DIS501

ΑΝΑΛΥΣΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Lecture 1: Εισαγωγή στο μάθημα και τα 5Ps

By: Δρ Αυγούστα Κυριακίδου Ζαχαρουδιού

ΕΙΣΑΓΩΓΗ: ΣΤΟΙΧΕΙΑ ΚΑΘΗΓΗΤΡΙΑΣ

Υπεύθυνη καθηγήτρια μαθήματος: Δρ Αυγούστα Κυριακίδου Ζαχαρουδιού

a.kyriakidou-zacharoudiou@nup.ac.cy

Καλύτερος τρόπος επικοινωνίας είναι μέσω email. Στη συνέχεια μπορούμε να κανονίσουμε μια συνάντηση, αν θέλετε να έρθετε να με δείτε

Assistant Professor in Computing and Information Systems, Fellow of Higher Education Academy

PhD, MSc, PGCert, BSc

Certified Scrum Master (Scrum Alliance)

AI in Human-Centered Design Certification (Interaction Design Foundation)

Agile Project Management Foundations Certification (Agile Business Consortium)

DevOps Foundation Certification (The DevOps Institute)

ΕΙΣΑΓΩΓΗ: ΤΙ ΕΙΝΑΙ ΑΥΤΟ ΤΟ ΜΑΘΗΜΑ?

Επιτυχημένα πληροφοριακά συστήματα ξεκινούν με προσεκτικό σχεδιασμό, εξασφαλίζοντας ορθότητα, αποδοτικότητα και αποδοχή από τους χρήστες.

Ανάλυση και σχεδίαση πληροφοριακών συστημάτων αποτελεί βασικό πυλώνα κάθε προσέγγισης ανάπτυξης λογισμικού, είτε ακολουθούνται παραδοσιακές είτε σύγχρονες μεθοδολογίες.

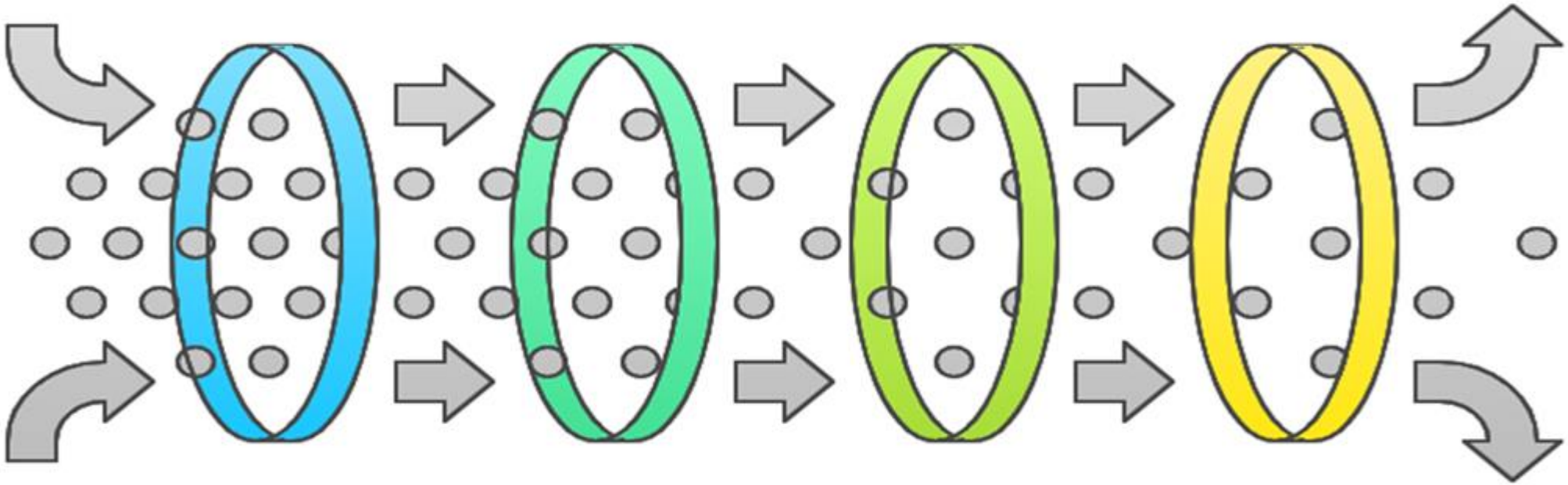
Ο επαρκής ("just enough") σχεδιασμός είναι κρίσιμος – αποτρέπει προβλήματα ικανοποίησης χρηστών και ενσωμάτωσης του συστήματος

Η μεθοδολογική δομή που παρέχει η ανάλυση και σχεδίαση συμβάλλει στην ανάπτυξη τεχνολογικών λύσεων, αποφεύγοντας σπατάλη χρόνου και πόρων.

Με τον κατάλληλο σχεδιασμό, τα συστήματα και οι εφαρμογές που δημιουργούμε είναι πιο ευέλικτα(flexible), συντηρήσιμα(maintainable) και επεκτάσιμα (extensible).

ΕΙΣΑΓΩΓΗ: ΤΙ ΕΙΝΑΙ ΑΥΤΟ ΤΟ ΜΑΘΗΜΑ?

ΑΝΘΡΩΠΟΚΕΝΤΡΙΚΗ ΣΧΕΔΙΑΣΗ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ



Κατανόηση Μεθόδων

Εξοικείωση με διάφορες μεθόδους σχεδιασμού

Ανάλυση Απαιτήσεων Χρηστών

Εξαγωγή και ανάλυση απαιτήσεων χρηστών

Μοντελοποίηση Απαιτήσεων

Μοντελοποίηση μέσω αντικειμενοστραφούς και δομημένης ανάλυσης

Αρχές Χρηστικότητας

Μάθηση βασικών αρχών σχεδιασμού αλληλεπίδρασης

ΕΙΣΑΓΩΓΗ: ΤΙ ΕΙΝΑΙ ΑΥΤΟ ΤΟ ΜΑΘΗΜΑ?

Ανάληψη ενός έργου πληροφορικής από την αρχή έως την ολοκλήρωση

- Εργασία σε ομάδες
- Ανάληψη διαφορετικών ρόλων

Ενσωμάτωση ενός συνόλου δεξιοτήτων

- Δεξιότητες διαχείρισης έργων
- Δεξιότητες ανάλυσης και σχεδιασμού συστημάτων
- Δεξιότητες μερικής υλοποίησης/πρωτοτύπου συστήματος

Αυτό ΔΕΝ είναι μάθημα προγραμματισμού και ΔΕΝ θα διδαχθείτε πώς να γράφετε κώδικα.

Όμως θα έχετε την ευκαιρία να δημιουργήσετε πρωτότυπα συστημάτων με την βοήθεια εργαλείων πρωτοτυποποίησης

ΕΙΣΑΓΩΓΗ: ΓΙΑΤΙ ΑΥΤΟ ΕΝΑΙ ΣΗΜΑΝΤΙΚΟ?

...γιατί αυτό θα αντιμετωπίσετε σε πραγματικά έργα πληροφορικής.

- Προσδιορίσετε τους ρόλους σε μια ομάδα ανάπτυξης σύμφωνα με τα βασικά σας δυνατά σημεία
- Διαπραγμάτευση και διαχείριση ενός έργου πληροφορικής
- Παράδοση προδιαγραφών και απαιτήσεων
- Ανάλυση, μοντελοποίηση και σχεδιασμός ενός προϊόντος
- Εφαρμογή και παράδοση

...επειδή η ανάλυση, σχεδίαση και ανάπτυξη λογισμικού πρέπει να γίνεται σε ένα συγκεκριμένο επιχειρηματικό πλαίσιο

- Ανώφελο αν το σύστημα/εφαρμογή δεν ανταποκρίνεται στις απαιτήσεις του πελάτη
- Εξίσου ανώφελο αν είναι πολύ ακριβό ή αν διαρκεί πολύ ο κύκλος ανάπτυξης του.

ΕΙΣΑΓΩΓΗ: ΠΩΣ

6 Διαδραστικές διαλέξεις (κάθε 2 εβδομάδες) - Πέμπτη 17:00-19:00

- Εισαγωγή στη θεωρία και την πρακτική της ανάλυσης και σχεδίασης πληροφοριακών συστημάτων
- Ερωτήσεις και απαντήσεις
- Διαδραστικές συζητήσεις, μελέτες περιπτώσεων, polls and quizzes

Κύρια Ομαδική Εργασία – Συνοπτική (βαθμολογούμενη) αξιολόγηση

- Βασιζόμενοι σε real-world case study, θα αναλύσετε και σχεδιάσετε σύστημα, καθορίζοντας και μοντελοποιώντας έτσι τις απαιτήσεις του συστήματος
- Χρησιμοποιήσετε κατάλληλα εργαλεία λογισμικού, π.χ. IBM Rational Architecture, Visio, Visual Paradigm κ.λπ. για την μοντελοποίηση, καθώς και εργαλεία π.χ Figma, AxurePR etc. για την δημιουργία πρωτότυπου συστήματος.

Διαδραστικές ασκήσεις και δραστηριότητες

- Κάποιες είναι μέρος της διαμορφωτική Αξιολόγησης - Παρακολούθηση της κατανόησης και των γνώσεων των φοιτητών
- 4 εντάσσονται σαν μέρος της Συνοπτικής Αξιολόγηση

Ασκήσεις Αυτοαξιολόγησης

- Συνοδεύονται από ενδεικτικές απαντήσεις
- Αυτοαξιολόγηση γνώσεων και κατανόησης που αποκτήθηκαν
- Διερευνητική διαδικασία
- Κάλυψη κενών γνώσεων

ΠΡΟΤΕΙΝΟΜΕΝΟ ΠΛΑΝΟ ΥΛΗΣ

- Το έργο, οι άνθρωποι, η διαδικασία, το προϊόν και το πρόβλημα.
- Παραδοσιακές και σύγχρονες διαδικασίες/μεθοδολογίες.
- Κατανόηση και σχεδιασμός προβλημάτων πριν να προχωρήσουμε στην ανάλυση και σχεδιασμό – με την βοήθεια της μεθοδολογίας Soft Systems.
- Καθορισμός και ανάλυση απαιτήσεων – Λειτουργικές, μη Λειτουργικές, Ιστορίες χρήστη κ.α
- Αντικειμενοστρεφής ανάλυση σε βάθος - με χρήση της UML και των σχετικών διαγραμμάτων και μοντέλων. (Use Case Modelling, Conceptual Class Diagram)
- Σε βάθος αντικειμενοστραφής σχεδιασμός με χρήση της UML. Χαρακτηριστικά της καλής σχεδίασης όπως encapsulation, κληρονομικότητα, επαναχρησιμοποίηση (reusability), κ.λπ. (Dynamic Modelling: Διαγράμματα ακολουθίας αντικειμένων, διαγράμματα καταστάσεων, διαγράμματα κλάσεων σχεδιασμού.)
- Πίσω στην Δομημένη Αναλυση και σχεδιασμό συστηματων – Data flow diagrams, Entity Relationship Diagrams and Entity Life Histories. Ζητήματα σχετικά με την εφαρμογή του σχεδιασμού UML σε κώδικα.
- Ανθρωποκεντρικός σχεδιασμός, Αλληλεπίδραση ανθρώπου υπολογιστή, αρχές χρηστικότητας, ευκολίας, προσβασιμότητας, εμπειρίας χρήστη κ.α
- Από τον σχεδιασμό σε αρχική υλοποίηση – Ψευδοκώδικας, θέματα ποιότητας και Έλεγχος συστήματος (software testing)
- Ομαδική εργασία, σχεδιασμός (planning), συνεδριάσεις, καταγραφή πρακτικών και παρουσιάσεις, καταγραφή απαιτήσεων και παραγωγή τεχνικών εγγράφων.

ΕΙΣΑΓΩΓΗ: ΑΞΙΟΛΟΓΗΣΗ ΚΑΙ ΠΑΡΑΔΟΤΕΑ

Συνοπτική αξιολόγηση

1. Ομαδική εργασία - 20%

Ημερομηνία υποβολής: Θα ανακοινωθεί στο Moodle

- **Παραδοτέα:** ομαδική έκθεση που θα παρέχει έναν λεπτομερή **αντικειμενοστρεφή ανάλυση και σχεδιασμό** και την **δημιουργία πρωτοτύπου** για ένα πληροφοριακό σύστημα με βάση μια ρεαλιστική μελέτη περίπτωσης.
  - Η ομάδα θα πρέπει να παρουσιάσει τα αποτελέσματα σε ένα 3-4-λεπτο βίντεο.
- Προδιαγραφή της ομαδικής εργασίας θα αναρτηθεί στο Moodle.
  - Σχέδιο βαθμολόγησης, κριτήρια βαθμολόγησης, σημαντικές ημερομηνίες

2. 4 Διαδραστικές δραστηριότητες - 20%

Ημερομηνίες υποβολής (θα ανοινωθούν στο Moodle)

- Ανατρέξτε και στον οδηγό σπουδών του μαθήματος
- Αν δεν ολοκληρωθεί έστω και μία υποχρεωτική δραστηριότητα, ο τελικός βαθμός θα είναι μηδέν (**0**). Σε αυτήν την περίπτωση, ο Πυλώνας «Δραστηριότητες» επαναλαμβάνεται με πέναλτι 64/100 στην επαναληπτική εξεταστική.
- Για να θεωρηθούν οι δραστηριότητες επιτυχείς, πρέπει να έχετε τουλάχιστον 50% σε κάθε μία.

3. Εξέταση - 60%

Ημερομηνία εξετάσεων: θα ανακοινωθεί

- Ελέγχει τις γνώσεις σας σχετικά με το θεωρητικό υλικό που καλύπτεται σε αυτό το μάθημα.

ΕΙΣΑΓΩΓΗ: ΑΞΙΟΛΟΓΗΣΗ ΚΑΙ ΠΑΡΑΔΟΤΕΑ

Διαμορφωτική αξιολόγηση

1. Κύρια Ομαδική εργασία – Online Ενδιάμεσο παραδοτέο

Ημερομηνία υποβολής: εβδομάδα 9

- Ομαδική εργασία θα εξεταστεί και θα δοθεί διαμορφωτική ανατροφοδότηση.
- Χωρίς βαθμολόγηση, η ανατροφοδότηση θα σας επιτρέψει να επιτύχετε το καλύτερο δυνατό αποτέλεσμα για την τελική ομαδική εργασία σας.

2. Coursework Feedback session κατά την διάρκεια του μαθήματος

- Ανατροφοδότηση για την ομαδική εργασία κατά την διάρκεια του μαθήματος. Στις ομάδες σας θα μπορείτε να παρουσιάσετε την δουλεία που έχετε κάνει, να ρωτήσετε απορίες, να πάρετε ανατροφοδότηση

3. Ασκήσεις κατά την διάρκεια του μαθήματος και διαμορφωτικές δραστηριότητες στο Moodle

- Διαμορφωτική ανατροφοδότηση για τις εβδομαδιαίες ασκήσεις και δραστηριότητές σας. Ενδυνάμωση της κατανόησης σας σε διάφορους τομείς και ανατροφοδότηση σε βασικά κεφάλαια πριν την παράδοση της κύριας σας εργασίας.

ΛΙΣΤΑ ΑΝΑΓΝΩΣΗΣ ΚΑΙ ΑΝΑΦΟΡΩΝ

- Κανένα εγχειρίδιο δεν μπορεί να καλύψει επαρκώς ολόκληρο το μάθημα.
- Αναμένεται να διαβάσετε ευρέως και να καλύψετε τα θέματα του μαθήματος από διάφορες οπτικές γωνίες και πηγές.
- *Λεπτομερείς συμβουλές ανάγνωσης θα παρέχονται κατά την πρόοδο του μαθήματος.*

1. Kendal E.K and Kendal E.J (2023). Ανάλυση & Σχεδίαση Συστημάτων. 10η έκδοση, Εκδόσεις Μ. Γκιούρδας
2. Kung, D.C (2024) Software Engineering, 2η έκδοση, McGraw Hill
3. Pressman, R.S. (2020). Software Engineering: Έκδοση, Mc-Graw Hill.
4. Ε. Α. Γιακουμάκης, Ν. Α. Διαμαντίδης, Τεχνολογία Λογισμικού, 2η έκδοση, Unibooks, 2021.
5. Sommerville, I. (2016). Software Engineering, Pearson

- **Παρακαλώ να ελέγχετε συχνά τη σελίδα του μαθήματος στο Moodle.**

  - **Εγχειρίδιο φοιτητή (Student Guide)**
  - Εβδομαδιαίες διαλέξεις, διδακτικό υλικό, διαμορφωτικές δραστηριότητες
  - Προδιαγραφές εργασιών, υποβολή και άλλες πληροφορίες

ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΑΝΑΛΥΣΗ ΚΑΙ ΣΧΕΔΙΑΣΜΟ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

LEARNING OBJECTIVES

- Τι είναι πληροφοριακό σύστημα, ποιοι οι βασικοί άξονες, παραδείγματα πληροφοριακών συστημάτων σε οργανισμούς
- Παρουσίαση της έννοιας της ανάλυσης και σχεδιασμού και ερμηνεία της σημασίας της
- Καθορισμός του πώς θα μελετήσουμε την Ανάλυση και Σχεδιασμό μέσω των 5Ps - Πρόβλημα, Άνθρωποι, Έργο, Διαδικασία, Προϊόν.

ΠΩΣ ΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ ΜΕΤΑΣΧΗΜΑΤΙΖΟΥΝ ΤΟ ΕΠΙΧΕΙΡΕΙΝ

- Το 2024 περισσότερες από 233.6 εκατομμύρια επιχειρήσεις είχαν κατοχυρωμένους διαδικτυακούς ιστότοπους .com.
- 266 εκατομμύρια ενήλικοι Αμερικανοί επισκέφθηκαν το Διαδίκτυο για τα ψώνια τους
- Οι πωλήσεις μέσω διαδικτύου (e-commerce) στην Αμερική έφτασε τα $300.05 εκατομμύρια στο τρίτο τρίμηνο του 2024, αύξηση της τάξης του 2.8% από το προηγούμενο τρίμηνο και 7.2% σε σύγκριση με τα ίδια τρίμηνα του 2023. to the same quarter the previous year.
- Η διαδικτυακή διαφήμιση συνεχίζει να αυξάνεται κατά περίπου 15% ετησίως.
- Νέοι νόμοι απαιτούν από τις επιχειρήσεις να αποθηκεύουν περισσότερα δεδομένα για μεγαλύτερες περιόδους.
- Οι αλλαγές στην επιχειρηματική δραστηριότητα έχουν ως αποτέλεσμα αλλαγές σε θέσεις εργασίας και σταδιοδρομίες.

Ζούμε σε έναν ψηφιακό κόσμο και τα πληροφοριακά συστήματα έχουν μετασχηματίσει την επιχειρηματική δραστηριότητα σε παγκόσμια κλίμακα

ΤΙ ΕΙΝΑΙ ΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ?

1. Go to **PollEv.com**
2. Enter **AVGOUSTAKYRI977**
3. Respond to activity

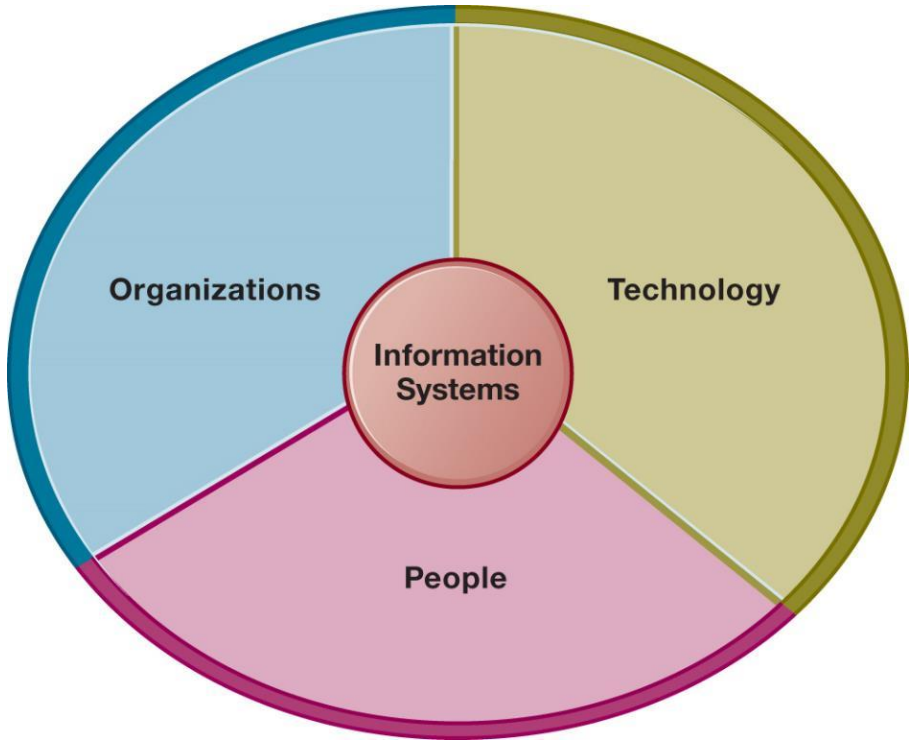
1. Text **AVGOUSTAKYRI977** to **22333**
2. Text in your message

ΤΙ ΕΙΝΑΙ ΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

Τεχνολογία πληροφοριών: το υλικό (hardware) και το λογισμικό (software) που χρησιμοποιεί μια επιχείρηση για την επίτευξη των στόχων

Information system: συσχετιζόμενα στοιχεία (software, hardware, people, networks, data recourses) που διαχειρίζονται πληροφορίες για να:

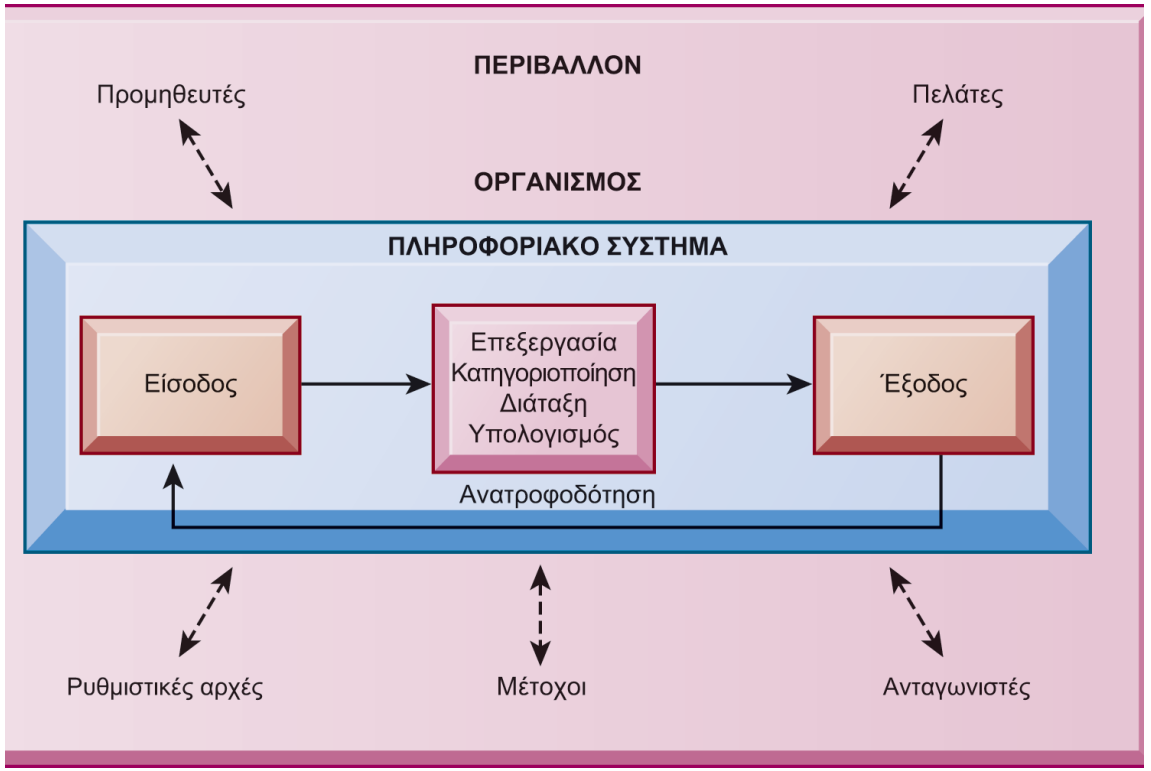
- Υποστήριξη λήψης αποφάσεων και ελέγχου
- Βοήθεια στην ανάλυση, οπτικοποίηση, και δημιουργία προϊόντων



Τα πληροφοριακά συστήματα δεν είναι απλώς τεχνολογία (υλικό, λογισμικό, δίκτυα); Διαθέτουν επίσης διαστάσεις **διοίκησης, οργάνωσης** και **τεχνολογίας**. Συμμετέχουν ενεργά άνθρωποι και οργανισμοί, καθιστώντας τα ένα πολυδιάστατο πεδίο.

ΤΙ ΕΙΝΑΙ ΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ

- Δραστηριότητες σε ένα πληροφοριακό σύστημα που παράγουν πληροφορίες:
  - Είσοδος
  - Επεξεργασία
  - Έξοδος
  - Ανατροφοδότηση
- Σαφής διάκριση μεταξύ **υπολογιστή** ή **προγράμματος υπολογιστή** και **πληροφοριακού συστήματος**



ΚΑΤΗΓΟΡΙΕΣ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΟΥΝΤΑΙ ΑΠΟ ΟΡΓΑΝΙΣΜΟΥΣ

Κατηγορία Πληροφοριακού Συστήματος	Σκοπός	Παραδείγματα Εφαρμογών
Συστήματα Επεξεργασίας Συναλλαγών (TPS)	Καταγραφή και διαχείριση καθημερινών συναλλαγών, όπως αγορές και πληρωμές.	Σύστημα POS (Point of Sale), Τραπεζικά Συστήματα Διαχείρισης Συναλλαγών
Συστήματα Διαχείρισης Πληροφοριών (MIS)	Παροχή αναφορών και πληροφοριών για τη λήψη αποφάσεων σε επίπεδο διαχείρισης.	Συστήματα Αναφορών για Επιχειρήσεις, Διοικητικά Συστήματα Πανεπιστημίων
Συστήματα Υποστήριξης Αποφάσεων (DSS)	Ανάλυση δεδομένων και υποστήριξη πολύπλοκων αποφάσεων.	Συστήματα Ανάλυσης Δεδομένων, Προβλέψεις Αγορών
Συστήματα Γραφείου & Συνεργασίας (OAS)	Διευκόλυνση της επικοινωνίας, της κοινής χρήσης πληροφοριών και της συνεργασίας.	Microsoft Office, Google Workspace, Microsoft Teams
Συστήματα Διαχείρισης Γνώσης (KMS)	Οργάνωση, αποθήκευση και διαμοιρασμός γνώσεων μέσα σε έναν οργανισμό.	Intranet γνώσης, SharePoint, Confluence
Συστήματα Επιχειρηματικής Ευφυΐας (BI)	Ανάλυση δεδομένων και εξαγωγή χρήσιμων επιχειρησιακών πληροφοριών.	Tableau, Power BI, SAS Analytics
Εκτελεστικά Συστήματα Υποστήριξης (ESS)	Υποστήριξη υψηλόβαθμων στελεχών με στρατηγικές πληροφορίες.	SAP Executive Dashboard, IBM Cognos Analytics
Συστήματα Διαχείρισης Πελατειακών Σχέσεων (CRM)	Βελτίωση των σχέσεων με τους πελάτες μέσω διαχείρισης επαφών και δεδομένων.	Salesforce CRM, HubSpot, Zoho CRM
Συστήματα Διαχείρισης Εφοδιαστικής Αλυσίδας (SCM)	Διαχείριση της εφοδιαστικής αλυσίδας για αποδοτικότητα και μείωση κόστους.	SAP SCM, Oracle SCM, Amazon Logistics System
Συστήματα Επιχειρησιακού Προγραμματισμού Πόρων (ERP)	Ολοκληρωμένη διαχείριση επιχειρησιακών πόρων, όπως χρηματοοικονομικά, παραγωγή και HR.	SAP ERP, Oracle ERP, Microsoft Dynamics 365
Εφαρμογές για Κινητά (Mobile Apps)	Παροχή λειτουργιών και υπηρεσιών μέσω κινητών συσκευών για επικοινωνία, αγορές, ψυχαγωγία και παραγωγικότητα.	Instagram, WhatsApp, Google Maps, Spotify, Mobile Banking Apps
Συστήματα Ηλεκτρονικού Εμπορίου (E-Commerce)	Διαχείριση ηλεκτρονικών πωλήσεων, πληρωμών και διανομής προϊόντων ή υπηρεσιών μέσω διαδικτύου.	Amazon, eBay, Shopify, WooCommerce, Skroutz

ΦΥΣΗ ΚΑΙ ΟΡΙΣΜΟΣ ΛΟΓΙΣΜΙΚΟΥ

Το λογισμικό αναφέρεται σε ένα **σύνολο οδηγιών, δεδομένων ή προγραμμάτων** που χρησιμοποιούνται για τη λειτουργία των υπολογιστών και την εκτέλεση συγκεκριμένων εργασιών. Είναι ένα **ζωτικό τμήμα του σύγχρονου υπολογιστή** και είναι αυτό που επιτρέπει στο υλικό (hardware) να εκτελεί διάφορες λειτουργίες. Σε αντίθεση με το φυσικό υλικό, το λογισμικό είναι **ασύλληπτο** και δημιουργείται με τη χρήση γλωσσών προγραμματισμού.

