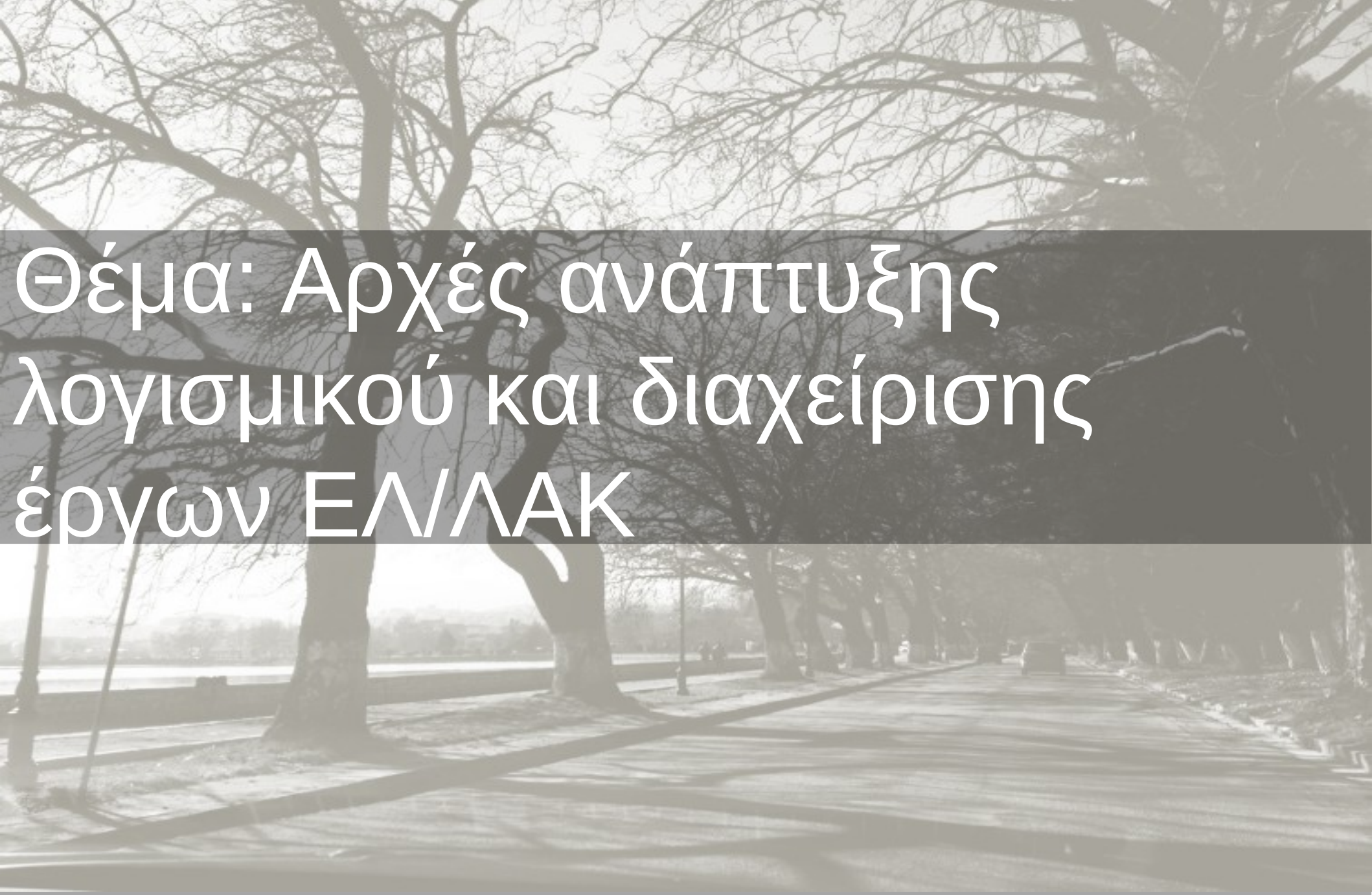


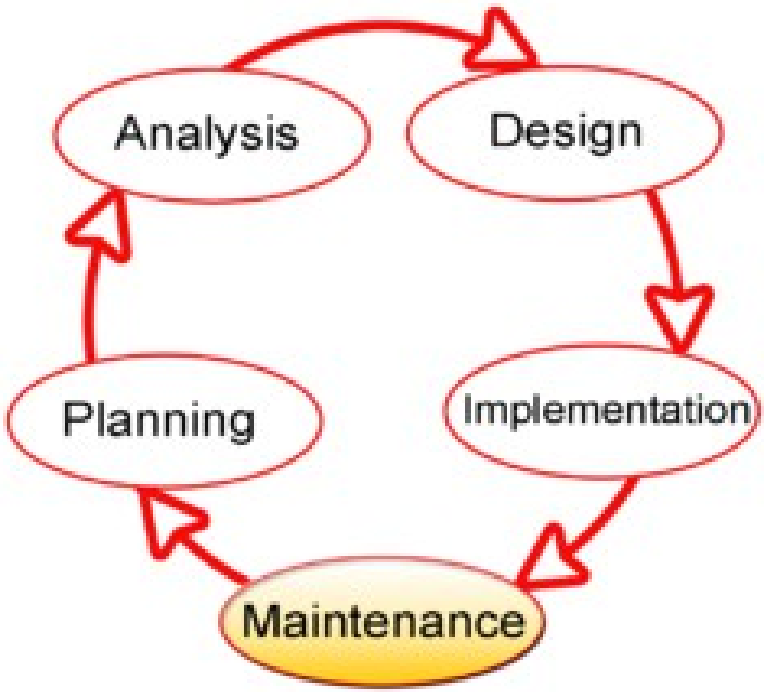
Θέμα: Αρχές ανάπτυξης λογισμικού και διαχείρισης έργων ΕΛ/ΛΑΚ

