ΜΗΧΑΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ

Ενότητα 1η - ΕΠΙΣΚΟΠΗΣΗ

Κεφάλαιο 2ο – Διαδικασίες Λογισμικού

Δρ. ΒΙΔΑΚΗΣ ΝΙΚΟΣ

Περασμένη εβδομάδα

- Εισαγωγή στο γνωστικό αντικείμενο
- Εισαγωγή στην διδακτική διαδικασία
- Επισκόπηση ορισμών μηχανικής λογισμικού
- Εξοικείωση με βασικούς όρους

Το σημερινό μάθημα ασχολείται

- Διαδικασία Λογισμικού
- Αναγκαίες διαδικασίες υλοποίησης ενός συστήματος λογισμικού
- Μοντέλα Διαδικασιών Λογισμικού
- Αυτοματοποίηση των Διαδικασιών του Software Engineering - CASE

Διαδικασία Λογισμικού
(Software Process)

Διαδικασίες Λογισμικού (Software Processes)

- Συνεκτικές δραστηριότητες για να ορίσουμε, να σχεδιάσουμε, να υλοποιήσουμε και να ελέγξουμε λογισμικά συστήματα.

Η Διαδικασία Λογισμικού
(The software process)

- Ένα δομημένο σετ από διαδικασίες οι οποίες είναι αναγκαίες στην υλοποίηση ενός συστήματος λογισμικού.
