα άτομα και τις αλληλεπιδράσεις πάνω από τις διαδικασίες και τα εργαλεία

Οι ελεγκτές λογισμικού θα πρέπει να δουλεύουν και να συνεργάζονται συνεχώς με τους προγραμματιστές και τους πελάτες του λογισμικού ώστε να καταλάβουν καλύτερα το λογισμικό που αναπτύσσεται. Γνωρίζοντας καλύτερα την πορεία ανάπτυξης του αλλά και τις απαιτήσεις του, θα τους κάνει να καταλάβουν καλύτερα που και πως πρέπει να εστιάσουν στην ελεγκτική διαδικασία. Επιπλέον θα βελτιωθεί η ολική τους εικόνα προς το έργο, αφού θα μπορούν να προσδιορίσουν τι θα το κάνει πετυχημένο.

II) Το λογισμικό που λειτουργεί πάνω από την εκτενή τεκμηρίωση

Τα κριτήρια σύμφωνα με τα οποία αξιολογείται το έργο από τους ελεγκτές δεν είναι και δεν θα έπρεπε να είναι αυστηρώς οριοθετημένα. Δεν υπάρχει μία στάνταρ λίστα με προϋποθέσεις που θα πρέπει να πληροί το λογισμικό. Αντιθέτως πρέπει να γίνεται δυναμικά ο καθορισμός των απαιτήσεων του λογισμικού σύμφωνα με τα συμπεράσματα που βγάζουν οι ελεγκτές από τις συνεχείς τους αλληλεπιδράσεις με την ομάδα παραγωγής έργου και τους πελάτες του

III) Τη συνεργασία με τον πελάτη πάνω από τις συμβατικές διαπραγματεύσεις

Ανεξαιρέτως αν οι ελεγκτές έχουν άμεση ή έμμεση επαφή με τους πελάτες του λογισμικού, σε ένα Agile περιβάλλον εργασίας, πρέπει να εστιάζουν στο τι είναι σημαντικό για τον πελάτη και όχι στην συνεχή διαπραγμάτευση για το τι θα του προσφέρει το λογισμικό. Ο τρόπος ικανοποίησης των αναγκών του καταναλωτή διαφέρει από έργο σε έργο οπότε, ειδικά η αυστηρή τοποθέτηση απαιτήσεων λογισμικού και η παρεμβολή αυτών μόνο μέσω διαπραγμάτευσης θα πρέπει να αποθαρρυνθεί. Η καλή συνεργασία μεταξύ του πελάτη και της ομάδας ανάπτυξης και ελέγχου λογισμικού κατά τους ευέλικτους τρόπους ανάπτυξης, είναι κρίσιμη.

IV) Την ανταπόκριση στην αλλαγή πάνω από την τήρηση ενός προδιαγεγραμμένου σχεδίου

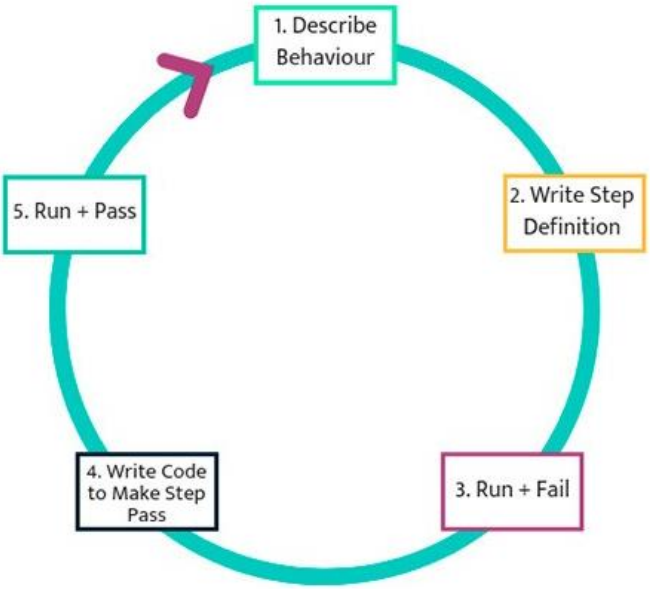
Στην πορεία ανάπτυξης κάθε project είναι σχεδόν αδύνατο να μην δημιουργηθούν προβλήματα ή απαιτήσεις οι οποίες δεν θα συμβαδίζουν με το αρχικό σχέδιο ανάπτυξης. Η σωστή αντιμετώπιση τέτοιων προβλημάτων σύμφωνα με την Agile προϋποθέτει τη διάθεση και την ενθάρρυνση, της παρέκκλισης της ομάδας ανάπτυξης από το προδιαγεγραμμένο σχέδιο καθώς και την ανταπόκριση και την προσαρμογή στις νέες αυτές απαιτήσεις. Η προσαρμογή αυτή όχι μόνο θα μειώσει το ρίσκο της επαυξητικής εμφάνισης προβλημάτων στη πορεία ανάπτυξης, αλλά και θα φέρει ως αποτέλεσμα τον υψηλότερο βαθμό ικανοποίησης του πελάτη από το τελικό προϊόν.
[19]

1.8. Agile Testing Μεθοδολογίες

Οι Agile testing μεθοδολογίες αποτελούν ένα σημαντικό μέρος των ευέλικτων τρόπων ανάπτυξης λογισμικού. Είναι ενταγμένες στους κύκλους ανάπτυξης του έργου και δεν σταματάνε ποτέ. Οι πιο δημοφιλής μέθοδοι Agile testing είναι [18, 19,27] :

- Behavior Driven Development (BDD)

Με αυτή τη μεθοδολογία, ενθαρρύνεται η συνεχής επικοινωνία μεταξύ των ενδιαφερόμενων του λογισμικού που παράγεται. Οι υπεύθυνοι εξέλιξης του λογισμικού σε συνδυασμό με τις υποστηρικτικές τους ομάδες (testers, οικονομικοί αναλυτές) δημιουργούν σενάρια που απεικονίζουν το πώς μια δοσμένη λειτουργικότητα θα πρέπει να ανταποκριθεί σε διαφορετικές καταστάσεις με διαφορετικά δεδομένα. Αυτά τα δεδομένα είναι γνωστά και ως "Προδιαγραφές εκτέλεσης".

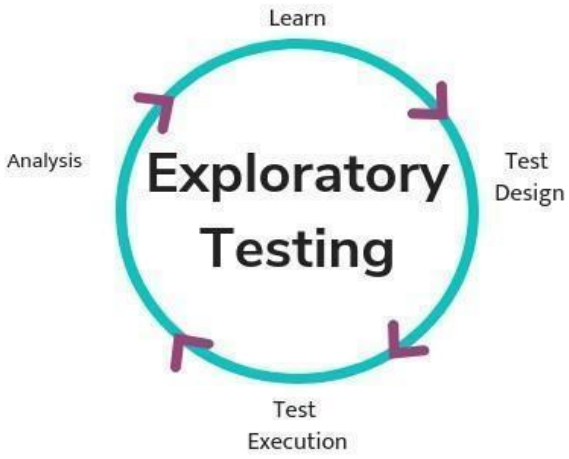


- Acceptance Test Driven Development (ATDD)

Η ATTD εστιάζει στην καλή επικοινωνία και ποιότητα ανάδρασης μεταξύ των πελατών, των προγραμματιστών και τον δοκιμαστών του λογισμικού. Ενθαρρύνεται ο έλεγχος του έργου από την οπτική γωνία του καθενός, δηλαδή ο πελάτης θα εστιάσει στο τι πρέπει να γίνει, ο προγραμματιστής στο πώς πρέπει να γίνει και ο δοκιμαστής στο τι μπορεί να πάει λάθος. Έτσι ελέγχεται το λογισμικό από όλες τις απαραίτητες οπτικές γωνίες καθώς και γίνεται πιο σαφής η ζητούμενη λειτουργικότητά του.

- Exploratory Testing

Με το exploratory testing η δημιουργία των ελεγκτικών τρόπων του λογισμικού και η εκτέλεσή τους γίνονται ταυτόχρονα. Αυτό επιτυγχάνεται με τον έλεγχο του ήδη υπάρχοντος λογισμικού και όχι με τη χρήση ξεχωριστών ελέγχων για κάθε φάση του λογισμικού. Καθώς το υλικό το οποίο τεστάρουν οι ελεγκτές είναι πολύ μεγαλύτερο από αυτά που εξετάζονται με άλλες μεθοδολογίες, η διαδικασία ελέγχου, συνήθως, είναι περισσότερο χαοτική με αποτέλεσμα να μην είναι εύκολο το documentation. Όμως, ελέγχεται καλύτερα η ολική εικόνα του λογισμικού και η αλληλεπίδραση των επιμέρους χαρακτηριστικών του.



- Session – Based Testing

Το session based testing ακολουθεί μία παρόμοια μεθοδολογία με αυτή του exploratory testing. Ως βασική διαφορά έχουν τον εξονυχιστικό έλεγχο του λογισμικού με πιο αυστηρές πρακτικές σε σχέση με αυτές του exploratory testing. Σε αυτό το είδους ελέγχου χρησιμοποιούνται καταστατικά ελέγχου, τα οποία λειτουργούν ως οδηγοί για το τι πρέπει να ελεγχθεί και με ποια σειρά. Επιπλέον, χρησιμοποιούνται αναφορές ελέγχου οι οποίες είναι ουσιαστικά έγγραφα, τα οποία αναγράφουν λεπτομερώς τι παρουσιάστηκε και αντιμετωπίστηκε κατά τη διαδικασία του testing. Οι έλεγχοι οργανώνονται και εκτελούνται ανεξάρτητα σε χρονικά ελεγμένες συνεδρίες. Κάθε μία από αυτές τις συνεδρίες ολοκληρώνεται με μία κατ' ιδίαν επικοινωνία μεταξύ των ελεγκτών και των υπεύθυνων ανάπτυξης του λογισμικού. Σε αυτές τις αλληλεπιδράσεις εξετάζονται και αναλύονται 5 βασικά ζητήματα :

1.Τι ελέγχθηκε και με τι πειραματίστηκαν οι ελεγκτές.
2.Τι αποτελέσματα είχαν αυτοί οι έλεγχοι.