Γιάννης Παππάς
Μονάδα Αριστείας ΕΛ/ΛΑΚ | 15/10/2014

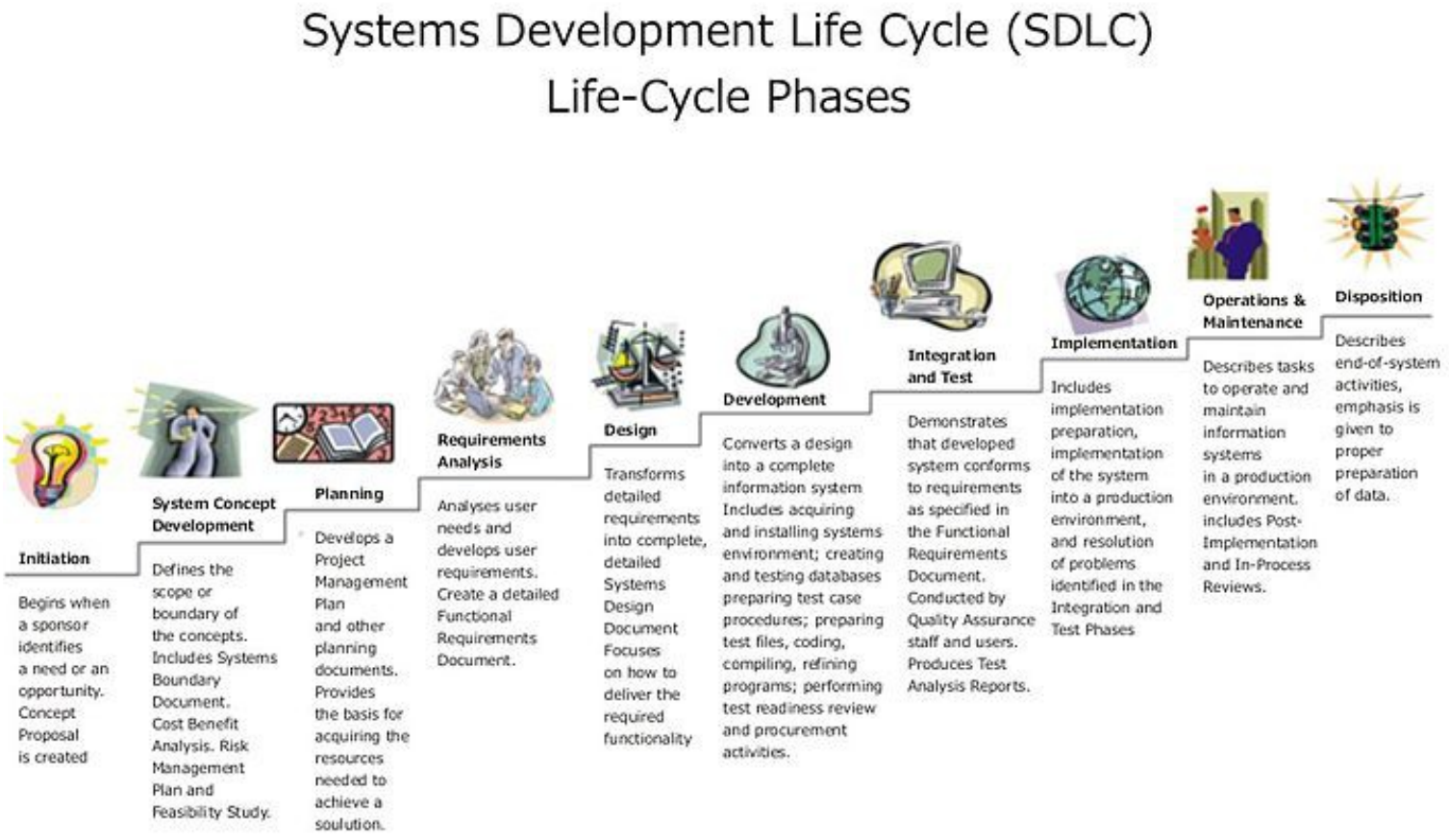
Σχεδιάγραμμα της παρουσίασης

- Systems development life cycle
- Πλαίσιο εργασίας για την ανάπτυξη λογισμικού
- The Agile System Development Life Cycle (SDLC)
  - Η μεθοδολογία Scrum
  - Η μεθοδολογία Extreme Programming
- Ποιότητα λογισμικού
- Θέματα ασφαλείας λογισμικού

Systems development life cycle



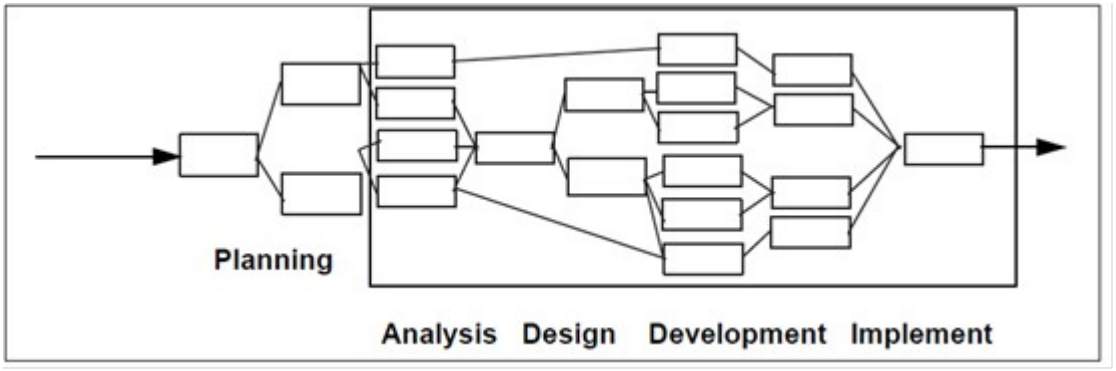
Systems development life cycle



Πλαίσιο εργασίας για την ανάπτυξη λογισμικού

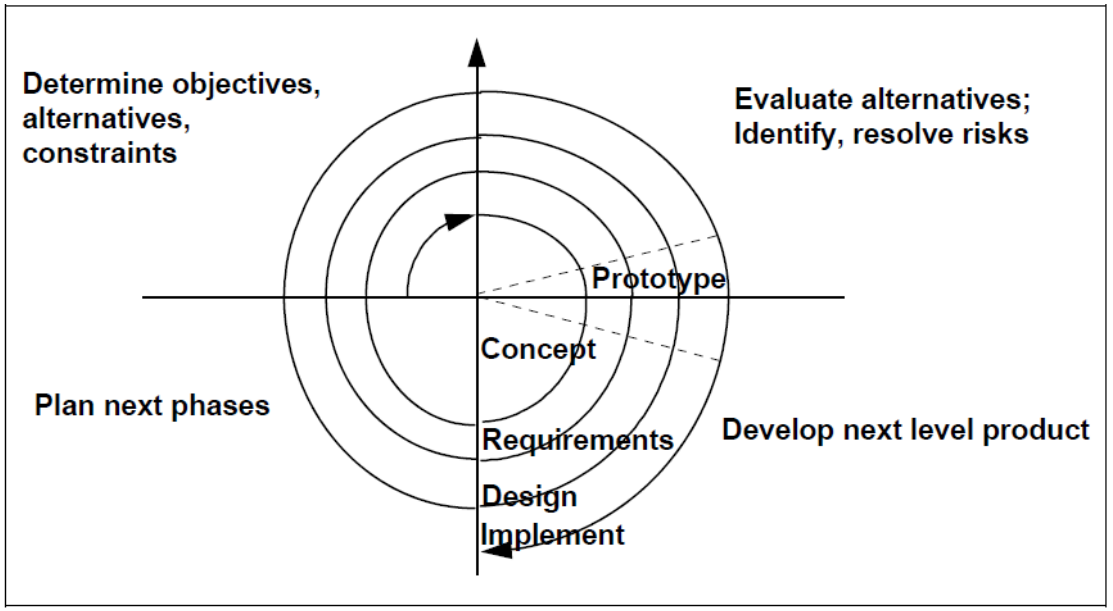
- Ενώ υπάρχουν διάφορες διαφορετικές μέθοδοι ανάπτυξης, όλες μοιράζονται τα παρακάτω κοινά στοιχεία:
- **Σύλληψη**
- **Απαιτήσεις**
- **Σχεδιασμός και τεκμηρίωση**
- **Προγραμματισμός**
- **Τεστ, Ολοκλήρωση και Εσωτερική Εκτίμηση**
- **Έκδοση**
- **Διατήρηση, Επιμήκυνση Μηχανικής και Ανταπόκριση σε Συμβάντα**