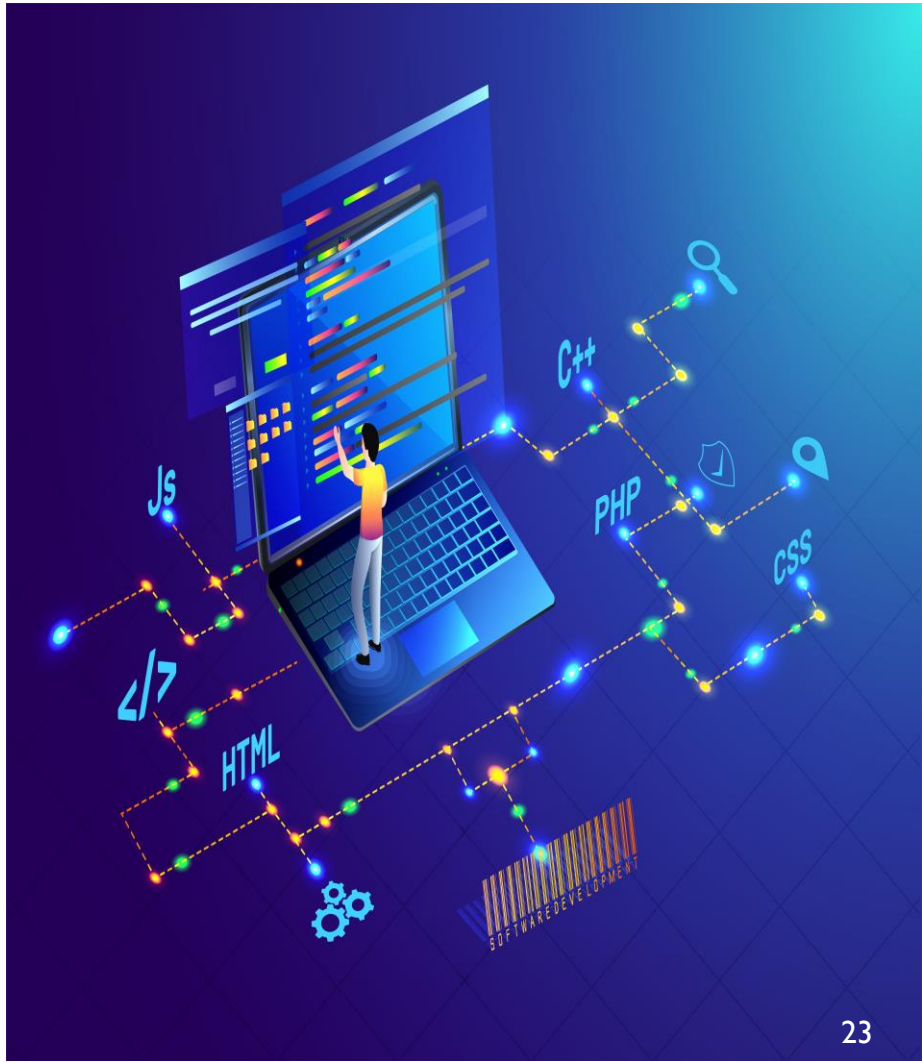
ΚΥΡΙΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΛΟΓΙΣΜΙΚΟΥ

Product characteristic	Description
Συντηρησιμότητα Maintainability	Είναι σημαντικό το λογισμικό να δημιουργείται με τρόπο που να του επιτρέπει να προσαρμόζεται και να αναπτύσσεται ώστε να ικανοποιεί τις μεταβαλλόμενες απαιτήσεις των πελατών του. Η συντηρησιμότητα είναι πολύ βασικό χαρακτηριστικό καθώς η εξέλιξη και η αλλαγή στο λογισμικό αποτελούν αναγκαιότητα σε ένα συνεχώς μεταβαλλόμενο επιχειρηματικό πλαίσιο.
Αξιοπιστία και Ασφάλεια Dependability and security	Η αξιοπιστία του λογισμικού περιλαμβάνει μια σειρά από χαρακτηριστικά, όπως η αξιοπιστία και η ασφάλεια. Ένα αξιόπιστο λογισμικό δεν πρέπει να προκαλεί φυσική ή οικονομική ζημία σε περίπτωση αποτυχίας του συστήματος. Οι κακόβουλοι χρήστες δεν πρέπει να είναι σε θέση να αποκτούν πρόσβαση ή να προκαλούν ζημιά στο σύστημα.
Αποδοτικότητα Efficiency	Το λογισμικό δεν πρέπει να κάνει σπατάλη πόρων του συστήματος όπως η μνήμη και οι κύκλοι του επεξεργαστή. Επομένως, η αποδοτικότητα περιλαμβάνει την ταχύτητα ανταπόκρισης, τον χρόνο επεξεργασίας, την χρήση μνήμης κ.λπ.
Αποδοχή και Χρηστικότητα Acceptability and Usability	Το λογισμικό πρέπει να είναι αποδεκτό από τους τύπους χρηστών για τους οποίους έχει σχεδιαστεί. Αυτό σημαίνει ότι πρέπει να είναι κατανοητό, εύκολο στη χρήση και συμβατό με άλλα συστήματα που το χρησιμοποιούν.

ΤΙ ΕΙΝΑΙ Η ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ

- Πριν από 30 χρόνια, μια έρευνα από έναν δημοσιογράφο έδειξε ότι οι περισσότεροι άνθρωποι πίστευαν πως η τεχνολογία λογισμικού ήταν η συγγραφή προγραμμάτων σε Fortran.

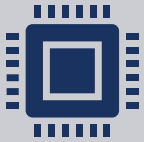
- Σήμερα, μια παρόμοια έρευνα δείχνει ότι οι περισσότεροι άνθρωποι πιστεύουν ότι η τεχνολογία λογισμικού είναι η συγγραφή κώδικα σε Java.



ΤΙ ΝΟΜΙΖΕΤΕ ΕΣΕΙΣ ΟΤΙ ΕΙΝΑΙ Η ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ?

Reply at:
PollEv.com/avgoustakyri977

ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ: ΟΡΙΣΜΟΣ



Κλάδος της πληροφορικής που ασχολείται με **όλες τις πτυχές της παραγωγής λογισμικού** από τα αρχικά στάδια της προδιαγραφής του συστήματος μέχρι τη συντήρηση του συστήματος μετά την ανάπτυξη του.

Engineering discipline

Στοχεύει στην ανάπτυξη αξιόπιστου λογισμικού που ικανοποιεί τις απαιτήσεις των χρηστών και των πελατών. Χρησιμοποιεί κατάλληλες θεωρίες και μεθόδους για την επίλυση προβλημάτων, λαμβάνοντας υπόψη τους οργανωτικούς και οικονομικούς περιορισμούς.

Ασχολείται με όλες τις πτυχές της παραγωγής λογισμικού

Όχι μόνο με την τεχνική διαδικασία ανάπτυξης, αλλά εμβαθύνει στη διαχείριση έργων και στην ανάπτυξη εργαλείων και μεθοδολογιών για την υποστήριξη της ανάπτυξης συστημάτων

ΓΙΑΤΙ ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ

Το λογισμικό επεκτείνεται σε όλους τους τομείς της κοινωνίας μας

Όλο και περισσότεροι άνθρωποι και η κοινωνία γενικά βασίζονται σε προηγμένα συστήματα λογισμικού. Πρέπει να είμαστε σε θέση να παράγουμε αξιόπιστα και ποιοτικά συστήματα οικονομικά και γρήγορα

Τα λογισμικά συστήματα γίνονται όλο και μεγαλύτερα και πιο πολύπλοκα – δεκάδες εκατομμύρια γραμμές κώδικα

Το κόστος του λογισμικού ανέρχεται στο **90 to 95%** του συνολικού κόστους του συστήματος (πριν 2 δεκαετίες το κόστος του λογισμικού ήταν μόνο **5 to 10%** του συνολικού κόστους του συστήματος).

Χρειαζόμαστε μια μηχανική προσέγγιση (engineering approach) στην ανάπτυξη λογισμικού

ΑΝΑΠΤΥΞΗ ΣΥΣΤΗΜΑΤΩΝ

Η ανάπτυξη συστημάτων είναι μια πολύπλοκη δραστηριότητα που απαιτεί:

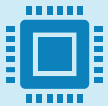
- Κατανόηση των αναγκών και απαιτήσεων των χρηστών από ένα σύστημα.
- Δημιουργία ρεαλιστικών μοντέλων του συστήματος, τα οποία μπορούν να μετατραπούν σε κώδικα λογισμικού.
- Εφαρμογή μιας μεθοδολογίας ανάπτυξης λογισμικού για τον σχεδιασμό, τη διαχείριση και την καθοδήγηση της ανάλυσης, του σχεδιασμού και της ανάπτυξης.
- Ανάπτυξη, δοκιμή, υλοποίηση και συντήρηση του συστήματος.
- Συνεργασία και ομαδική εργασία μεταξύ ατόμων με διαφορετικές δεξιότητες.
- Δέσμευση και υποστήριξη από τη διοίκηση.

Σε κάθε έργο ανάπτυξης που στοχεύει στην παραγωγή χρήσιμων πληροφοριακών συστημάτων, **η κύρια έμφαση δίνεται στις δραστηριότητες ανάλυσης και σχεδιασμού.**

ΑΝΑΓΚΗ ΓΙΑ ΑΝΑΛΥΣΗ ΚΑΙ ΣΧΕΔΙΑΣΜΟ ΣΥΣΤΗΜΑΤΩΝ

Η υλοποίηση ενός συστήματος χωρίς την κατάλληλη σκέψη και σωστό σχεδιασμό οδηγεί σε μεγάλη δυσαρέσκεια των χρηστών και συχνά προκαλεί την εγκατάλειψή του.

Η ανάλυση και ο σχεδιασμός συστημάτων προσδίδουν δομή στην ανάλυση και τον σχεδιασμό των πληροφοριακών συστημάτων. Απαιτεί μια σειρά διαδικασιών που έχουν αναληφθεί συστηματικά για τη βελτίωση μιας επιχείρησης μέσω της χρήσης πληροφοριακών συστημάτων.



Η εμπλοκή των χρηστών καθ' όλη τη διάρκεια ενός έργου συστήματος είναι ζωτικής σημασίας για την επιτυχή ανάπτυξη συστημάτων.

Το λογισμικό συνήθως αναπτύσσεται από ομάδες ανθρώπων που πρέπει να μπορούν να επικοινωνούν και να κατανοούν ο ένας τον άλλον.

Πρέπει να δημιουργηθούν μοντέλα και διαγράμματα του συστήματος

Απαιτείται κοινή γλώσσα και κατανόηση μεταξύ αναλυτών συστημάτων και προγραμματιστών

ΑΝΑΛΥΣΗ ΣΥΣΤΗΜΑΤΩΝ

Είναι μια διαδικασία συλλογής και ερμηνείας δεδομένων, αναγνώρισης προβλημάτων και ανάλυσης ενός συστήματος στα επιμέρους συστατικά του.

- Κατανόηση του προβλήματος
  - *Γιατί υπάρχει ανάγκη για την ανάπτυξη και εισαγωγή ενός νέου συστήματος στον οργανισμό;*
- Καθορισμός και περιγραφή της λειτουργικότητας του συστήματος (Τι πρέπει να κάνει το σύστημα)
  - *Διερεύνηση, ανάλυση και μοντελοποίηση των απαιτήσεων/λειτουργιών του συστήματος*
- **Δημιουργία Λογικού (αφηρημένο) μοντέλου:** *Πτυχές του συστήματος που μπορούν να σχεδιαστούν χωρίς να απαιτείται γνώση των εργαλείων ή των γλωσσών προγραμματισμού που θα χρησιμοποιηθούν για την υλοποίησή του.*

ΣΧΕΔΙΑΣΗ ΣΥΣΤΗΜΑΤΩΝ

Εστιάζει στο πώς θα επιτευχθεί ο στόχος του συστήματος (πώς θα λειτουργήσει το σύστημα)

- καθορίζοντας πώς θα υλοποιηθεί, χωρίς να περιλαμβάνει τη συγγραφή κώδικα.

Επικεντρώνεται στην κατανόηση της λύσης:

- Εργαλεία, περιβάλλοντα και γλώσσες προγραμματισμού που θα χρησιμοποιηθούν.
- Αλληλεπίδραση ανθρώπου-υπολογιστή και σχεδιασμός διεπαφής χρήστη (UI).
- Τα μοντέλα και τα διαγράμματα που δημιουργούνται περιέχουν περισσότερες λεπτομέρειες σχεδίασης, έτοιμες να μετατραπούν σε κώδικα.

Ο ΡΟΛΟΣ ΤΟΥ ΑΝΑΛΥΤΗ ΣΥΣΤΗΜΑΤΩΝ

Ο αναλυτής συστημάτων λειτουργεί ως διαμεσολαβητής μεταξύ των πελατών και της τεχνικής ομάδας.

- Είναι υπεύθυνος για την ενσωμάτωση των επιχειρησιακών απαιτήσεων στην τεχνολογία και τη διασφάλιση της ομαλής λειτουργίας των επιχειρησιακών διαδικασιών.
- Χρησιμοποιεί τόσο επιχειρηματική όσο και τεχνική γνώση για την ανάλυση επιχειρησιακών διαδικασιών, υπολογιστικών συστημάτων και υποδομών, ώστε να αναπτύξουν αποτελεσματικές στρατηγικές που εξυπηρετούν τις καθημερινές ανάγκες του οργανισμού.
- Πρέπει να είναι ικανός να συνεργάζεται με άτομα διαφορετικών προφίλ και να έχει εμπειρία στη χρήση υπολογιστών.
- Συνεργάζεται με προγραμματιστές για την ανάπτυξη του αρχικού λογισμικού, συνεργάζεται με τους χρήστες για την ανάπτυξη της τεκμηρίωσης, σχεδιάζει την ομαλή μετάβαση των χρηστών στο νέο σύστημα

Τρεις βασικοί ρόλοι:

- Σύμβουλος (Consultant)
- Ειδικός Υποστήριξης (Supporting Expert) – Δρα ως μεσολαβητής και αρχιτέκτονας.
- Πράκτορας αλλαγής (Agent of Change)

Ο ΡΟΛΟΣ ΤΟΥ ΑΝΑΛΥΤΗ ΣΥΣΤΗΜΑΤΩΝ

- **Ιδιότητες**
  - Επιδεξιότητα στην επίλυση προβλημάτων
  - Ικανότητα επικοινωνίας
  - Ισχυρή προσωπική και επαγγελματική ηθική
  - Αυτοπειθαρχία και αυτοπαρακίνηση

- **Καθήκοντα**
  - Συλλογή δεδομένων και πληροφοριών
  - Ανάλυση του προβλήματος
  - Καθορισμός των απαιτήσεων των χρηστών
  - Προτεραιοποίηση των απαιτήσεων
  - Μοντελοποίηση απαιτήσεων
  - Σχεδιασμός και αξιολόγηση συστημάτων
  - Στρατηγικές υλοποίησης (ενσωμάτωση του συστήματος στη χρήση) και συντήρησης

ΕΝΣΩΜΑΤΩΣΗ ΤΩΝ ΑΝΑΓΚΩΝ ΑΛΛΗΛΕΠΙΔΡΑΣΗΣ ΑΝΘΡΩΠΟΥ ΥΠΟΛΟΓΙΣΤΉ (HCI)

Η ζήτηση για **αναλυτές συστημάτων** που είναι σε θέση να ενσωματώσουν το **HCI** στη διαδικασία ανάπτυξης συστημάτων συνεχίζει να αυξάνεται, καθώς οι εταιρείες αρχίζουν να συνειδητοποιούν ότι η **ποιότητα των συστημάτων** και η **ποιότητα της εργασιακής ζωής** μπορεί να βελτιωθεί λαμβάνοντας υπόψη μια από την αρχή κιόλας ενός έργου λογισμικού **ανθρωποκεντρική προσέγγιση.**

ΠΩΣ ΘΑ ΜΕΛΕΤΗΣΟΥΜΕ ΤΟ ΜΑΘΗΜΑ ΑΥΤΟ

5Ps

Process	**Διαδικασίες/μεθοδολογίες**
Project	**Έργο**
Product	**Προϊόν**
People	**Άνθρωποι**
Problem	**Πρόβλημα**

5PS

- Εισαγωγή στην τεχνολογία λογισμικού από τη σκοπιά του **'έργου - project', 'προβλήματος- problem', 'διαδικασίες - process', ' προιόντος - product', και 'ανθρώπων- people'**.

- Αντιστεκόμαστε σε οποιοδήποτε στενό πλαίσιο αναφοράς

Επειδή η δημιουργία πληροφοριακών συστημάτων…

- Δεν αφορά μόνο την κατασκευή ενός τεχνικού συστήματος για την ικανοποίηση μιας δεδομένης απαίτησης
- Δεν αφορά μόνο το υλικό (hardware) και το λογισμικό (software)

Αλλά…

- Είναι ένα οργανωτικό φαινόμενο που εντάσσεται στα πλαίσια ενός οργανισμού

ΠΡΕΠΕΙ ΕΠΟΜΕΝΩΣ ΝΑ ΔΙΕΥΡΥΝΟΥΜΕ ΤΟ ΟΡΑΜΑ ΜΑΣ…

Και να δώσουμε σημασία σε 5 βασικές πτυχές

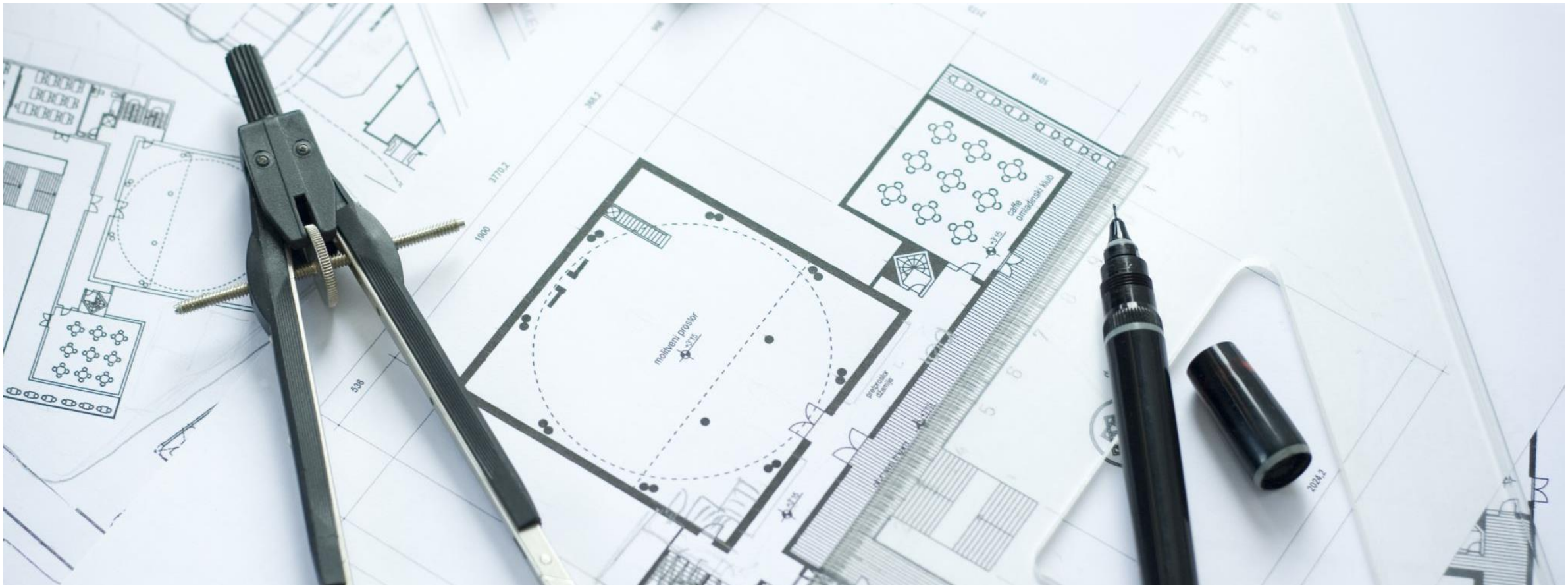
Process/Διαδικασία - Μοντέλα διαδικασίας παραγωγής λογισμικού - πως τα χρησιμοποιούμε

Project – ποιες είναι οι δομές, οι πόροι, ο προϋπολογισμός που θα χρησιμοποιήσουμε για την διαχείριση ανάπτυξης του συστήματος

Product – ποια είναι τα συστήματα που παράγουμε

People - πολλοί άνθρωποι και σε διάφορους ρόλους

Problem – γιατί ξεκινήσαμε με την ανάπτυξη αυτού του συστήματος εξαρχής



THE PROJECT P – ΤΟ ΕΡΓΟ

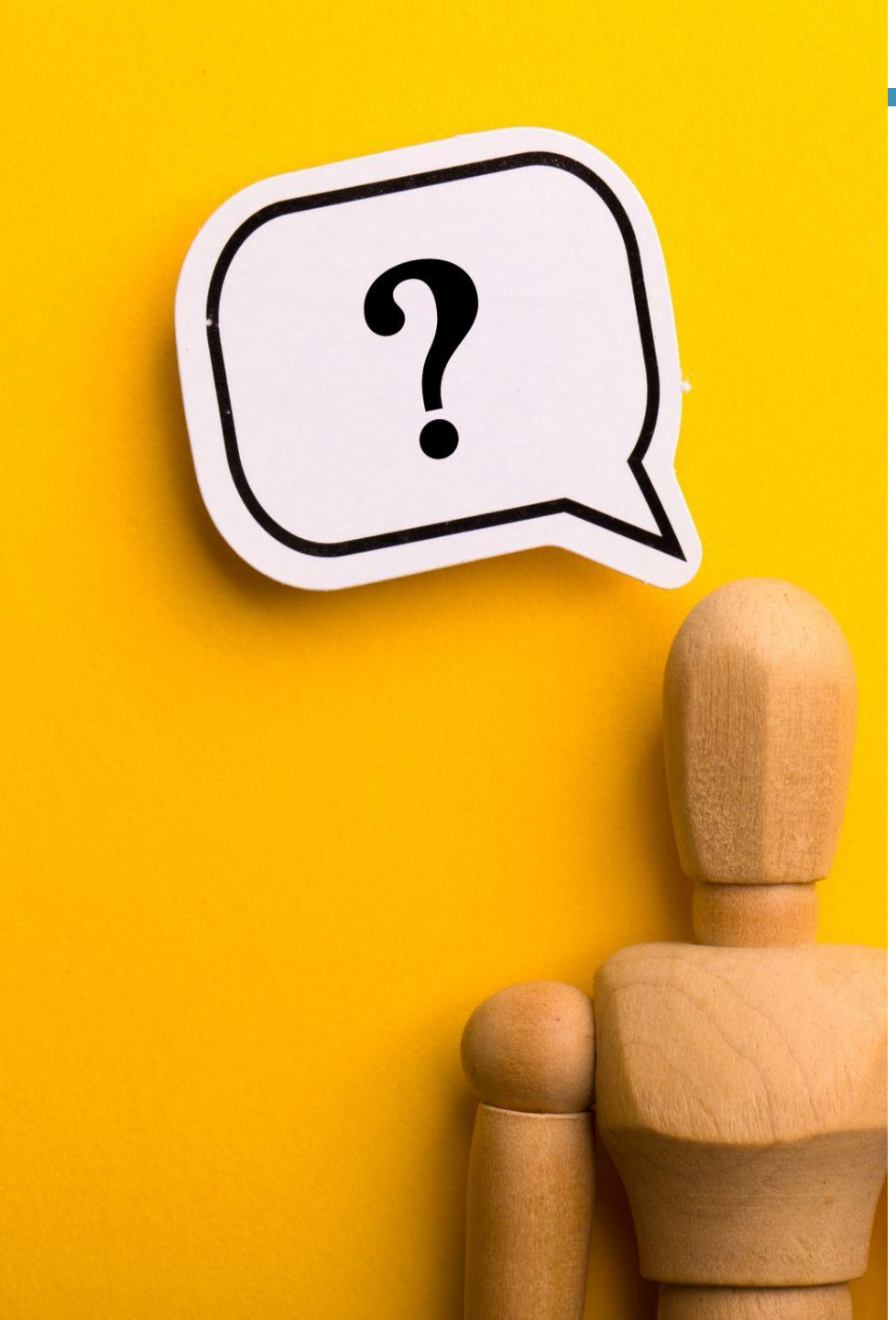
"A planned undertaking"

"A large undertaking"

ΤΙ ΕΙΝΑΙ ΕΝΑ ΕΡΓΟ

"Ένα έργο είναι ένα μοναδικό σύνολο συντονισμένων δραστηριοτήτων, με συγκεκριμένα σημεία έναρξης και λήξης, που αναλαμβάνονται από ένα άτομο ή μια ομάδα για την επίτευξη συγκεκριμένων στόχων εντός καθορισμένου χρόνου, κόστους και παραμέτρων απόδοσης, όπως καθορίζονται στην επιχειρηματική υπόθεση."

(Office of Government Commerce)



ΓΙΑΤΙ ΧΡΕΙΑΖΟΜΑΣΤΕ ΕΡΓΑ?

- Οποιαδήποτε ανάπτυξη συστήματος αποφασίσουμε να αναλάβουμε, χρειάζεται ένα έργο που θα πρέπει να δομηθεί και να δημιουργηθεί γύρω από αυτή την προσπάθεια.
- Το έργο θα καθορίσει στη συνέχεια όλες τις λεπτομέρειες όπως, τους πόρους, τα εργαλεία που θα χρησιμοποιηθούν, τα οφέλη, τους περιορισμούς, τους κινδύνους, το κόστος κ.λπ.

ΠΑΡΑΔΕΙΓΜΑΤΑ ΕΡΓΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ?

ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΩΝ: ΔΡΑΣΤΗΡΙΟΤΗΤΕΣ

Οι δραστηριότητες διαχείρισης έργων περιλαμβάνουν:

- Εκτιμήσεις προσπάθειας, κόστους και χρόνου
- Σχεδιασμός και χρονοπρογραμματισμός έργων
- Διαχείριση κινδύνων
- Διοίκηση έργου, διαχείριση επικοινωνιών και διαχείριση των ενδιαφερομένων μερών
- Διαχείριση ποιότητας

Οι δραστηριότητες αυτές διασφαλίζουν ότι το σύστημα λογισμικού παραδίδεται εγκαίρως και εντός του προϋπολογισμού.

ΤΙ ΠΗΓΕ ΛΑΘΟΣ?

- **Τα έργα πληροφορικής έχουν τρομερό ιστορικό**
- **Μια μελέτη της Standish Group* του 2020 (CHAOS) διαπίστωσε ότι:**
  - Μόνο το 31% των έργων λογισμικού είναι επιτυχημένα, και από αυτά τα επιτυχημένα έργα μόνο το 46% επιστρέφει υψηλή αξία όσον αφορά την επίτευξη των στόχων για το πεδίο εφαρμογής (scope), το χρόνο και το κόστος,
  - Το 66% των τεχνολογικών έργων (με βάση την ανάλυση 50.000 έργων παγκοσμίως) καταλήγουν σε μερική ή ολική αποτυχία. Ενώ τα μεγαλύτερα έργα είναι πιο επιρρεπή στο να αντιμετωπίσουν προκλήσεις ή να αποτύχουν εντελώς, ακόμη και τα μικρότερα έργα λογισμικού αποτυγχάνουν μία στις δέκα φορές. Τα μεγάλα έργα είναι επιτυχή σε λιγότερο από το 10% των περιπτώσεων.
  - Το 31% των αμερικανικών έργων πληροφορικής ακυρώθηκε εντελώς και η απόδοση του 53% "ήταν τόσο ανησυχητική που αμφισβητήθηκε".

**Η Standish Group είναι μια ανεξάρτητη διεθνής συμβουλευτική εταιρεία ερευνών πληροφορικής που ιδρύθηκε το 1985, γνωστή από τις εκθέσεις της σχετικά με έργα υλοποίησης πληροφοριακών συστημάτων στο δημόσιο και ιδιωτικό τομέα.*

ΤΙ ΠΗΓΕ ΛΑΘΟΣ?

- Μια μελέτη της PricewaterhouseCoopers διαπίστωσε ότι πάνω από το ήμισυ όλων των έργων αποτυγχάνουν και μόνο το **2,5% των επιχειρήσεων επιτυγχάνουν σταθερά τους στόχους** τους όσον αφορά το πεδίο εφαρμογής, το χρόνο και το κόστος για όλους τους τύπους έργων.
- Τα έργα λογισμικού και υλικού αποτυγχάνουν σε ποσοστό 65%.
- **Πάνω από τα μισά** από όλα τα έργα πληροφορικής γίνονται ανεξέλεγκτα.
- **Μόνο το 30% των έργων πληροφορικής είναι επιτυχημένα**.
- **Ένα στα έξι έργα** πληροφορικής έχει μέση υπέρβαση κόστους 200% και υπέρβαση χρονοδιαγράμματος 70%.
- Δέκα μεγάλες κυβερνητικές συμβάσεις έχουν υπέρβαση κόστους άνω των 16 δισεκατομμυρίων δολαρίων και είναι συνολικά 38 χρόνια πίσω από το χρονοδιάγραμμα.

ΓΙΑΤΙ ΝΟΜΙΖΕΤΕ ΟΤΙ Η ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΩΝ ΕΙΝΑΙ ΔΥΣΚΟΛΗ?

Reply at:
PollEv.com/avgoustakyri977

ΠΑΡΑΓΟΝΤΕΣ ΑΠΟΤΥΧΙΑΣ ΣΤΗ ΔΙΑΧΕΙΡΙΣΗ ΕΡΓΩΝ

- Σχέσεις με τον πελάτη
- Συμβάσεις και νομικές συμφωνίες
- Πολιτική και συγκρούσεις
- Μειωμένη κερδοφορία
- Μη ρεαλιστικοί στόχοι
- Κακή επικοινωνία
- Ανταγωνιστικό μειονέκτημα
- Δυσαρέσκεια πελατών
- Αντιλαμβανόμενη αξία του έργου
- Μη κατανόηση απαιτήσεων

- Λάθος διαχειριστής έργου
- Ανεπαρκές σκεπτικό, σκοποί, καθήκοντα και στόχοι
- Μη υποστηρικτική ανώτατη διοίκηση
- Έλλειψη ή κακή χρήση τεχνικών διαχείρισης έργων
- Ανεπαρκής σχεδιασμός του έργου
- Έλλειψη δέσμευσης για το έργο
- Τα ενδιαφερόμενα μέρη είναι ποικίλα και δεν εντοπίζονται πάντα εύκολα
- Πολλά έργα λογισμικού είναι νέα και καινοτόμα και συνεπώς δεν μπορούμε να επωφεληθούμε από την εμπειρία του παρελθόντος
- Τα έργα συχνά έχουν αντίκτυπο πέρα από τα παραδοσιακά οργανωτικά όρια

ΕΙΝΑΙ ΤΑ ΕΡΓΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΠΡΑΓΜΑΤΙΚΑ ΤΟΣΟ ΔΙΑΦΟΡΕΤΙΚΑ ΑΠΟ ΑΛΛΑ ΕΙΔΗ ΕΡΓΩΝ?

Αορατότητα

το λογισμικό δεν είναι τόσο ορατό όσο μια γέφυρα (εντός χρόνου και προϋπολογισμού λόγω της ακραίας λεπτομέρειας στο σχεδιασμό, παγωμένος σχεδιασμός με μικρή ευελιξία στις μεταβαλλόμενες απαιτήσεις)

Πολυπλοκότητα

Περισσότερη πολυπλοκότητα ανά δολάριο από ό,τι άλλες τεχνικές

Συμμόρφωση

τα συστήματα πρέπει να συμμορφώνονται με τις πολλές και συχνά παράλογες απαιτήσεις των οργανισμών-πελατών

Ευελιξία

Επειδή είναι πολύ εύκολο να αλλάξει το λογισμικό, θα υπόκειται σε μεγάλο βαθμό αλλαγών

Η πρωταρχική πρόκληση της διαχείρισης της ανάπτυξης λογισμικού είναι **η σταδιακή ελαχιστοποίηση του κινδύνου και η μεγιστοποίηση της παραγωγικότητας, της ποιότητας και της ανταγωνιστικότητας**. Αυτό καθιστά την κατασκευή συστημάτων ΤΠ και λογισμικού πιο προβληματική από άλλα είδη έργων

THE PEOPLE P
ΟΙ ΑΝΘΡΩΠΟΙ

Ενδιαφερόμενα μέρη, χρήστες, πελάτες και άλλοι…

ΟΙ ΑΝΘΡΩΠΟΙ

- Όσοι έχουν κάποιο ενδιαφέρον για την ανάπτυξη και τη μορφή που θα πάρει το τελικό προϊόν.
- Διάφορα ενδιαφερόμενα μέρη, οι αναλυτές που σχεδιάζουν το σύστημα, η ομάδα ανάπτυξης κ.λπ.
- **Ενδιαφερόμενα μέρη**
  - άτομα τα οποία ενδιαφέρονται για ένα υπάρχον ή νέο σύστημα
  - μπορεί να είναι τεχνικοί ή μη τεχνικοί εργαζόμενοι
  - μπορεί να περιλαμβάνονται εξωτερικοί και εσωτερικοί εργαζόμενοι
- **Διάφοροι τρόποι ονομασίας τους**

Χρήστες… ή Συμμετέχοντες…

Πελάτες… Μέλη

THE PROBLEM P
ΤΟ ΠΡΟΒΛΗΜΑ

Πως ξεκίνησαν όλα?

ΤΕΛΟΣ...ΤΟ ΠΡΟΒΛΗΜΑ

- **Γιατί αρχίσαμε να μιλάμε για την ανάγκη ενός συστήματος;**

  Για να πετύχουμε κάτι νέο, καλύτερο, διαφορετικό, αλλά τι;

- **Τι είναι το πρόβλημα;**

  "Μια κατάσταση στην οποία κάποιος μπορεί να επιθυμεί να δράσει".'

Πρέπει να εντοπίσουμε το πρόβλημα, να κατανοήσουμε το πρόβλημα και να το σχεδιάσουμε πριν προχωρήσουμε στη λύση.

ΓΙΑΤΙ ΠΡΕΠΕΙ ΝΑ ΚΑΤΑΝΟΗΣΟΥΜΕ ΤΟ ΠΡΟΒΛΗΜΑ?

- Οι οργανισμοί είναι λίγο ακατάστατοι και ασυνεπείς.
- **Το "ανθρώπινο" στοιχείο αποτελεί συχνά το πρόβλημα και την αιτία αποτυχίας** στην ανάλυση και την εφαρμογή πληροφοριακών συστημάτων.

Μπορούμε να κατασκευάσουμε νέο σύστημα και να προσθέσουμε τεχνολογία σε αυτή την ακατάστατη κατάσταση και να περιμένουμε καλά αποτελέσματα;

- Χρειαζόμαστε κάποια δραστηριότητα για να φέρουμε στο επίκεντρο το πρόβλημα (τα προβλήματα) ή την προβληματική κατάσταση
- Να ανακαλύψουμε τι είδους συστήματα μπορούμε να κατασκευάσουμε για να μετριάσουμε τα προβλήματα.

ΠΩΣ ΜΟΝΤΕΛΟΠΟΙΟΥΜΕ ΠΡΟΒΛΗΜΑΤΑ?

Μια απάντηση μας δίδεται από τον Peter Checkland και την **Soft Systems Methodology (SSM)**

THE PRODUCT P
ΤΟ ΠΡΟΪΟΝ

Το παραδοτέο ενός έργου

ΤΟ ΠΡΟΪΟΝ

Το προϊόν περιλαμβάνει:

- Το συγκεκριμένο υλικό και λογισμικό, το οποίο πρόκειται να αναπτυχθεί και να παραδοθεί στους πελάτες.

- ***Ένα πρόγραμμα, μια βάση δεδομένων, ένας ιστότοπος, μια εφαρμογή για κινητά, ένα εργαλείο τεχνητής νοημοσύνης κ.λπ.***

- Την τεκμηρίωση, τους πόρους δεδομένων, το εκπαιδευτικό υλικό, τη γνώση και την εμπειρογνωμοσύνη, τα εργαλεία όπως τα ολοκληρωμένα περιβάλλοντα, τα εργαλεία δοκιμών και τα εργαλεία σχεδιασμού έργων που χρησιμοποιούνται για τη δημιουργία λύσεων για τους ενδιαφερόμενους φορείς

ΕΡΓΑΛΕΙΑ CASE (COMPUTER-AIDED SOFTWARE ENGINEERING)

- Εργαλεία παραγωγικότητας για αναλυτές συστημάτων που έχουν δημιουργηθεί για να βελτιώσουν την καθημερινή τους εργασία μέσω της χρήσης μιας αυτοματοποιημένης υποστήριξης

Μπορούν να χωριστούν σε δύο γενικές ομάδες:

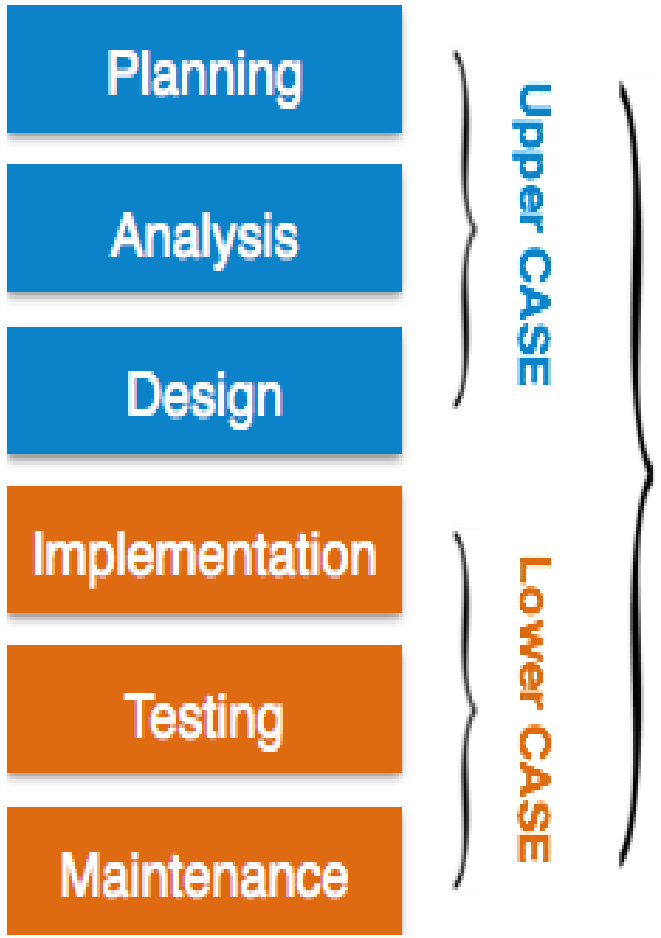
- **Upper-CASE**

  Εργαλεία που υποστηρίζουν τα αρχικά στάδια ανάπτυξης (απαιτήσεις και σχεδίαση, μοντελοποίηση)
  - IBM Rational Architect, Visio, Visual Paradigm, LucidChart

- **Lower-CASE**

  Εργαλεία που υποστηρίζουν τα επόμενα στάδια (π.χ., προγραμματισμό, αποσφαλμάτωση / debugging, έλεγχο)

– Παρέχουν την δυνατότητα παραγωγής πηγαίου κώδικα από τον σχεδιασμό CASE
– Ο πηγαίος κώδικας παράγεται συνήθως σε πολλές γλώσσες
– Μειώνουν το χρόνο συντήρησης
– Βοηθούν στη δημιουργία κώδικα χωρίς σφάλματα



ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΠΡΟΪΌΝΤΟΣ

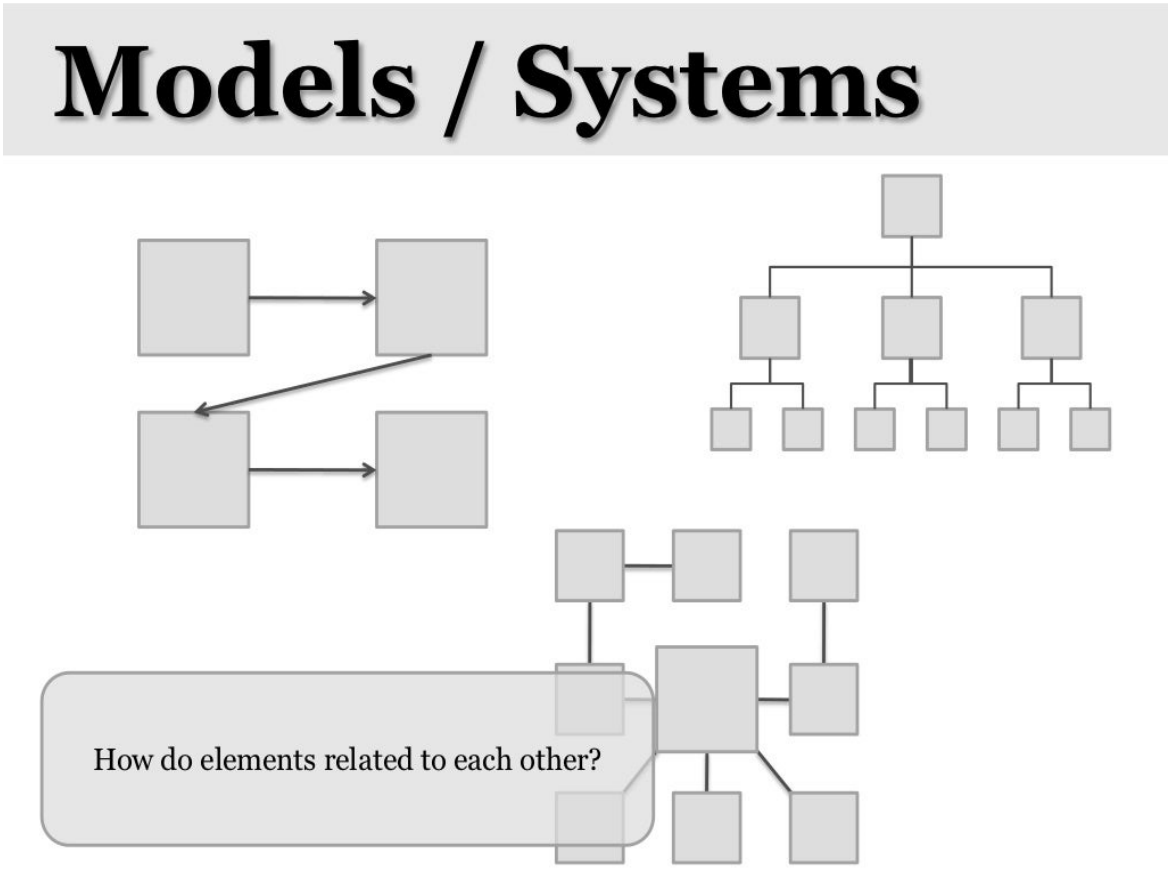
Η μοντελοποίηση αποτελεί βασικό μέρος της ανάλυση, σχεδίασης και ανάπτυξης πληροφοριακών συστημάτων

- Σε κάθε έργο ανάπτυξης πληροφοριακών συστημάτων, η **ανάλυση και ο σχεδιασμός** επικεντρώνονται στη δημιουργία **μοντέλων και διαγραμμάτων** του προτεινόμενου συστήματος.
- Όπως και στην κατασκευή αεροσκαφών, και άλλων κατασκευαστικών έργων, όπου χρησιμοποιούνται μοντέλα κλίμακας πριν την υλοποίηση.
- Το λογισμικό **δεν είναι ορατό (δεν μπορούμε να το αγγίξουμε)**
  - Αλλά οι ομάδες ανάπτυξης χρειάζονται **οπτικές αναπαραστάσεις** για να μπορούν να συνεργαστούν και να συντονίζονται.
  - Ακόμη και ένας μεμονωμένος προγραμματιστής ωφελείται από ορατά μοντέλα.
- Τα **μοντέλα βοηθούν** στην καλύτερη κατανόηση του συστήματος και στη μετατροπή τους σε κώδικα.
  - Είναι καλύτερο οι προγραμματιστές να κατανοήσουν το σύστημα μέσω των μοντέλων και μετά να προχωρήσουν στη δημιουργία κώδικα.

ΓΙΑΤΙ ΠΡΕΠΕΙ ΝΑ ΜΟΝΤΕΛΟΠΟΙΟΥΜΕ ΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ?

- Τα πληροφοριακά συστήματα είναι **πολύπλοκα** και πρέπει να ενσωματώνονται με άλλα συστήματα.
- Η ανάπτυξη λογισμικού είναι **αναπόσπαστο μέρος της στρατηγικής μιας επιχείρησης** και δεν πρέπει να υποτιμάται.
- Η **μοντελοποίηση συστημάτων** βοηθά στη **διαχείριση της πολυπλοκότητας** και στη **μείωση του ρίσκου,** προβλέποντας διαφορετικές συμπεριφορές.
  - Επιτρέπει την **οπτικοποίηση ολόκληρων συστημάτων,** την **αξιολόγηση επιλογών** και την **καλύτερη επικοινωνία του σχεδιασμού** πριν από την ανάπτυξη.
  - Έμπειροι αναλυτές, σχεδιαστές και προγραμματιστές **επενδύουν περισσότερο χρόνο** στη δημιουργία μοντέλων παρά στη συγγραφή κώδικα.
- **Καλά σχεδιασμένα μοντέλα** διευκολύνουν την έγκαιρη και εντός προϋπολογισμού **παράδοση μεγάλων, πολύπλοκων επιχειρησιακών συστημάτων.**

ΤΙ ΕΙΝΑΙ ΕΝΑ ΜΟΝΤΕΛΟ



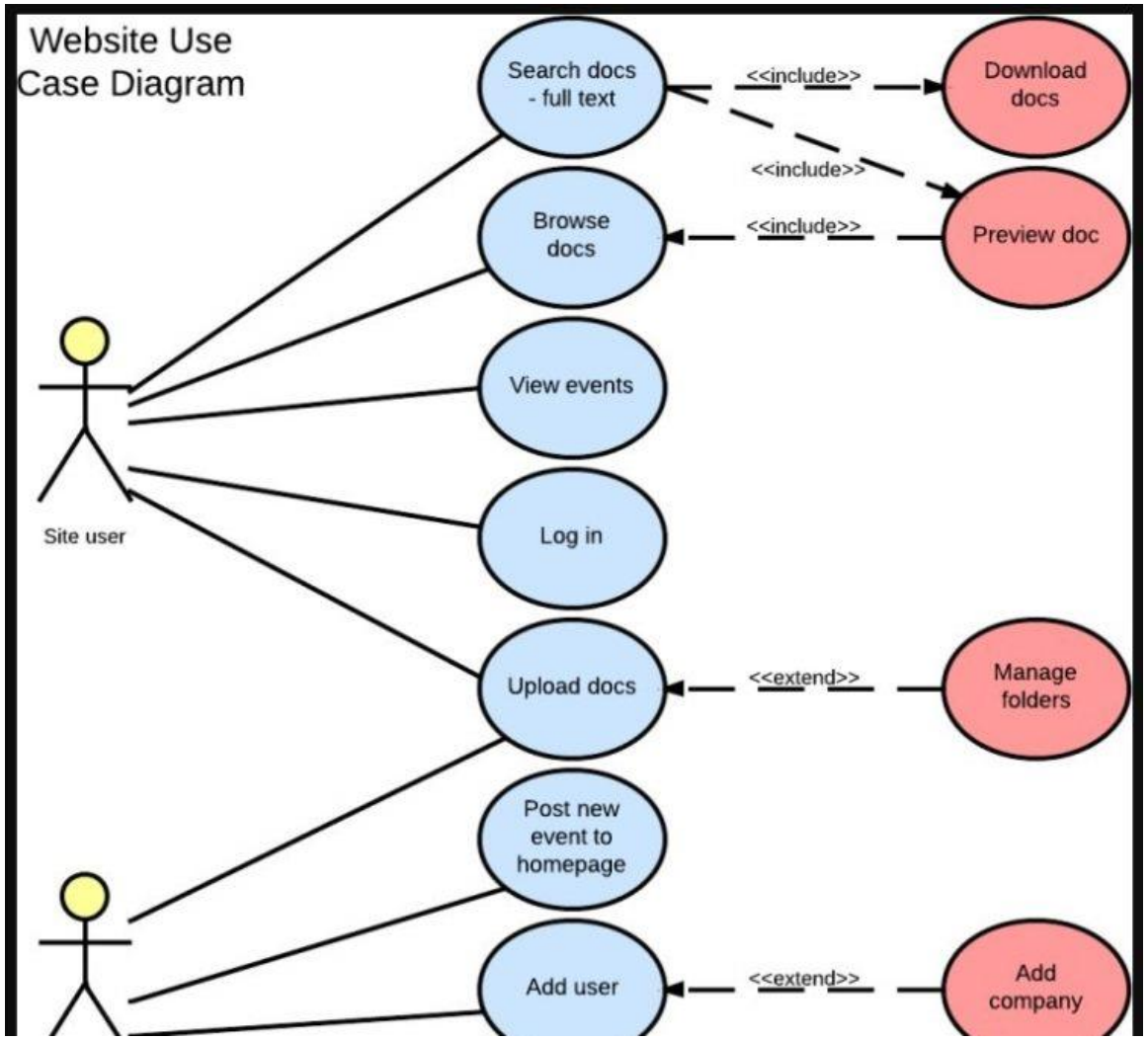
- Μια αφηρημένη αναπαράσταση κάποιου πράγματος. Όπως οι χάρτες τα μοντέλα αναπαριστούν κάτι άλλο.
- Μια προσέγγιση του πραγματικού αντικειμένου που κατασκευάζεται.

Ένα χρήσιμο μοντέλο:

- έχει τη σωστή ποσότητα λεπτομέρειας και δομής και αναπαριστά μόνο ό,τι είναι σημαντικό για την εκάστοτε εργασία
- είναι πιο γρήγορο και εύκολο στην κατασκευή
- μπορεί να εξελίσσεται καθώς μαθαίνουμε περισσότερα για την εργασία ή το πρόβλημα

Τα περισσότερα μοντέλα ανάπτυξης συστημάτων διατηρούνται ως δεδομένα σε εργαλεία μοντελοποίησης, και πολλά από αυτά τα δεδομένα αναπαρίστανται οπτικά με τη μορφή ΔΙΑΓΡΑΜΜΑΤΩΝ, με υποστηρικτικές περιγραφές κειμένου.

ΤΙ ΕΙΝΑΙ ΤΟ ΔΙΑΓΡΑΜΜΑ?



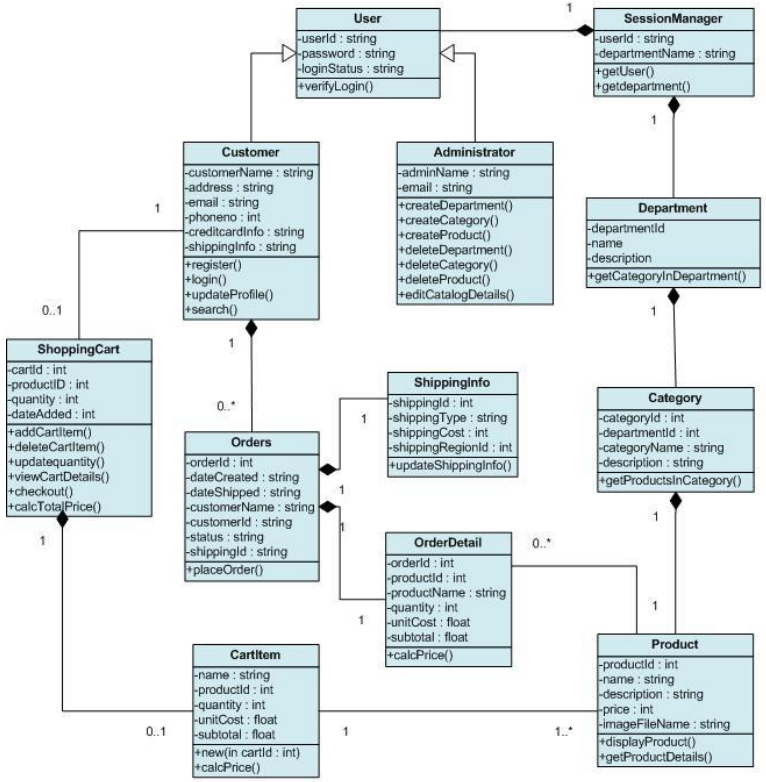
- Οπτική αναπαράσταση κάποιου τμήματος ενός μοντέλου.
- Οι αναλυτές και οι σχεδιαστές χρησιμοποιούν διαγράμματα για την απεικόνιση μοντέλων συστημάτων με τον ίδιο τρόπο που οι αρχιτέκτονες χρησιμοποιούν σχέδια και διαγράμματα για τη μοντελοποίηση κτιρίων.
- Για παράδειγμα, ένα **μοντέλο απαιτήσεων ενός συστήματος** θα δώσει μια πλήρη εικόνα των απαιτήσεων του εν λόγω συστήματος.
  - Μπορεί να χρησιμοποιεί έναν ή περισσότερους τύπους διαγραμμάτων προκειμένου να καλύψει όλες τις πτυχές των απαιτήσεων, π.χ. διαγράμματα περιπτώσεων χρήσης (use case diagrams), διαγράμματα δραστηριοτήτων (activity diagrams) κ.λπ.