Σε αυτό το σημείο θα αναφερθούν τυχών ανακαλύψεις που έγιναν κατά τη διαδικασία ελέγχου του λογισμικού, καθώς και τα αποτελέσματα των ελέγχων που έγιναν. Καταγράφονται και οι επιτυχείς έλεγχοι, αλλά και τα προβλήματα που εντοπίστηκαν.

3.Τα προβλήματα και τα εμπόδια που παρουσιάστηκαν στους ελεγκτές.

Να σημειωθεί πως δεν αναφέρονται σε αυτό το στάδιο τα προβλήματα του λογισμικού που αφορούν τη λειτουργικότητα του, αλλά τα προβλήματα που εμπόδισαν ή περιόρισαν τους ελεγκτές από το να κάνουν σωστά τη δουλειά τους.

4.Τα μέρη του λογισμικού τα οποία θα ελεγχθούν στο μέλλον.
5.Η ολική εντύπωση των ελεγκτών για τα μέρη του λογισμικού τα οποία έλεγξαν.

Επιπλέον στο session – based testing ενθαρρύνονται οι παρακάτω πρακτικές :

- Ο προσδιορισμός του στόχου που θα πρέπει να βάλουν οι ελεγκτές για το τρέχων sprint.

Η σωστή τοποθέτηση του στόχου, επιτρέπει στους ελεγκτές να ξεκαθαρίσουν που και πως πρέπει να εστιάσουν κατά τη διάρκεια του ελέγχου. Επιτρέπει επίσης την ευκολότερη οργάνωση των προτεραιοτήτων των ελεγκτών και για το τρέχων sprint αλλά και για τη συνέχεια του project.

- Δημιουργία καταστατικού που ορίζει τα μέρη του λογισμικού που πρέπει να ελεγχθούν.

Το καταστατικό αυτό μπορεί να περιέχει περισσότερες οδηγίες, όπως τη χρονική στιγμή που θα ελεγχθεί το κάθε μέρος του λογισμικού, τη χρονική διάρκεια του κάθε ελέγχου, καθώς και ποιοι θα είναι οι υπεύθυνοι ελεγκτές για το κάθε αυτό μέρος.

- Καταγραφή της διαδικασίας αλληλεπίδρασης ελεγκτών - υπευθύνων ανάπτυξης λογισμικού.

Η καταγραφή των εντυπώσεων και των συμπερασμάτων που λήφθηκαν από αυτές τις αλληλεπιδράσεις, μπορεί να είναι χρήσιμες μελλοντικά καθώς θα υπάρχει καλύτερη εικόνα και για την ίδια τη φύση του λογισμικού, αλλά και για τι εντυπώσεις και τα προβλήματα που εμφανίστηκαν στη διαδικασία του ελέγχου. Αυτές οι εγγραφές, σε περίπτωση ανάγκης για αλλαγή στο μέλλον ενός μέρος του λογισμικού που κατηγοριοποιήθηκε ως ελλαττωματικό, θα φανούν πολύ χρήσιμες λόγω της ενορατικότητας που θα προσφέρουν.

2η ενότητα: Μεταβατική παράγραφος

Από μόνη της η επίγνωση των ευέλικτων μεθόδων ανάπτυξης λογισμικού δεν είναι αρκετή για να προσφέρει σε μία ομάδα όλα τα πλεονεκτήματα της Agile. Μια ομάδα πέρα από το να ξέρει πώς να λειτουργήσει με τους παραπάνω τρόπους, θα πρέπει να είναι σε θέση να πειραματιστεί ώστε να δει τι δουλεύει καλύτερα για εκείνη. Όπως προαναφέρθηκε, δεν είναι όλοι οι μέθοδοι το ίδιο αποτελεσματικοί σε κάθε ομάδα, οι υπεύθυνοι θα πρέπει να δοκιμάσουν αρκετές τεχνικές προτού καταλήξουν στις καλύτερες, μάλιστα ενθαρρύνεται να μην χρησιμοποιείται η ίδια τεχνική για πολύ μεγάλα χρονικά διαστήματα καθώς είναι πιθανό να μειώνεται η αποτελεσματικότητά της σταδιακά.

Η φαντασία είναι επίσης σημαντική.
Απλές τεχνικές όπως η καθιέρωση εβδομαδιαίων συσκέψεων και η εναλλαγή των ρόλων των μελών της ομάδας μπορεί να μην είναι αρκετές. Έχει παρατηρηθεί πως ακόμα και πρακτικές που στην αρχή δεν φαίνονται πολύ καλές, έχουν τη δυνατότητα να αυξήσουν την παραγωγικότητα και την αποτελεσματικότητα μιας ομάδας. Πρακτικές όπως μείωση του ωραρίου εργασίας, αλλαγή των γραφείων με όρθιες θέσεις εργασίας ή ακόμα και αντικατάσταση των καρεκλών των εργαζομένων με μπάλες που χρησιμοποιούνται για τη yoga, μπορεί ακόμα και αν γίνεται για λίγες ώρες μέσα στην εβδομάδα να αυξήσει την παραγωγικότητα όλων των μελών της ομάδας. Στην κάθε ομάδα θα δουλέψει κάτι διαφορετικό γι' αυτό λοιπόν ενθαρρύνεται ο πειραματισμός στους τρόπους εργασίας.

Φυσικά, πέρα από ό,τι μόλις αναφέρθηκε, είναι πολύ σημαντική η χρήση των σωστών εργαλείων για όλες τις φάσεις ανάπτυξης λογισμικού. Τα σωστά εργαλεία είναι τα καλύτερα όπλα που έχει μια ομάδα για την εκπόνηση κάθε έργου. Παρακάτω, αναφέρονται τα πλέον διαδεδομένα στην αγορά εργαλεία, τα πλεονεκτήματά τους καθώς και αναλυτική σύγκριση του τι προσφέρουν σε σχέση με τα υπόλοιπα.

3η ενότητα: Πρακτικές μέθοδοι και εργαλεία

3.1. [1] WORKSOFT

Το Worksoft είναι ένα εργαλείο στον τομέα του αυτοματισμού συνεχών ελέγχων για επιχειρηματικές εφαρμογές (enterprise applications), είναι μια επιλογή για διεθνείς επιχειρήσεις που θέλουν να επιταχύνουν τις πρωτοβουλίες Agile-plus-DevOps και να επιτύχουν αυτοματισμό επιχειρηματικής διαδικασίας σε τεράστια κλίμακα.
Το Worksoft Certify παρέχει ένα πλαίσιο Agile testing που επιτρέπει σε μη τεχνικούς χρήστες, προγραμματιστές και επαγγελματίες αυτοματισμού ελέγχων να συνεργάζονται με Agile τρόπο και να ενσωματώνονται εύκολα σε εργαλεία, αλυσίδες και διαδικασίες DevOps.

Βασικά χαρακτηριστικά του Worksoft:

- Ενεργοποιεί την υιοθέτηση του Agile, κατασκευάζοντας αυτοματισμό πιο κοντά στο development sprint.
- Παρέχει γρήγορη αναγνώριση και τεκμηρίωση υπαρχουσών επιχειρηματικών διαδικασιών και παραλλαγών.
- Επιτυγχάνει τον έλεγχο διεργασιών end-to-end σε εταιρικές εφαρμογές.
- Ακολουθεί αποτελεσματικά τις σύνθετες εφαρμογές και τις συχνές ενημερώσεις εφαρμογών.
- Αποκαλύπτει και επιλύει προβλήματα μεταφοράς δεδομένων και ζητήματα συστήματος πριν από τη διακοπή της παραγωγής.

Τα Agile testing ανταποκρίνονται στην υπόσχεση για ταχεία ανάπτυξη χωρίς να θυσιάζουν ούτε τη λειτουργικότητα ούτε την ποιότητα. Το Worksoft Certify επιτρέπει λειτουργικές δοκιμές τόσο για να καθοδηγήσει όσο και να ελέγξει ολόκληρη τη διαδικασία ανάπτυξης στο πλαίσιο μιας ομαλής, δομημένης προσέγγισης που ικανοποιεί ευέλικτους αναπτυξιακούς στόχους συνοχής και γρήγορης ανάκαμψης.
Η ευκολία χρήσης του ξεπερνά τις συνηθισμένες αντιρρήσεις για υπερβολική επένδυση πολύ νωρίς και οι αυτοματοποιημένες δυνατότητες συντήρησης αντιμετωπίζουν τις πιθανότητες μεταγενέστερων αλλαγών να επηρεάσουν τα operating test.
Τέλος, η χρήση του αυτοματισμού της Worksoft σε περιβάλλον DevOps μειώνει το χρόνο ανάπτυξης και επιταχύνει την ανάπτυξη λογισμικού υψηλής ποιότητας. Δεν πρέπει όμως περιμένουμε έως ότου «σπάσει» κάτι στην παραγωγή για να το διορθώσουμε. Το Worksoft μας βοηθά να μειώσουμε δραματικά τον χρόνο δοκιμής έως και 90% - ακόμη και ενόψει συνεχούς ανάπτυξης, ενσωμάτωσης και συνολικής αλλαγής.

3.2. [10] PractiTest

Το PractiTest χρησιμοποιείται ως εργαλείο διαχείρισης ελέγχων του Agile Testing. Είναι εύκολο να το μάθει κάποιος, είναι προσιτό και ευέλικτο και βοηθά τόσο τους προγραμματιστές όσο και τους υπεύθυνους ελέγχων. Το PractiTest περιλαμβάνει απαιτήσεις, δοκιμές, αποτελέσματα, ζητήματα, αναφορές και παρέχει τη λεπτομερή κατάσταση του έργου. Διαχειρίζεται τις διαδικασίες ανάπτυξης και δοκιμών και έχει τον έλεγχο κάθε εργασίας.

Βασικά χαρακτηριστικά:

- Μπορεί να ενσωματωθεί με εργαλεία όπως JIRA, Jenkins, Selenium, TestComplete κ.λπ.
- Παρέχει ιεραρχική δομή δέντρου για τη διαχείριση και την εύρεση πληροφοριών.
- Ο ισχυρός και προσαρμοσμένος πίνακας εργαλείων του παρέχει σχετικές και ακριβείς πληροφορίες.
- Μπορούμε να εισάγουμε εύκολα τα υπάρχοντα δεδομένα μας.
- Πολύπλοκα ερωτήματα βάσης δεδομένων μπορούν να δημιουργηθούν εύκολα.
- Δημιουργεί χειροκίνητα τεστ και οργανώνει με βάση sprint.
- Το PractiTest έχει τη δυνατότητα να δημιουργεί σφάλμα απευθείας από το test run() .