Πλαίσιο εργασίας για την ανάπτυξη λογισμικού



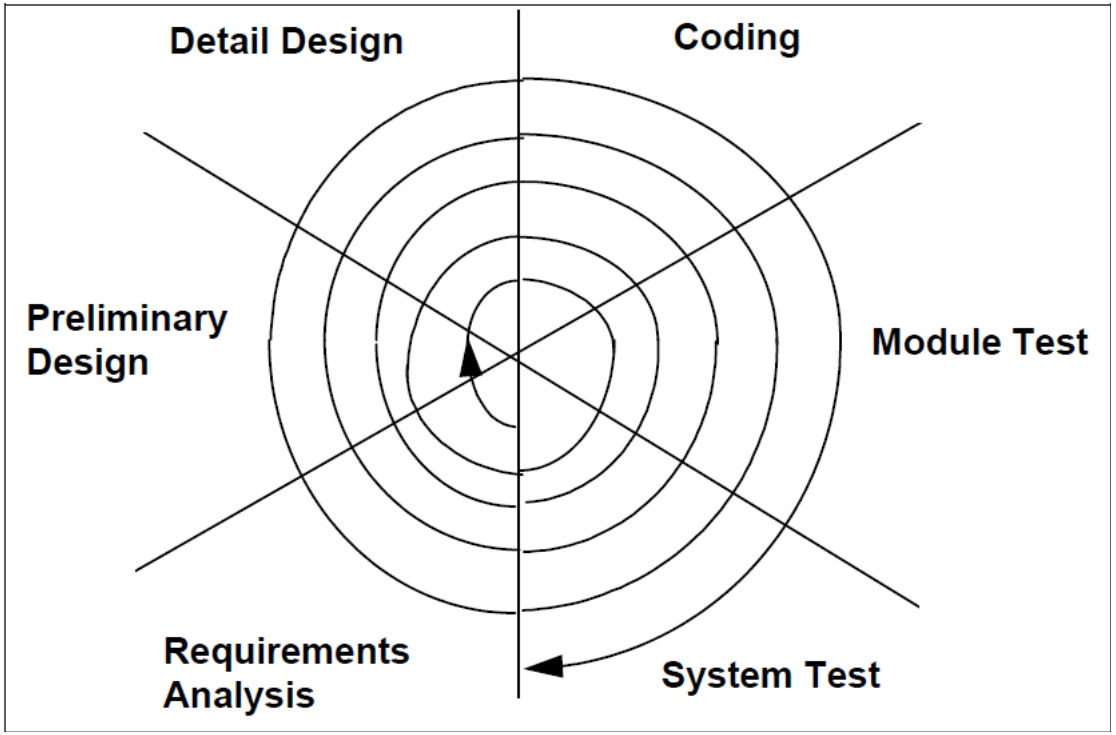
Η μεθοδολογία του Καταρράκτη

Πλαίσιο εργασίας για την ανάπτυξη λογισμικού



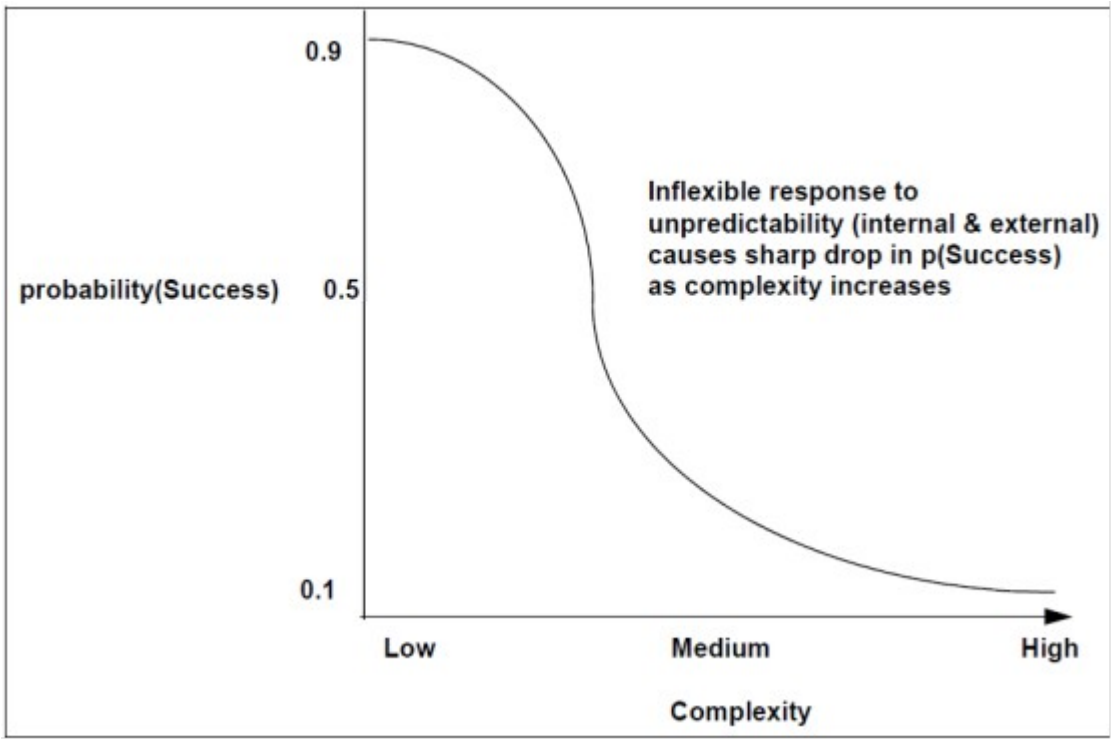
Η μεθοδολογία του Σπιράλ

Πλαίσιο εργασίας για την ανάπτυξη λογισμικού



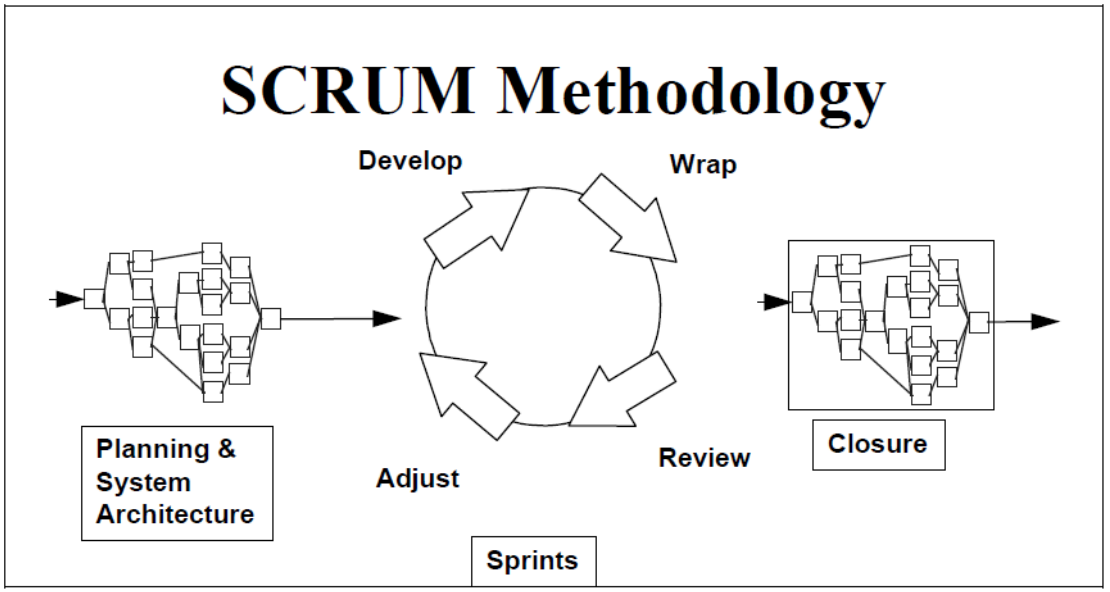
Η Επαναληπτική μεθοδολογία

Πλαίσιο εργασίας για την ανάπτυξη λογισμικού



Γράφος Ορισμένης Διαδικασίας Ρίσκου/Πολυπλοκότητας

Η μεθοδολογία SCRUM



Η μεθοδολογία Scrum

Η μεθοδολογία SCRUM

- Τα χαρακτηριστικά της μεθοδολογίας Scrum είναι:
  - Οι πρώτες και οι τελευταίες φάσεις (Σχεδιασμού και Κλεισίματος) αποτελούνται από καθορισμένες διαδικασίες, όπου όλες οι διαδικασίες, είσοδοι και έξοδοι είναι καλά ορισμένες. Η γνώση του πώς θα εκτελέσεις τις διαδικασίες αυτές, είναι ρητά ορισμένη. Η ροή είναι γραμμική, με κάποιες επαναλήψεις στη φάση του σχεδιασμού.
  - Η φάση Sprint είναι μια εμπειρική διαδικασία. Πολλές από τις διαδικασίες στη φάση Sprint είναι μη αναγνωρίσιμες η μη ελέγξιμες. Η φάση αυτή θεωρείται ως μαύρο κουτί(black box) που απαιτεί εξωτερικούς ελέγχους. Συγκεκριμένα, έλεγχοι, που περιλαμβάνουν διαχείριση ρίσκου, τοποθετούνται σε κάθε επανάληψη(iteration) της Sprint φάσης, για την αποφυγή χάους ενώ αυξάνεται η ευκαμψία.

Η μεθοδολογία SCRUM

- Τα Sprint είναι μη γραμμικά και ευέλικτα. Όπου είναι ικανό, χρησιμοποιείται ρητή διαδικασία γνώσης. Διαφορετικά, υπονοημένη γνώση καθώς και δοκιμές όπως και λάθη, χρησιμοποιούνται για να χτίσουν τη διαδικασία γνώσης. Τα Sprint χρησιμοποιούνται για να εξελίξουν το τελικό προϊόν.
- Το έργο είναι ανοιχτό στο περιβάλλον μέχρι τη φάση Κλεισίματος(Closure phase). Το παραδοτέο μπορεί να αλλάξει οποιαδήποτε στιγμή κατά τη διάρκεια των φάσεων σχεδιασμού(Planning) και Sprint. Το έργο παραμένει ανοιχτό στην πολυπλοκότητα του περιβάλλοντος, συμπεριλαμβάνοντας πιέσεις ανταγωνισμού, χρόνου, ποιότητας όπως και οικονομικές, κατά τη διάρκεια των φάσεων αυτών.
- Το παραδοτέο καθορίζεται κατά τη διάρκεια του έργου ανάλογα με το περιβάλλον.