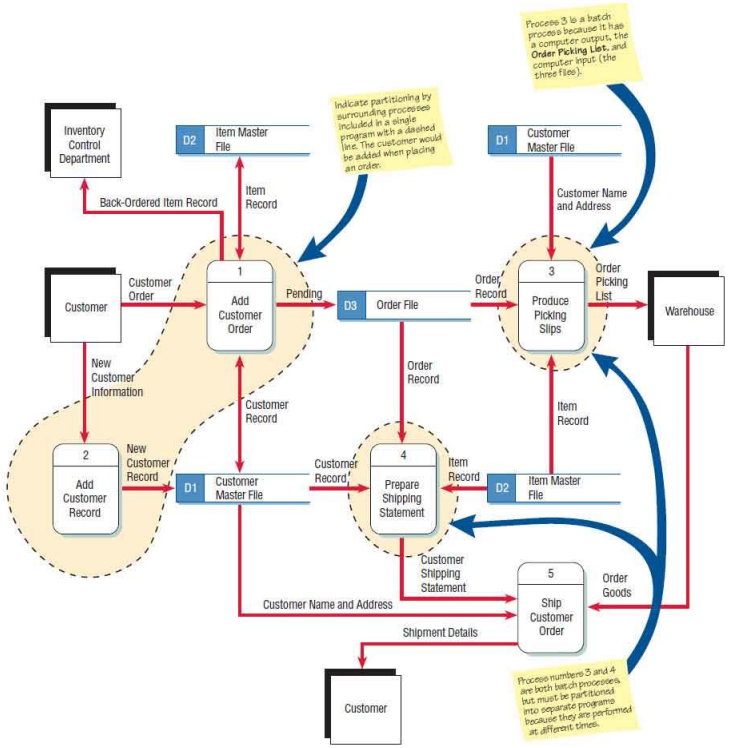
ΤΙ ΕΙΝΑΙ ΤΟ ΔΙΑΓΡΑΜΜΑ

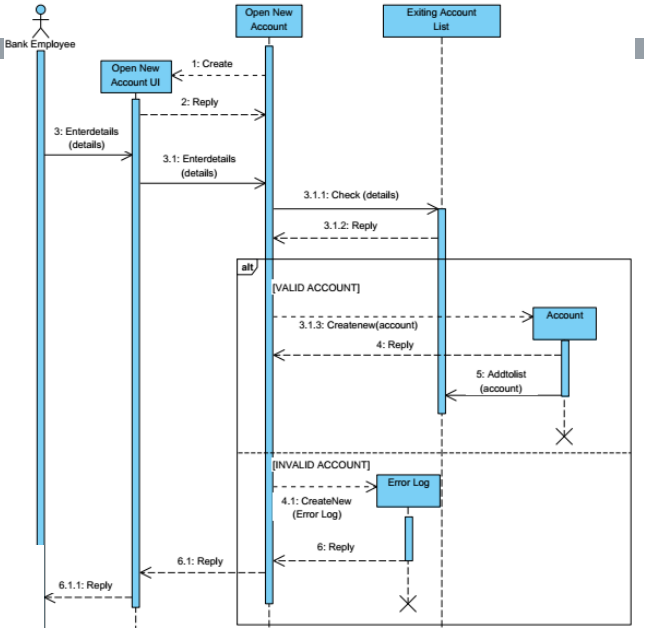
- Ένα **μεμονωμένο διάγραμμα** μπορεί να απεικονίσει ή να τεκμηριώσει μια συγκεκριμένη πτυχή ενός συστήματος.
- Ένα **μοντέλο**, όμως, παρέχει **πλήρη εικόνα** του συστήματος σε μια συγκεκριμένη φάση και από μια συγκεκριμένη οπτική.

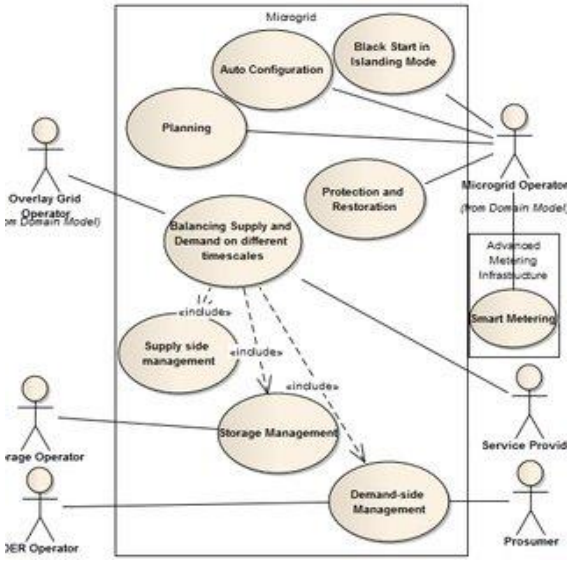
- **ΠΑΡΑΔΕΙΓΜΑ**
  - Ένα **UML activity diagram** μπορεί να δείχνει μέρος της διαδικασίας παραγωγής ενός βιβλίου.
  - Όμως, αυτό **δεν αποτελεί πλήρες μοντέλο**.
  - Ένα πλήρες μοντέλο παραγωγής βιβλίου θα πρέπει να περιλαμβάνει και άλλα διαγράμματα, όπως τη **διαπραγμάτευση συμβολαίων** και το **μάρκετινγκ** του βιβλίου.

ΕΙΔΗ ΔΙΑΓΡΑΜΜΑΤΩΝ









ΤΙ ΜΟΝΤΕΛΟΠΟΙΟΥΜΕ ΣΥΝΗΘΩΣ ΣΤΟ ΠΕΡΙΒΑΛΛΟΝ ΕΝΟΣ ΣΥΣΤΗΜΑΤΟΣ;

Reply at:
PollEv.com/avgoustakyri977

1. Go to **PollEv.com**
2. Enter **AVGOUSTAKYRI977**
3. Respond to activity

1. Text **AVGOUSTAKYRI977** to **22333**
2. Text in your message

ΤΟ ΒΑΣΙΚΟ ΚΟΙΝΟ ΠΡΟΒΛΗΜΑ – ΤΙ ΚΑΤΑΛΑΒΑΙΝΕΙ Ο ΚΑΘΕΝΑΣ ΣΤΗΝ ΟΜΑΔΑ?

as proposed in project proposal;

as the analyst described it;

as it was designed;

as it was implemented;

as it was installed;

what was expected!!!

ΠΩΣ ΣΧΕΔΙΑΖΟΥΜΕ/ΜΟΝΤΕΛΟΠΟΙΟΥΜΕ ΤΟ ΠΡΟΙΟΝ?

- **Δύο (2) βασικά παραδείγματα στην Τεχνολογία Λογισμικού με τα σχετικά εργαλεία, μοντέλα και διαγράμματα**
  - Δομημένη ανάλυση και σχεδιασμός συστημάτων - Structured Systems Analysis and Design
  - Αντικειμενοστρεφής ανάλυση και σχεδιασμός- **Object Oriented Analysis and Design**

ΠΩΣ ΣΧΕΔΙΑΖΟΥΜΕ/ΜΟΝΤΕΛΟΠΟΙΟΥΜΕ ΤΟ ΠΡΟΙΟΝ?

Συμβατικές απαντήσεις από την SSAD

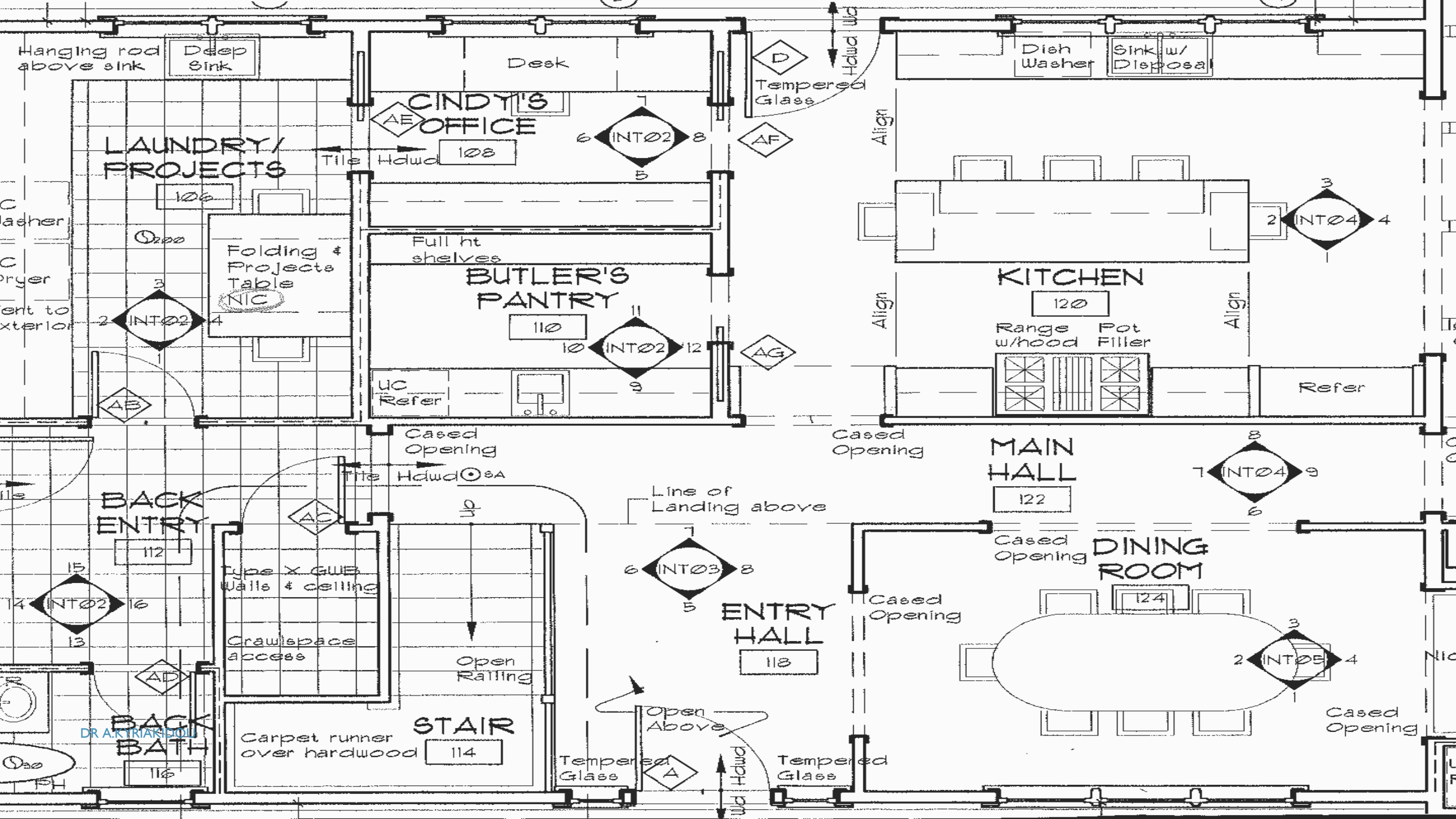
- Μοντέλο διαδικασίας Process model - το διάγραμμα ροής δεδομένων και το minispec
- Μοντέλο δεδομένων Data model - διαγράμματα οντοτήτων - συσχετίσεων
- Μοντέλο συμπεριφοράς - entity life history

Each model taken separately and with an orderly transition (decomposition)

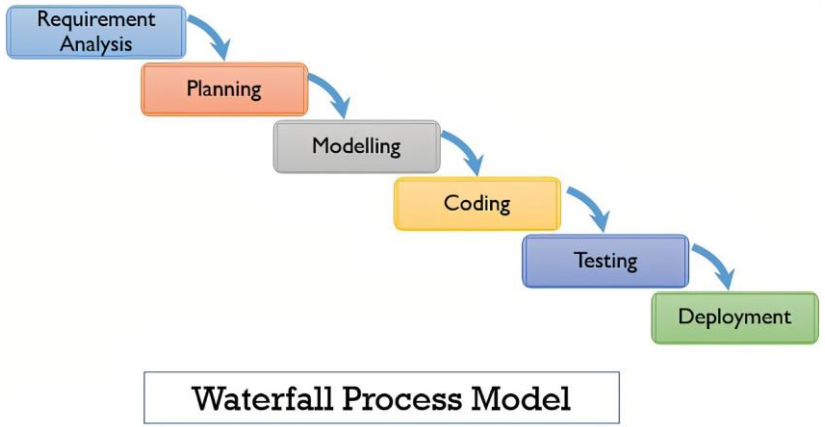
Πιο μοντέρνες απαντήσεις από το OOAD

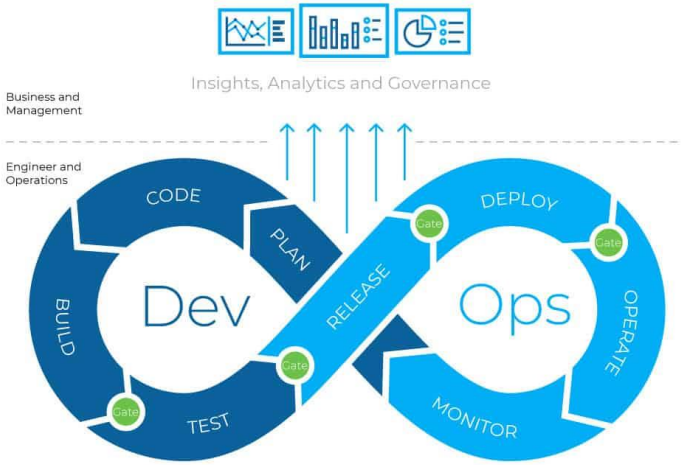
- Μοντέλο περίπτωσης χρήσης – Use case diagrams
- Μοντέλο Αντικειμένων (Object model) – Class diagrams
- Δυναμικό Μοντέλο – Object sequence diagrams, State chart diagrams
- Διάγραμμα Κλάσεων Σχεδιασμού - Design Class Diagram
- Περιπτώσεις χρήσης, κλάσεις και αντικείμενα, χαρακτηριστικά, μεταβίβαση μηνυμάτων και συμπεριφορά. Use Cases, Classes and Objects, attributes, message passing and behaviour

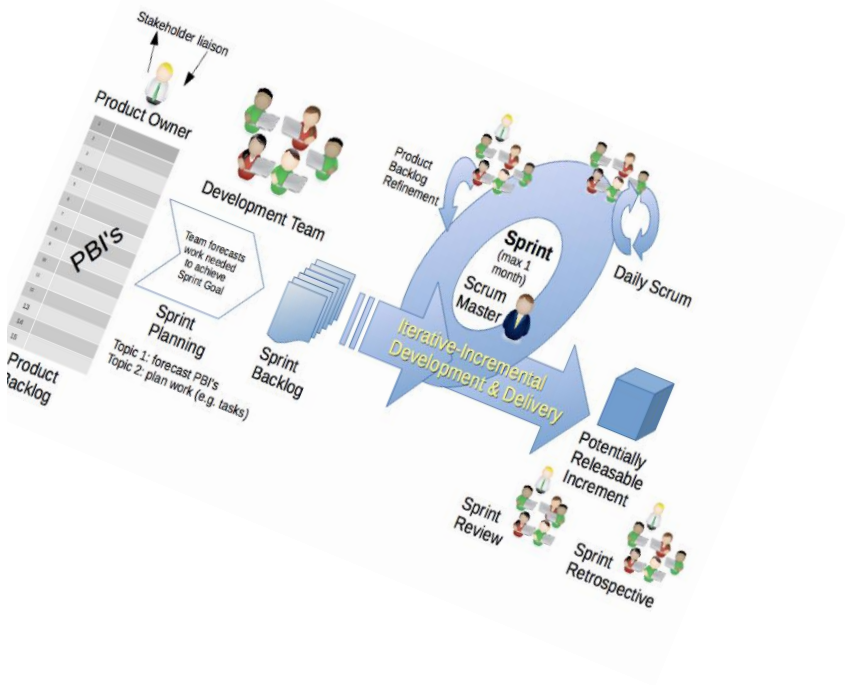
  όλα σε ένα ολοκληρωμένο μοντέλο, η μοντελοποίηση γίνεται με επαναληπτικό και σταδιακό τρόπο, υποστηρίζοντας την αφαίρετικότητα (abstraction) και όχι την αποσύνθεση(***decomposition)***



THE PROCESS P – ΜΕΘΟΔΟΛΟΓΙΕΣ ΚΑΙ ΔΙΑΔΙΚΑΣΙΕΣ





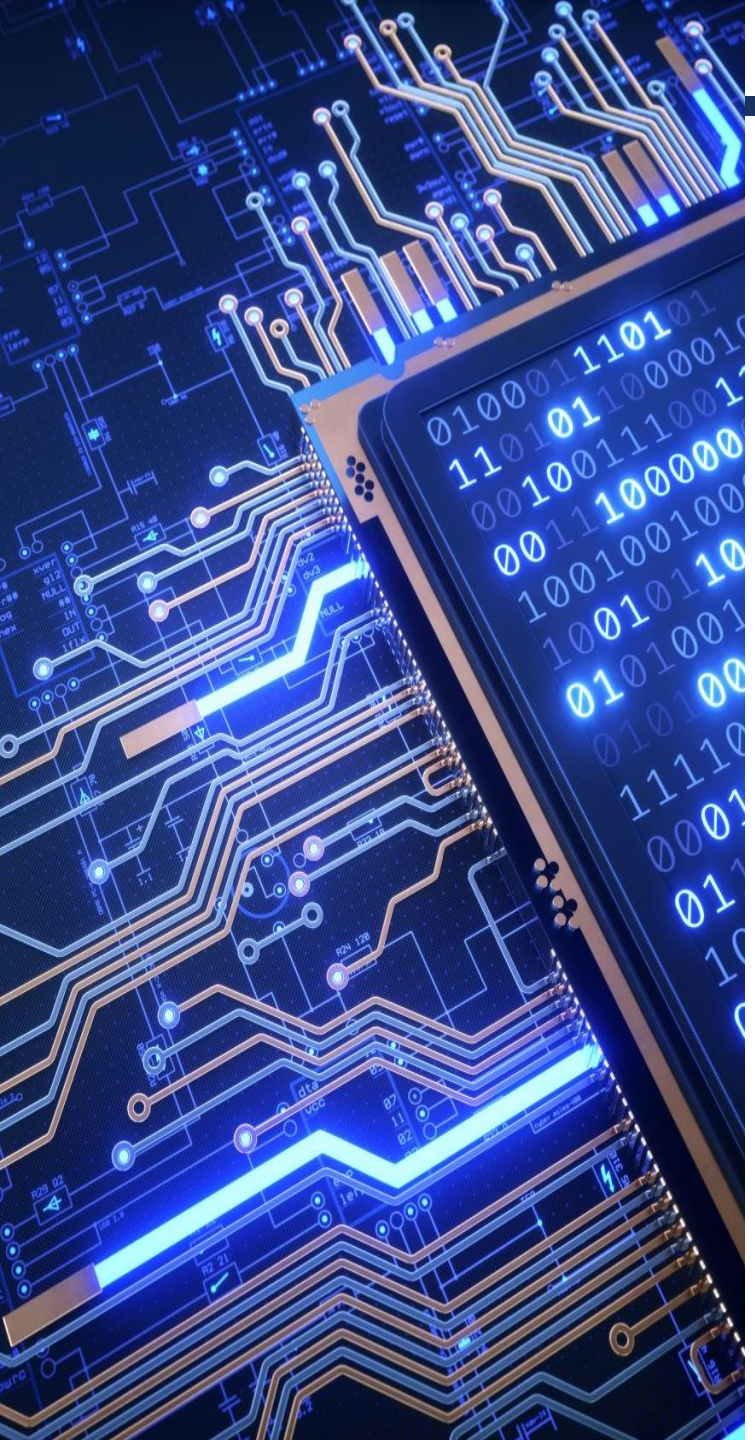


Μεθοδολογικές διαδικασίες που ακολουθούν οι μηχανικοί λογισμικού προκειμένου να προχωρήσουν από μια ιδέα η ένα πρόβλημα σε ένα πλήρες λογισμικό που θα χρησιμοποιηθεί από τους χρήστες

THE PROCESS P

- Κάθε ανάπτυξη πληροφοριακών συστημάτων πρέπει να ακολουθεί κάποιου είδους μεθοδολογία για να καθοδηγείται σχετικά με τον τρόπο ανάπτυξης του συστήματος, ώστε να διασφαλίζεται ότι το **σύστημα θα παραδοθεί εγκαίρως**, **εντός του προϋπολογισμού** και θα **ικανοποιεί τις απαιτήσεις του πελάτη**.

- **Μεθοδολογίες αποσκοπούν στην υποστήριξη της ανάπτυξης ενός συστήματος με επίκεντρο τους πελάτες και τη διασφάλιση της επιτυχίας.**
  - καλύτερα τελικά προϊόντα που ανταποκρίνονται στις απαιτήσεις των χρηστών
  - καλύτερη διαδικασία ανάπτυξης που βελτιώνει τον έλεγχο της ανάπτυξης και την παραγωγικότητα
  - χρήση κοινής προσέγγισης στον οργανισμό, προϋποθέτει κάποια δομή στην ανάπτυξη
  - Εύκολες να διδαχθούν και χρησιμοποιηθούν
  - Μπορούν να ελεγχθούν ως προς την ποιότητα, δεδομένου ότι απαιτούν ρητά παραδοτέα
  - Ενισχύουν την επικοινωνία μεταξύ των μελών στη διαδικασία ανάπτυξης.



ΔΙΑΔΙΚΑΣΙΑ/ΜΕΘΟΔΟΛΟΓΙΑ ΑΝΑΠΤΥΞΗΣ ΛΟΓΙΣΜΙΚΟΥ: ΟΡΙΣΜΟΣ

- Τυπική βηματική περιγραφή δραστηριοτήτων για την παραγωγή ενός συστήματος.
  - Συνδυάζεται με εργαλεία, βοηθήματα τεκμηρίωσης (documentation), κανόνες, σχεδιαστικές συμβουλές τα οποία βοηθούν τους προγραμματιστές συστημάτων να σχεδιάσουν και να υλοποιήσουν ένα σύστημα.

- Αποτελούνται από φάσεις (phases), οι οποίες οι ίδιες αποτελούνται από υπο-φάσεις και έτσι οι software developers είναι σε θέση να επιλέξουν ποια διαδικασία να χρησιμοποιήσουν στην ανάπτυξη του συστήματος τους.
  - Κάποιες διαδικασίες είναι πιο ευέλικτες και προσαρμόσιμες ενώ κάποιες άλλες δίνουν πιο πολλή έμφαση στον σχεδιασμό και δημιουργία λεπτομερούς πλάνου.
  - **Plan-based vs Agile processes**

- **Βοηθούν τους developers να σχεδιάζουν, διαχειρίζονται, να ελέγχουν και να αξιολογούν τα έργα ανάπτυξης λογισμικού**.

ΔΙΑΔΙΚΑΣΙΑ/ΜΕΘΟΔΟΛΟΓΙΑ ΑΝΑΠΤΥΞΗΣ ΛΟΓΙΣΜΙΚΟΥ: **ΟΡΙΣΜΟΣ**

Ένα πλαίσιο που χρησιμοποιείται για την διάρθρωση, το σχεδιασμό (planning) και τον έλεγχο της ανάπτυξης πληροφοριακών συστημάτων.

Ένα σύνολο δραστηριοτήτων (φάσεων) με στόχο την ανάπτυξη ή εξέλιξη ενός προϊόντος λογισμικού

4 θεμελιώδεις δραστηριότητες που είναι κοινές σε όλες τις διαδικασίες ανάπτυξης συστημάτων.

Προδιαγραφές Λογισμικού Τι πρέπει να κάνει το σύστημα

Ανάπτυξη λογισμικού - Καθορισμός της οργάνωσης του συστήματος και υλοποίησή του

Επικύρωση και επαλήθευση λογισμικού - Κάνει το σύστημα αυτό που θέλει ο πελάτης; λειτουργεί σωστά;

Εξέλιξη και συντήρηση λογισμικού - Αλλαγές ως απόκριση σε αλλαγές των αναγκών του χρήστη

Μια διαδικασία δεν είναι απαραίτητα κατάλληλη για χρήση σε όλα τα έργα ανάπτυξης λογισμικού

Κάθε μια από τις διαθέσιμες μεθοδολογίες είναι η καταλληλότερη για συγκεκριμένα είδη έργων με βάση διάφορα στοιχεία, όπως το **είδος της τεχνολογίας που αναπτύσεται, ο οργανισμός και η κουλτούρα της ομάδας, τις απαιτήσεις του πελάτη**

Ο μεταβαλλόμενος ρόλος των μεθοδολογιών ανά τα χρόνια

- **Pre-Methodology Era** ('60s –'70s)
  - Έμφαση στον προγραμματισμό και τα τεχνικά ζητήματα, χωρίς έμφαση στην επιτυχία του έργου ή στην παράδοση στον χρόνο, έμφαση στην κατασκευή και την επιδιόρθωση (build and fix)
- **Early Methodology Era** ('70s-80s)
  - Κύκλος ζωής – waterfall model, φάσεις ανάπτυξης, πρώιμες τεχνικές προσεγγίσεις.
- **Methodology Era** ('80s-)
  - Δομημένη, προσανατολισμένη στα δεδομένα, προτυποποίηση, Αντικειμενοστραφής προσέγγιση, Soft systems methodology, Rational Unified Process, Spiral model
- **Post-Methodology Era** (late '90s-)
  - Επαναξιολόγηση της έννοιας και της χρησιμότητας της "Μεθοδολογίας".
  - Ανάπτυξη με εργαλεία (RAD), αντικειμενοστραφές παράδειγμα, επαναληπτική και αυξητική ανάπτυξη, Outsourcing, ευέλικτες μέθοδοι
  - Αυτή η εποχή, για ορισμένους, αντιπροσωπεύει βελτιωμένες μεθοδολογίες, ενώ απομακρύνεται από τους γραφειοκρατικούς τύπους της εποχής της μεθοδολογίας.Για άλλους αντιπροσωπεύει την εγκατάλειψη των μεθοδολογιών, ακόμη και την πλήρη απομάκρυνση από την εσωτερική ανάπτυξη (π.χ. outsourcing).