Τα κύρια αντικείμενα που έχουμε την δυνατότητα να διαχειριστούμε ως μέρος της διαδικασίας Agile Testing είναι:
1)User Stories
2)Acceptance & Additional Tests
3)Issues (tasks, bugs, etc)
4)Graphs & Reports
Ο συντονισμός της εργασίας γύρω από αυτά τα αντικείμενα με τη χρήση του PractiTest είναι απλός και εύκολος. Μπορούν να δημιουργηθούν και βαθύτερα επίπεδα ανάλυσης (π.χ. Ανάλυση Κινδύνου, Λειτουργική Ανάλυση, Ανάλυση Αγοράς κ.λπ.) οργανώνοντας τις ιστορίες χρηστών σε παράλληλα φίλτρα με βάση τα πεδία που επιλέγουμε να προσθέσουμε στις προσαρμογές της ιστορίας χρήστη. Επίσης, έχουμε τη δυνατότητα να γράψουμε και να διαχειριστούμε τις δοκιμές με μια βιβλιοθήκη δοκιμών, μπορούμε ακόμη να φτιάξουμε Test Sets και, τέλος, να χρησιμοποιήσουμε φίλτρα για να οργανώσουμε τα προβλήματά μας βάσει Sprints, User Stories, Modules κ.λπ.

3.3. [6] 

Το JunoOne είναι ένα ακόμη διαθέσιμο εργαλείο για διαχείριση Agile testing και για την παρακολούθηση των προβλημάτων με ισχυρή ενοποίηση JIRA.
Μπορούμε να εργαστούμε αποτελεσματικά με το σύστημα διαχείρισης JunoOne, το οποίο είναι σχεδιασμένο για να βελτιώσει το Test Management και incident Management. Το JunoOne, που ουσιαστικά είναι ακόμα ένα Test Management Software, προσφέρει μια σειρά εργαλείων που θα κάνουν όλες τις δοκιμαστικές δραστηριότητες καλά οργανωμένες, θα μας βοηθήσουν να οργανώσουμε την εργασία μας, να ελέγξουμε τα μεμονωμένα βήματα και τη συνολική κατάσταση των έργων ενδελεχώς.
Το JunoOne προσφέρει πολλά χαρακτηριστικά σε όλη τη διαδικασία. Ξεκινά με τη δημιουργία της καμπάνιας και την εκτέλεση του test analysis. Μετά από αυτό το στάδιο, είμαστε σε θέση να δοκιμάσουμε και να εκτελέσουμε τις δοκιμές (run tests). Τέλος, σαν επιπλέον λειτουργία, επιλύει τα προβλήματα και προστατεύει τα δεδομένα μας.

Στο Agile Testing ένα test case γράφεται από μια ιστορία χρήστη (history user) και, στη συνέχεια, αρκετές ταχύτητες αργότερα, μια νέα ιστορία χρήστη αλλάζει τη ροή εργασίας ή την προβλεπόμενη λειτουργικότητα και γράφεται μια νέα test case. Εάν η προηγούμενη test case δεν ενημερωθεί ή διαγραφεί, μπορεί να προκαλέσει σύγχυση από το QA στο Dev στο Product.
Το JunoOne:

- Παρέχει αποτελεσματική εφαρμογή της διαδικασίας δοκιμής
- Έχει απλή ενσωμάτωση με το JIRA
- Παρέχει στους χρήστες αποτελεσματικές αναφορές
- Τα Widgets του είναι διαμορφώσιμα ανάλογα με της ανάγκες του χρήστη
- Γίνεται αυτόματη και λεπτομερής χαρτογράφηση απαιτήσεων
- Παρέχει μια επισκόπηση σχετικά με την κατάστασή τους
- Είναι ανοιχτό και αξιόπιστο.

3.4. [3,9,24] JIRA

Το JIRA είναι ένα Agile Testing καθώς και ένα Project Management Tool που αναπτύχθηκε από την Atlassian. Η Atlassian Corporation Plc είναι μια εταιρεία λογισμικού που αναπτύσσει προϊόντα για διαχειριστές έργων, προγραμματιστές λογισμικού κ.λπ.
Το Atlassian είναι γνωστό για το εργαλείο του - JIRA που χρησιμοποιείται για την παρακολούθηση ελαττωμάτων, το σχεδιασμό, τις αναφορές και τη διαχείριση όλων των ευέλικτων έργων ανάπτυξης λογισμικού.

Βασικά χαρακτηριστικά:

- Το JIRA υποστηρίζει μια ευέλικτη μεθοδολογία όπως Scrum, Kanban κ.λπ.
- Έχει μια ισχυρή δυνατότητα αναφοράς που παρέχει πρόσβαση σε δεκάδες αναφορές με απόδοση ομάδας σε πραγματικό χρόνο.
- Σχεδιάζει και προβλέπει χάρτη πορείας και είναι σε θέση να λαμβάνει τεκμηριωμένες αποφάσεις.
- Παρέχει προσαρμοσμένα φίλτρα χρησιμοποιώντας JIRA Query Language.
- Μπορεί να ενσωματωθεί με τα εργαλεία προγραμματιστή για end-to-end ανιχνευσιμότητα.

Το λογισμικό Jira έχει σχεδιαστεί για να μοντελοποιεί κάθε μέλος της ομάδας λογισμικού με 3 απλά βήματα : plan, track, and release great software.
Plan: Δημιουργήστε user stories and issues, plan sprints και διανείμετε εργασίες σε όλη την ομάδα του λογισμικού.
Track: Δώστε προτεραιότητα και συζητήστε την εργασία της ομάδας σε πλήρες πλαίσιο με πλήρη προβολή.
Release: Αποστολή λογισμικού γνωρίζοντας ότι οι πληροφορίες που έχουμε είναι πάντα ενημερωμένες.

- Πώς είναι το JIRA ευκίνητο και χρήσιμο για τους δοκιμαστές(testers);

Το πεδίο εφαρμογής του JIRA είναι είτε για εντοπισμό σφαλμάτων είτε για παρακολούθηση εργασιών - αυτά δεν αλλάζουν είτε είναι Agile Projects είτε κανονικά. Η καλή γνώση αυτής της μεθοδολογίας και το εργαλείο που παρέχει υποστήριξη είναι σημαντική, επειδή ο κόσμος της πληροφορικής μετατοπίζει την εστίασή του στο Agile - και θέλουμε να μάθουμε όλα όσα πρέπει να γνωρίζουμε για να αξιοποιήσουμε το καλύτερο.

3.5. [9] TestRail

Το TestRail είναι ένα σύγχρονο Agile Testing Tool που ανήκει στην Gurock Software Company. Η Gurock ιδρύθηκε το 2004 με έδρα το Βερολίνο της Γερμανίας. Το Gurock Software ειδικεύεται στην ανάπτυξη λογισμικού και στο QA. Τα προϊόντα που αναπτύχθηκαν στο Gurock είναι χρήσιμα για τους πελάτες παγκοσμίως για την κατασκευή του λογισμικού τους.
Επί του παρόντος, πολλοί μεγάλοι οργανισμοί όπως η Microsoft, η Dell, η Oracle, η Intel, η HP, η Adobe κλπ χρησιμοποιούν τα προϊόντα τους. Το TestRail είναι ένα διαδικτυακό εργαλείο που χρησιμοποιείται από την ομάδα δοκιμών για τη διαχείριση του Agile Testing Efforts. Το TestRail έχει σχεδιαστεί ειδικά για την υποστήριξη ενός ευέλικτου ορόσημου έργου χρησιμοποιώντας sprint, διάφορες αναφορές και μετρήσεις.

Βασικά χαρακτηριστικά:

- Το TestRail διαχειρίζεται τις δοκιμαστικές περιπτώσεις, τις δοκιμές, παρακολουθεί τα αποτελέσματα των δοκιμών-ελέγχων και διαχειρίζεται όλους τους ελέγχους.
- Το TestRail ακόμη ενσωματώνεται με εργαλεία αυτοματισμού όπως JIRA, Jenkins και Selenium.
- Χρησιμοποιώντας την έκδοση cloud, μπορούμε να τη χρησιμοποιήσουμε χωρίς εγκατάσταση ή ρύθμιση.
- Επιλέγει εύκολα test cases με βάση το φίλτρο για εκτέλεση και αποθηκεύει το αποτέλεσμα χρησιμοποιώντας σύγχρονη διεπαφή.
- Διάφοροι πίνακες ελέγχου, αναφορές και ειδοποιήσεις μέσω email ενημερώνουν τον χρήστη καθ' όλη τη διάρκεια του κύκλου δοκιμών.
- Μπορεί να ενσωματωθεί με Bugzilla, GitHub, Rally, TFS, FogBugz κ.λπ.

Το TestRail οργανώνει test cases, διαχειρίζεται τις δοκιμές (test runs), παρακολουθεί τα αποτελέσματα των δοκιμών και μετρά την πρόοδο με το δημοφιλές διαδικτυακό εργαλείο Agile testing. Το TestRail μας βοηθά να επιτύχουμε τους ποιοτικούς μας στόχους και να ολοκληρώσουμε τις δοκιμές μας εγκαίρως.
Τέλος, είναι το ολοκληρωμένο λογισμικό που βασίζεται στον ιστό και μπορεί να ενσωματωθεί στο JIRA. Το TestRail βοηθά τους δοκιμαστές, τους προγραμματιστές και τους οδηγούς ομάδων να διαχειρίζονται αποτελεσματικά, να παρακολουθούν και να συντονίζουν τις προσπάθειες δοκιμών λογισμικού, όλα από μια κεντρική και εύχρηστη εφαρμογή Ιστού. Η έξυπνη διεπαφή χρήστη που βασίζεται στον Ιστό καθιστά εύκολη τη δημιουργία δοκιμαστικών περιπτώσεων, τη διαχείριση δοκιμαστικών δοκιμών και τον συντονισμό ολόκληρης της διαδικασίας.