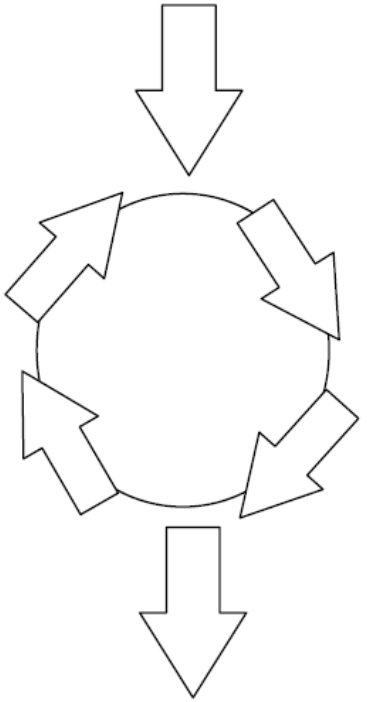
Σύγκριση μεθοδολογιών ανάπτυξης λογισμικού

	Waterfall	Spiral	Iterative	SCRUM
Defined processes	Required	Required	Required	Planning & Closure only
Final product	Determined during planning	Determined during planning	Set during project	Set during project
Project cost	Determined during planning	Partially variable	Set during project	Set during project
Completion date	Determined during planning	Partially variable	Set during project	Set during project
Responsiveness to environment	Planning only	Planning primarily	At end of each iteration	**Throughout**
Team flexibility, creativity	Limited - cookbook approach	Limited - cookbook approach	Limited - cookbook approach	**Unlimited during iterations**
Knowledge transfer	Training prior to project	Training prior to project	Training prior to project	**Teamwork during project**
Probability of success	Low	Medium low	Medium	**High**

Φάσεις του SCRUM

Scrum Methodology

- **Pregame**
  - Planning
  - System Architecture/High Level Design
- **Game**
  - Sprints (Concurrent Engineering)
  - Develop (Analysis,Design,Develop)
  - Wrap
  - Review
  - Adjust
- **Postgame**
  - Closure



Φάσεις του SCRUM

- ***Pregame***
  - Σχεδιασμός: Ορισμός μιας νέας έκδοσης στηριζόμενη στο υπάρχον γνωστό backlog(απόθεμα), μαζί με μια εκτίμηση του χρόνου και του κόστους του. Εάν ένα νέο σύστημα αναπτύσσεται, η φάση αυτή αποτελείται από αντιληπτική ικανότητα και ανάλυση. Εάν ένα υπάρχων σύστημα επεκτείνεται, η φάση αυτή αποτελείται από περιορισμένη ανάλυση.
  - Αρχιτεκτονική: Σχεδιασμός το πώς τα backlog αντικείμενα θα υλοποιηθούν. Η φάση αυτή περιλαμβάνει τροποποίηση της αρχιτεκτονικής του συστήματος και σχεδιασμός υψηλού επιπέδου.

Φάσεις του SCRUM

- ***Game***
  - Sprint Ανάπτυξης: Η ανάπτυξη λειτουργικότητας της νέας έκδοσης, με σταθερό σεβασμό στις μεταβλητές του χρόνου, απαιτήσεων, ποιότητας, κόστους και ανταγωνισμού. Η διαδραστικότητα με αυτές τις μεταβλητές ορίσει και το τέλος της φάσης. Υπάρχουν πολλαπλά, επαναλαμβανόμενα, αναπτυξιακά Sprint, ή κύκλοι, που χρησιμοποιούνται για να εξελίξουν το σύστημα.
- ***Postgame***
  - Κλείσιμο: Προετοιμασία για έκδοση, που περιλαμβάνει τα τελικά εγχειρίδια, προ-έκδοσης στάδιο ελέγχου, και έκδοση(release).

Έλεγχοι SCRUM

- Backlog: Απαιτήσεις λειτουργικότητας προϊόντος που δεν δρομολογούνται ικανοποιητικώς από την υπάρχουσα έκδοση προϊόντος. Προβλήματα(bugs), ελαττώματα(defects), επεκτάσεις που ζήτησε ο πελάτης, ανταγωνιστική λειτουργικότητα προϊόντος, ακραία ανταγωνιστική λειτουργικότητα, και αναβαθμίσεις τεχνολογίας. Όλα τα παραπάνω αποτελούν τα backlog αντικείμενα.
- Έκδοση/Επέκταση: Τα αντικείμενα backlog που σε μια συγκεκριμένη χρονική στιγμή αντικατοπτρίζουν μια βιώσιμη έκδοση στηριζόμενοι στις μεταβλητές των απαιτήσεων, χρόνου, ποιότητας και ανταγωνισμού.
- Πακέτα: Τμήματα του προϊόντος ή αντικείμενα που πρέπει να αλλαχθούν για την υλοποίηση ενός αντικειμένου backlog σε μια νέα έκδοση.

Έλεγχοι SCRUM

- Αλλαγές: Αλλαγές που πρέπει να συμβούν σε ένα πακέτο για την υλοποίηση ενός backlog αντικειμένου.
- Προβλήματα: Τεχνικά προβλήματα που συμβαίνουν και πρέπει να λυθούν για την υλοποίηση μιας αλλαγής.
- Ρίσκο: Ρίσκο που επηρεάζουν την επιτυχία ενός έργου συνεχώς εκτιμούνται και δρομολογούνται δράσεις. Άλλοι έλεγχοι επηρεάζονται ως αποτέλεσμα της διαχείρισης ρίσκου.
- Λύσεις: Λύσεις στα προβλήματα και τα ρίσκο, που έχουν συνήθως ως αποτέλεσμα τις αλλαγές.
- Θέματα: Γενικά θέματα έργου που δεν ορίζονται με όρους πακέτου, αλλαγών και προβλημάτων.

Χαρακτηριστικά SCRUM

- Ευέλικτο παραδοτέο: Το περιεχόμενο του παραδοτέου υπαγορεύεται από το περιβάλλον.
- Ευέλικτος προγραμματισμός: Το παραδοτέο μπορεί να απαιτηθεί νωρίτερα ή αργότερα από τον αρχικό προγραμματισμό.
- Μικρές ομάδες: Κάθε ομάδα δεν έχει παραπάνω από έξι άτομα. Μπορεί να υπάρχουν πολλαπλές ομάδες σε ένα έργο.
- Συχνές αναθεωρήσεις: Η εξέλιξη της ομάδας αναθεωρείται όπως ορίζεται από την περιβαλλοντική πολυπλοκότητα και το ρίσκο(συνήθως κύκλοι μίας με τεσσάρων εβδομάδων). Ένα λειτουργικό εκτελέσιμο πρέπει να ετοιμαστεί από κάθε ομάδα για κάθε αναθεώρηση.
- Συνεργασία: Εσωτερική και εξωτερική συνεργασία αναμένεται, κατά τη διάρκεια του έργου.
- Αντικειμενοστραφής(Object-Oriented): Κάθε ομάδα θα δρομολογήσει ένα σύνολο από συσχετιζόμενα αντικείμενα και καθαρές διεπαφές και συμπεριφορά.

Πίνακας προβλημάτων SCRUM

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης
ψηφιακήελλάδα
Όλα είναι δυνατά
Επιχειρησιακό Πρόγραμμα "Ψηφιακή Σύγκλιση"
grnet
Διασυνδέοντας την Έρευνα και τ
ΕΣΠΑ 2007-2013
Ποιότητα ζωής για όλους
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
Επιστημονικό & ΤΕχνολογικό Πάρκο Ηπείρου
MedLab
INTELLIGENT INFORMATION SYSTEMS

Extreme Programming

- Η φιλοσοφία του Extreme Programming δεν αποτελεί ένα ευρύ σύστημα διαχείρισης έργου, αλλά ένα σύνολο από τις καλύτερες πρακτικές ανάπτυξης λογισμικού που έχουν κοινά χαρακτηριστικά με τις καλύτερες πρακτικές της περιοχής της διαχείρισης έργων