EARLY METHODOLOGY ERA (70S-80S)

Prescriptive Process Models

ΚΥΚΛΟΣ ΖΩΗΣ ΑΝΑΠΤΥΞΗΣ ΣΥΣΤΗΜΑΤΩΝ Ή ΤΟ ΜΟΝΤΕΛΟ ΚΑΤΑΡΡΑΚΤΗ

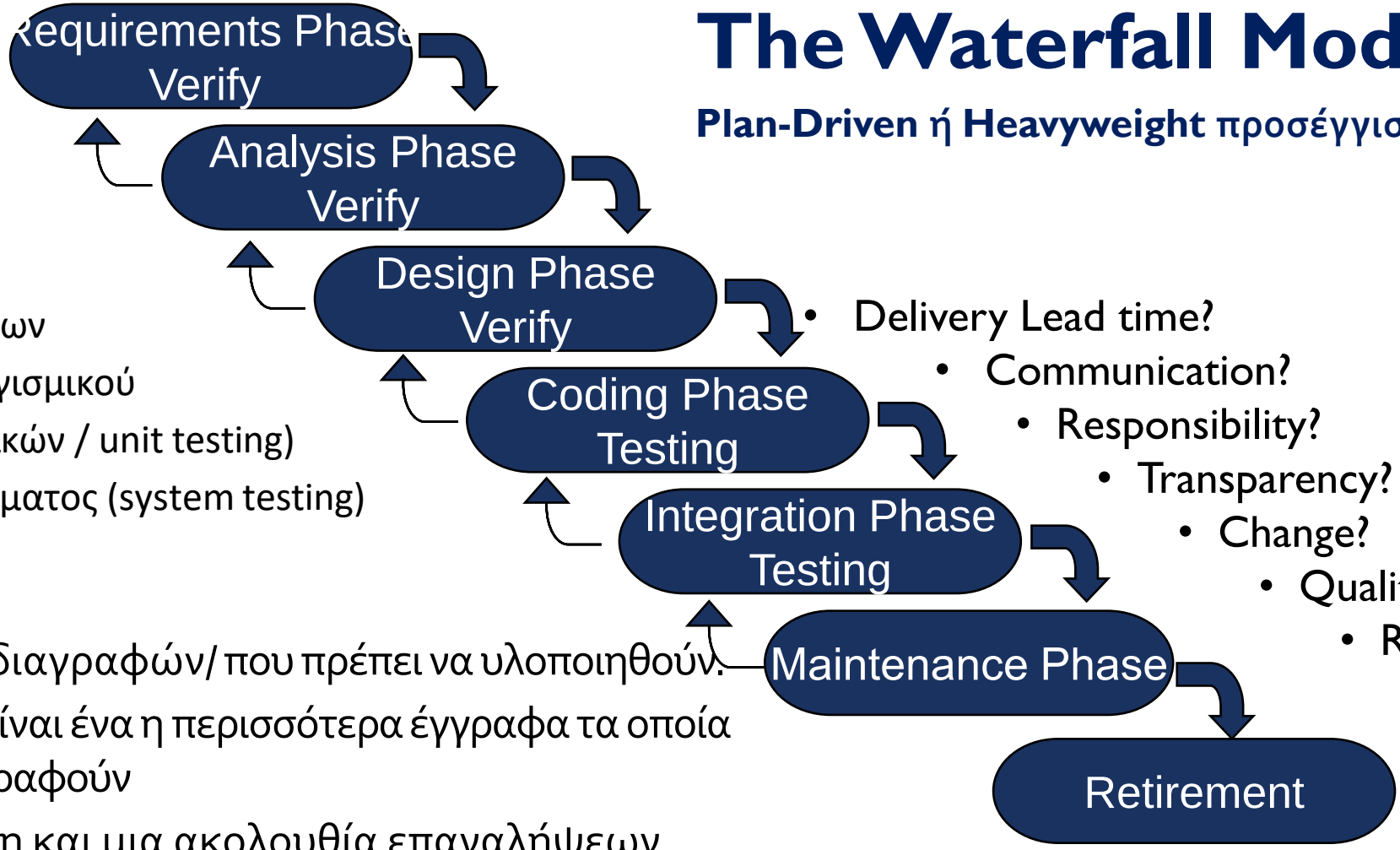
“Code late, get it right first time”
"Κωδικοποιήστε αργά, κάντε το σωστά απο την πρώτη φορά"

The Waterfall Model

Plan-Driven ή Heavyweight προσέγγιση

- **Χαρακτηρίζεται από**
  - Σειριακά βήματα (phases)

- **Υπάρχουν διακριτές φάσεις**
  - Ανάλυση και ορισμός απαιτήσεων
  - Σχεδιασμός συστήματος και λογισμικού
  - Υλοποίηση (και δοκιμή συστατικών / unit testing)
  - Ενσωμάτωση και δοκιμή συστήματος (system testing)
  - Λειτουργία και συντήρηση



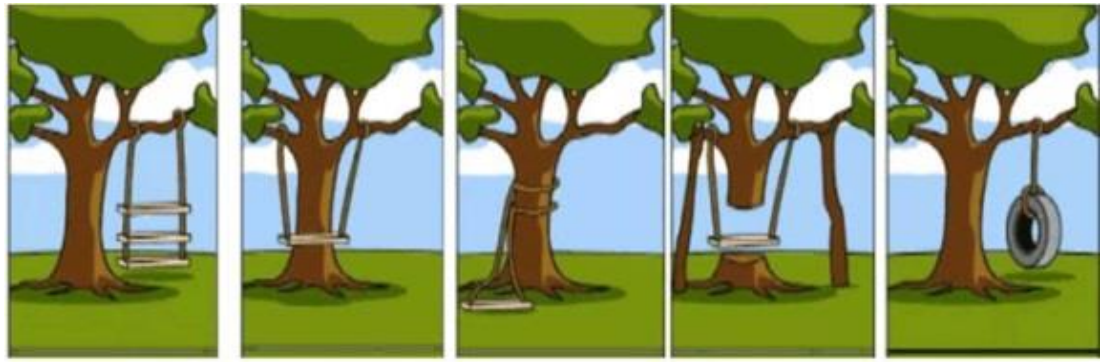
- Delivery Lead time?
- Communication?
- Responsibility?
- Transparency?
- Change?
- Quality?
- ROI?

- Βασίζεται στη δημιουργία προδιαγραφών/ που πρέπει να υλοποιηθούν.
- Το αποτέλεσμα της κάθε φάσης είναι ένα η περισσότερα έγγραφα τα οποία πρέπει να εγκριθούν και να υπογραφούν
- Περιλαμβάνει ανατροφοδότηση και μια ακολουθία επαναλήψεων, ωστόσο εξακολουθεί να είναι σε γενικές γραμμές διαδοχικό (είναι εύκολο να κατεβαίνεις τον καταρράκτη παρά να ανεβαίνεις!). Η οπισθοδρόμηση θεωρείται παραδοσιακά δαπανηρή και πρέπει να αποφεύγεται.

THE WATERFALL MODEL - ΜΕΙΟΝΕΚΤΗΜΑΤΑ

Το πρόβλημα των κινεζικών ψιθύρων

- Χαμηλή επικοινωνία και τα έγγραφα που δημιουργούνται στο τέλος κάθε σταδίου οδηγούν σε
  - Παρερμηνεία των προδιαγραφών
  - Μεγάλη εξάρτηση από το **documentation**
  - Κακή συνεργασία ή ακόμη και επίρριψη ευθυνών όταν τα πράγματα πάνε στραβά
  - Ο πελάτης συμμετέχει μόνο στην αρχή
  - Ο πελάτης πρέπει να περιμένει έως ότου υλοποιηθεί και αναπτυχθεί ολόκληρο το σύστημα για να αποκομίσει τα οφέλη



THE WATERFALL MODEL - ΜΕΙΟΝΕΚΤΗΜΑΤΑ

Δύσκολη, αργή και δαπανηρή προσαρμογή στην αλλαγή

- Υποθέτει ότι όλες οι ανάγκες και απαιτήσεις μπορούν να καταγραφούν από την αρχή και ότι δεν θα χρειαστεί να γίνουν αλλαγές στη συνέχεια.
- Προδιαγραφές που δεν μπορούν να αλλάξουν στη πορεία δεν είναι ρεαλιστική παραδοχή. Αν χρειαστεί να αλλάξουν είναι δύσκολο, χρονοβόρο και κοστίζει πολύ.
- Σειριακή και πλήρης ολοκλήρωση κάθε βήματος δεν είναι πάντα ενδεδειγμένη

Η ανεπαρκής δοκιμή (testing) υπονομεύει το σύστημα και την απόδοση της επένδυσης

- Η φάση των δοκιμών συχνά συμπιέζεται, με αποτέλεσμα τη μείωση της παροχής οφέλους και την αύξηση του κόστους στη συνέχεια.
- Η μεγάλη διάρκεια ανάπτυξης σημαίνει ότι το σύστημα είναι ξεπερασμένο όταν παραδίδεται.
- Παρέχει φτωχά αποτελέσματα (έργα καθυστερημένα, συστήματα που δεν ικανοποιούν τους χρήστες και πάνω από τον προϋπολογισμό).

THE WATERFALL MODEL- ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΣΥΝΗΘΩΣ

Most appropriate

- Στην ανάπτυξη μεγάλων έργων λογισμικού, πολύ γραφειοκρατικών, όπου ένα σύστημα αναπτύσσεται σε διάφορες τοποθεσίες
  - Η δομή του μοντέλου του καταρράκτη (προγραμματισμός) διευκολύνει το συντονισμό των εργασιών
- Σε έργα που δεν υπάρχει πίεση για άμεση υλοποίηση.
- Σε έργα όπου οι χρήστες έχουν καλή γνώση του οργανισμού και οι απαιτήσεις είναι σταθερές και δεν θα αλλάξουν.
- Κρίσιμα συστήματα (critical systems)
  - Απαιτείται εκτεταμένη ανάλυση ασφάλειας στα στάδια των απαιτήσεων/προδιαγραφών και σχεδίασης
- Σε μικρά και απλά έργα, π.χ database systems, που υπάρχει πολύ καλή κατανόηση απαιτήσεων.

THE WATERFALL MODEL- ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΕΙΤΑΙ ΣΥΝΗΘΩΣ

Least appropriate

- Μεγάλα έργα όπου οι απαιτήσεις δεν είναι καλά κατανοητές ή μεταβάλλονται, π.χ. λόγω εξωτερικών αλλαγών, μεταβαλλόμενων προσδοκιών, αλλαγών στον προϋπολογισμό ή ταχέως μεταβαλλόμενης τεχνολογίας.
- Web applications and mobile applications
- Συστήματα πραγματικού χρόνου ή συστήματα που καθοδηγούνται από συμβάντα.
- Εφαρμογές αιχμής.

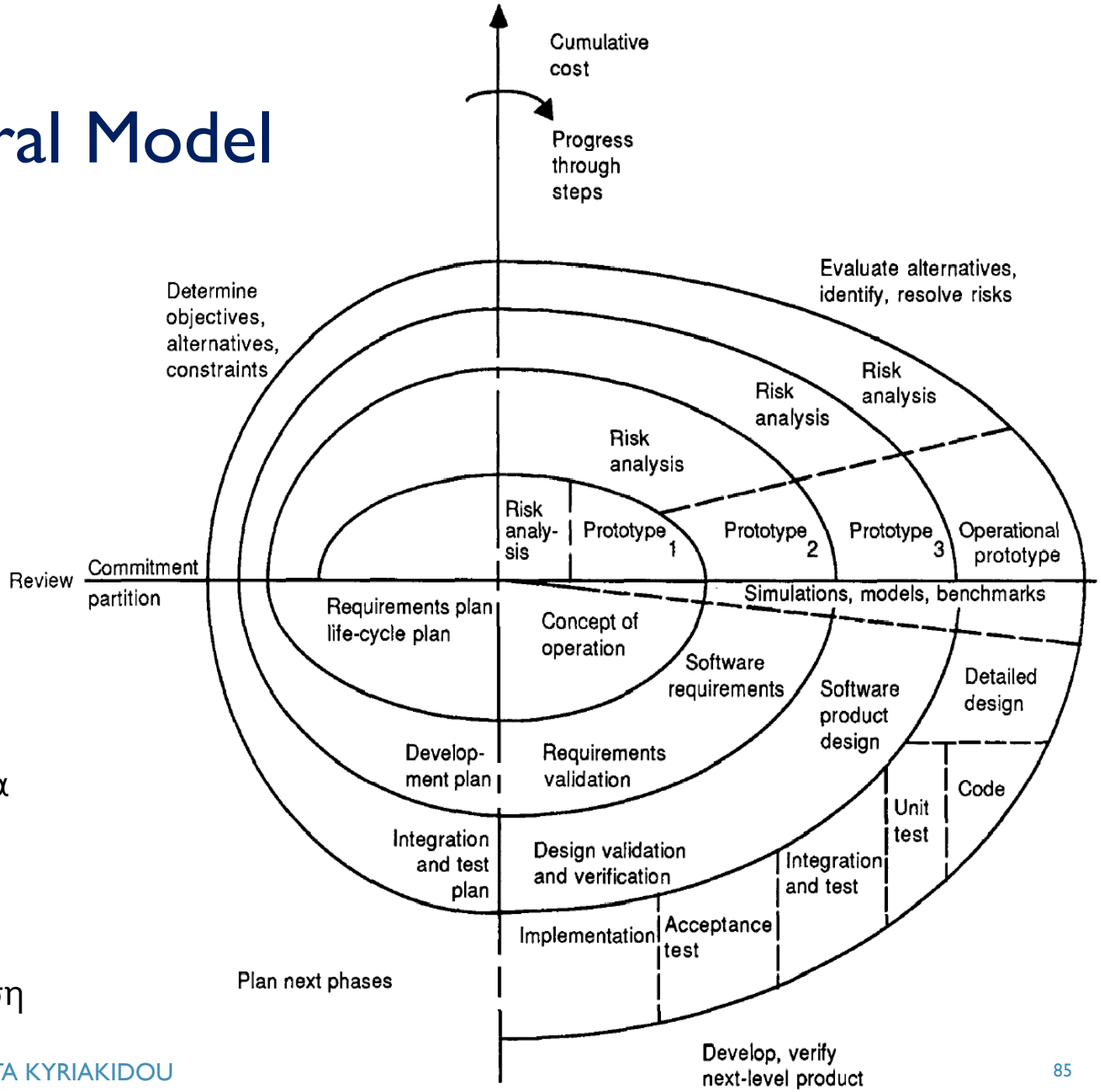
METHODOLOGY ERA (80S)

Σπειροειδές Μοντέλο - Spiral Model

- Κάθε σπιράλ αντιπροσωπεύει μια φάση, όπως ο προσδιορισμός των απαιτήσεων, ο σχεδιασμός κ.λπ.

- Συνδυάζει την πρωτυποποίηση (εξελικτική ανάπτυξη) με τη σειριακή προσέγγιση του κύκλου ζωής υπό τη μορφή σπιράλ.

- Βασικό σημείο η διαχείριση κινδύνων
  - Risk-driven διαδικασία
  - Αναπτύσσει **prototypes** για να μειώσει τα ρίσκα
- Εάν η ανάλυση ρίσκου αποτύχει τότε το έργο διακόπτεται
  - Αν το έργο έχει ήδη προχωρήσει πολύ είναι δύσκολο να τερματιστεί ακόμη και αν η ανάλυση ρίσκου αποτύχει

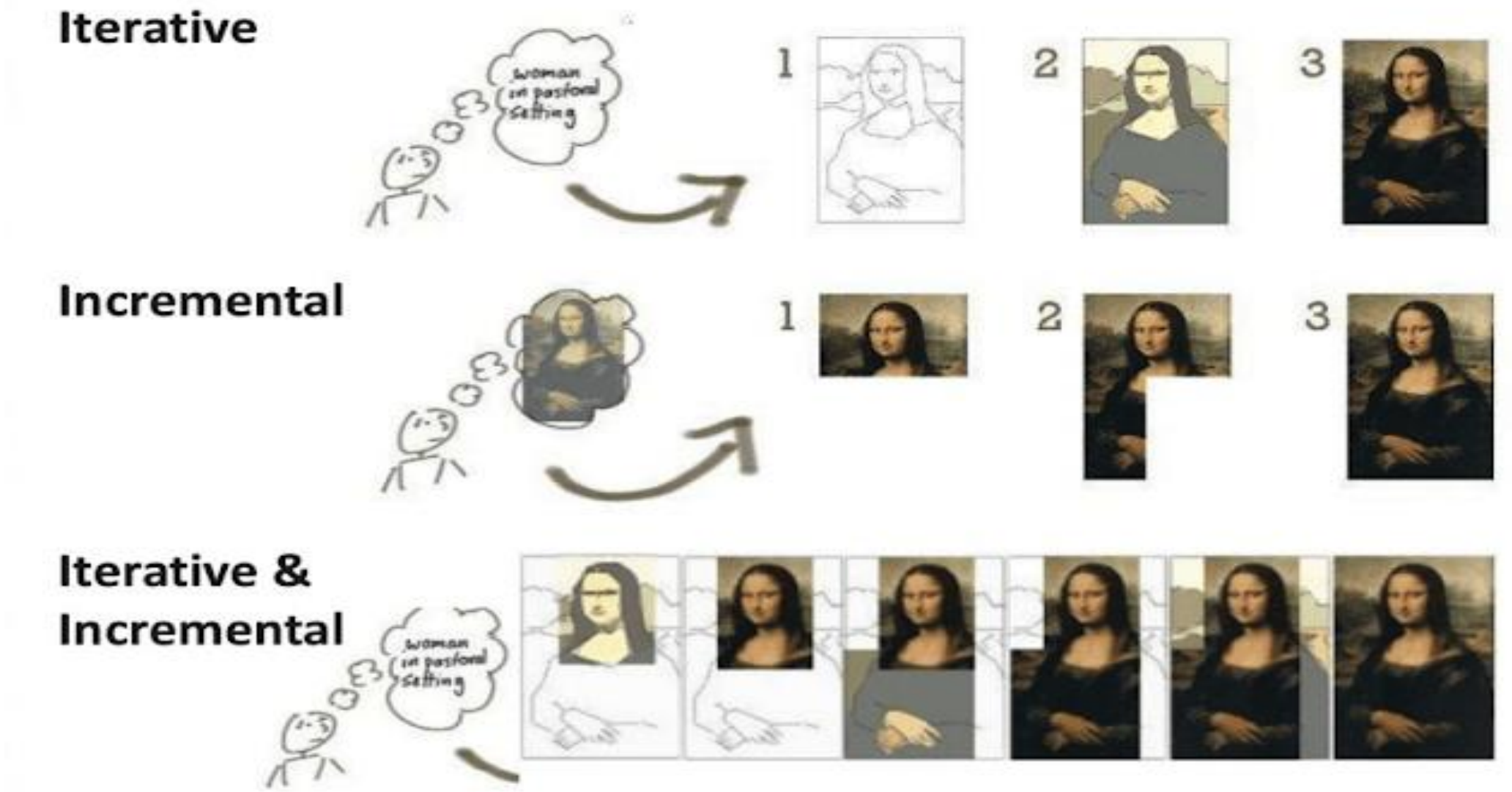


Iterative and Incremental Development – Επαναληπτική/αυξητική ανάπτυξη

- **Το σύστημα παραδίδεται βαθμιαία σε μικρές προσαυξήσεις.**
  - Ξεκινούμε με τις πιο σημαντικές λειτουργίες και ενσωματώνουμε σιγά σιγά περαιτέρω λειτουργίες.
- **Σε κάθε επανάληψη (iteration) οι λειτουργίες ορίζονται, σχεδιάζονται, αναπτύσσονται και δοκιμάζονται.**
  - Οι επαναληπτικού κύκλοι επαναλαμβάνονται μέχρι να έχουμε ένα πλήρως λειτουργικό σύστημα το οποίο να είναι έτοιμο για παράδοση
  - Στο τέλος κάθε επαναληπτικού κύκλου το σύστημα δίνεται στους χρήστες για feedback.
  - Η επικύρωση/validation των ολοκληρωμένων κομματιών του συστήματος μπορεί να ενθαρρύνει την τελειοποίησή των υπόλοιπων κομματιών. Οι χρήστες όμως απαγορεύεται να ζητήσουν περαιτέρω αλλαγές σε κομμάτια που θεωρούνται ολοκληρωμένα.

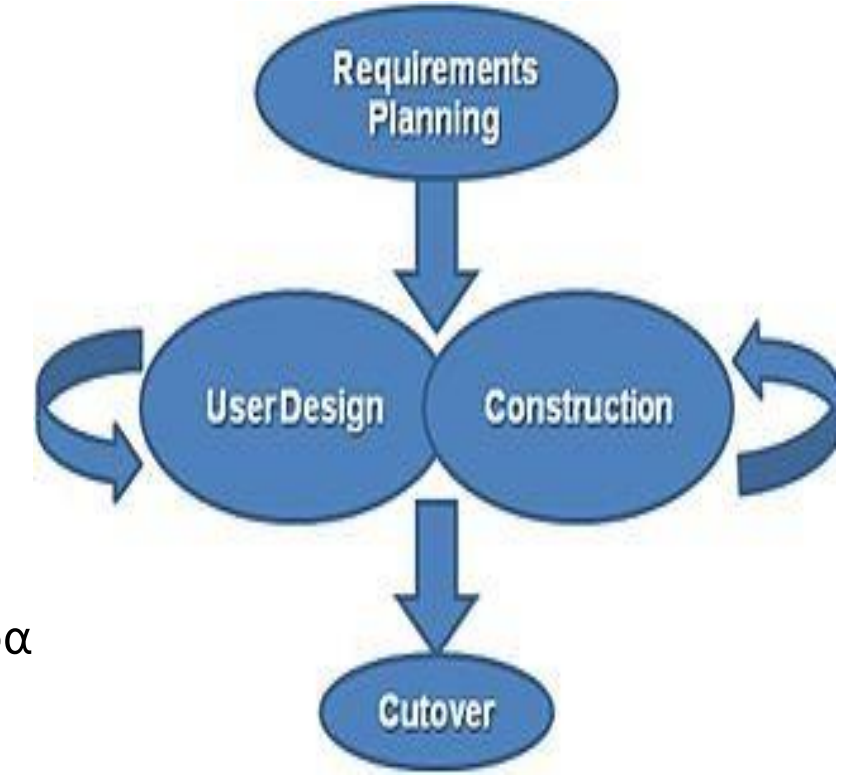
- **Η διαδικασία δεν προσπαθεί να ξεκινήσει με το να έχει μπροστά της το πλήρες σύνολο των απαιτήσεων και του σχεδιασμού.** Αντ' αυτού, η ομάδα προσπαθεί να προετοιμάσει μόνο ό,τι χρειάζεται για την επιτυχή παράδοση της επόμενης επανάληψης.

Iterative and Incremental Development – Επαναληπτική/αυξητική ανάπτυξη



RAPID APPLICATION DEVELOPMENT (RAD)

- Πολλά κοινά με τα επαναληπτικά μοντέλα
- Έμφαση σε **συμμετοχή χρηστών**, **πρωτοτυτοποίηση**, **επαναχρησιμοποίηση** και χρήση αυτοματοποιημένων εργαλείων
  - Ελάχιστος προγραμματισμός
- **Έννοια time box**
  - Συγκεκριμένα χρονικά όρια για την πραγματοποίηση δραστηριοτήτων
  - Priotirisation των απαιτήσεων – Joint application workshops (JAD)
  - Η διάρκειά του αποφασίζεται από την αρχή
  - Έπειτα προσπαθεί κάποιος να κάνει όσο το δυνατόν περισσότερα μέσα σε αυτό
  - Ολοκληρωμένο σύστημα σε 60-90 μέρες.
- Χρήση ομάδας SWAT (Skilled Workers with Advanced Tools)



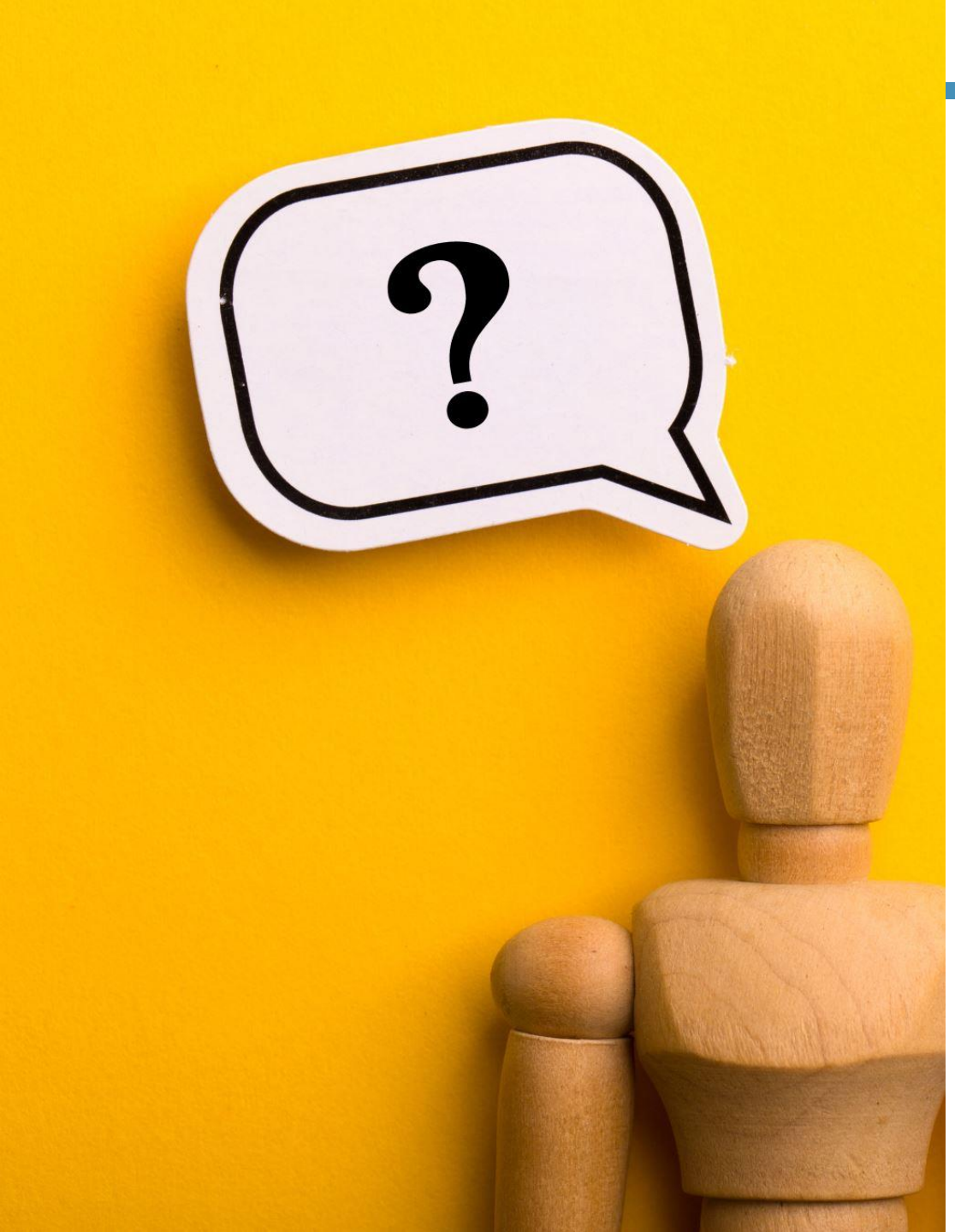
RAPID APPLICATION DEVELOPMENT (RAD) – ΣΧΟΛΙΑ

[+] Προτερήματα

- Περισσότερο ποιοτικό σύστημα
- Έλεγχος κινδύνων (αν και εδώ γίνεται μόνο στην αρχή βάσει εμπειρίας)
- Περισσότερα έργα σε μικρότερο χρονικό διάστημα και εντός προϋπολογισμού

[-] Μειονεκτήματα

- Απαιτεί παρουσία συνεχή του χρήστη
- Τελικά κακή σχεδίαση (πολλές μικρές αλλαγές χωρίς να ληφθεί υπόψη ο γενικότερος αρχιτεκτονικός σχεδιασμός)
- Όχι κατάλληλο για μεγάλα συστήματα



BREAKOUT SESSION – TUTORIAL EXERCISE

(YOU CAN FIND THIS ON MOODLE)

POST METHODOLOGY ERA (90S TO DATE)

ΠΟΙΕΣ ΕΙΝΑΙ ΜΕΡΙΚΕΣ ΑΠΟ ΤΙΣ ΣΥΓΧΡΟΝΕΣ ΠΙΕΣΕΙΣ/ΠΡΟΚΛΗΣΕΙΣ ΣΤΗΝ ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ;

Reply at:
PollEv.com/avgoustakyri977

1. Go to **PollEv.com**
2. Enter **AVGOUSTAKYRI977**
3. Respond to activity

1. Text **AVGOUSTAKYRI977** to **22333**
2. Text in your message

ΣΥΓΧΡΟΝΕΣ ΠΙΕΣΕΙΣ/ΠΡΟΚΛΗΣΕΙΣ ΣΤΗΝ ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ;

- Πολλές "heavyweight" μεθοδολογίες
- Παγκοσμιοποίηση, ταχέως μεταβαλλόμενο περιβάλλον, νέες ευκαιρίες.
- Αδύνατο να μπορούμε να παράγουμε όλες τις απαιτήσεις.
- Μικρότερες εταιρείες που αναπτύσσουν συστήματα. Εταιρείες πρόθυμες να συμβιβαστούν με την ποιότητα του λογισμικού και τις απαιτήσεις για την ταχεία παράδοση λογισμικού.
- Πολύ δύσκολο να κάνεις πλάνα για το μέλλον
- Επιταχυνόμενες* αλλαγές στην τεχνολογία (Grid Computing/ Cloud Computing, mob apps, AI, AR κ.λπ. (*πάντα ένα επιχείρημα)

Agile software development Manifesto

Δημιουργήθηκε το 2001 σε ένα καταφύγιο σκι στο Snowbird της Γιούτα.