  - Καθορισμός Προδιαγραφών / Απαιτήσεων (Specification)
  - Σχεδιασμός (Design)
  - Τεκμηρίωση (Validation)
  - Εξέλιξη (Evolution)
- Μοντέλο διαδικασίας λογισμικού είναι μια αφηρημένη απεικόνιση της διαδικασίας. Παρουσιάζει την περιγραφή μιας διαδικασίας από μια συγκεκριμένη διάσταση.

Προσδιορισμός Λογισμικού
(Software specification)

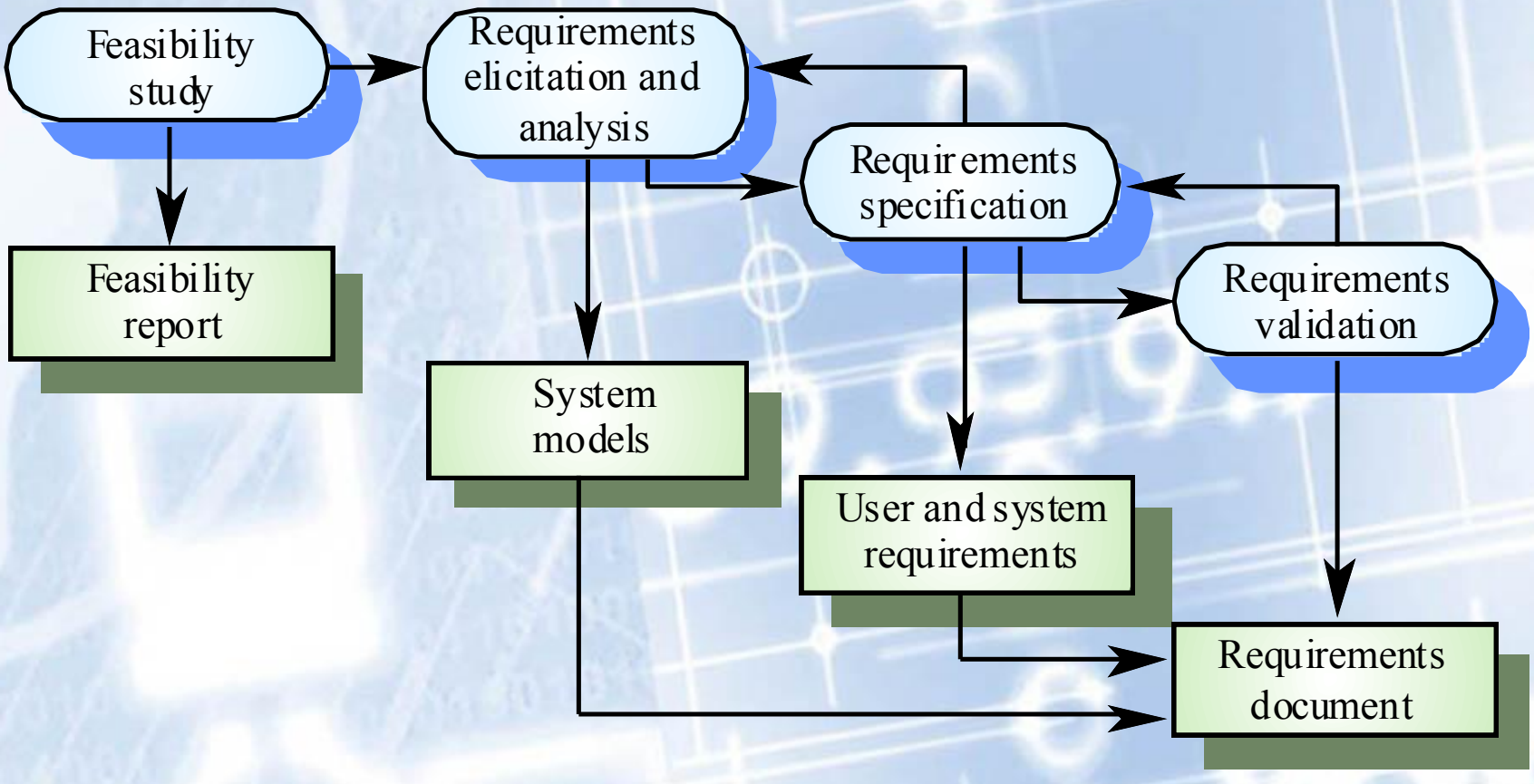
- Είναι η διαδικασία προσδιορισμού των απαιτήσεων που πρέπει να ληφθούν υπ' όψιν κατά την υλοποίηση και λειτουργία του συστήματος.
- Τεχνική της Διαδικασίας του προσδιορισμού των απαιτήσεων (Requirements engineering process)
  - Μελέτη σκοπιμότητας (Feasibility study)
  - Καταγραφή και ανάλυση Απαιτήσεων (Requirements elicitation and analysis)
  - Ορισμός Απαιτήσεων (Requirements specification)
  - Αξιολόγηση Απαιτήσεων (Requirements validation)