Extreme Programming

- Διαχείριση ρίσκου
  - Μικρές εκδόσεις:
    - Το Extreme Programming βασίζεται στο ότι τα μικρότερα έργα έχουν υψηλότερο ποσοστό επιτυχίας. Σπάει όλα τα έργα λογισμικού σε πολλαπλές μικρές εκδόσεις, όπου κάθε έκδοση του λογισμικού εμπεριέχει μόνο ένα υποσύνολο από την απαιτούμενη λειτουργικότητα. Αυτές οι μικρές εκδόσεις, είναι αυξανόμενες εκδόσεις προϊόντος του τελικού αναμενόμενου έργου, παρέχοντας περιορισμένα υποσύνολα της λειτουργικότητας στους χρήστες συστήματος. Κάθε έκδοση της επιπρόσθετης λειτουργικότητας παρέχει στους τελικούς χρήστες μια δυνατότητα να χρησιμοποιήσουν τις εξελικτικές δυνατότητες του λογισμικού και παρέχουν υψηλής ποιότητας ανάδραση, έτσι βελτιώνοντας την ποιότητα της προοδευτικής επεξεργασίας.
    - Ο στοχευόμενος χρόνος ανάμεσα στις εκδόσεις είναι δύο με έξι βδομάδες, με μια ισχυρή προσπάθεια για τη μικρότερη πιθανή χρονική περίοδο. Ενώ μπορεί να είναι δύσκολο να αναγνωρίσεις κατάλληλες λειτουργικότητες για τις πρώτες εκδόσεις, η σημασία των πρώτων εκδόσεων δεν μπορεί να μεγαλοποιηθεί. Τα έργα χρησιμοποιούν, μικρές αυξανόμενες εκδόσεις που επωφελούνται από την ανάδραση του χρήστη και από την πολύ καλά κατανοητή κατάσταση του έργου, και παρέχουν μια επιστροφή της επένδυσης(return-on-investment) πριν το έργο ολοκληρωθεί πλήρως.

Extreme Programming

- Ενοποιημένη Διαχείριση
  - Συνεχής ενοποίηση

Extreme Programming

- Διαχείριση Σταδίων και Διαχείριση Χρόνου

Properties of	
Work Packages	**Story Cards**
Are deliverable-oriented.	Describe something of value to the user.
Usually contains no more than eighty hours of effort to complete.	Need to be of a size that you can build a few of them every two weeks.
Are independent.	Should be independent of each other.
Are testable.	Can be tested.
Can be broken into activity lists.	Are broken down into tasks.

Extreme Programming

- Προγραμματισμός των κανόνων

Wedding.com >> Summary Report
Create a report that lists the number of guests invited, the number of guests that have RSVP'd in the affirmative, the number of guests that have RSVP'd regrets, and the number of invitations that as yet have not RSVP'd. Show the total budget for the wedding, the funds already spent, the funds required to pay for items already ordered or booked, and the budget amount still remaining. Also show the date the report was printed and the number of days until the wedding.

Extreme Programming

- Διαχείριση των ανθρώπινων πόρων
  - Συλλεκτική ιδιοκτησία
  - Υποφερτή προσπάθεια

Extreme Programming

- Διαχείριση Ποιότητας
  - Απλός Σχεδιασμός και Επανάληψη Αναλύσεων
  - Έλεγχος
  - Συνδυαζόμενος προγραμματισμός

Extreme Programming

- Διαχείριση επικοινωνίας
  - «Όρθια» συνάντηση
  - Κοινός χώρος εργασίας
  - Πελάτης «On site»
  - «Μεταφορά»

Συνδυασμός Scrum με Extreme Programming

- PAIR PROGRAMMING
- TEST-DRIVEN DEVELOPMENT (TDD)
- ΑΥΞΗΤΙΚΟΣ ΣΧΕΔΙΑΣΜΟΣ
- ΣΥΝΕΧΗΣ ΟΛΟΚΛΗΡΩΣΗ
- ΣΥΛΛΕΚΤΙΚΗ ΙΔΙΟΚΤΗΣΙΑ ΚΩΔΙΚΑ
- ΠΛΗΡΟΦΟΡΗΜΕΝΟΣ ΧΩΡΟΣ ΕΡΓΑΣΙΑΣ
- ΠΡΟΤΥΠΟΠΟΙΗΣΗ ΚΩΔΙΚΑ
- ΥΠΟΦΕΡΤΟΣ ΡΥΘΜΟΣ/ΕΝΕΡΓΟΠΟΙΗΜΕΝΗ ΕΡΓΑΣΙΑ

DevOps

- Μέθοδος ανάπτυξης κώδικα
- Αναπτύχθηκε για να τονίσει την επικοινωνία, τη συνεργασία και την ενοποίηση ανάμεσα στους μηχανικούς λογισμικού και του IT operations professional
- Χρησιμοποιεί μεθοδολογίες Agile καθώς και άλλες μεθοδολογίες ανάπτυξης λογισμικού
- Δημιουργήθηκε για να καλύψει την απαίτηση για υψηλό ρυθμό εκδόσεων παραγωγής κώδικα
- Χρησιμοποιεί πολλές virtualized και cloud δομές
- Χρησιμοποιεί σε μεγάλο βαθμό data center automation εργαλεία και εργαλεία διαχείρισης έργων

Ποιότητα Λογισμικού

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης
ψηφιακήελλάδα
Όλα είναι δυνατά
Επιχειρησιακό Πρόγραμμα "Ψηφιακή Σύγκλιση"
grnet
Διασυνδέοντας την Έρευνα και τ
ΕΣΠΑ 2007-2013
Ποιότητα ζωής για όλους
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
Επιστημονικό & Τεχνολογικό Πάρκο Ηπείρου
MEDICAL TECHNOLOGY
Medlab
INTELLIGENT INFORMATION SYSTEMS

Εισαγωγή

- Οι μετρήσεις χωρίζονται σε άμεσες και έμμεσες:
  - Άμεση μέτρηση ενός χαρακτηριστικού είναι η μέτρηση που δεν βασίζεται σε μέτρηση κάποιου άλλου χαρακτηριστικού. Αντιθέτως,
  - έμμεση μέτρηση ενός χαρακτηριστικού είναι η μέτρηση που περιλαμβάνει μετρήσεις ενός ή περισσοτέρων άλλων χαρακτηριστικών.

Εισαγωγή

Κατά την παραγωγή ενός προϊόντος στη βιομηχανία ακολουθείται πρόγραμμα εξασφάλισης ποιότητας, σύμφωνα με το οποίο:

- καθορίζονται τα χαρακτηριστικά-ιδιότητες του προϊόντος που θα μετρηθούν, καθώς και των επιθυμητών ορίων,
- καθορίζονται οι διαδικασίες μέτρησης,
- εντοπίζονται και απορρίπτονται τα προϊόντα που δεν πληρούν τις ποιοτικές προδιαγραφές και
- βελτιώνεται η διαδικασία παραγωγής ώστε να ελαχιστοποιηθεί ο αριθμός των προϊόντων που απορρίπτονται.

Εισαγωγή

- Οι βασικές αρχές της εξασφάλισης ποιότητας λογισμικού είναι ίδιες με την εξασφάλιση ποιότητας κάθε άλλου προϊόντος. Παρόλα αυτά υπάρχει μία βασική αντίθεση που διαμορφώνει την εξής ιδιαιτερότητα κατά την παραγωγή προϊόντων λογισμικού:
  - Στη μαζική παραγωγή καθορίζονται αρχικά οι προδιαγραφές και τα χαρακτηριστικά ενός πρωτοτύπου και στη συνέχεια παράγονται ποσότητες προϊόντων ίδιων φαινομενικά με το αρχικό πρωτότυπο, με σκοπό να μην αποκλίνουν τα αντίγραφα αυτά από τις αρχικές προδιαγραφές.
  - Αντίθετα, στην περίπτωση του λογισμικού, το τελικό προϊόν (αν εξαιρέσουμε τις διαδικασίες συσκευασίας και το συνοδευτικό υλικό, όπως π.χ. εγχειρίδια και μέσο αποθήκευσης) παράγεται μία μόνο φορά. Τα αντίτυπα που αναπαράγονται στη συνέχεια είναι πιστά αντίγραφα του πρωτοτύπου.

Εισαγωγή

- Μία ορθή στρατηγική για τη βελτίωση της ποιότητας λογισμικού πρέπει να προτείνει στις εταιρείες παραγωγής λογισμικού
  - να υλοποιούν πολύ λιγότερο κώδικα
  - να επιλέγουν πολύ προσεκτικότερα τι ακριβώς υλοποιούν και
  - να θέτουν ελαστικότερα όρια ολοκλήρωσης και παράδοσης των έργων τους.