3.6. [21] 

Το SoapUI είναι ένα ευέλικτο εργαλείο δοκιμών και είναι το πιο προηγμένο REST και Service Oriented Architecture (SOA) που αναπτύχθηκε από τη SmartBear. Η SmartBear είναι μια Εταιρεία Πληροφορικής Λογισμικού που ξεκίνησε το 2009 με έδρα τη Βοστώνη των ΗΠΑ.
Το SoapUI χρησιμοποιείται βασικά για λειτουργικές δοκιμές υπηρεσιών διαδικτύου. Περιλαμβάνει ανάπτυξη υπηρεσιών διαδικτύου, επίκληση υπηρεσιών διαδικτύου κ.λπ. Χρησιμοποιώντας το SoapUI, τα test cases δημιουργούνται εύκολα για υπηρεσίες διαδικτύου. Έχει ισχυρό GUI που βοηθά στην εκτέλεση δοκιμών και στην ανάλυση των αναφορών στο ίδιο το GUI.
Βασικά χαρακτηριστικά:

- Το SoapUI είναι ένα δωρεάν και ανοιχτού κώδικα εργαλείο.
- Το SoapUI υποστηρίζει λειτουργικές δοκιμές υπηρεσίας διαδικτύου SOAP και λειτουργικές δοκιμές API REST.
- Επιτρέπει τη δημιουργία και εκτέλεση λειτουργικών δοκιμών functional, regression and load tests.
- Test cases μπορούν να δημιουργηθούν εύκολα χρησιμοποιώντας drag and drop διεπαφή (μεταφοράς και απόθεσης).
- Το SoapUI παρέχει πλήρη δοκιμαστική κάλυψη και υποστηρίζει όλα τα τυπικά πρωτόκολλα.
- Υποστηρίζει πολλαπλές δοκιμές περιβάλλοντος και δεδομένων.

Το SOAPUI δεν είναι απλώς ένα λειτουργικό εργαλείο δοκιμής Api, αλλά μας επιτρέπει, επίσης, να εκτελούμε μη λειτουργικές δοκιμές, όπως δοκιμές απόδοσης και ασφάλειας. Υποστηρίζει τον εντοπισμό σφαλμάτων δοκιμών και επιτρέπει στους δοκιμαστές να αναπτύξουν ελέγχους βάσει δεδομένων. Σήμερα, το SoapUI υποστηρίζει επίσης IDEA, Eclipse και NetBeans. Το SoapUI μπορεί να δοκιμάσει υπηρεσίες διαδικτύου SOAP και REST, JMS, AMF, καθώς και να κάνει οποιεσδήποτε κλήσεις HTTP (S) και JDBC.
Το API TESTING είναι ένας τύπος δοκιμής λογισμικού που επικυρώνει τις διεπαφές προγραμματισμού εφαρμογών (API). Ο σκοπός του API Testing είναι να ελέγξει τη λειτουργικότητα, την αξιοπιστία, την απόδοση και την ασφάλεια των διεπαφών προγραμματισμού.

3.7. [5] VERSIONONE Accelerate Software Delivery

Το VersionOne κυκλοφόρησε το 2002, αλλά από το 2017, αποκτήθηκε από την Collabnet Inc. Το VersionOne αναπτύσσει ευέλικτες λύσεις λογισμικού και υπηρεσίες. Έχει την έδρα της στην Alpharetta της Γεωργίας και άλλα γραφεία βρίσκονται στην Ατλάντα και το Άμστερνταμ. Το VersionOne είναι ένα ευέλικτο εργαλείο δοκιμών, καθώς και ένα εργαλείο διαχείρισης έργων Agile για την υποστήριξη ευέλικτων μεθοδολογιών όπως Scrum, Kanban XP, Lean, κ.λπ. Παρέχει μια κεντρική προβολή του backlog με drag and drop interface.
Βασικά χαρακτηριστικά:

- Το VersionOne παρέχει κεντρική διαχείριση έργων και βελτιώνει την προβολή του έργου.
- Η διαχείριση Stories and epics μπορεί να γίνει σε πολλά έργα και ομάδες έργων.
- Ορίζει εύκολα και παρακολουθεί την παράδοση σε όλα τα έργα και το έργο έχει την ορατότητα σε ολόκληρο τον κύκλο ζωής του.
- Επιτρέπει στον χρήστη να δώσει προτεραιότητα σε ιστορίες και ελαττώματα χρησιμοποιώντας τη διεπαφή μεταφοράς και απόθεσης.
- Το VersionOne μπορεί να ενσωματωθεί με JIRA, Jenkins, TFS, GitHub κ.λπ.

Είναι ένα μέσο που προσφέρει μια καλή πλατφόρμα σχεδιασμού και παρακολούθησης για να υποστηρίξει την Agile testing ως προσέγγιση bottom-up. Το VersionOne έχει ενσωματώσει τη διασφάλιση ποιότητας ως ουσιαστικό μέρος ολόκληρης της διαδικασίας ανάπτυξης λογισμικού.
Τα οφέλη του ποικίλουν καθώς το VersionOne διευκολύνει μια ευέλικτη πλατφόρμα από άκρη σε άκρη για να σχεδιάζουμε και να παρακολουθούμε όλες τις ιστορίες, τα ελαττώματα, τις εργασίες και τις δοκιμές μας. Μας δίνει εύκολη πρόσβαση και ορατότητα για να εργαστούμε με πολλές ομάδες και πολλά έργα ταυτόχρονα. Έχει ενοποιήσει την ανάπτυξη λογισμικού, την παράδοση και τη ρύθμιση της ροής εργασίας σε ένα μόνο πακέτο για τους χρήστες του.

3.8. [20] 

Το Selenium WebDriver είναι ένα ευέλικτο εργαλείο αυτοματισμού που χρησιμοποιείται ευρέως σε ολόκληρη τη βιομηχανία λογισμικού. Στη σημερινή τεχνολογία, η λέξη "Agile" και "Automation" χρησιμοποιούνται συχνά λόγω των τεράστιων πλεονεκτημάτων τους.
Στη βιομηχανία λογισμικού απαιτείται να εκτελείται οποιοδήποτε έργο με ευέλικτη μεθοδολογία και να εκτελούνται επαναλαμβανόμενες δοκιμές χρησιμοποιώντας αυτοματισμό - Selenium WebDriver για μια διαδικτυακή εφαρμογή (αυτοματοποιημένος έλεγχος εφαρμογών)
Βασικά χαρακτηριστικά:

- Το Selenium WebDriver χρησιμοποιείται μόνο για την αυτοματοποίηση εφαρμογών που βασίζονται σε πρόγραμμα περιήγησης και δεν υποστηρίζει εφαρμογές που βασίζονται σε επιτραπέζιους υπολογιστές.
- Το πλαίσιο βάσει λέξεων-κλειδιών μπορεί να δημιουργηθεί εύκολα χρησιμοποιώντας το Selenium WebDriver.
- Προσομοιώνει πώς ένας πραγματικός χρήστης μπορεί να αλληλεπιδράσει με την εφαρμογή χρησιμοποιώντας αυτοματοποιημένες περιπτώσεις δοκιμών.
- Υποστηρίζει γλώσσες προγραμματισμού JAVA, C #, Python, Ruby, PHP για τη σύνταξη σεναρίων αυτοματισμού.

Η υλοποίηση μια σουΐτας ελέγχων με χρήση του Selenium περιλαμβάνει τα εξής στοιχεία
– Driver: εκτελέσιμο το οποίο υλοποιεί το Web Driver πρωτόκολλο και αλληλεπιδρά άμεσα με τον browser (διανέμεται από τον κάθε κατασκευαστή browser)
– Βιβλιοθήκη Selenium: παρέχει προγραμματιστική διεπαφή (WebDriver) για αλληλεπίδραση με τον browser, (κάθε μέθοδος του API υλοποιείται ως μια σειρά κλήσεων προς τον Driver και υπάρχει διαφορετική υλοποίηση ανάλογα με τη γλώσσα προγραμματισμού)
– Testing framework: πλατφόρμα για την υλοποίηση και εκτέλεση ελέγχων, π.χ. Junit
– Κώδικας ελέγχου: ο κώδικας ελέγχου της εφαρμογής (χρησιμοποιεί τη βιβλιοθήκη Selenium και δυνατότητες που προσφέρονται από το Testing Framework)

Ακόμα υποστηρίζει συγγραφή ελέγχων σε διάφορες γλώσσες προγραμματισμού. Η αλληλεπίδραση με τον browser στα πλαίσια των ελέγχων βασίζεται σε δυο κυρίως interfaces
– WebDriver: αντιπροσωπεύει τον browser και παρέχει λειτουργίες για 1. έλεγχο του browser, 2. αναζήτηση και επιλογή HTML στοιχείων, 3. υποστήριξη debugging, 4. διαφορετική υλοποίηση για κάθε τύπο browser, π.χ. FirefoxDriver, ChromeDriver
– WebElement: αντιπροσωπεύει ένα HTML στοιχείο μιας σελίδας που προβάλλεται στον browser και παρέχει λειτουργίες για 1. λήψη πληροφοριών για το στοιχείο (π.χ. κείμενο), 2. έλεγχο της κατάστασής του, 3. εκτέλεση ενεργειών (π.χ. κλικ, επιλογή, εισαγωγή κειμένου)

3.9. [2] 

Το Apache JMeter είναι ένα ευέλικτο εργαλείο δοκιμής απόδοσης γραμμένο σε γλώσσα JAVA. Έχει σχεδιαστεί για να μετρά την απόδοση μιας εφαρμογής και τη λειτουργική της συμπεριφορά υπό ένα συγκεκριμένο φορτίο. Χρησιμοποιώντας το JMeter μπορούμε να προσομοιώσουμε ένα μεγάλο φορτίο στον διακομιστή ιστού και να αναλύσουμε την απόδοσή του.
Βασικά χαρακτηριστικά:

- Το JMeter είναι ένα εργαλείο ανοιχτού κώδικα.
- Η γραφική ανάλυση είναι δυνατή για τη μέτρηση της απόδοσης της εφαρμογής υπό διαφορετικούς τύπους φορτίου.
- Προσφέρει ικανότητα εκτέλεσης δοκιμών φόρτωσης και απόδοσης σε διαφορετικές εφαρμογές, διακομιστές κ.λπ.
- Παρέχει εκτεταμένη υποστήριξη plug-in για ανάλυση δεδομένων και οπτικοποίηση.
- Το JMeter μπορεί να χρησιμοποιηθεί για στατικούς και δυναμικούς πόρους όπως Servlets, Java Objects και διακομιστές FTP για τη μέτρηση της απόδοσής τους.