- Κορυφαίοι επαγγελματίες από τις διάφορες νέες μεθόδους της εποχής συναντήθηκαν
- Αναγνώρισαν ότι είχαν κοινά στοιχεία που τους διαφοροποιούσαν από τις τότε κυρίαρχες παραδοσιακές προσεγγίσεις καταρράκτη της εποχής.

"Ανακαλύπτουμε καλύτερους τρόπους ανάπτυξης λογισμικού κάνοντάς το και βοηθώντας άλλους να το κάνουν. Αναγνωρίζουμε ότι:

- Τα μεμονωμένα άτομα και οι αλληλεπιδράσεις τους έχουν προτεραιότητα έναντι των διεργασιών και των εργαλείων
- Ένα λογισμικό που είναι λειτουργικό είναι πιο σημαντικό από κατανοητή τεκμηρίωση
- Η συνεργασία με τον πελάτη είναι πιο σημαντική από τη διαπραγμάτευση του συμβολαίου
- Η σωστή απόκριση σε αλλαγές είναι πιο σημαντική από κάποιο συγκεκριμένο πλάνο

Ενώ υπάρχει αξία στα αντικείμενα στα δεξιά, εμείς εκτιμούμε περισσότερο τα αντικείμενα στα αριστερά.

(Υπογράφηκε από 17 διάσημα ονομάτα στο χώρο της τεχνολογίας λογισμικού)

http://agilemanifesto.org/

- **Kent Beck - αναλογία οδήγησης**

Agile Software Development

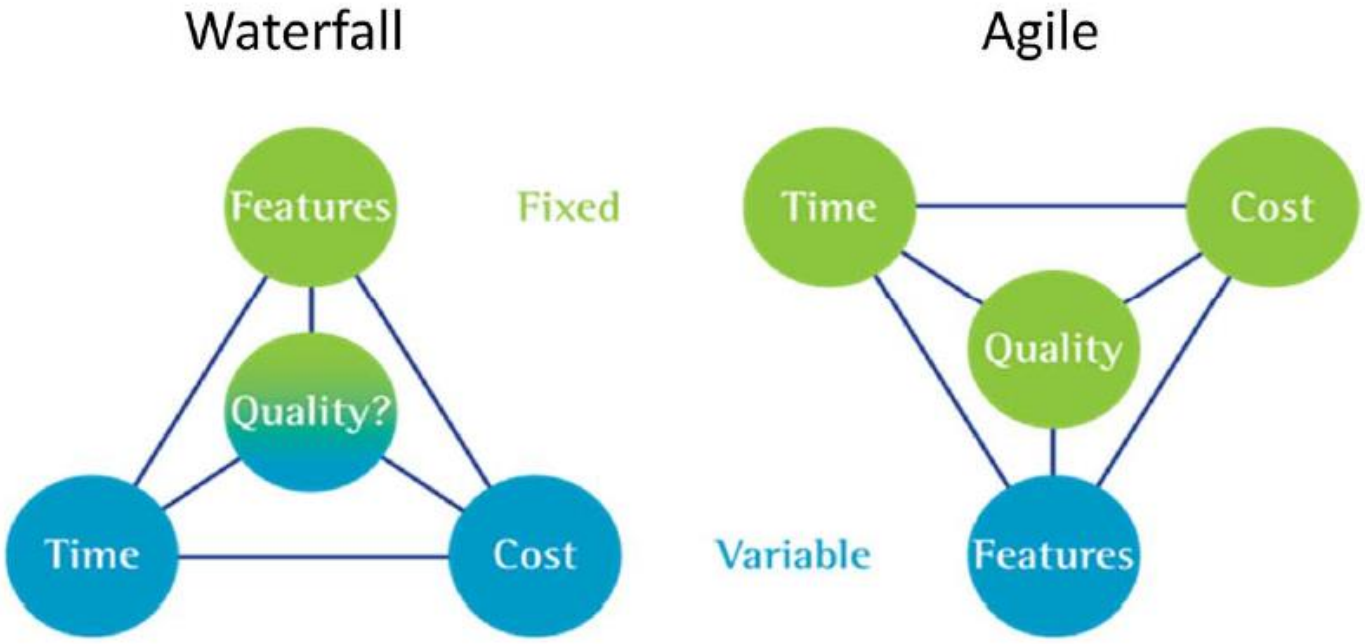
- **Ένας "γενικός όρος" για ένα σύνολο μεθόδων και πρακτικών που δίνουν έμφαση:**
  - Ομαδική εργασία
  - Αναγνώριση της συνεχής ανάγκης για προσαρμογή στις αλλαγές
  - Ελαφριά τεκμηρίωση
  - Ταχεία και συχνή ανάπτυξη μικρών κομματιών του συστήματος σε μικρά χρονικά διαστήματα
  - Πελάτες και χρήστες μέρος της ομάδας ανάπτυξης

- **Αυτό που διαχωρίζει την ευέλικτη ανάπτυξη από άλλες προσεγγίσεις**
  - Η εστίαση στους ανθρώπους και στην υποστήριξη της συνεργασίας.
  - Τα συστήματα εξελίσσονται μέσω της συνεργασίας μεταξύ αυτοοργανωμένων, διαλειτουργικών ομάδων (***self-organizing, cross-functional*** teams)

- **Μεγάλη έμφαση στη συνεργασία και την αυτοοργανωμένη ομάδα (self-organizing team):**
  - Οι ομάδες είναι διαλειτουργικές (cross-functional) - δεν έχουν συγκεκριμένους ρόλους, αλλά όταν συναντώνται όλες οι σωστές δεξιότητες υπάρχουν μέσα στην ομάδα.

Μεταβλητές έργου (Triple constraint) στην ανάπτυξη συστημάτων

Εάν τα πράγματα δεν πάνε σύμφωνα με το σχέδιο, τότε η μόνη επιλογή είναι να μετατοπιστεί το χρονοδιάγραμμα ή να ανατεθούν πρόσθετοι πόροι σε αυτό (αυξάνοντας το κόστος) προκειμένου να παραδοθεί εγκαίρως.



Επικεντρωθείτε στο ελάχιστο βιώσιμο προϊόν **Minimum Viable Product** : προϊόν με επαρκή χαρακτηριστικά για να ικανοποιήσει τους πρώτους χρήστες. Το τελικό, πλήρες σύνολο χαρακτηριστικών σχεδιάζεται και αναπτύσσεται μόνο αφού ληφθούν υπόψη τα σχόλια των αρχικών χρηστών του προϊόντος. **Δεν είναι όλες οι απαιτήσεις το ίδιο σηματνικές - πρέπει να δοθεί προτεραιότητα. **Οι απαιτήσεις είναι βέβαιο ότι θα αλλάξουν**

Minimum Viable Product

Agile methods focus on…

- Κώδικας που λειτουργεί και όχι λεπτομερής σχεδιασμός - ελάχιστη δημιουργία τεκμηρίωσης(**documentation**)
- Συχνή/γρήγορη παράδοση νέων εκδόσεων λειτουργικού λογισμικού για αξιολόγηση - για την κάλυψη μεταβαλλόμενων απαιτήσεων
- Εκτεταμένη υποστήριξη εργαλείων (π.χ. εργαλεία αυτοματοποιημένων δοκιμών - **testing**) που χρησιμοποιούνται για την υποστήριξη της ανάπτυξης - **test driven development**
- Δημιουργία απαιτήσεων/προδιαγραφών, σχεδιασμός και υλοποίηση και δοκιμές/**testing** διαπλέκονται μεταξύ τους και λαμβάνουν μέρος σε κάθε επανάληψη.
- Το σύστημα αναπτύσσεται ως μια σειρά εκδόσεων ή προσαυξήσεων με τους χρήστες να συμμετέχουν ενεργά.

	Traditional	Agile
Fundamental Assumptions	Systems are fully specifiable, predictable, and can be built through meticulous and extensive planning.	High-quality, adaptive software can be developed by small teams using the principles of continuous design improvement and testing based on rapid feedback and change.
Control	Process centric	People centric
Management Style	Command-and-control	Leadership-and-collaboration
Knowledge Management	Explicit	Tacit
Role Assignment	Individual—favors specialization	Self-organizing teams—encourages role interchangeability
Communication	Formal	Informal
Customer's Role	Important	Critical
Project Cycle	Guided by tasks or activities	Guided by product features
Development Model	Life cycle model (Waterfall, Spiral, or some variation)	The evolutionary-delivery model
Desired Organizational Form/Structure	Mechanistic (bureaucratic with high formalization)	Organic (flexible and participative encouraging cooperative social action)
Technology	No restriction	Favors object-oriented technology

"Κωδικοποιήστε νωρίς, διορθώστε και βελτιώστε καθώς προχωράτε".

Code early, fix and improve it as you go along".

EXTREME PROGRAMMING (XP)

Source: dilbert.com

EXTREME PROGRAMMING (XP)

- **Το XP υιοθετεί μια "ακραία" προσέγγιση στην επαναληπτική και αυξητική ανάπτυξη (iterative and incremental).**
  - Νέες εκδόσεις του συστήματος μπορεί να δημιουργούνται αρκετές φορές την ημέρα. Το σύστημα θα πρέπει να έχει κατασκευαστεί και όλες οι δοκιμές θα πρέπει να έχουν ολοκληρωθεί μέσα σε 10 λεπτά (10-minute builds).
  - Όλα γίνονται σε μικρά βήματα. Σε κάθε βήμα έχουμε μια λειτουργική έκδοση του συστήματος
  - Οι προσαυξήσεις (increments – working products) παραδίδονται στους πελάτες κάθε 1-2 εβδομάδες.
  - Όλες οι δοκιμές πρέπει να εκτελούνται για κάθε build και το build γίνεται αποδεκτό μόνο αν οι δοκιμές εκτελούνται με επιτυχία.
  - Βασίζεται στη συνεχή βελτίωση του κώδικα και στο pair programming
  - Ο καθένας στην ομάδα μπορεί να αλλάξει τον κώδικα οπουδήποτε και οποτεδήποτε

- Συμμετοχή του πελάτη σημαίνει ΠΛΗΡΗΣ-ΧΡΟΝΙΚΗ εμπλοκή του πελάτη με την ομάδα.
- Εργασία ≤ 40 ώρες την εβδομάδα

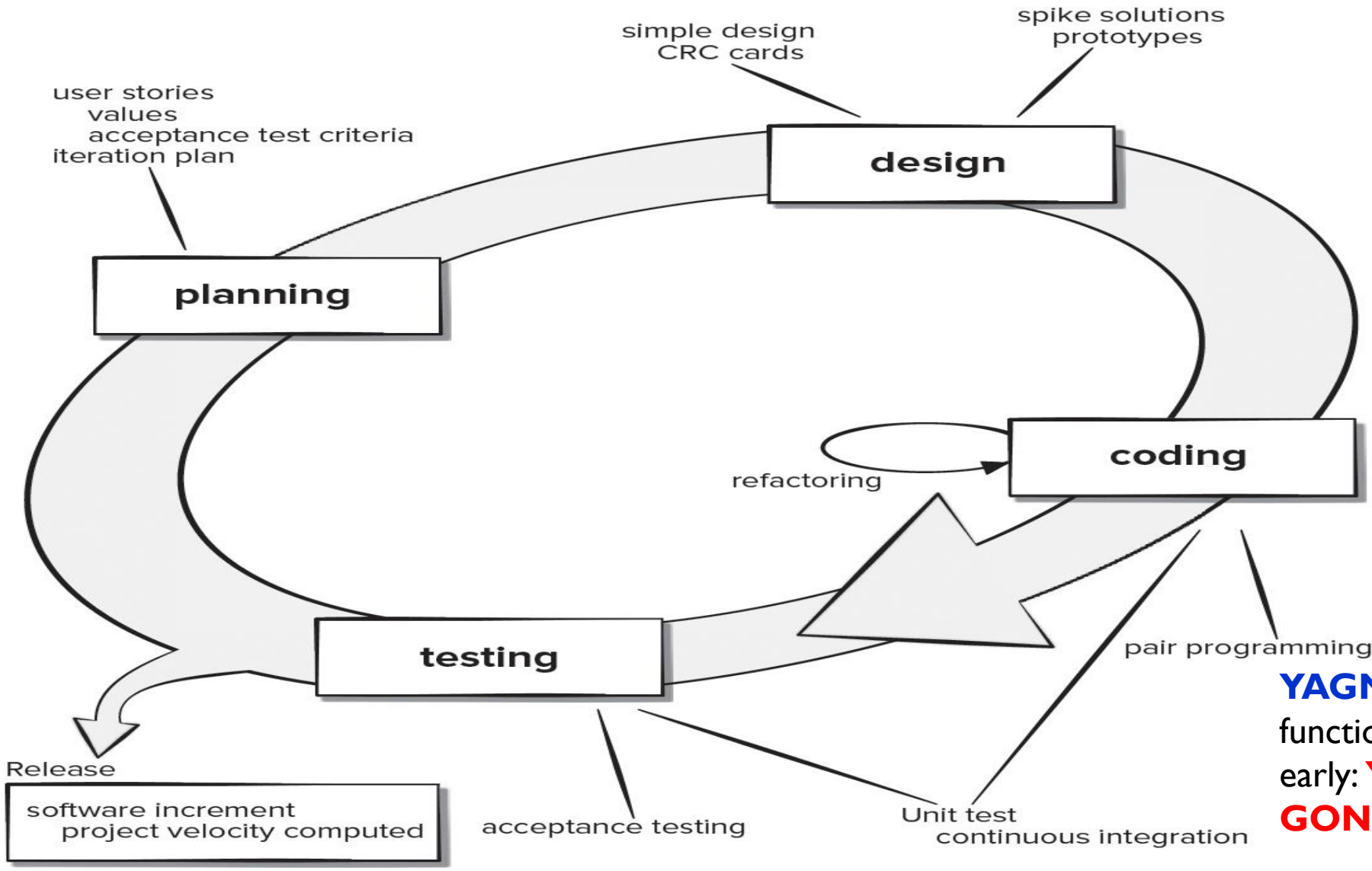
EXTREME PROGRAMMING (XP)

- **Πρακτικές**

  - Ολόκληρη ομάδα – Μικρή ομάδα μεταξύ 2 and 12 ατόμων αλλα αυτό έχει επεκταθεί και στα 20 άτομα
  - Απαιτήσεις παρουσιάζονται σαν user stories
  - Απλή σχεδίαση
  - Μικρές περίοδοι υλοποίησης (sprints or increments) (π.χ. 2 βδομάδες)
  - Έλεγχοι από τον πελάτη
  - Πρότυπα υλοποίησης (Coding standards)

- Ανάπτυξη προσανατολισμένη από τους ελέγχους (Test-driven development)
  - Περιπτώσεις ελέγχων αναπτύσσονται πρώτα
- Βελτίωση της σχεδίασης
- Συνιδιοκτησία κώδικα
- Pair programming
- Συνεχής ενσωμάτωση
- Ήπιοι ρυθμοί
- Χωρίς υπερωρίες

XP FRAMEWORK



YAGNI : No functionality is added early: **YOU AINT' GONNA NEED IT!**

SCRUM


Sprint
Backlog
Scrum
Items
Release
Stakeholders
Team
Team's
Owner
Product
work
needs
process
wants
working
Goals
done
often
desires
Planning
Members
Strategy
accountable
ScrumMaster
important
Business
called
people
Goal
Review
works
Sprints
provides
Visioning
book
project
next
ScrumMaster's
Roadmap
Releases
Results
Products
set
Daily
usually
Often
priorities
planning
figure
discussed
responsibility
requirements
direction
business
agreeing
simple
order
may
must
release
Section
new
Stakeholder
support
thus
part
first
get
Teams
agree
help
goals
ready
continuously
result
agreed
Vision
short
throughout

- Το Scrum είναι μια ευέλικτη μεθοδολογία που μας επιτρέπει να εστιάζουμε στην παροχή της υψηλότερης επιχειρηματικής αξίας στον συντομότερο δυνατό χρόνο.
- Μας επιτρέπει να αναπτύσσουμε και να επιθεωρούμε γρήγορα και επανειλημμένα το προϊόν (κάθε δύο εβδομάδες έως ένα μήνα).
- Η επιχείρηση θέτει τις προτεραιότητες. Οι ομάδες αυτοοργανώνονται για να καθορίσουν τον καλύτερο τρόπο για την παράδοση των απαιτήσεων υψηλότερης προτεραιότητας.
- Κάθε δύο εβδομάδες έως ένα μήνα ο καθένας μπορεί να δει πραγματικό λογισμικό που λειτουργεί και να αποφασίσει να το κυκλοφορήσει ως έχει ή να συνεχίσει να το βελτιώνει για άλλο ένα sprint

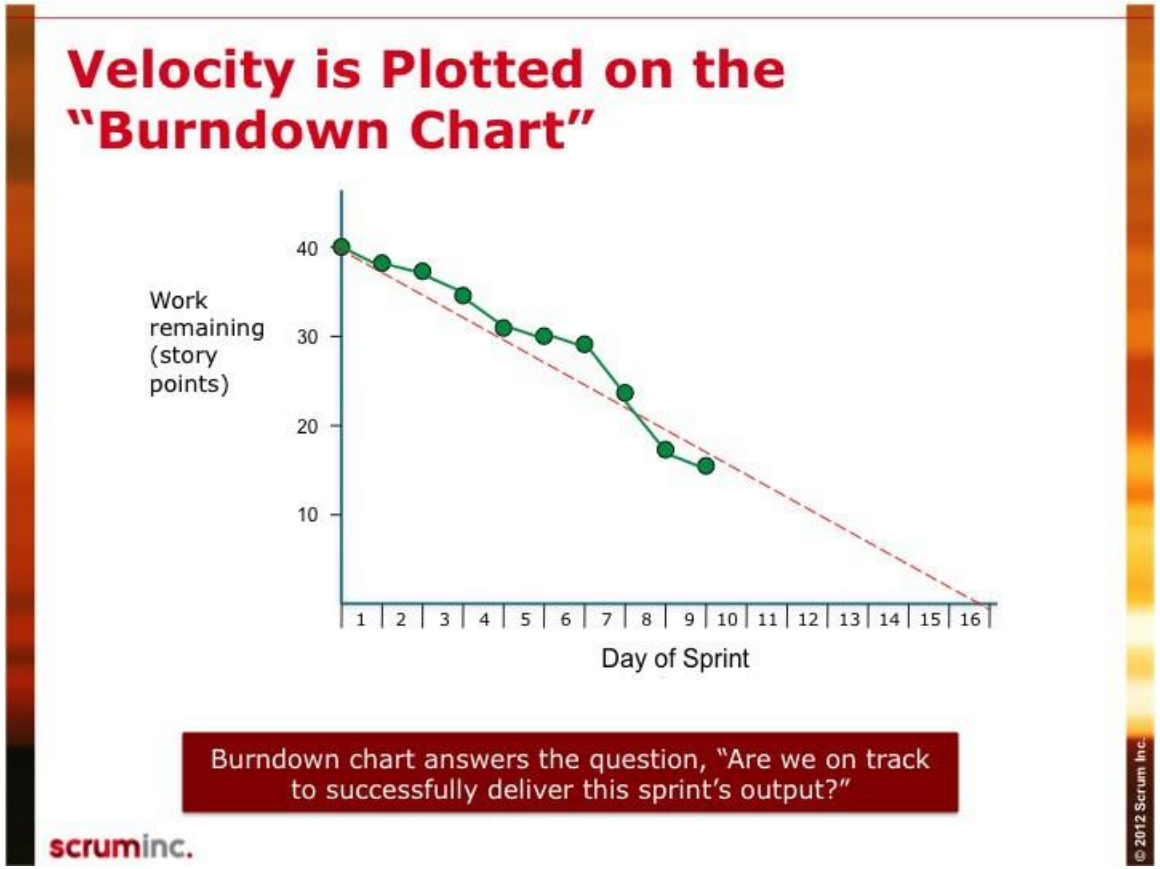
SCRUM IN 100 WORDS

SCRUM CHARACTERISTICS

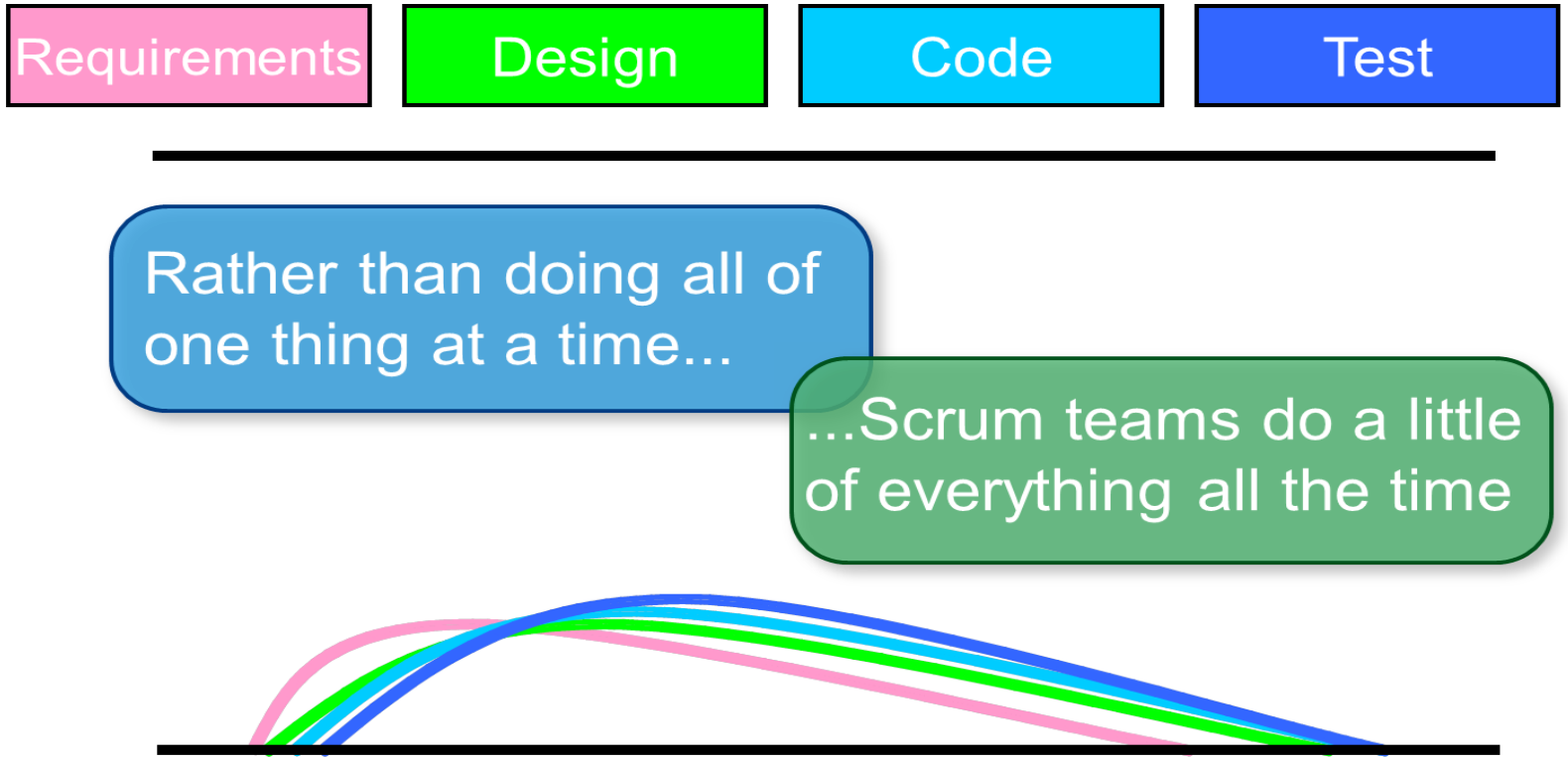
- Δίνει έμφαση σε ένα σύνολο αξιών και πρακτικών διαχείρισης έργων και όχι στις απαιτήσεις, το σχεδιασμό και την υλοποίηση. **Χρειάζεται να συνδυαστεί με άλλες μεθόδους ανάπτυξης συστημάτων, π.χ. με το Extreme Programming για ανάλυση, μοντελοποίηση, κωδικοποίηση και δοκιμές.**

- Small, Self-organizing and cross-functional teams

- **Καθημερινή μέτρηση της προόδου και της ομάδας**
  - 15 λεπτά καθημερινές συνεδριάσεις - Daily scrum stand-up meetings
  - Burndown/up charts τα οποία μετρούν την πρόοδο που έχει γίνει όσο αφορά τις απαιτήσεις που πρέπει να υλοποιηθούν στο Sprint - work completed or work remaining
  - Velocity: σημαντική μετρική – προσθέτουμε τα story points των user stories που έχουν ολοκληρωθεί. Το average από μερικά Sprints, δείχνει πόσα user stories περίπου μπορεί να διαχειριστεί η ομάδα.



SCRUM CHARACTERISTICS



- Το πλεονέκτημα του Scrum είναι ότι η κατεύθυνση ενός έργου προσαρμόζεται με βάση το ολοκληρωμένο έργο και όχι με βάση εικασίες ή προβλέψεις.
- Έμφαση στη γρήγορη διάθεση του **Minimum Viable Product**

Three SCRUM Roles

- **Product owner:**
  - Υπεύθυνος για την εκπροσώπηση των συμφερόντων των πελατών στην ομάδα.
  - Γράφει τις ιστορίες χρηστών, τις ιεραρχεί (μετά από συζήτηση με την ομάδα) και τις προσθέτει στο Product Backlog.
  - Μπορεί να είναι πελάτης, αλλά και διαχειριστής προϊόντος ή άλλος εκπρόσωπος των ενδιαφερομένων μερών.
- **Development Team :**
  - Είναι υπεύθυνη για την υλοποίηση του συστήματος
  - Προγραμματιστές, ελεγκτές, QA και άλλο προσωπικό. Όλοι μαζί σε ένα γραφείο, αυτοοργανωμένη (self-organising) αυτοδιαχειριζόμενη, (self managing), διαλειτουργική (cross-functional). 8 ή λιγότερα μέλη.
- **Scrum master:**
  - Υπεύθυνος για τη βοήθεια της ομάδας στην επίτευξη των στόχων της, εφαρμόζοντας πρακτικές και κανόνες SCRUM.
  - Διευκολυντής (Facilitator), ΔΕΝ πρέπει να θεωρείται διαχειριστής έργου (δεν είναι εύκολο να βλέπουμε πάντα τη διαφορά).

The Agile: Scrum Framework at a glance

Inputs from Executives, Team, Stakeholders, Customers, Users

Product Owner

The Team

Scrum Master

Burndown/up Charts

Daily Scrum Meeting

Every 24 Hours

1-4 Week Sprint

Sprint Review

1
2
3
4
5
6
7
8

Ranked list of what is required: features, stories, ...