- Διεθνή πρότυπα:
  - ISO9000,
  - IEEE,
  - τα βραβεία Baldrige,
  - το Capability Maturity Model (CMM)
  - και το Capability Maturity Model Integration (CMMI),

- Τα πρότυπα αυτά παρέχουν βασικούς κανόνες στις εταιρείες παραγωγής λογισμικού και ενθαρρύνουν την εφαρμογή μετρήσεων, χωρίς όμως να προτείνουν συγκεκριμένες λύσεις για τις μετρήσεις.

Εργαλεία ποιότητας λογισμικού

- Η καταγραφή των προδιαγραφών έχει ως επακόλουθο χρονικά τη διαδικασία ανάπτυξης όπου θα πρέπει να επιλεγούν και να χρησιμοποιηθούν οι κατάλληλες τεχνικές και διαδικασίες για την παραγωγή της εφαρμογής λογισμικού.

- Πρώτα απ' όλα πρέπει να επιλεγεί ο τρόπος με τον οποίο θα παραδοθεί η εφαρμογή σε ένα προβλεπόμενο τρόπο
- Το επόμενο βήμα είναι να εξασφαλίσουμε ότι η εφαρμογή που βρίσκεται υπό ανάπτυξη δουλεύει – ιδανικά κατά τη διάρκεια της ανάπτυξης
- Χρειαζόμαστε ένα μηχανισμό να συναρμολογεί την εφαρμογή μας σε επαναλαμβανόμενο και αξιόπιστο τρόπο
- Τέλος θα θέλαμε να ενεργοποιήσουμε μια εύκολη διατήρηση της βάσης του κώδικα (code base) έτσι ώστε να προστίθενται χαρακτηριστικά, συχνά με τον ίδιο γρήγορο και επαναλαμβανόμενο τρόπο που η εφαρμογή χτίζεται

Εργαλεία ποιότητας λογισμικού

- Εργαλεία κατασκευής:
  - Ακρογωνιαίος λίθος του SDLC. Είναι το εργαλείο που συνεργάζεται, συνδέει όλα τα άλλα SDLC εργαλεία σε μία συνεκτική διαδικασία. Επίσης το εργαλείο κατασκευής εξασφαλίζει ότι το έργο μπορεί να χτιστεί σε οποιοδήποτε μηχάνημα, σε οποιοδήποτε περιβάλλον.
  - Παραδείγματα: Ant, Maven 2
- Εργαλεία ελέγχου έκδοσης:
  - Παρέχει αντίγραφα ασφαλείας για τον κώδικα και επιτρέπει στους προγραμματιστές να δουλεύουν παρέα στο ίδιο έργο. Επίσης το σύστημα αυτό επιτρέπει να αναγνωριστούν οι εκδόσεις και οι συνεργαζόμενες releases, και αν κριθεί απαραίτητο rollbacks του κώδικα
  - Παραδείγματα: Cvs, Subversion, Git

Εργαλεία ποιότητας λογισμικού

- Τεστ μονάδος:
  - εξασφαλίζει ότι δουλεύει ο κώδικας, και ενισχύει πιο καθαρό, πιο modular και καλύτερα σχεδιασμένο κώδικα
  - Παραδείγματα: Junit, TestNG
- Τεστ ενοποίησης φόρτωσης και απόδοσης
- Εργαλεία μέτρησης ποιότητας
  - Η ποιότητα κώδικα έχει άμεση σχέση με τον αριθμό των σφαλμάτων καθώς και την ευκολία διατήρησης αργότερα. Επίσης οι μετρικές ποιότητας κώδικα είναι ένας καλός τρόπος να βελτιώσουμε μη έμπειρους προγραμματιστές ακολουθώντας συμφωνίες κώδικα (code conventions) και τις καλύτερες πρακτικές (best practices).
  - Παραδείγματα: CheckStyle, PDM, FindBugs, Jupiter

Εργαλεία ποιότητας λογισμικού

- Εργαλεία τεχνικής τεκμηρίωσης
- Εργαλεία διαχείρισης ζητημάτων
  - Το σύστημα αυτό μπορεί να χρησιμοποιηθεί από τους εκτελεστές τεστ για να αναφέρουν σφάλματα (bugs) και τους προγραμματιστές για να τεκμηριώσουν τις διορθώσεις σφαλμάτων. Επίσης μπορεί να χρησιμοποιηθεί για να οργανωθούν και να τεκμηριωθούν εκδόσεις, να σχεδιαστούν επαναλήψεις και να ανατεθούν δουλειές σε μέλη της ομάδας.
  - Παραδείγματα: Bugzilla, Trac, Redmine (ma.ellak.gr/forge)
- Εργαλεία συνεχής ενοποίησης
  - Ο στόχος είναι να τα καλύψουμε όλα κάτω από μία μοναδική διαδικασία. Η διαδικασία αυτή ονομάζεται διαδικασία συνεχής ενοποίησης (Continuous Integration or CI).
  - Παραδείγματα: Continuum, CruiseControl, LuntBuild και Hudson

Μετρικές ποιότητας λογισμικού

- Οι μετρικές που χρησιμοποιούνται για την μέτρηση της ποιότητας ενός συστήματος λογισμικού ποικίλουν και εξαρτώνται από πολλούς παράγοντες, όπως:
  - Την τεχνολογία υλοποίησης του συστήματος: αντικειμενοστραφής ή ακολουθιακός προγραμματισμός, γλώσσα προγραμματισμού κτλ.
  - Τα συγκεκριμένα χαρακτηριστικά που μας ενδιαφέρουν: εσωτερικά ή εξωτερικά.

- Εσωτερικές Μετρικές
  - Οι εσωτερικές μετρικές λογισμικού χρησιμοποιούνται για τη μέτρηση χαρακτηριστικών του λογισμικού για τα οποία υπάρχει απτή αντίληψη για τη φυσική τους σημασία και δυνατότητα άμεσης μέτρησης

Μετρικές ποιότητας λογισμικού

- Εσωτερικές Μετρικές
  - Μια βασική τους κατηγοριοποίηση είναι σε μετρικές **μεγέθους** (όπως οι μετρικές του Halstead), **δομής** (όπως η μετρική πολυπλοκότητας του McCabe) και **δεδομένων** (όπως η μετρική πολυπλοκότητας δομών δεδομένων του Tsai).

  - Οι μετρικές μεγέθους (Halstead) μετρούν αριθμήσιμα στοιχεία του λογισμικού σχετιζόμενα με το μέγεθος του πηγαίου κώδικα. Οι τέσσερις βασικές μετρήσιμες ποσότητες που προτείνει είναι οι ακόλουθες:
  - **n1** ο αριθμός των διακριτών τελεστών που εμφανίζονται στο πρόγραμμα.
  - **n2** ο αριθμός των διακριτών εντέλων που εμφανίζονται στο πρόγραμμα.
  - **N1** ο αριθμός των συνολικών εμφανίσεων τελεστών στο πρόγραμμα.
  - **N2** ο αριθμός των συνολικών εμφανίσεων εντέλων στο πρόγραμμα.

Μετρικές ποιότητας λογισμικού

- Εσωτερικές Μετρικές
  - Η μετρική κυκλωματικής πολυπλοκότητας του McCabe είναι μία μετρική δομής και μετράει πόσο πολύπλοκος είναι ο γράφος ροής του προγράμματος. Επικεντρώνει το ενδιαφέρον της στα σημεία του προγράμματος που μπορεί να ληφθεί απόφαση και σχετίζεται με τα πιθανά μονοπάτια που μπορεί να ακολουθήσει η ροή του προγράμματος. Η μετρική αυτή ονομάζεται και κυκλωματικός αριθμός V(G), όπου G ο γράφος του προγράμματος, και δίνεται από την παρακάτω σχέση:
  - **V(G) = e – n + 2•p,**
  - όπου **e** ο αριθμός των ακμών του γράφου,
  - **n** ο αριθμός των κόμβων του γράφου και
  - **p** ο αριθμός των συνεκτικών συνιστωσών του γράφου.
  - Στον παραπάνω γράφο, που είναι ένας απλοποιημένος γράφος ελέγχου, οι κόμβοι είναι τα σημεία απόφασης και οι ακμές είναι οι εναλλακτικές δράσεις. Προφανώς, όσο μεγαλύτερος είναι ο κυκλωματικός αριθμός μίας ρουτίνας, τόσο πιο πολύπλοκη είναι αυτή. Σε περιπτώσεις ρουτινών με πολύ μεγάλο κυκλωματικό αριθμό συνίσταται να γίνεται διάσπασή τους σε περισσότερες.