3.10. [9,16,23] 

Το σενάριο qTest είναι ένα πρόσθετο JIRA που χρησιμοποιείται για ευέλικτες δοκιμές. Το σενάριο qTest διαχειρίζεται τη δημιουργία δοκιμαστικών περιπτώσεων, εκτέλεσης, παρακολούθησης ελαττωμάτων και αποτελεσμάτων.
Βασικά χαρακτηριστικά:

- Είναι κατασκευασμένο ειδικά για BDD, TDD και ATDD προσέγγιση.
- Μπορεί εύκολα να εισαγάγει και να εξαγάγει τα υπάρχοντα αρχεία.
- Παρέχει πρόγραμμα επεξεργασίας Gherkin για σενάριο και ανάπτυξη χαρακτηριστικών.
- Test scenarios μπορούν να εκτελεστούν χρησιμοποιώντας χειροκίνητη μέθοδο μέσω διεπαφής JIRA ή χρησιμοποιώντας αυτοματοποίηση μέσω Maven ή Ruby.
- Αναλύει την κάλυψη και τις δυνατότητες των δοκιμών.
- Μπορεί να δημιουργήσει προσαρμοσμένες αναφορές χρησιμοποιώντας το JIRA και μπορεί να συγκρίνει την αναφορά ταχύτητας. Για αυτοματοποιημένα αποτελέσματα δοκιμών, μπορεί να ενσωματωθεί με Cucumber μέσω Maven.

3.13. [17] QMETRY

Το QMetry είναι ένα εργαλείο ευέλικτης δοκιμής ανοιχτού κώδικα και αναπτύχθηκε επίσης για DevOps. Το QMetry βοηθά την ευέλικτη ομάδα να δημιουργήσει, να διαχειριστεί και να αναπτύξει το λογισμικό πιο γρήγορα σε σύγκριση με άλλα εργαλεία.
Παρέχει μια ευέλικτη λύση με ισχυρή διαχείριση δοκιμών, προσαρμοσμένες μετρήσεις, αυτοματοποιημένο έλεγχο και συνεχή ολοκλήρωση. Το QMetry παρέχει μια προσθήκη για το JIRA και μετατρέπει τα αποτελέσματα αυτοματισμού σε ποιοτικές μετρήσεις και λεπτομερή συνοπτική αναφορά.
Βασικά χαρακτηριστικά:

- Το QMetry είναι ενσωματωμένο με ALM, Rally, Bamboo, VersionOne, Selenium κ.λπ.
- Υποστηρίζει ευέλικτες δοκιμές για τη δημιουργία και οργάνωση μεγάλου όγκου ιστοριών χρηστών και δοκιμαστικών περιπτώσεων.
- Επιτρέπει τον σχεδιασμό δοκιμών, την εκτέλεση, τη διαχείριση ελαττωμάτων, την ιχνηλασιμότητα και την πλήρη κάλυψη χρησιμοποιώντας ευέλικτη δοκιμή.
- Η διεπαφή μεταφοράς και απόθεσης μπορεί να συνδέσει τις ιστορίες χρηστών με τα σενάρια.
- Παρέχει προσαρμοσμένο πίνακα ελέγχου και αναλυτικές αναφορές ανάλυσης που παρέχουν πρόοδο στις δραστηριότητες δοκιμών.

3.11. [15] PivotalTracker

Το Pivotal Tracker είναι ένα εργαλείο διαχείρισης έργου που χρησιμοποιείται για τους σκοπούς του σχεδιασμού έργου. Βασίζεται στη μεθοδολογία ανάπτυξης Agile, ωστόσο, λειτουργεί καλά σε όλους τους τύπους μοντέλων έργων όπως το μοντέλο Waterfall, το επαναληπτικό μοντέλο, το μοντέλο V κ.λπ.
Το Tracker βοηθά επίσης στον προγραμματισμό επαναλήψεων κατά τη διάρκεια ευέλικτων δοκιμών (Agile testing). Βοηθά στη διάσπαση ενός μεγάλου έργου σε μικρές εργασίες ή κομμάτια και μπορεί να διαχειριστεί και να παραδοθεί εύκολα. Η ευέλικτη ομάδα παραδίδει το έργο σύμφωνα με τη βοήθεια ενός ιχνηλάτη.
Τέλος, βοηθά στη λήψη της συλλογικής προβολής ολόκληρης της ομάδας και αναλύει την πρόοδο της ομάδας στο τέλος της προόδου του έργου και δείχνει την πλήρη κατάσταση του έργου.

Βασικά χαρακτηριστικά:

- Η κοινή προβολή της ομάδας βοηθά στην προετοιμασία του έργου.
- Μπορούμε να οπτικοποιήσουμε εύκολα το εύρος και την εστιασμένη ομαδική εργασία.

- Η ομάδα του έργου λαμβάνει την προβολή προόδου σε πραγματικό χρόνο και δείχνει την κατάσταση της ομάδας.
- Μπορεί να προσφέρει σαφή και ενιαία αντίληψη για το ποιος είναι υπεύθυνος για την εργασία και ποια είναι η επόμενη εργασία ενός μέλους της ομάδας.
- Βοηθά να παρακολουθούμε όλες τις ιστορίες από την έναρξη του έργου έως την επιτυχή παράδοσή του.
- Το Pivotal Tracker παρέχει ενημερώσεις διαμόρφωσης.
- Η ισχυρή υποστήριξη αναζήτησης στον ιχνηλάτη αποθηκεύει επίσης τις κανονικές αναζητήσεις για χρήση αργότερα όταν απαιτείται.
- Παρέχει εύκολη δυνατότητα μεταφοράς και απόθεσης για επισύναψη αρχείων, παρουσιάσεων κ.λπ.

3.12. [30] ZEPHYR

Το Zephyr είναι ένα Εργαλείο Διαχείρισης Δοκιμών σε πραγματικό χρόνο που αναπτύχθηκε λαμβάνοντας υπόψη τη σημερινή τεχνολογική εποχή, τις πρόσφατα αναπτυγμένες πολύπλοκες επιχειρηματικές εφαρμογές και σχεδιάστηκε για να ικανοποιεί δυναμικές αλλαγές στις διαδικασίες δοκιμών. Διαχειρίζεται όλες τις διαδικασίες, εργασίες που σχετίζονται με τον κύκλο ζωής δοκιμών λογισμικού και παρέχει ένα προϊόν υψηλής ποιότητας.
Μαζί με την ευέλικτη μεθοδολογία, το Zephyr είναι χρήσιμο για διάφορες άλλες μεθοδολογίες όπως το Waterfall, το V-Model κ.λπ. Το Zephyr είναι ενσωματωμένο στο JIRA έτσι ώστε οι δραστηριότητες δοκιμών να διαχειρίζονται εύκολα. Με αυτό το πρόσθετο Zephyr στο JIRA, μπορούμε να παρακολουθούμε την κατάσταση του έργου και να παραδώσουμε ένα ποιοτικό προϊόν.
Βασικά χαρακτηριστικά:

- Μπορεί να ενσωματωθεί με JIRA, Selenium, Jenkins, QTP, Bamboo κ.λπ.
- Εισάγει ιστορίες χρηστών και διατηρεί την ιχνηλασιμότητα μέσα στο Zephyr για το JIRA.
- Οι δοκιμές μπορούν να δημιουργηθούν και να τροποποιηθούν και επίσης σχεδιάζονται κύκλοι εκτέλεσης δοκιμών.
- Οι δοκιμές οργανώνονται στη διαχείριση φακέλων δομής έτσι ώστε να μπορούν να επαναχρησιμοποιηθούν εύκολα.
- Το Zephyr δημιουργεί κύκλους δοκιμών, ευθυγραμμίζει το sprint και εκχωρεί πόρους για εκτέλεση.
- Εκτελεί δοκιμές, αποθηκεύει τα αποτελέσματα των δοκιμών στο Zephyr και παρακολουθεί εύκολα ποιοτικές μετρήσεις.
- Το Zephyr Enterprise παρέχει όλες τις πληροφορίες, όπως έργο, sprint και κυκλοφορία.

3.14. [30] planbox AGILE WORK INNOVATION

Ένα από τα πιο σημαντικά μέρη του agile cycle είναι τα γραφήματα(burndown charts). Το Planbox παρακολουθεί την πρόοδο των γραφημάτων εξουθένωσης, ώστε όλοι να γνωρίζουν πόσο μακριά-κοντά βρισκόμαστε από τους στόχους/την ολοκλήρωση του Sprint. Το Planbox ενσωματώνει επίσης τα σχόλια πελατών, τα bug reports, και τις διορθώσεις, καθιστώντας το χρήσιμο για ένα ευρύ φάσμα χρηστών. Το Planbox είναι ιδιαίτερα διαμορφώσιμο, εξασφαλίζοντας ότι οποιεσδήποτε ευκίνητες βασισμένες ομάδες(agile teams) θα είναι σε θέση να εργαστούν για τις συγκεκριμένες περιστάσεις τους με συγκεκριμένες προδιαγραφές και ευελιξία. Η βελτίωση βασίζεται στο καθημερινό Scrum.

What is Planbox Work?

Μια φιλοξενούμενη διαδικτυακή εφαρμογή που φέρνει τις διάφορες λειτουργίες μιας εταιρείας στον ευέλικτο κύκλο ζωής, το Planbox Work επιτρέπει στις ομάδες να σχεδιάζουν έργα, να αναθέτουν εργασίες, να διαχειρίζονται πόρους, να παρακολουθούν το χρόνο, να συνεργάζονται με τα ενδιαφερόμενα μέρη εντός και εκτός της εταιρείας και να μοιράζονται αρχεία και έγγραφα – όλα σε ένα εύχρηστο, διαισθητικό πλαίσιο. Με την τεχνολογία μεταφοράς και απόθεσης(drag-and-drop technology), οι αναθέσεις εργασιών και οι ιεράρχες γίνονται στην ίδια σελίδα.