Product Backlog

Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting

Task Breakout

Sprint Backlog

Sprint end date and team deliverable do not change

Finished Work

Sprint Retrospective

neon rain interactive

AGILE FOR ALL

SCRUM PROCESS (1)

Product Backlog – Κατάλογος Προϊόντος

- Τα μέλη της ομάδας και οι πελάτες μοιράζονται σκέψεις, ανησυχίες, λειτουργίες και χαρακτηριστικά του προϊόντος, η τις γενικές απαιτήσεις του έργου.
- Ο **Ιδιοκτήτης Προϊόντος (Product Owner)** καταγράφει τις απαιτήσεις (που ονομάζονται επίσης στοιχεία backlog προϊόντος ή user stories) στο **backlog προϊόντος (product backlog)** και τις **κατατάσσει ως προς τη σημαντικότητά τους (ποιά πρέπει να υλοποιηθούν πρώτα)**.
- Τα μέλη της ομάδας **εκτιμούν πόση προσπάθεια χρειάζεται για να υλοποιηθούν αυτές οι απαιτήσεις (user stories)**, χρησιμοποιώντας την **τεχνική του πόκερ σχεδιασμού.**

Sprint planning and Sprint Backlog

- Το Sprint ξεκινά με το λεγόμενο **Sprint planning meeting** το οποίο καθορίζει και τον στόχο του Sprint - που θα επικεντρωθεί η δουλειά **(Sprint goal).**
- Η ομάδα, με βάση αυτό το **Sprint goal** και την κατάταξη των user stories από τον Product owner αποφασίζει **ποιες ιστορίες από το product backlog** θα **συμπεριληφθούν** στο **Sprint backlog**.
- Τα user stories υψηλής προτεραιότητας προστίθενται στο **Sprint backlog**.
  - Η ομάδα μετατρέπει τα user stories σε μικρότερα tasks που θα αναπτυχθούν κατά τη διάρκεια του Sprint. Δημιουργεί το Scrum Board για να οπτικοποιεί την εξέλιξη της δουλειάς μέχρι να γίνει Done.

SCRUM PROCESS (2)

Sprint Cycle (fixed 2-4 weeks) and Potentially Shippable Increment

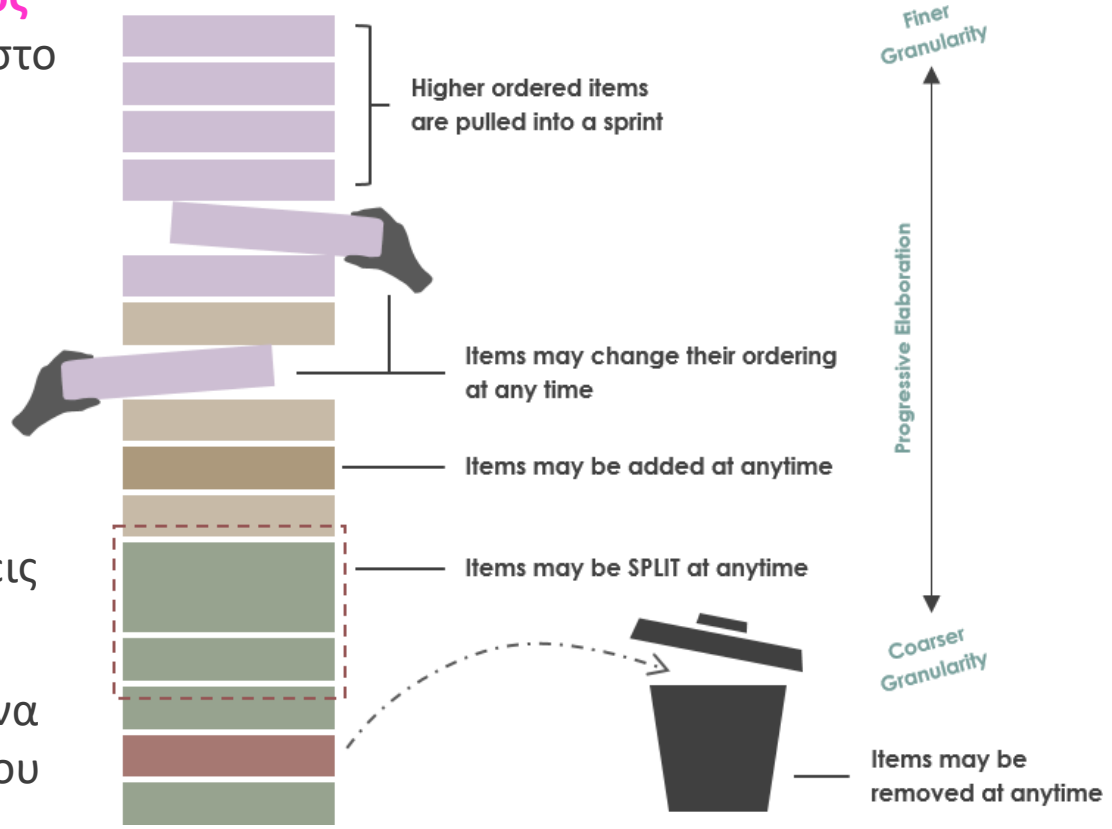
- Μόλις συμφωνηθούν τα βασικά tasks, η ομάδα αυτό-οργανώνεται για να αναπτύξει το προϊόν.
- Το προϊόν σχεδιάζεται, αναπτύσσεται και ελέγχεται κατά τη διάρκεια του Sprint.
- **Potentially Shippable Increment:** είναι το παραδοτέο που παραδίδεται στο τέλος του Sprint.
  - Βρίσκεται σε τελική κατάσταση και δεν χρειάζεται περαιτέρω δουλειά η έλεγχο, για να ενσωματωθεί στο τελικό προϊόν.
  - Το παραδοτέο του πρώτου Sprint, δεν σημαίνει απαραίτητα ότι είναι το MVP, μπορεί να είναι απλά ένα λειτουργικό κομμάτι του έργου. Το MVP συνήθως απαιτεί περισσότερα από ένα Sprint.

PRODUCT BACKLOG- REFINEMENT/GROOMING

- **Δεν επιτρέπεται καμία αλλαγή κατά τη διάρκεια ενός Sprint!** Νέα user stories δεν πρέπει να προστίθενται στο Sprint κατά τη διάρκεια ενός Sprint!

- **5-10% of Sprint time** του χρόνου του Sprint αφιερώνεται στο Product Backlog Refinement.

- Η ομάδα αναγνωρίζει ότι ανά πάσα στιγμή ο Product Owner μπορεί να συμπεριλάβει καινούριες απαιτήσεις μέσα στο Product backlog.
- Πρέπει να ξαναμπούν σε **σειρά προτεραιότητας** και να **ξανα-εκτιμηθούν** για να μπορεί η ομάδα στο τέλος του Sprint να αποφασίσει ποια θα είναι τα επόμενα user stories για υλοποίηση.



DAILY SCRUM IN PRACTICE

- **Scrum Master** εξασφαλίζει ότι η Ομάδα Ανάπτυξης πραγματοποιεί τη συνάντηση, αλλά η Ομάδα Ανάπτυξης είναι υπεύθυνη για τη διεξαγωγή του Daily Scrum.

- Το Daily Scrum είναι μια εσωτερική καθημερινή συνάντηση για την Ομάδα Ανάπτυξης.

- Ονομάζεται και 15 Minute Stand up.

  - **3 Ερωτήσεις πρέπει να απαντηθούν:**
    1. Τι έχει ολοκληρωθεί από την προηγούμενη συνάντηση;
    2. Τι θα ολοκληρωθεί σήμερα και πριν από την επόμενη συνάντηση;
    3. Ποια εμπόδια/δυσκολίες υπάρχουν;

- Σύνθετα ζητήματα συζητούνται εκτός συνάντησης.

SCRUM PROCESS (3)

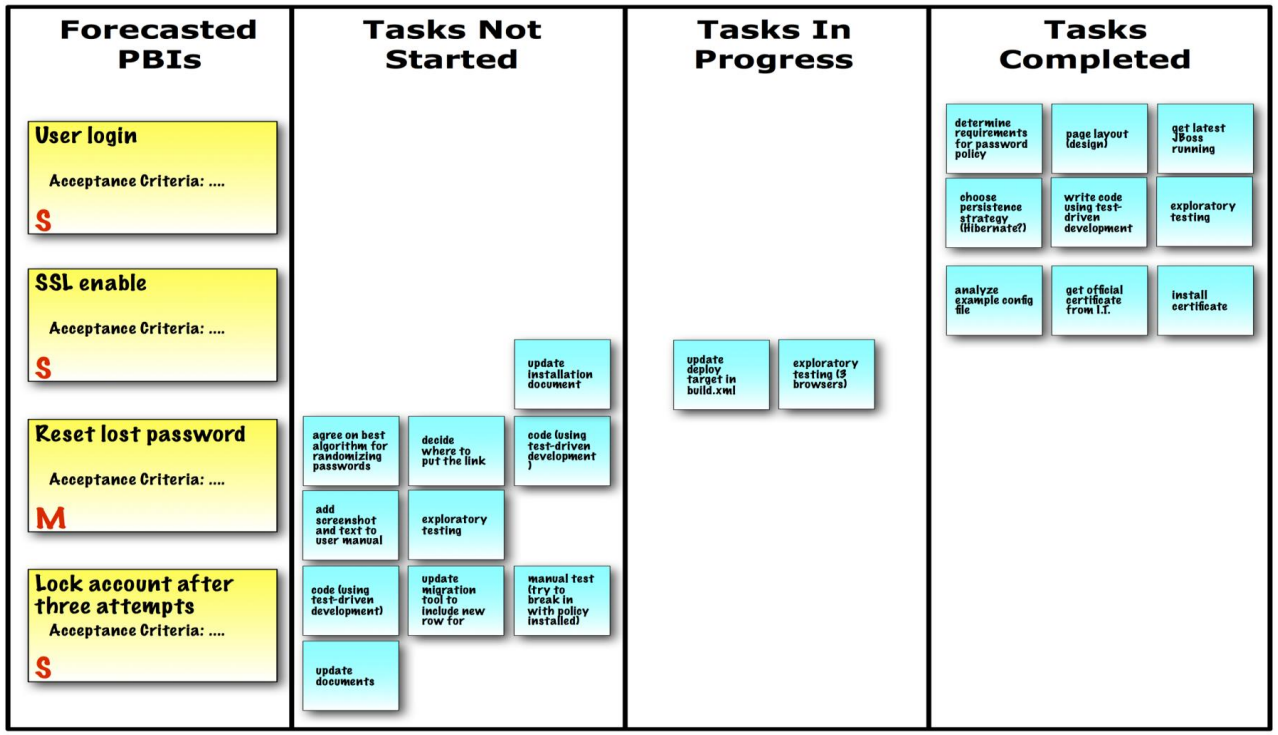
Sprint Review Meeting

- Η ομάδα παρουσιάζει αυτό που έχει επιτευχθεί κατά τη διάρκεια του Sprint στον Product Owner και άλλους stakeholders
- Συνήθως παίρνει τη μορφή ενός demo των νέων χαρακτηριστικών ή υποκείμενης αρχιτεκτονικής που έχει υλοποιηθεί σε αυτό το Sprint
- Η συνάντηση είναι συγκεκριμένης ώρας (timeboxed), περιπου 2 ώρες

Sprint Retrospective Meeting

- Στο τέλος του Sprint, ο Scrum Master, ο Product Owner και η development ομάδα συζητούν για το πώς πήγε το σπριντ.
- **Τι πήγε καλά, τι δεν πήγε καλά και τι μπορεί να βελτιωθεί στο επόμενο sprint**
- Τυπικά 15–30 λεπτά και αυτό βοηθά την ομάδα να γίνεται καλύτερη και να λειτουργά καλυτέρα στο επόμενο σπριντ.
- Αυτές οι βελτιώσεις ενσωματώνονται στο επόμενο Sprint.

FURTHER SPRINT DETAILS – THE SPRINT BACKLOG



- Το Spring backlog είναι μια άκρως ορατή εικόνα σε πραγματικό χρόνο της δουλειάς που η ομάδα ανάπτυξης σχεδιάζει να ολοκληρώσει κατά τη διάρκεια του Sprint.

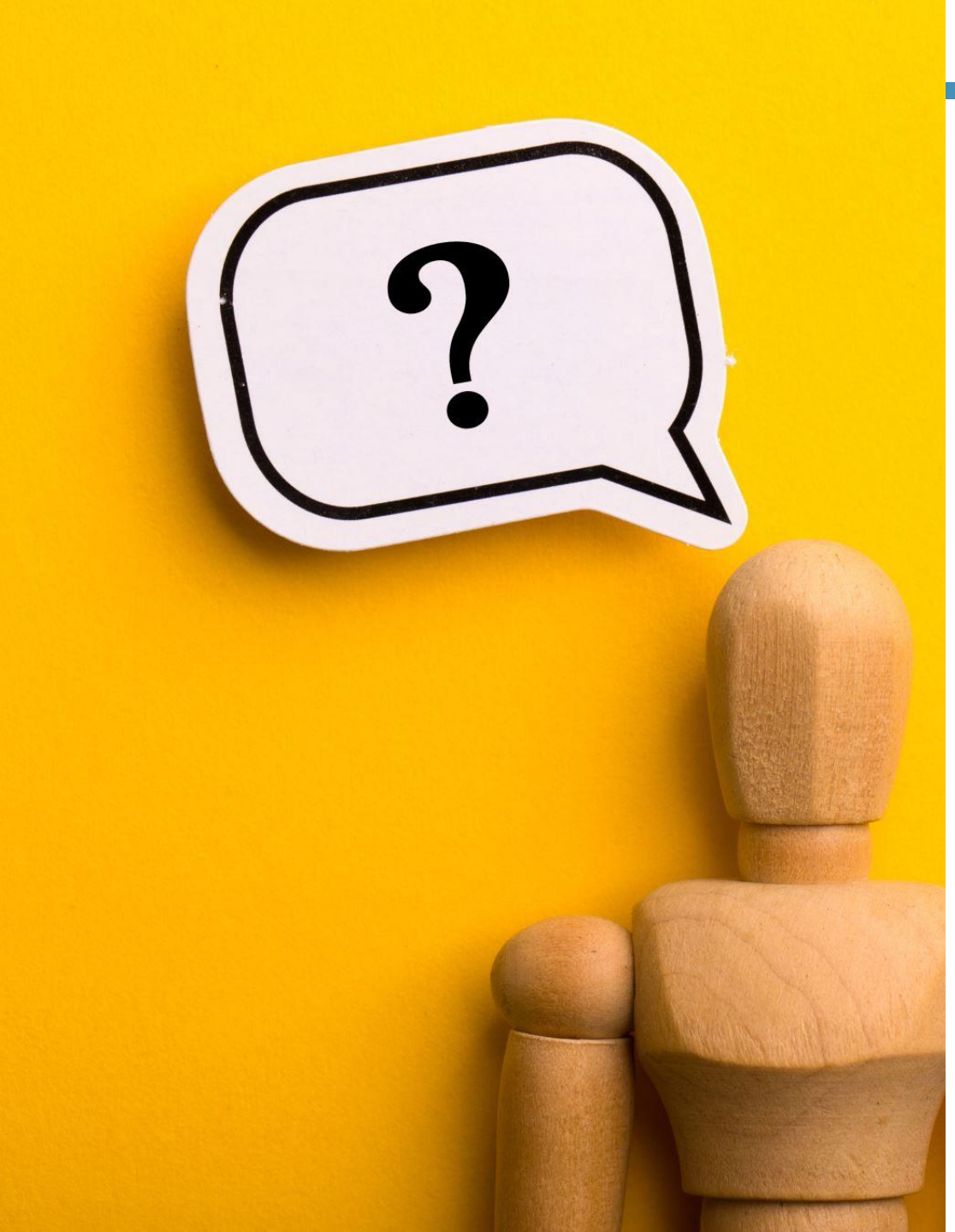
- Κατά την διάρκεια του **Sprint**, το Sprint Backlog (περιλαμβάνει τις ιστορίες χρήστη που θα υλοποιηθούν), χρησιμοποιεί ένα πίνακα Scrum για να οπτικοποιήσει την πρόοδο της εργασίας καθώς μια ιστορία χρήστη προχωρά από το "προς εκτέλεση (to do)" μέχρι την «υλοποίηση της (done)"

SPRINT FURTHER DETAILS

Το επόμενο **Sprint** αρχίζει αμέσως μετά την ολοκλήρωση του προηγούμενο χωρίς καθυστέρηση.

Κάθε Sprint περιλαμβάνει

- **Sprint Planning**
- **Daily Scrum**
- **Development work**
- **Product Backlog Refinement**
- **Sprint Review**
- **Sprint Retrospective**



BREAKOUT SESSION

Instructions: Fill in the blanks with the appropriate letter of the Scrum terms

____ Is a time-box - a specific set of time

____ Value: Do the right thing and work on tough problems

____ User story contains enough details to be worked

____ Vertical slice of a potentially shippable product

____ Visual aid that shows trending of user stories completed

____ Share lessons learned from the previous iteration

____ Servant leader that removes roadblocks

____ Value: Complete the right work based on sprint goals

____ Meeting to demo progress made during the iteration

____ Simple description of a feature from a user's perspective

____ Committed user stories for upcoming iteration

____ Tasked with understanding the product vision

____ Discussion and commitment for the next iteration's work

____ Value: Treat everyone as a capable team member

____ Prioritized list of user stories that grows and changes

____ Common understanding of when something is complete

____ Discussion of yesterday's progress, today's plan, and issues

____ Team-created list of rules, expectations, and procedures

____ Value: Discuss the work and challenges of the project

____ Self-organizing, cross-functional group of people

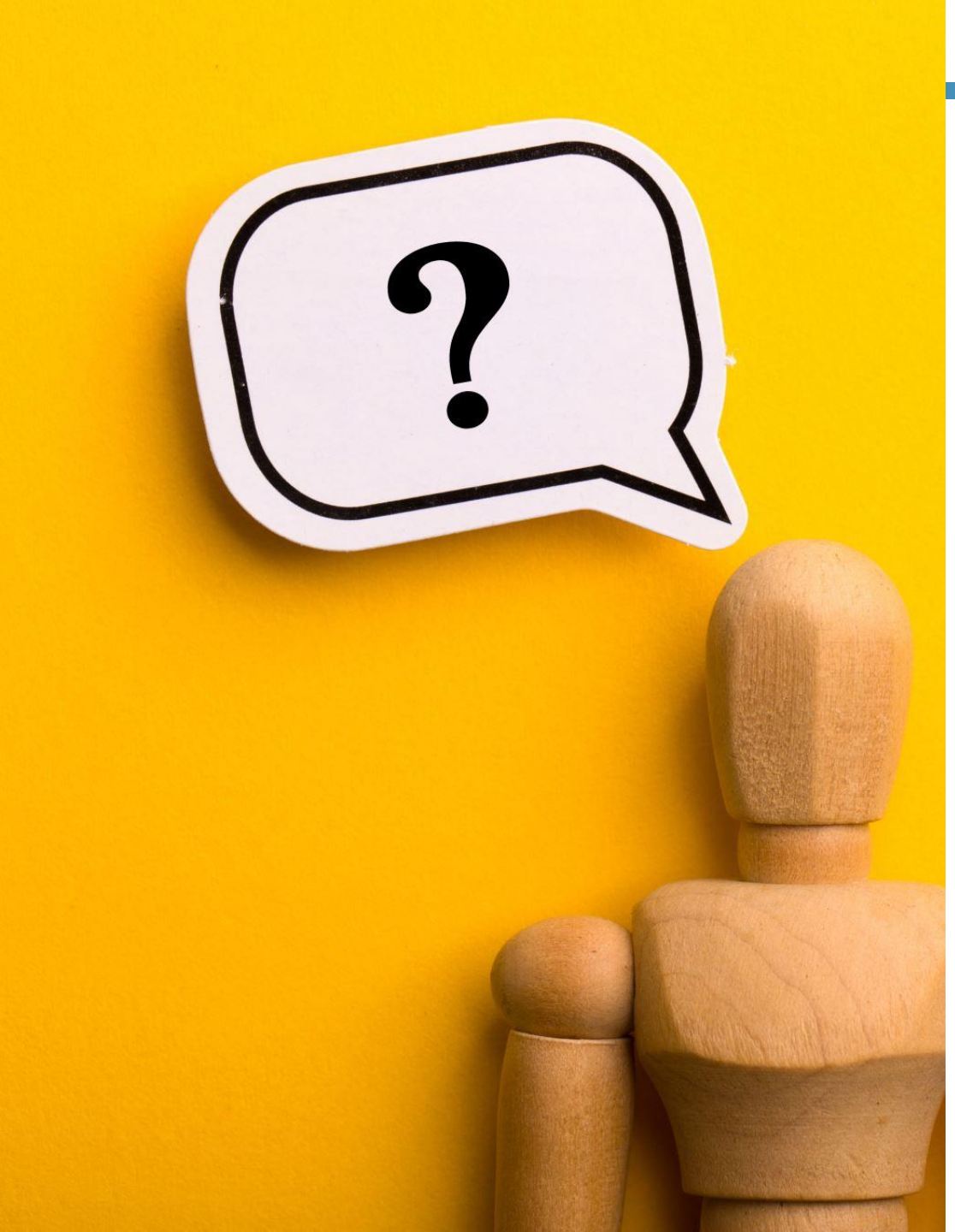
____ Compares total user stories versus completed user stories

____ Number of story points that are completed per iteration

____ Value: Give 100% effort to achieve project team's goals

____ Work that is currently being completed

A. Burn Up Chart
B. Burndown Chart
C. Commitment
D. Courage
E. Daily Standup
F. Definition of Done
G. Definition of Ready
H. Focus
I. Openness
J. Product Increment
K. Product Backlog
L. Product Owner
M. Respect
N. Development Team
O. Scrum Master
P. Sprint
Q. Sprint Backlog
R. Sprint Planning
S. Sprint Retrospective
T. Sprint Review
U. Team Velocity
V. User Story
W. Work in Progress
X. Working Agreement



BREAKOUT SESSION (Product Backlog με User Stories)

User Story	Story Points	Προτεραιότητα
Ως χρήστης, θέλω να μπορώ να εγγραφώ στην εφαρμογή με email και κωδικό, ώστε να έχω προσωπικό λογαριασμό.	5	Υψηλή
Ως χρήστης, θέλω να επαναφέρω τον κωδικό μου μέσω email, ώστε να έχω πρόσβαση στον λογαριασμό μου εάν τον ξεχάσω.	3	Μεσαία
Ως διαχειριστής, θέλω να μπορώ να βλέπω όλους τους εγγεγραμμένους χρήστες σε έναν πίνακα, ώστε να μπορώ να διαχειρίζομαι το σύστημα.	8	Υψηλή
Ως χρήστης, θέλω να μπορώ να προσθέτω προϊόντα στο καλάθι αγορών μου, ώστε να ολοκληρώσω τις αγορές μου ευκολότερα.	5	Υψηλή
Ως χρήστης, θέλω να μπορώ να αφαιρώ προϊόντα από το καλάθι αγορών μου, ώστε να κάνω αλλαγές πριν την ολοκλήρωση της παραγγελίας.	3	Μεσαία
Ως χρήστης, θέλω να μπορώ να βλέπω το ιστορικό παραγγελιών μου, ώστε να γνωρίζω τις προηγούμενες αγορές μου.	8	Υψηλή
Ως διαχειριστής, θέλω να λαμβάνω ειδοποιήσεις όταν ένα προϊόν έχει χαμηλό απόθεμα, ώστε να διατηρώ τα αποθέματα επαρκή.	13	Χαμηλή
Ως χρήστης, θέλω να λαμβάνω ειδοποίηση μέσω email όταν η παραγγελία μου αποσταλεί, ώστε να γνωρίζω την κατάσταση της παραγγελίας μου.	5	Μεσαία
Ως διαχειριστής, θέλω να έχω αναφορές πωλήσεων ανά μήνα, ώστε να μπορώ να αναλύω την απόδοση του καταστήματος.	13	Χαμηλή

Sprint Goal: "Να επιτρέψουμε στους χρήστες να εγγράφονται, να διαχειρίζονται το καλάθι τους και να παρακολουθούν τις παραγγελίες τους, ενώ οι διαχειριστές μπορούν να διαχειρίζονται χρήστες και αποθέματα."

1. Σε ομάδες επιλέξετε ποια User Stories θα ενσωματώσετε στο Sprint Backlog για να υλοποιηθούν στο Sprint. Λάβεται υπόψη το Sprint Goal και τη **σημασία των User Stories** (τι έχει υψηλότερη προτεραιότητα)

Για τους χρήστες (User Stories)

1.Εγγραφή χρήστη – (Story Points: 5)

"Ως χρήστης, θέλω να μπορώ να εγγραφώ στην εφαρμογή με email και κωδικό, ώστε να έχω προσωπικό λογαριασμό."

2.Προσθήκη προϊόντων στο καλάθι – (Story Points: 5)

"Ως χρήστης, θέλω να μπορώ να προσθέτω προϊόντα στο καλάθι αγορών μου, ώστε να ολοκληρώσω τις αγορές μου ευκολότερα."

3.Ιστορικό παραγγελιών – (Story Points: 8)

"Ως χρήστης, θέλω να μπορώ να βλέπω το ιστορικό παραγγελιών μου, ώστε να γνωρίζω τις προηγούμενες αγορές μου."

Για τους διαχειριστές (Admin User Stories)

4.Προβολή εγγεγραμμένων χρηστών – (Story Points: 8)

"Ως διαχειριστής, θέλω να μπορώ να βλέπω όλους τους εγγεγραμμένους χρήστες σε έναν πίνακα, ώστε να μπορώ να διαχειρίζομαι το σύστημα."

5.Ειδοποιήσεις για χαμηλό απόθεμα – (Story Points: 13)

"Ως διαχειριστής, θέλω να λαμβάνω ειδοποιήσεις όταν ένα προϊόν έχει χαμηλό απόθεμα, ώστε να διατηρώ τα αποθέματα επαρκή."

Όμως τι είναι το πρόβλημα εδώ?

"Ως διαχειριστής, θέλω να λαμβάνω ειδοποιήσεις όταν ένα προϊόν έχει χαμηλό απόθεμα, ώστε να διατηρώ τα αποθέματα επαρκή. → *Έχει χαμηλή προτεραιότητα.*

Όταν το Sprint Goal περιλαμβάνει User Stories χαμηλής προτεραιότητας, σημαίνει ότι **ίσως ο στόχος δεν είναι καλά ευθυγραμμισμένος με τις προτεραιότητες του Product Owner**.
➞ **Σε μια πραγματική Sprint Planning συνάντηση,** η ομάδα και ο **Product Owner** θα το συζητούσαν και πιθανότατα **θα άλλαζαν το Sprint Goal** για να αντικατοπτρίζει καλύτερα τις πιο σημαντικές απαιτήσεις.