  - Η μετρική της πολυπλοκότητας δομών δεδομένων του Tsai εφαρμόζεται στα δεδομένα του προγράμματος, επομένως το μόνο που χρειάζεται είναι να υπάρχουν οι λεπτομερείς κατάλογοι των δεδομένων που θα επεξεργαστεί το πρόγραμμα και όχι η τελική έκδοση του κώδικα του προγράμματος.

Μετρικές ποιότητας λογισμικού

- Εσωτερικές Μετρικές
- Άλλα γνωστά παραδείγματα εσωτερικών μετρικών λογισμικού αυτής της κατηγορίας είναι τα εξής:
  - μετρική γραμμών κώδικα,
  - μετρική γραμμών σχολίων προς γραμμές κώδικα,
  - μετρικές για λειτουργικά σημεία,
  - μετρικές τμημάτων,
  - μετρικές ζωντανών μεταβλητών,
  - μετρική ελαχίστου αριθμού μονοπατιών,
  - μετρικές διασταυρώσεων,
  - μετρικές εισόδων – εξόδων,
  - μετρικές πολυπλοκότητας,
  - μετρική ουσιώδους πολυπλοκότητας.
- Οι μετρικές αυτές είναι αξιοποιήσιμες για κάθε γλώσσα προγραμματισμού που βασίζεται σε πηγαίο κώδικα και δεν σχετίζονται με ειδικά χαρακτηριστικά κάποιας συγκεκριμένης γλώσσας προγραμματισμού.

Μετρικές ποιότητας λογισμικού

- Εξωτερικές Μετρικές

- Οι εξωτερικές μετρικές λογισμικού σχετίζονται άμεσα με τα εξωτερικά ποιοτικά χαρακτηριστικά του λογισμικού

- Τεχνικές Μέτρησης: Ένας πρώτος διαχωρισμός τους μπορεί να γίνει σε **αναλυτικές** και **εμπειρικές** τεχνικές.
  - Οι αναλυτικές στηρίζονται σε θεωρητικά μοντέλα που προσομοιώνουν τη συμπεριφορά του χρήστη ή σε πρότυπα και κανόνες. Παραδείγματα αποτελούν η ανάλυση πληκτρολογήσεων, το γνωστικό περιδιάβασμα, μέθοδοι ευρετικής αξιολόγησης, έλεγχος συμβατότητας με κανόνες σχεδιασμού και πρότυπα

  - Οι εμπειρικές στηρίζονται στην κατασκευή και αξιολόγηση της συμπεριφοράς ή των χαρακτηριστικών ενός πρωτοτύπου ή ενός ολοκληρωμένου συστήματος. Παραδείγματα αποτελούν η μέτρηση απόδοσης, πρωτόκολλο ομιλούντων χρηστών, καταγραφή ενεργειών των χρηστών, συμπλήρωση ερωτηματολογίων, συνεντεύξεις χρηστών, ομαδική αξιολόγηση.

ΔΙΕΘΝΕΣ ΠΡΟΤΥΠΟ ISO/IEC 9126

- Το πρότυπο ISO 9126 αποτελεί ένα μοντέλο ποιότητας που είναι απόλυτα ιεραρχικό και δεν υπάρχουν επικαλύψεις, μιας και κάθε χαρακτηριστικό ανήκει αποκλειστικά σε ένα παράγοντα ποιότητας. Παράλληλα, κάθε χαρακτηριστικό δεν είναι εσωτερικό του προϊόντος αλλά είναι ορατό στον χρήστη. Έτσι με τον τρόπο αυτό έχουμε μια θεώρηση περισσότερο από την πλευρά του χρήστη και όχι από την ομάδα ανάπτυξης του λογισμικού. Χαρακτηριστικά του είναι:

- Η λειτουργικότητα
- Η αξιοπιστία
- Η ευχρηστία
- Η αποδοτικότητα
- Η συντηρησιμότητα
- Η μεταφερσιμότητα

ΔΙΕΘΝΕΣ ΠΡΟΤΥΠΟ ISO/IEC 9126

- ΤΟ ΠΡΟΤΥΠΟ ISO/IEC 25010
- Το νέο πρότυπο έχει οκτώ χαρακτηριστικά ποιότητας (σε σύγκριση με τα έξη του ISO/IEC 9126) και τριάντα ένα υποχαρακτηριστικά. Περιγράφει ποια χαρακτηριστικά πρέπει να οριστούν, μετρηθούν και εκτιμηθούν όχι το πως.
- Το νέο πρότυπο έχει δύο κύριες διστάσεις που ορίζονται ως εξής:

- Ένα **μοντέλο ποιότητας προϊόντος λογισμικού** (software product quality model) που αποτελείται από οκτώ χαρακτηριστικά που περαιτέρω υποδιαιρούνται σε υποχαρακτηριστικά που μπορούν να μετρηθούν εσωτερικά ή εξωτερικά. Ένα μοντέλο δηλαδή που αναφέρεται στις στατικές και στις δυναμικές ιδιότητες του λογισμικού. Ένα μοντέλο για τα εγγενή, εσωτερικά και ουσιαστικά δηλαδή, χαρακτηριστικά του προϊόντος.

- Ένα **μοντέλο ποιότητας συστήματος σε χρήση** (system quality in use model) αποτελείται από πέντε χαρακτηριστικά, τα οποία στη συνέχεια υποδιαιρούνται σε υποχαρακτηριστικά που μπορούν να μετρηθούν όταν ένα προϊόν χρησιμοποιείται σε ένα ρεαλιστικό πλαίσιο χρήσης. Το μοντέλο αυτό χαρακτηρίζει την επίδραση του προϊόντος στα ενδιαφερόμενα μέρη. Καθορίζεται η ποιότητά του από την ποιότητα του λογισμικού, του υλικού και του λειτουργικού συστήματος. Επίσης χαρακτηρίζει τους χρήστες, τις λειτουργίες το κοινωνικό περιβάλλον και εφαρμόζεται σε ένα πλήρως σύστημα ανθρώπου-υπολογιστή (human computer system). Ένα μοντέλο δηλαδή για τα χαρακτηριστικά που σχετίζονται με την ανθρώπινη χρήση του προϊόντος.

ΔΙΕΘΝΕΣ ΠΡΟΤΥΠΟ ISO/IEC 9126

- Ορισμός της **ποιότητας σε χρήση** (quality in use):
  - Προκύπτει δοκιμάζοντας ή παρατηρώντας τα αποτελέσματα μια πραγματικής ή εξομοιωμένης χρήσης
  - Μέτρηση εγγενών ιδιοτήτων ενός συστήματος που μπορεί να περιλαμβάνει, υλικό, λογισμικό, επικοινωνίες και χρήστες
  - Επίσης μετράμε ιδιότητες εξαρτώμενες από το σύστημα ενός λογισμικού προϊόντος
  - Επιτυγχάνεται μόνο σε ρεαλιστικά περιβάλλοντα (σε χρήση)

- Οι μετρικές ποιότητας σε χρήση σχετίζονται με την ολοκλήρωση ρεαλιστικών εργασιών από τους χρήστες (είτε από δοκιμές χρηστών είτε από κανονική χρήση). Οι εξωτερικές μετρικές χρήσης σχετίζονται με τη συμπεριφορά μοναδικών λειτουργιών, και μπορούν να εκτιμηθούν μοναδικά, ή ως ένα μέρος ενός ευρύτερου user testing που μπορεί να μετρά επίσης την γενική χρηστικότητα.

ΔΙΕΘΝΕΣ ΠΡΟΤΥΠΟ ISO/IEC 9126

- Το πρότυπο ISO 25010 είναι διαφορετικό από το ISO 9126 στα εξής:
  - Ορίζονται πια οι σχέσεις ανάμεσα στο σύστημα και το λογισμικό στο νέο μοντέλο με την εισαγωγή του μοντέλου δεδομένων (data model) (ISO 25012)
  - Η ποιότητα σε χρήση έχει πέντε χαρακτηριστικά αντί για τέσσερα χωρίς την «παραγωγικότητα» και τη «συμβατότητα» αλλά προσθέτοντας την «αποτελεσματικότητα», την «ικανοποίηση» και τη «χρηστικότητα»
  - Η ενοποίηση των εξωτερικών και των εσωτερικών χαρακτηριστικών και των υποκατηγοριών τους με δύο νέα χαρακτηριστικά: την «ασφάλεια» και τη «συμβατότητα».