Μέσω της δομής τεσσάρων επιπέδων του Planbox που αποτελείται από πρωτοβουλίες, έργα, στοιχεία και εργασίες, οι ομάδες μπορούν να δημιουργήσουν τις δικές τους προσαρμοσμένες ροές εργασίας. Η Access βασίζεται σε ρόλους και η εργασία planbox χρησιμοποιεί ασφάλεια SSL/HTTPS για να διατηρεί τα δεδομένα μας ασφαλή. Το εγγενές API προσφέρει πολλούς διαφορετικούς τρόπους επέκτασης της λειτουργικότητας του εργαλείου.

Κύρια Χαρακτηριστικά:

- Project and Portfolio Management
- Real-Time Collaboration
- Time Tracking: Η ενσωματωμένη λειτουργία παρακολούθησης χρόνου του Planbox Work διευκολύνει τους χρήστες να παρακολουθούν το χρόνο που ξοδεύουν σε κάθε εργασία. Μπορείτε να εισαγάγετε χρόνο χειροκίνητα ή χρησιμοποιώντας χρονοδιακόπτη.
- Reporting and Analytics: Οι αναφορές, πίνακες και γραφήματα του Planbox Work δείχνουν πώς παρακολουθούνται-εξελίσσονται οι διαδικασίες και πώς προχωρούν τα έργα ή οι πρωτοβουλίες.
- Integrations: Εκτός από το ανοιχτό API για την ενσωμάτωση εφαρμογών τρίτων μερών και τις υπάρχουσες ροές εργασίας σας, το Planbox Work ενσωματώνεται εύκολα με εφαρμογές όπως Zendesk, UserVoice, SnapEngage, GitHub, Microsoft Outlook, Google Calendar κ.α.

3.15. ActiveCollab

Το ActiveCollab είναι ένα ολοκληρωμένο εργαλείο διαχείρισης έργου που προάγει τη συνεργασία και συνδυάζει τη διαχείριση εργασιών, την παρακολούθηση του χρόνου, και φυσιολογική χρέωση. Είναι εύκολο στη χρήση και καλά σχεδιασμένο app. Βοηθά να οργανώσουμε τα έργα, τις εργασίες και τα αρχεία μας σε ένα σημείο ώστε να υπάρχει οργάνωση και αξιοπιστία.

Ένα ακόμη ισχυρό ευέλικτο εργαλείο διαχείρισης έργων, ιδανικό για μικρές επιχειρήσεις.
Χαρακτηριστικά εργαλείου:

- Διαχείριση εργασιών: Όλη η εργασία είναι σε ένα σημείο. Μπορούμε να παρακολουθούμε όλες τις ενημερώσεις κοιτάζοντας τον πίνακα εργαλείων.
- Φιλτράρισμα εργασιών: Αναζητήστε αμέσως αυτό που θέλετε και ανακτήστε πληροφορίες άμεσα(task filtering).
- Seamless workflow
- Προσαρμόζεται στη ροή εργασίας (adapts to workflow).
- Ενσωμάτωση ηλεκτρονικού ταχυδρομείου (email integration).
- Υπάρχει καλή συνεργασία της ομάδας με χαρακτηριστικά και εξωτερικές εφαρμογές όπως: το ημερολόγιο "all in one", @mentions και συνεργατική γραφή.
- Παρακολούθηση χρόνου: Ώρα σύνδεσης σε έργα, εφαρμογή χρονοδιαγραμμάτων, αναφορές παρακολούθησης χρόνου(time tracking).

Πλεονεκτήματα:

- Γρήγορο και εύκολο στη χρήση.
- Αξιόπιστο εργαλείο με μια εύχρηστη διεπαφή.
- Πλούσια χαρακτηριστικά με μέτρια τιμή.

Μειονεκτήματα:

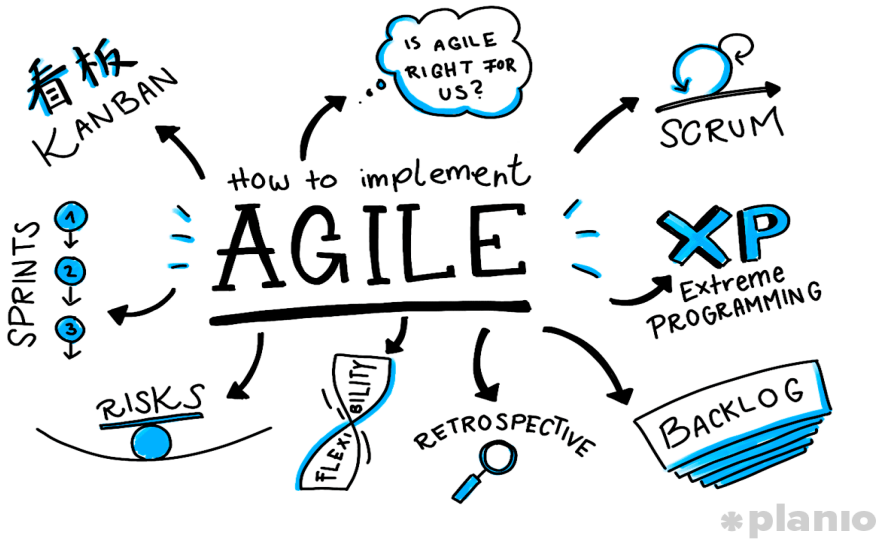
- Does not have a sprint planning option.

Agile Software Features:

- Labels, Project progress, Set recurring tasks, Workload reports

3.16. Χρήσιμες ορολογίες και επεξηγήσεις για την λειτουργικότητα των παραπάνω εργαλείων.

[22]



I. <u>Sprints:</u>
Το Sprint είναι μια επανάληψη ενός χρονικού πλαισίου ενός κύκλου συνεχούς ανάπτυξης (timeboxed iteration). Μέσα σε ένα Sprint, η προγραμματισμένη ποσότητα εργασίας πρέπει να ολοκληρωθεί από την ομάδα και να είναι έτοιμη για έλεγχο. Ο όρος χρησιμοποιείται κυρίως στη μεθοδολογία Scrum Agile, αλλά βασική ιδέα της συνεχούς παράδοσης του Kanban είναι επίσης η ουσία του Sprint Scrum.

II. <u>DevOps:</u>
Είναι μια πρόταση ανάπτυξης εταιρικού λογισμικού που χρησιμοποιείται για μια ‘Agile’ σχέση μεταξύ ανάπτυξης και λειτουργιών πληροφορικής. Ο στόχος του DevOps είναι να αλλάξει και να βελτιώσει τη σχέση υποστηρίζοντας καλύτερη επικοινωνία και συνεργασία μεταξύ αυτών των δύο επιχειρηματικών μονάδων.

III. <u>Scrum: (agile μεθοδολογία)</u>
Το Scrum είναι ένα πλαίσιο για τη διαχείριση έργων που δίνει έμφαση στην ομαδική εργασία, την υπευθυνότητα και την επαναληπτική πρόοδο προς έναν σαφώς καθορισμένο στόχο. Υπάρχουν τρεις κύριοι ρόλοι που έχει μια ομάδα Scrum:
Ο κάτοχος του έργου έχει την ευθύνη να διαχειριστεί την καθυστέρηση προϊόντων, συνεργάζεται με τελικούς χρήστες και πελάτες και παρέχει κατάλληλες απαιτήσεις στην ομάδα για την κατασκευή του κατάλληλου προϊόντος. Το Scrum Master συνεργάζεται με την ομάδα scrum για να διασφαλίσει ότι κάθε sprint ολοκληρώνεται εγκαίρως. Το Scrum master εξασφαλίζει τη σωστή ροή εργασίας για την ομάδα.

Scrum Team: Κάθε μέλος της ομάδας πρέπει να είναι αυτο-οργανωμένο, αφοσιωμένο και υπεύθυνο για την υψηλή ποιότητα της εργασίας.

IV. Kanban: (agile μεθοδολογία)
Το Kanban είναι μια ευέλικτη μεθοδολογία που δεν είναι απαραίτητα επαναλαμβανόμενη. Διαδικασίες όπως το Scrum έχουν σύντομες επαναλήψεις που μιμούνται έναν κύκλο ζωής έργου σε μικρή κλίμακα, έχοντας ξεχωριστή αρχή και τέλος για κάθε επανάληψη. Το Kanban επιτρέπει στο λογισμικό να αναπτυχθεί σε έναν μεγάλο κύκλο ανάπτυξης.

V. Διαφορά μεταξύ burn-up και burn-down γραφημάτων:
Τα γραφήματα Burn-up και Burn-down χρησιμοποιούνται για να παρακολουθούν την πρόοδο του έργου. Τα διαγράμματα Burn-up αντιπροσωπεύουν πόση δουλειά έχει ολοκληρωθεί σε οποιοδήποτε έργο, ενώ τα Burn-down αντιπροσωπεύουν την υπόλοιπη εργασία σε ένα έργο.

VI. Παραδοσιακό μοντέλο Waterfall vs τεστ Agile:
Η ευέλικτη δοκιμή γίνεται παράλληλα με την αναπτυξιακή δραστηριότητα, ενώ στην παραδοσιακή δοκιμή μοντέλου waterfall γίνεται στο τέλος της ανάπτυξης. Όπως γίνεται παράλληλα, η ευέλικτη δοκιμή γίνεται σε μικρά χαρακτηριστικά ενώ, σε ένα μοντέλο καταρράκτη, οι δοκιμές πραγματοποιούνται σε ολόκληρη την εφαρμογή.

VII. Ποια είναι η διαφορά μεταξύ Epic, User stories & Tasks;
User Stories: Ορίζουν την πραγματική επιχειρηματική απαίτηση. Δημιουργούνται συνήθως από τον ιδιοκτήτη της επιχείρησης.
Task: Για την επίτευξη των επιχειρηματικών απαιτήσεων, η ομάδα ανάπτυξης δημιουργεί εργασίες.
Epic: Μια ομάδα σχετικών ιστοριών χρηστών ονομάζεται Epic.