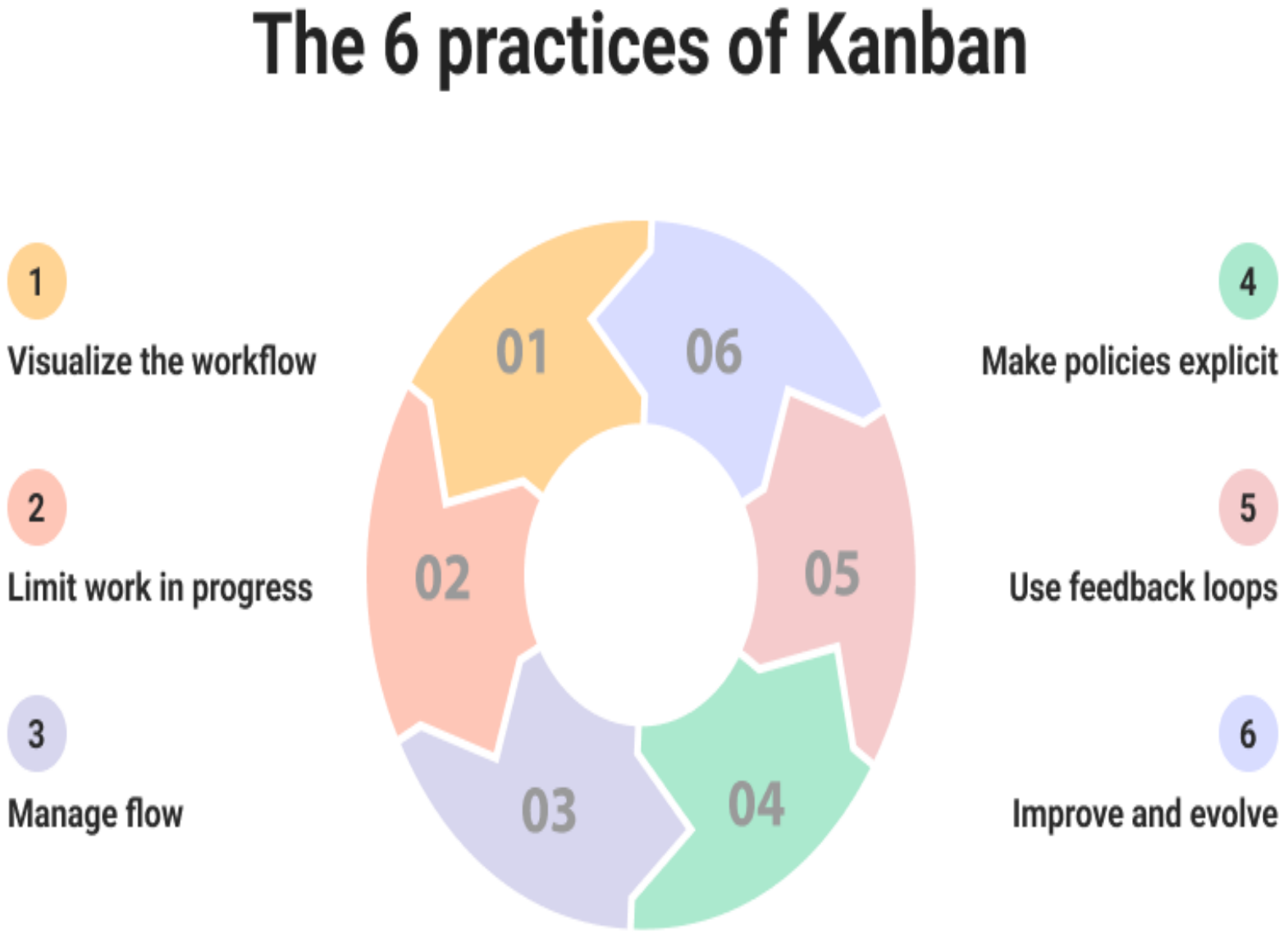
Άρα, η σωστή διαδικασία είναι:
1️⃣ Ο Product Owner προτεραιοποιεί το Product Backlog.
2️⃣ Η ομάδα επιλέγει υψηλής προτεραιότητας User Stories.
3️⃣ Αν το Sprint Goal περιλαμβάνει χαμηλής προτεραιότητας ιστορίες, τότε πρέπει να αναθεωρηθεί.
4️⃣ Η ομάδα προσαρμόζει το Sprint Goal ώστε να αντικατοπτρίζει καλύτερα τις πραγματικές προτεραιότητες.

Νέο Sprint Goal (χωρίς τη χαμηλής προτεραιότητας ιστορία):
✅ "Να επιτρέψουμε στους χρήστες να εγγράφονται, να διαχειρίζονται το καλάθι τους και να παρακολουθούν τις παραγγελίες τους, ενώ οι διαχειριστές μπορούν να διαχειρίζονται τους χρήστες του συστήματος."

APPLICABILITY OF SCRUM

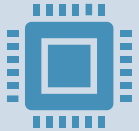
- **Δεν είναι κατάλληλο για κάθε έργο ανάπτυξης συστημάτων, όπως**
  - Μεγάλες και κατανεμημένες ομάδες (αν και υπάρχουν σήμερα άλλα πλαίσια που προσπαθούν να το αντιμετωπίσουν αυτό, π.χ. LESS και SAFE)
  - Κουλτούρες στις οποίες η ομάδα δεν μπορεί να ενδυναμωθεί
  - Έργα στα οποία δεν είναι εφικτή η συνεχής ολοκλήρωση(**continuous integration**) και δοκιμή του συστήματος
  - Κουλτούρες που αντιστέκονται σε σημαντικές αλλαγές
- **Ελάχιστες κατευθυντήριες γραμμές για το κομμάτι της τεχνολογίας λογισμικού**
  - Ο τρόπος επέκτασης των απαιτήσεων όσον αφορά την ανάλυση, τη σχεδίαση, τη μοντελοποίηση και τον προγραμματισμό δεν αντιμετωπίζεται από το SCRUM
  - Να συμπληρώνετε πάντα το SCRUM με άλλη μεθοδολογία, όπως η XP

KANBAN


The 6 practices of Kanban
1
Visualize the workflow
2
Limit work in progress
3
Manage flow
01
02
03
04
05
06
4
Make policies explicit
5
Use feedback loops
6
Improve and evolve

KANBAN

Το Kaban είναι ένα Agile πλαίσιο διαχείρισης έργου που χρησιμοποιείται για την οπτικοποίηση της ροής εργασίας. Μια οπτική μέθοδος διαχείρισης εργασιών που βοηθά τις ομάδες να βελτιώσουν τη ροή εργασίας και να μειώσουν τις καθυστερήσεις.



Αναπτύχθηκε από την **Toyota** για τη βελτιστοποίηση της παραγωγής.

Εφαρμόζεται ευρέως στη **διαχείριση έργων και ανάπτυξη λογισμικού**.

Οπτικοποίηση της εργασίας μέσω ενός **Kanban Board**.

Περιορισμός του Work In Progress (WIP) για αποφυγή υπερφόρτωσης.

Συνεχής βελτίωση των διαδικασιών μέσω ανάλυσης των ροών εργασίας.

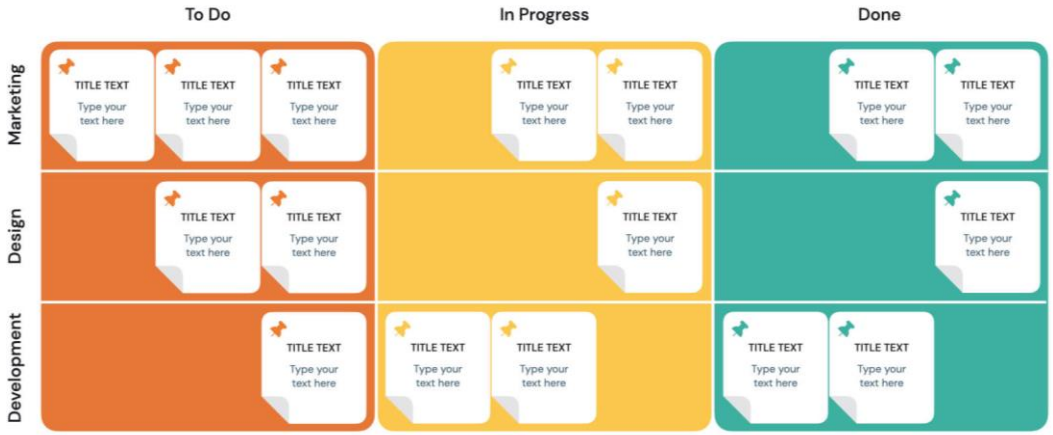
ΒΑΣΙΚΕΣ ΑΡΧΕΣ ΤΗΣ KANBAN

📌 **Οπτικοποίηση της εργασίας** – Χρήση Kanban board για να δούμε τη ροή εργασίας.
📌 **Περιορισμός του Work In Progress (WIP)** – Μείωση του αριθμού των εργασιών που εκτελούνται ταυτόχρονα.
📌 **Διαχείριση της ροής** – Παρακολούθηση και βελτίωση της διαδικασίας εργασίας.
📌 **Ρητές πολιτικές διαδικασίας** – Καθορισμός ξεκάθαρων κανόνων λειτουργίας.
📌 **Εφαρμογή ανατροφοδότησης** – Συνεχής βελτίωση με βάση τα δεδομένα και τα σχόλια των ομάδων.
📌 **Συνεργασία και πειραματισμός** – Προσαρμογή στις αλλαγές με συλλογική μάθηση.

KANBAN BOARD - ΟΠΤΙΚΟΠΟΙΗΣΗ ΤΗΣ ΡΟΗΣ ΕΡΓΑΣΙΑΣ

Το **Kanban Board** αποτελείται από **στήλες** που αντιπροσωπεύουν τις φάσεις μιας εργασίας.

- **Backlog** – Ανάθεση νέων εργασιών
- **To Do** – Εργασίες προς έναρξη
- **In Progress** – Εργασίες σε εξέλιξη
- **Review / Testing** – Έλεγχος και αξιολόγηση
- **Done** – Ολοκληρωμένες εργασίες



Διαχείριση Ροής

•Οι εργασίες κινούνται **από αριστερά προς τα δεξιά**.

•Οι ομάδες **παρακολουθούν** τη ροή εργασίας σε πραγματικό χρόνο.

•**Περιορίζεται** ο αριθμός των εργασιών σε κάθε στάδιο για αποφυγή bottlenecks.

•Οι ομάδες **αναλύουν και βελτιστοποιούν** συνεχώς την παραγωγικότητά τους.

•**Προσαρμογή προτεραιοτήτων** ανάλογα με τις ανάγκες της ομάδας

Kanban in 1 minute

SCRUM VS KANBAN

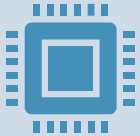
Χαρακτηριστικό	Kanban	Scrum
Τρόπος εργασίας	Συνεχής ροή εργασιών (**flow-based**)	Επαναλήψεις εργασίας σε **sprints**
Sprint	❌ Δεν έχει sprints	✅ Χρησιμοποιεί sprints (π.χ., 2 εβδομάδες)
Ανάθεση εργασιών	📌 Τα μέλη επιλέγουν εργασίες από το Kanban Board βάσει ροής και προτεραιοτήτων	📌 Η ομάδα επιλέγει tasks κατά το Sprint Planning (όχι ο Scrum Master)
WIP Limits	✅ Υπάρχουν (Work In Progress limits για έλεγχο ροής)	❌ Δεν υπάρχουν αυστηρά όρια WIP
Σταθερό backlog	❌ Όχι, το backlog είναι δυναμικό και μπορεί να αλλάξει συνεχώς	✅ Ναι, το Sprint Backlog κλειδώνει μέχρι το τέλος του sprint
Ρόλοι στην ομάδα	Ευέλικτοι, χωρίς αυστηρούς ρόλους	Συγκεκριμένοι ρόλοι: **Scrum Master, Product Owner, Development Team**
Αλλαγές στις εργασίες	✅ Μπορούν να αλλάξουν ανά πάσα στιγμή	❌ Δεν επιτρέπονται αλλαγές στα tasks μέσα στο sprint
Τρόπος βελτίωσης	Συνεχής βελτίωση (**Kaizen**) μέσω δεδομένων και ροής εργασιών	Ανασκόπηση μετά από κάθε sprint (**Sprint Retrospective**)
Καταλληλότητα	✅ Ιδανικό για ομάδες με **μεταβαλλόμενες προτεραιότητες**	✅ Ιδανικό για ομάδες που δουλεύουν με **σταθερούς στόχους** ανά sprint

ΜΕΙΟΝΕΚΤΗΜΑΤΑ ΤΗΣ KANBAN

- **Έλλειψη χρονικών πλαισίων (Timeboxing)**
- **Ασαφείς ρόλοι και ευθύνες**
- **Δύσκολη επιβολή του Work in Progress (WIP) Limit**
- **Δεν είναι ιδανική για αυστηρά δομημένα έργα:** Για ομάδες που απαιτούν προβλέψιμα χρονοδιαγράμματα, αυστηρό σχεδιασμό και καθορισμένες προθεσμίες, η Kanban μπορεί να φαίνεται υπερβολικά ευέλικτη και να δίνει την αίσθηση έλλειψης ελέγχου.
- **Απαιτεί υψηλό επίπεδο αυτοπειθαρχίας:** Οι εργασίες δεν ανατίθενται από κάποιον υπεύθυνο αλλά επιλέγονται από τα μέλη της ομάδας. Αν η ομάδα δεν έχει αρκετή πρωτοβουλία και υπευθυνότητα, η παραγωγικότητα μπορεί να μειωθεί.
- **Μπορεί να είναι δύσκολη στη διαχείριση για μεγάλες ομάδες**: Σε μεγάλες ομάδες, ο πίνακας Kanban μπορεί να γίνει **πολύπλοκος και γεμάτος**, καθιστώντας δύσκολη την παρακολούθηση της προόδου.
- **Δυσκολία στη μέτρηση της προόδου σε υψηλό επίπεδο:** Χωρίς sprints, velocity tracking ή καθορισμένα ορόσημα, η Kanban δεν προσφέρει ξεκάθαρη εικόνα της συνολικής προόδου για τους stakeholders, κάνοντας δύσκολη την αναφορά εξέλιξης.
- **Μπορεί να οδηγήσει σε υπερφόρτωση εργασιών (Task Hoarding)**: Τα μέλη της ομάδας ενδέχεται να επιλέγουν πολλές εργασίες ταυτόχρονα, γεγονός που μπορεί να προκαλέσει multitasking, καθυστερήσεις και μειωμένη εστίαση.

DEBATES WITHIN AGILE METHODS

Το γεγονός ότι δεν υπάρχει λεπτομερής τεκμηρίωση και επίσης κάποιες διαδικασίες είναι πιο άτυπες(Informal) καθιστά τις ευέλικτες μεθοδολογίες ως ασύμβατες με νομικές διαδικασίες που πρέπει να γίνουν για τη δημιουργία συμβατικής συμφωνίας με μεγάλες εταιρίες.

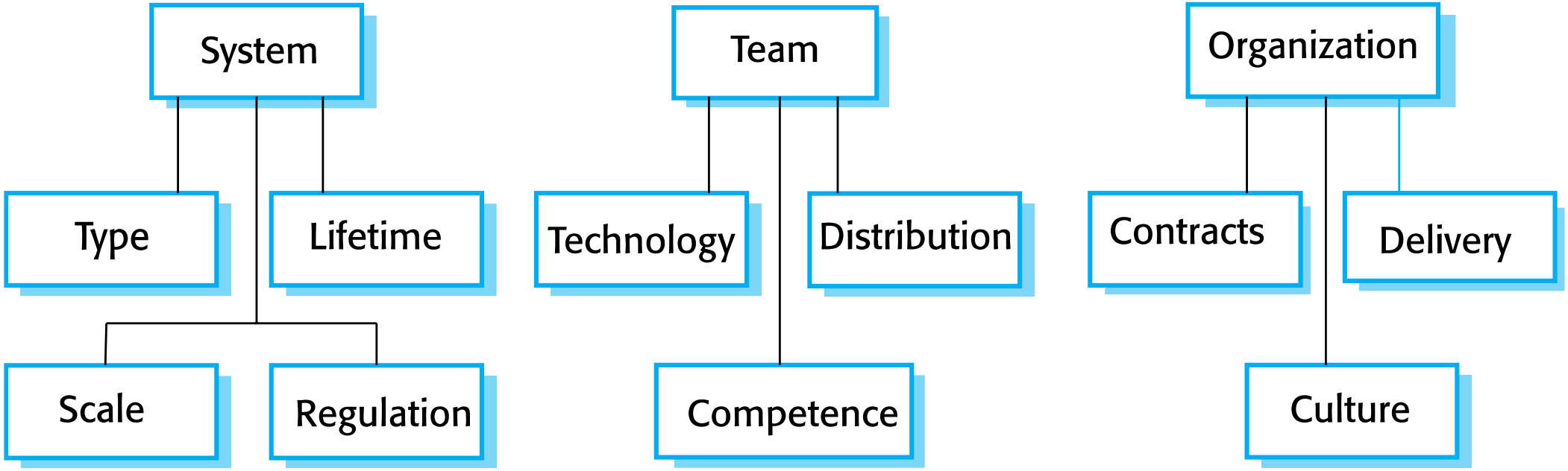


Οι ευέλικτες μέθοδοι είναι καταλληλότερες για την ανάπτυξη νέου λογισμικού παρά για τη συντήρηση λογισμικού.

Ωστόσο, η πλειονότητα του κόστους λογισμικού στις μεγάλες εταιρείες προέρχεται από τη συντήρηση των υφιστάμενων συστημάτων λογισμικού.

Οι ευέλικτες μέθοδοι έχουν σχεδιαστεί για μικρές ομάδες που βρίσκονται μαζί, αν και υπάρχει τώρα μια κίνηση για την κλιμάκωση των ευέλικτων μεθόδων σε κατανεμημένα πλαίσια, π.χ. τα πλαίσια LESS και SAFE.

Agile or Plan based development
Factors to consider



DEVOPS

DevOps

Απαλλαγμένη από όλη την ορολογία, η μεθοδολογία DevOps είναι αρκετά απλή. Βάζει απλώς τις ομάδες ***Ανάπτυξης (Development)*** *και* ***Λειτουργιών (Operations)*** *στην ίδια σελίδα. Οι προγραμματιστές και οι ειδικοί των λειτουργιών κατανοούν τις ανάγκες και τους περιορισμούς ο ένας του άλλου (και απαλλάσσονται από τυχόν ανταγωνιστική νοοτροπία, εάν υπάρχει) και σχεδιάζουν ομαλά χρονοδιαγράμματα για την επίτευξη επιχειρηματικών στόχων. Η ανάπτυξη, ο έλεγχος και η επικύρωση(testing) προχωρούν σχεδόν ταυτόχρονα, επιτρέποντας ευέλικτη ανάπτυξη (agile development)."*

DevOps
Γεφυρώνοντας την Ανάπτυξη & τις Λειτουργίες

- Το DevOps είναι ένα σύνολο πρακτικών, εργαλείων και πολιτιστικής φιλοσοφίας που ενσωματώνει την ανάπτυξη λογισμικού (**Dev**) και τις IT λειτουργίες (**Ops**) με στόχο τη βελτίωση της αποδοτικότητας, της ταχύτητας και της ποιότητας.
- Καταργεί τα σιλό μεταξύ των ομάδων ανάπτυξης και λειτουργιών, επιτρέποντας τη συνεχή ενσωμάτωση (CI) και συνεχή παράδοση (CD).
- Αυτοματοποιεί και παρακολουθεί κάθε στάδιο της ανάπτυξης λογισμικού, από τον προγραμματισμό έως την υλοποίηση.

- **Βασικές Αρχές του DevOps:**

🛠 **Συνεργασία** – Οι προγραμματιστές και οι IT ομάδες εργάζονται μαζί.
🔁 **Αυτοματοποίηση** – Μειώνει τις χειροκίνητες διαδικασίες και τα σφάλματα.
🚀 **Συνεχής Ενσωμάτωση & Παράδοση (CI/CD)** – Επιτρέπει πιο γρήγορες κυκλοφορίες.
📊 **Παρακολούθηση & Ανατροφοδότηση** – Διασφαλίζει απόδοση, ασφάλεια και επεκτασιμότητα.

DevOps
Building on best industry practices

- Χρησιμοποιεί στοιχεία και εμπειρία από την Agile ανάπτυξη, τη Lean παραγωγή και το ITSM.
- Προσφέρει μεγαλύτερη ευελιξία πάνω στο Agile μοντέλο, διασφαλίζοντας ότι η ομάδα λειτουργιών (Operations) θα αποτελεί μέρος της ομάδας ανάπτυξης (Development).
  - Στόχος είναι η ελαχιστοποίηση του χάσματος μεταξύ των ομάδων και η βελτίωση της επικοινωνίας.

- Εστιάζει σε πρακτικές όπως η **Συνεχής Ενσωμάτωση (Continuous Integration)** και η **Συνεχής Παράδοση (Continuous Delivery)**
  - Οι συχνές αναπτύξεις (deployments) διασφαλίζουν τακτικές κυκλοφορίες και ότι οι νέες εκδόσεις λειτουργούν σωστά και καλύπτουν τις ανάγκες των πελατών.
- Με τη χρήση αποτελεσματικών εργαλείων, η επαναλαμβανόμενη εργασία μπορεί να αυτοματοποιηθεί και να βελτιωθεί η διαφάνεια.
  - Έτσι, όλα τα εμπλεκόμενα μέρη έχουν πλήρη ορατότητα στο έργο.

DevOps central concept is to manage end-to-end engineering processes.

DevOps
Οφέλη και μειονεκτήματα

✅ **Γρηγορότερη διάθεση στην αγορά** – Οι αυτοματοποιημένοι μηχανισμοί και οι CI/CD pipelines επιταχύνουν την παράδοση λογισμικού.

✅ **Βελτιωμένη Συνεργασία** – Ενισχύει τη συνεργασία μεταξύ προγραμματιστών, δοκιμαστών και λειτουργιών.

✅ **Υψηλότερη Ποιότητα & Αξιοπιστία** – Οι αυτοματοποιημένες δοκιμές μειώνουν τα σφάλματα και βελτιώνουν τη σταθερότητα.

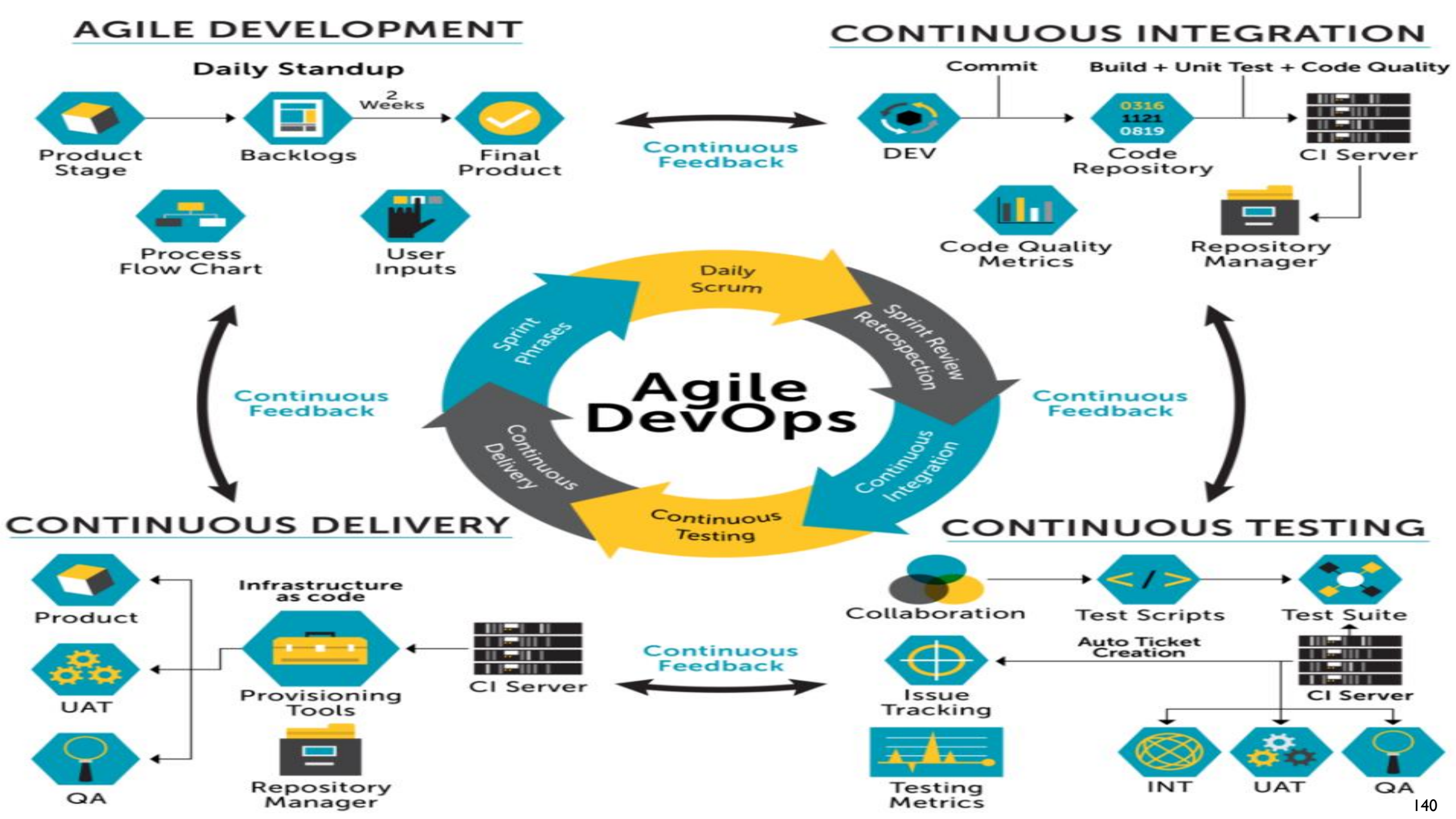
✅ **Επεκτασιμότητα & Ευελιξία** – Το cloud-native DevOps επιτρέπει την εύκολη κλιμάκωση εφαρμογών.

✅ **Ασφάλεια & Συμμόρφωση** – Ενσωματώνει πρακτικές ασφαλείας (**DevSecOps**) για ασφαλείς κυκλοφορίες λογισμικού.

Cons

- Πίεση για εργασία τόσο σε παλιό όσο και σε νέο κώδικα.
- Μεγάλη εξάρτηση από αυτοματοποιημένα εργαλεία για την αποτελεσματικότητα της διαδικασίας.
- Η ανάπτυξη κώδικα μπορεί να επηρεάσει το περιβάλλον παραγωγής.
- Απαιτείται εξειδικευμένη ομάδα ανάπτυξης.

Το DevOps μεταμορφώνει τον κλάδο του IT, επιτρέποντας γρήγορη, αποδοτική και αξιόπιστη ανάπτυξη λογισμικού.


AGILE DEVELOPMENT
Daily Standup
Product Stage
Backlogs
2 Weeks
Final Product
Process Flow Chart
User Inputs
Continuous Feedback
CONTINUOUS INTEGRATION
Commit
Build + Unit Test + Code Quality
DEV
0316
1121
0819
Code Repository
CI Server
Code Quality Metrics
Repository Manager
Continuous Feedback
Daily Scrum
Sprint Review Retrospection
Continuous Integration
Continuous Testing
Continuous Delivery
Sprint Phrases
Agile DevOps
Continuous Feedback
CONTINUOUS DELIVERY
Product
UAT
QA
Infrastructure as code
Provisioning Tools
Repository Manager
CI Server
Continuous Feedback
CONTINUOUS TESTING
Collaboration
Test Scripts
Test Suite
Auto Ticket Creation
CI Server
Issue Tracking
Testing Metrics
INT
UAT
QA

HOW DID WE DO THIS WEEK?
WHAT DID WE LEARN?

Reply at:
PollEv.com/avgoustakyri977

1. Go to **PollEv.com**
2. Enter **AVGOUSTAKYRI977**
3. Respond to activity

1. Text **AVGOUSTAKYRI977** to **22333**
2. Text in your message

ΑΝΑΓΝΩΣΕΙΣ/ΑΝΑΦΟΡΕΣ

- Kung, D.C (2024) Software Engineering, 2nd Edition, McGraw Hill – Chapter 1
- Ε. Α. Γιακουμάκης, Ν. Α. Διαμαντίδης, Τεχνολογία Λογισμικού, 2η έκδοση, Unibooks, 2021 – Κεφάλαιο 1
- Pressman, R.S. (2020). Software Engineering: A practitioner's approach, 9th Edition, Mc-Graw Hill – Chapter 1
- Sommerville, I. (2016). Software Engineering, Pearson – Chapter 1