ΔΙΕΘΝΕΣ ΠΡΟΤΥΠΟ ISO/IEC 9126

- Το πρότυπο ISO 25010 είναι διαφορετικό από το ISO 9126 στα εξής:
  - Ορίζονται πια οι σχέσεις ανάμεσα στο σύστημα και το λογισμικό στο νέο μοντέλο με την εισαγωγή του μοντέλου δεδομένων (data model) (ISO 25012)
  - Η ποιότητα σε χρήση έχει πέντε χαρακτηριστικά αντί για τέσσερα χωρίς την «παραγωγικότητα» και τη «συμβατότητα» αλλά προσθέτοντας την «αποτελεσματικότητα», την «ικανοποίηση» και τη «χρηστικότητα»
  - Η ενοποίηση των εξωτερικών και των εσωτερικών χαρακτηριστικών και των υποκατηγοριών τους με δύο νέα χαρακτηριστικά: την «ασφάλεια» και τη «συμβατότητα».

Θέματα Ασφαλείας Λογισμικού

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης
ψηφιακήελλάδα
Όλα είναι δυνατά
Επιχειρησιακό Πρόγραμμα "Ψηφιακή Σύγκλιση"
grnet
Διασυνδέοντας την Έρευνα και τ
ΕΣΠΑ 2007-2013
Ποιότητα ζωής για όλους
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
Επιστημονικό & Τεχνολογικό Πάρκο Ηπείρου
MEDICAL TECHNOLOGY
Medlab
INTELLIGENT INFORMATION SYSTEMS

Η πρόκληση της εξασφάλισης λογισμικού και της ασφάλειας

- Τα ρίσκα της εξασφάλισης λογισμικού που αντιμετωπίζονται σήμερα από τους χρήστες μπορούν να κατηγοριοποιηθούν σε τρεις περιοχές:
  - **Τυχαίος σχεδιασμός ή λάθη υλοποίησης** που οδηγούν σε εκμεταλλεύσιμες τρωτότητες κώδικα.
  - **Το μεταβαλλόμενο τεχνολογικό περιβάλλον,** που εκθέτει νέες τρωτότητες και παρέχει νέα εργαλεία για την εκμετάλλευσή τους.
  - **Μοχθηροί μυημένοι(malicious insiders)** που ψάχνουν να κάνουν κακό στους χρήστες.

Η πρόκληση της εξασφάλισης λογισμικού και της ασφάλειας

- Πλαίσιο εργασίας για την ανάπτυξη λογισμικού
  - Όπως παρουσιάστηκε νωρίτερα
- Οι καλύτερες πρακτικές για την ασφάλεια λογισμικού
  - **Εκπαίδευση Ασφαλείας**
  - **Ορισμός Απαιτήσεων Ασφαλείας**
  - **Σχεδιασμός ασφαλείας**
  - **Ασφάλεια στον κώδικα**
  - **Ασφαλής Διαχείριση κώδικα**
  - **Έλεγχος**

Η πρόκληση της εξασφάλισης λογισμικού και της ασφάλειας

- Οι καλύτερες πρακτικές για την ασφάλεια λογισμικού
  - **Τεκμηρίωση**
  - **Ετοιμότητα ασφαλείας**
  - **Ανταπόκριση**
  - **Πιστοποίηση Ακεραιότητας**
  - **Έρευνα ασφαλείας**
  - **Ευαγγελισμός ασφαλείας**

Συσχετιζόμενοι ρόλοι για την ασφάλεια λογισμικού

- Το ευρύτερο οικοσύστημα των ολοκληρωτών λογισμικού, των διαχειριστών και των τελικών χρηστών που αγοράζουν και υλοποιούν τις εφαρμογές, συνεισφέρουν στην ολική εξασφάλιση ενός προϊόντος ή ενός συστήματος.
  - **Ολοκληρωτές:** Καθώς οι εφαρμογές κλιμακώνονται σε πολύ μεγάλα περιβάλλοντα και ολοκληρώνονται (συνεργάζονται) με άλλα προϊόντα και συστήματα, νέα τρωτά τμήματα που δεν υπήρχαν στο προϊόν από μόνο του, μπορεί να εμφανιστούν. Οι ολοκληρωτές πρέπει να δουλέψουν σε συνεργασία με τους κατασκευαστές λογισμικού για να βρουν και να μετριάσουν τα τρωτά αυτά σημεία.

Συσχετιζόμενοι ρόλοι για την ασφάλεια λογισμικού

- **Διαχειριστές:** Οι διαχειριστές πρέπει να εξασφαλίσουν ότι τα συστήματα παραμένουν κατάλληλα ρυθμισμένα. Αυτοματοποιημένη διόρθωση προβλημάτων πρέπει να ενεργοποιηθεί για να αυξήσει την ταχύτητα της διόρθωσης των τρωτών σημείων. Οι διαχειριστές επίσης πρέπει να υλοποιούν πρότυπα πολυεπίπεδα αμυντικά μέτρα για ασφάλεια, όπως τοίχοι ασφαλείας(firewall), αντιβιοτικά(antivirus), anti-malware, anti-phising, ανίχνευση εισβολής και αποτροπή, virtual private networks, ισχυρή αυθεντικοποίηση και διαχείριση αναγνώρισης(identity management).
- **Τελικοί χρήστες:** Οι τελικοί χρήστες πρέπει να αναλάβουν την ευθύνη να αναφέρουν πιθανά προβλήματα(bugs) ή τρωτά σημεία και δεν πρέπει να εισάγουν λογισμικό από μη έμπιστες πηγές στα συστήματα. Η υπεύθυνη χρήση του λογισμικού είναι μια σημαντική συνεχιζόμενη απαίτηση, εξασφάλιση και ασφάλεια.

Στόχοι εξασφάλισης λογισμικού

- **Μια Εκτεταμένη Βάση Γνώσης:** Οι μηχανικοί ανάπτυξης εκπαιδεύονται σε πρακτικές συγγραφής ασφαλούς κώδικα με σκοπό να αναπτυχθούν πιστοποιημένα προγράμματα. Οι πελάτες και οι προγραμματιστές κατανοούν την σημασία της εξασφάλισης λογισμικού και συνειδητοποιούν την Επιστροφή Της Επένδυσης(Return On Investment).
- **Ισχυρές Διαδικασίες Ανάπτυξης:** Οι προγραμματιστές υλοποιούν διαδικασίες που παρουσιάζονται να είναι αποτελεσματικές στην βελτίωση της ασφαλείας και έχουν ένα καθαρό μονοπάτι στην έναρξη της διαδικασίας για την κατασκευή υγιών εξασφαλισμένων προγραμμάτων λογισμικού.

Στόχοι εξασφάλισης λογισμικού

- **Συνεπής Εκτίμηση, Μετρικές, και Πιστοποίηση:** Η ποιότητα του κώδικα μπορεί να κριθεί στηριζόμενη τόσο στις εισόδους (ισχύς των διαδικασιών ανάπτυξης, εκπαίδευση και ενημέρωση των προγραμματιστών) και εξόδους(τρωτά σημεία και νέος κακός κώδικας)
- **Διαδικασίες Αποτελεσματικής Αντίδρασης:** Διαδικασίες αναγνώρισης και διόρθωσης νεοανακαλυπτόμενων τρωτών σημείων, δρομολογούνται διαμέσου του οικοσυστήματος του λογισμικού.
- **Αυστηρό R&D:** Οι πιο σημαντικές ανάγκες R&D εξασφάλισης λογισμικού αναγνωρίζονται και υποστηρίζονται με τους κατάλληλους πόρους.

Ερωτήσεις;

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης
ψηφιακήελλάδα
Όλα είναι δυνατά
Επιχειρησιακό Πρόγραμμα "Ψηφιακή Σύγκλιση"
grnet
Διασυνδέοντας την Έρευνα και τ
ΕΣΠΑ 2007-2013
Ποιότητα ζωής για όλους
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
Επιστημονικό & Τεχνολογικό Πάρκο Ηπείρου
MEDICAL TECHNOLOGY
Medlab
INTELLIGENT INFORMATION SYSTEMS