VIII. Τι είναι το Taskboard στην Agile;
Το Taskboard είναι ένας πίνακας ελέγχου που δείχνει την πρόοδο του έργου.
Περιέχει: User stories, tasks, εργασίες σε εξέλιξη, εκκρεμείς εργασίες για επαλήθευση ή δοκιμή και ολοκληρωμένες εργασίες.

IX. Τι είναι το Test Driven Development (TDD);
Πρόκειται για μια δοκιμαστική τεχνική ανάπτυξης στην οποία προστίθεται μια δοκιμή πριν γραφεί ο πλήρης κώδικας παραγωγής. Στη συνέχεια, εκτελείται η δοκιμή και, με βάση το αποτέλεσμα, αναπαράγεται ο κώδικας ώστε να πληροί τις απαιτήσεις δοκιμής.

X. Τι είναι η Scrum ban;
Πρόκειται για ένα μοντέλο ανάπτυξης λογισμικού που είναι ένας συνδυασμός Scrum και Kanban. Το Scrumban θεωρείται ότι συντηρεί έργα στα οποία υπάρχουν συχνές αλλαγές ή απρόσμενες ιστορίες χρηστών. Μπορεί να μειώσει τον ελάχιστο χρόνο ολοκλήρωσης για ιστορίες χρηστών.

3.17. Συμπεράσματα:
[16,28]

3.17.1. Σύγκριση βασικών χαρακτηριστικών των προαναφερθέντων Agile Εργαλείων.

	Γρήγορη ανατροφοδότηση	Διαδικτυακές διεπαφές	Χαμηλή επιβάρυνση	Οπτικοποιημένα μέσα	End of trial roles	Ενσωμάτωση με jira	Τερματισμός της φάσης δοκιμής
Worksoft	✓		✓		✓		✓
PractiTest	✓		✓		✓	✓	✓
Juno	✓		✓		✓	✓	✓
Jira	✓		✓		✓	✓	✓
TestRail	✓	✓	✓		✓	✓	✓
SoapUI	✓	✓	✓		✓		✓
VersionOne	✓		✓		✓	✓	✓
Selenium WebDriver	✓	✓	✓		✓		✓
Apache JMeter	✓		✓	✓	✓		✓
QTest	✓		✓	✓	✓		✓
Privotal Tracker	✓		✓		✓		✓
Zephyr	✓		✓		✓	✓	✓
Ometry	✓		✓		✓	✓	✓
ActiveCollab	✓		✓	✓	✓		
Planbox	✓	✓	✓	✓			

3.17.2. Συμπεράσματα σύγκρισης.

Η παραπάνω ανάλυση αφορούσε τα διάφορα εργαλεία Agile testing και τα βασικά χαρακτηριστικά τους. Η λίστα δεν είναι εξαντλητική, καθώς υπάρχουν πολλά άλλα εργαλεία που χρησιμοποιούνται με βάση τις απαιτήσεις και τον προϋπολογισμό του έργου.
Οι πέντε βασικές ομοιότητες που παρέχουν τα εργαλεία Agile testing είναι οι εξής:

1) Γρήγορη ανατροφοδότηση

Βέλτιστο σενάριο: Ελέγχεται το πλήρες εύρος λειτουργικότητας για κάθε αλλαγή κώδικα. Όταν εκτελούνται συνεχείς δοκιμές είναι σημαντικό όλη η ομάδα ανάπτυξης να λάβει άμεσα σχόλια σχετικά με αυτές που εκτελέστηκαν. Επομένως, ένας εύκολος στην πρόσβαση και ευανάγνωστος μηχανισμός αναφοράς είναι απαραίτητος. Εδώ εμπίπτουν οι αναφορές δοκιμών. Μπορούμε να ενσωματώσουμε πλήρως τα εκτελέσιμα αποτελέσματα των δοκιμών μας σε οποιοδήποτε σύστημα συνεχούς ενοποίησης και να μοιραστούμε τα αποτελέσματά μας με εύκολα κατανοητά αρχεία αναφορών με ολόκληρη την ομάδα για να εξασφαλίσουμε υψηλό επίπεδο διαφάνειας.
Δεν είναι αρκετά γρήγορο; Μπορούμε επίσης να λάβουμε ζωντανά σχόλια κατά την εκτέλεση της δοκιμής:
Απλώς ανοίγουμε την αναφορά δοκιμής ενώ η δοκιμή εξακολουθεί να εκτελείται για να δούμε αν έχουν αποτύχει οι μέχρι στιγμής δοκιμαστικές περιπτώσεις. Όσο νωρίτερα γνωρίζουμε, τόσο πιο γρήγορα μπορούμε να αντιδράσουμε.

2) Υψηλό επίπεδο αυτοματισμού

Γεγονός είναι ότι οι χειροκίνητες δοκιμές είναι αργές, απαιτούν εργασία, είναι ασυνεπείς και έχουν λάθη. Εάν θέλουμε να ανταποκριθούμε γρήγορα σε μεταβαλλόμενες απαιτήσεις και συνεχείς αλλαγές κώδικα, είναι απαραίτητο ένα υψηλό επίπεδο αυτοματισμού. Δίπλα στις βασικές δοκιμές μονάδας, οι δοκιμές αποδοχής και οι δοκιμές ενσωμάτωσης είναι πολύ σημαντικές για τη δοκιμή όλου του εύρους λειτουργικότητας. Για να μειωθεί ο χρόνος και να αυξηθεί η ποιότητα, ο αυτοματοποιημένος έλεγχος πρέπει, από την αρχή, να αποτελεί αναπόσπαστο μέρος του έργου. Μπορούμε να χρησιμοποιήσουμε το Ranorex Studio για να αυτοματοποιήσουμε ολόκληρο το φάσμα των δοκιμών UI.

3) Χαμηλή επιβάρυνση

Μερικές φορές, είναι λογικό να δημιουργηθεί ένα απλό τεστ, το οποίο χρησιμοποιείται μόνο σε ένα συγκεκριμένο sprint. Σε αυτές τις περιπτώσεις, δεν υπάρχει ούτε χρόνος ούτε πόροι για μια μεγάλη εγκατάσταση. Χρησιμοποιώντας Agile testing, μπορούμε να αυτοματοποιήσουμε εύκολα αυτές τις δοκιμές και να τις ενσωματώσουμε στο υπάρχον περιβάλλον δοκιμών μας. Τα ελαφριά έργα αυτοματοποίησης δοκιμών έχουν ως αποτέλεσμα εκτελέσιμα αρχεία, τα οποία μπορούμε να τα ενεργοποιήσουμε απευθείας από τη γραμμή εντολών. Τα εκτελέσιμα αρχεία μας ενημερώνουν εάν ο έλεγχος απέτυχε ή πέτυχε. Επιπλέον, ένα αρχείο αναφοράς μπορεί να παρέχει λεπτομερείς πληροφορίες σχετικά με τη δοκιμαστική περίοδο. Αυτές είναι οι τέλειες συνθήκες για ενσωμάτωση σε οποιαδήποτε διαδικασία συνεχούς ολοκλήρωσης.

4) Τερματισμός δοκιμαστικών ρόλων (end of trial roles)

Σε μια ευέλικτη διαδικασία ανάπτυξης λογισμικού, ολόκληρη η ομάδα είναι υπεύθυνη για την ποιότητα. Τα όρια μεταξύ της παραδοσιακής κατανόησης των υπευθύνων δοκιμών και των προγραμματιστών θολώνουν. Έτσι, οι τυπικοί δοκιμαστές θα πρέπει να είναι σε θέση να γράφουν τεστ μονάδας ή απλές δοκιμές ολοκλήρωσης. Αντίθετα, οι προγραμματιστές θα πρέπει να καταγράφουν δοκιμές UI. Χρησιμοποιώντας το ευρύ σύνολο εργαλείων ιδανικό για όλα τα σετ δεξιοτήτων, μπορούμε να ορίσουμε ιδανικές συνθήκες για τους προγραμματιστές και τους δοκιμαστές για να συνεργάζονται σε έργα.

5) Τερματισμός της φάσης δοκιμής

Τέλος, σε ευέλικτους κύκλους επανάληψης, συνήθως δεν υπάρχει χρόνος για διαδοχική επεξεργασία όλων των επιπέδων δοκιμών (δοκιμές μονάδας, δοκιμές ολοκλήρωσης, δοκιμές συστήματος, δοκιμές αποδοχής…). Επομένως, ο αυτοματοποιημένος έλεγχος πρέπει να αποτελεί αναπόσπαστο μέρος του έργου από την αρχή. Συνεπώς, μπορούμε να δημιουργήσουμε δοκιμαστικά σενάρια ανά πάσα στιγμή μέσω των Agile testing εργαλείων. Μόλις μάθουμε πώς θα μοιάζει το περιβάλλον εργασίας χρήστη, μπορούμε να εφαρμόσουμε μια αυτοματοποιημένη δοκιμή και να βελτιώσουμε τις λεπτομερείς διαδρομές των στοιχείων αποθετηρίου καθώς προχωράμε.

Τα γενικά χαρακτηριστικά των Agile testing εργαλείων αναφέρονται παραπάνω καθώς και τα βασικά πλεονεκτήματά τους. Οι Agile testing μέθοδοι βασίζονται σε υψηλό επίπεδο αυτοματισμού δοκιμών. Για να ορίσουμε, όμως, τις ιδανικές προϋποθέσεις για την ομάδα μας, πρέπει να φροντίσουμε να αποκτήσουμε ένα εργαλείο ή ένα σύνολο εργαλείων, που επιτρέπουν να εξασφαλίσουμε υψηλό επίπεδο διαφάνειας και συνεργασία εντός της ομάδας.
Προφανώς, η λίστα των εργαλείων Agile testing είναι πολύ μεγαλύτερη. Τα εργαλεία που αναφέρονται σε αυτήν την εργασία είναι δείγματα εργαλείων και δυνατοτήτων που παρέχονται από Agile δοκιμές. Το κλειδί για την επιτυχία στο Agile development είναι να επιτραπεί η ευελιξία παράλληλα με τη διατήρηση της οργάνωσης. Ο καλύτερος τρόπος για να γίνει αυτό είναι να αναπτύξουμε μια σειρά από καλά εργαλεία που βοηθούν στην παρακολούθηση του έργου και στην οργάνωση της προόδου της ομάδας. Δεν θα επιβάλλουν αυστηρά χρονοδιαγράμματα και ρόλους, αλλά απλώς θα διευκολύνουν τους προγραμματιστές να αυτοδιαχειρίζονται και να συγκλίνουν στους στόχους τους.
Τα εργαλεία υποστηρίζουν το έργο βοηθώντας την ομάδα να προσδιορίσει τις απαιτήσεις και να τις χωρίσει σε μια σειρά από μικρότερες εργασίες. Στη συνέχεια, παρακολουθεί τους προγραμματιστές καθώς συνεργάζονται στα μέρη. Η διαδικασία χωρίζεται συχνά σε μικρούς κύκλους που συγκλίνουν σταδιακά στο τελικό αποτέλεσμα. Οι κύκλοι εναλλάσσονται μεταξύ προγραμματιστικών συνεδριών και κωδικών sprint. Διατηρώντας τον κύκλο σύντομο και συμπεριλαμβάνοντας πολλά σχόλια προγραμματιστών κατά την διάρκεια ανάπτυξης του κώδικα, η ομάδα επιτρέπει την προσαρμογή και την εστίαση.

Άλλα Agile Testing εργαλεία:

- Agile Manager
- Active Collab
- Agile Bench
- Telerik TeamPulse
- Planbox
- LeanKit Axosoft
- Hive
- Agilo for Scrum
- κ.ά.

4η ενότητα: Συμπεράσματα εργασίας.

Ύστερα από τη μελέτη της Agile και των εργαλείων της μπορούμε να συμπεράνουμε πως χρειαζόμαστε να κάνουμε διεξοδική μελέτη προτού εφαρμόσουμε τις αρχές και τις μεθοδολογίες της στην δουλειά μας. Οφείλουμε να μάθουμε σε τι πρέπει να εστιάσουμε και πώς να αντιμετωπίσουμε τυχόν προβλήματα. Πρέπει να είμαστε έτοιμοι να πειραματιστούμε με τις ήδη υπάρχουσες τεχνικές ή και να δημιουργήσουμε αυτές που εξυπηρετούν τις δικές μας ανάγκες. Τέλος, πρέπει να εντάξουμε και τα κατάλληλα Agile εργαλεία στην τρόπο εργασίας μας ώστε να αυξηθεί ακόμα περισσότερο η αποδοτικότητά μας. Είναι σημαντικός ο έλεγχος των εργαλείων που πιθανώς θα χρησιμοποιήσουμε, προκειμένου να επιλέξουμε εκείνα που ανταποκρίνονται καλύτερα στις ανάγκες μας, καθώς, όπως αναλύσαμε προηγουμένως, διαφέρουν πολύ το ένα από το άλλο. Σε όλα χρειάζονται πάνω απ’ όλα διάθεση και προθυμία για αλλαγή προς το καλύτερο.

5η ενότητα: Βιβλιογραφία:

1. Agile Alliance, *Advancing the practice of Agile*. Διαθέσιμο από: https://www.Agilealliance.org/ [πρόσβαση 5 Μαΐου 2020].
2. Apache Software Foundation, *Apache JMeter - Apache JMeter™*. Διαθέσιμο από: https://jmeter.apache.org/index.html [πρόσβαση 5 Μαΐου 2020].
3. Atlassian, Agile tools for software teams, 2020. Διαθέσιμο από: https://www.atlassian.com/software/jira/Agile [πρόσβαση 5 Μαΐου 2020].
4. Cohen, David, Lindvall , Mikael and Patricia Costa, An introduction to Agile methods. Στο: *Advances in Computers*, 2004, v. 62, 1-66p. Διαθέσιμο από: https://books.google.gr/books?hl=el&lr=&id=N-06uoJ9iSsC&oi=fnd&pg=PA1&dq=Agile&ots=l6gY8T-nSm&sig=efDg1s4lxNpKEBOMXkIweKGAIqo&redir_esc=y#v=onepage&q&f=false. [πρόσβαση 5 Μαΐου 2020].
5. CollabNet VersionOne, *DevOps and Agile Solutions for the Enterprise*. Διαθέσιμο από: https://www.collab.net/ [πρόσβαση 5 Μαΐου 2020].
6. Deveny, JunoOne. *Great tool for Agile Project Management, Test Management and Issue Tracking with powerful Integrations.* Διαθέσιμο από: https://juno.one/ [πρόσβαση 5 Μαΐου 2020].
7. Dingsoyr, Torgeir κ.ά. A decade of Agile methodologies: Towards explaining Agile software development. *Journal of Systems and Software*, 2012, vol. 85 (6), 1213-1221p. Διαθέσιμο από: https://www.sciencedirect.com/science/article/pii/S0164121212000532 [πρόσβαση 5 Μαΐου 2020].
8. Gilley, Cliff, *Pros and Cons of Agile Product Development*, 2019. Διαθέσιμο από: https://community.uservoice.com/blog/the-pros-and-cons-of-Agile-product-development/ [πρόσβαση 5 Μαΐου 2020].
9. Gurock, *Agile Testing Software Tool*. Διαθέσιμο από: https://www.gurock.com/testrail/Agile-testing [πρόσβαση 5 Μαΐου 2020].
10. Guru99, *What is Agile Testing? Process, Strategy, Test Plan, Life Cycle Example*. Διαθέσιμο από: https://www.guru99.com/Agile-testing-a-beginner-s-guide.html [πρόσβαση 5 Μαΐου 2020].
11. Hales, Laury, *Pros & Cons of Agile Development Methods*, 2019. Διαθέσιμο από: https://study.com/academy/lesson/pros-cons-of-Agile-development-methods.html [πρόσβαση 5 Μαΐου 2020].
12. Krüger, Nico, *What Is Agile Testing? Agile Test Strategy*. Διαθέσιμο από: https://www.perforce.com/blog/alm/what-Agile-testing-5-examples [πρόσβαση 5 Μαΐου 2020].
13. Kukhnavets, Pavel, *Why Agile is So Popular in Project Management World, 2018*. Διαθέσιμο από: https://hygger.io/blog/why-Agile-is-so-popular-in-project-management/ [πρόσβαση 5 Μαΐου 2020].
14. *Manifesto for Agile Software Development*. Διαθέσιμο από: https://Agilemanifesto.org/ [πρόσβαση 5 Μαΐου 2020].
15. Pivotal Tracker, Agile Development, Στο: *Pivotal Tracker is changing how teams build software—one story at a time*. Διαθέσιμο από: https://www.pivotaltracker.com/Agile [πρόσβαση 5 Μαΐου 2020].

16. PractiTest, *Test management tool*. Διαθέσιμο από: https://www.practitest.com/?utm_medium=listings&utm_source=sth&utm_campaign=Agile+testing+tools [πρόσβαση 5 Μαΐου 2020].
17. QMetry, *Best Test Management and Automated Testing Tools*. Διαθέσιμο από: https://www.qmetry.com/ [πρόσβαση 5 Μαΐου 2020].
18. ReQtest, *Agile Testing, Agile Testing Methods, Principles and Advantages,* 2020. Διαθέσιμο από: https://reqtest.com/testing-blog/Agile-testing-principles-methods-advantages/ [πρόσβαση 5 Μαΐου 2020].
19. Sealights, *Understanding Agile Testing Methodology and 4 Agile Testing Methods*. Διαθέσιμο από: https://www.sealights.io/Agile-testing/understanding-Agile-testing-methodology-and-4-Agile-testing-methods/ [πρόσβαση 5 Μαΐου 2020].
20. Selenium, *Selenium Projects*. Διαθέσιμο από: https://www.selenium.dev/projects/ [πρόσβαση 5 Μαΐου 2020].
21. SmartBear Software, *SoapUI: The World's Most Popular API Testing Tool*. Διαθέσιμο από: https://www.soapui.org/ [πρόσβαση 5 Μαΐου 2020].
22. Software Testing Help, *25 Best Agile Testing Interview Questions And Answers*, 2020. Διαθέσιμο από: https://www.softwaretestinghelp.com/25-best-Agile-testing-interview-questions-and-answers/ [πρόσβαση 5 Μαΐου 2020].
23. Software Testing Help, *Top 10 Agile Testing Tools In 2020*, April 16, 2020. Διαθέσιμο από: https://www.softwaretestinghelp.com/Agile-testing-tools/ [πρόσβαση 5 Μαΐου 2020].
24. Tricentis, End-to-end JIRA plugin for TDD, BDD and ATDD Testing. Στο: *qTest Scenario: The Enterprise BDD Solution for Jira Software*. Διαθέσιμο από: https://www.qasymphony.com/software-testing-tools/qtest-scenario/bdd-tdd-jira-plugin/ [πρόσβαση 5 Μαΐου 2020].
25. Verma, Anushree, *5 Good Reasons Why Agile Should Be Applied*, 2016. Διαθέσιμο από: https://dzone.com/articles/you-will-thank-us-5-good-reasons-why-Agile-should [πρόσβαση 5 Μαΐου 2020].
26. Wikipedia, *Agile software development, 2020.* Διαθέσιμο από: https://en.wikipedia.org/wiki/Agile_software_development [πρόσβαση 5 Μαΐου 2020].
27. Wikipedia, *Test automation management tools,* 2018. Διαθέσιμο από: https://en.wikipedia.org/wiki/Test_automation_management_tools [πρόσβαση 5 Μαΐου 2020].
28. Worksoft, *Agile & DevOps Test Automation*. Διαθέσιμο από: https://www.worksoft.com/solutions#devops [πρόσβαση 5 Μαΐου 2020].
29. Wrike, *Why Use Agile Project Management?* Στο: *Wrike Project Management Guide* . Διαθέσιμο από: https://www.wrike.com/project-management-guide/faq/why-use-Agile-project-management/ [πρόσβαση 5 Μαΐου 2020].
30. Zephyr, Test Management Tools | Jira Test Case Management Software. Στο: *Software Testing As Agile As You Need It to Be*. Διαθέσιμο από: https://www.getzephyr.com/ [πρόσβαση 5 Μαΐου 2020].