Αναγκαίες διαδικασίες υλοποίησης ενός συστήματος λογισμικού

The software process

- Καθορισμός Προδιαγραφών / Απαιτήσεων (Specification)
- Σχεδιασμός (Design)
- Υλοποίηση (Implementation)
- Επικύρωση/Τεκμηρίωση (Validation / Verification)
- Εξέλιξη (Evolution)

The requirements engineering process



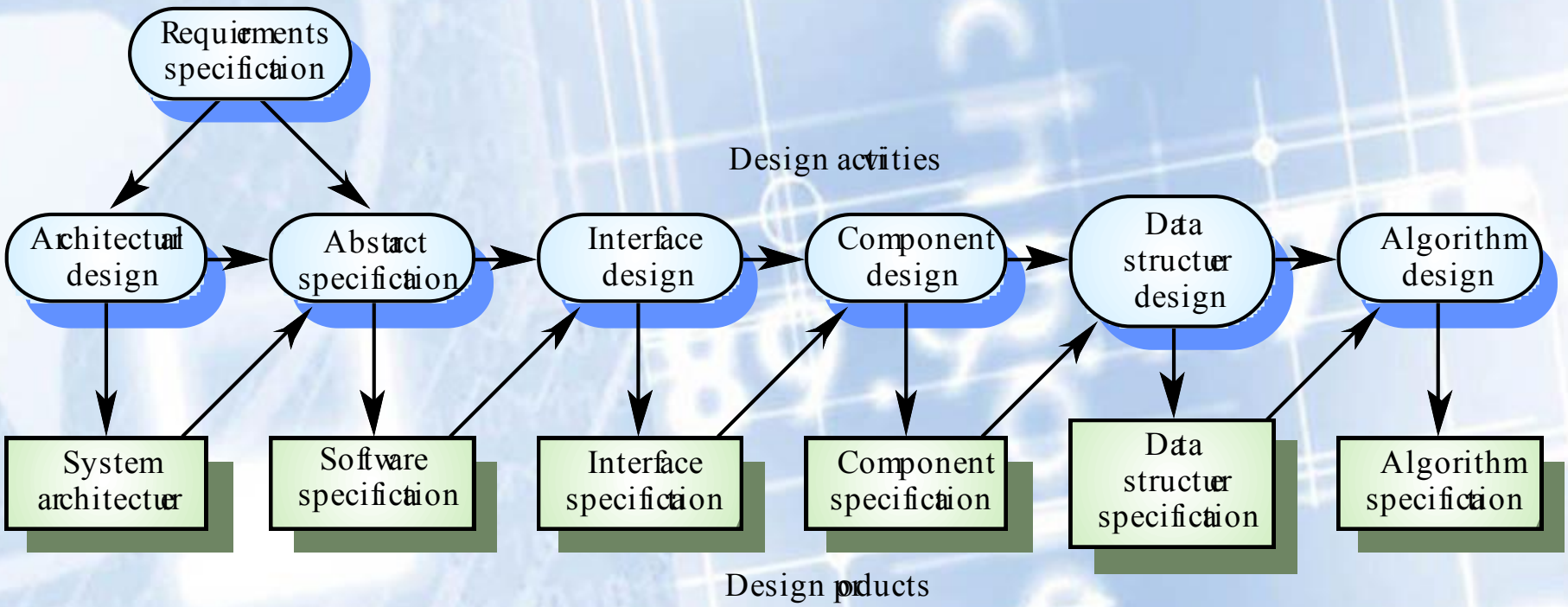
Software design and implementation

- Η Διαδικασία μετατροπής των προδιαγραφών του συστήματος σε πραγματοποιήσιμο σύστημα.
- Σχεδιασμός Λογισμικού
  - Σχεδιασμός λογισμικών δομών οι οποίες απεικονίζουν τις προδιαγραφές και τις απαιτήσεις του συστήματος.
- Υλοποίηση
  - Μετάφραση των σχεδιαστικών δομών σε κώδικα.
- Η σχεδίαση και η υλοποίηση είναι δύο στάδια με πολλά κοινά μεταξύ τους.

Δραστηριότητες της Διαδικασίας Σχεδιασμού.

- Αρχιτεκτονικός σχεδιασμός
- Αφηρημένος προσδιορισμός
- Σχεδιασμός διεπαφής
- Σχεδιασμός Κομματιών
- Σχεδιασμός δομών δεδομένων
- Σχεδιασμός αλγορίθμων

Διαδικασία Σχεδιασμού Λογισμικού



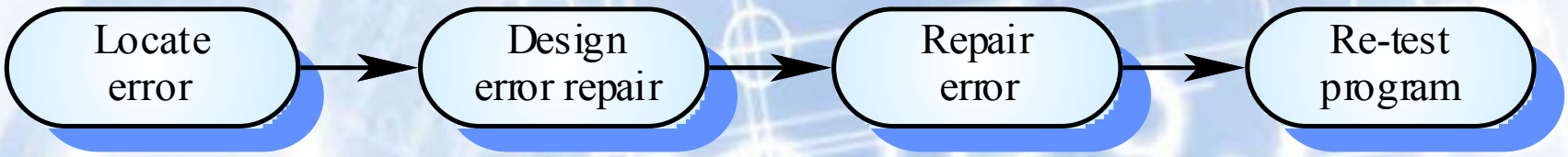
Μέθοδοι Σχεδιασμού

- Μέθοδος σχεδιασμού είναι η συστηματική προσέγγιση για την υλοποίηση του σχεδιασμού λογισμικού
- Ο σχεδιασμός συνήθως τεκμηριώνετε με γραφικά μοντέλα.
- Μοντέλα Σχεδιασμού
  - Data-flow model
  - Entity-relation-attribute model
  - Structural model
  - Object models

Προγραμματισμός & debugging

- Προγραμματισμός & debugging είναι η μετάφραση/μετατροπή του σχεδίου σε πρόγραμμα και ο προσδιορισμός και διόρθωση των λαθών.
- Για τον προγραμματισμό δεν υπάρχει μια γενική διαδικασία.
- Οι προγραμματιστές κάνουν έλεγχο στο κώδικα για να εντοπίσουν λάθη και να τα διορθώσουν στην διαδικασία του debugging.

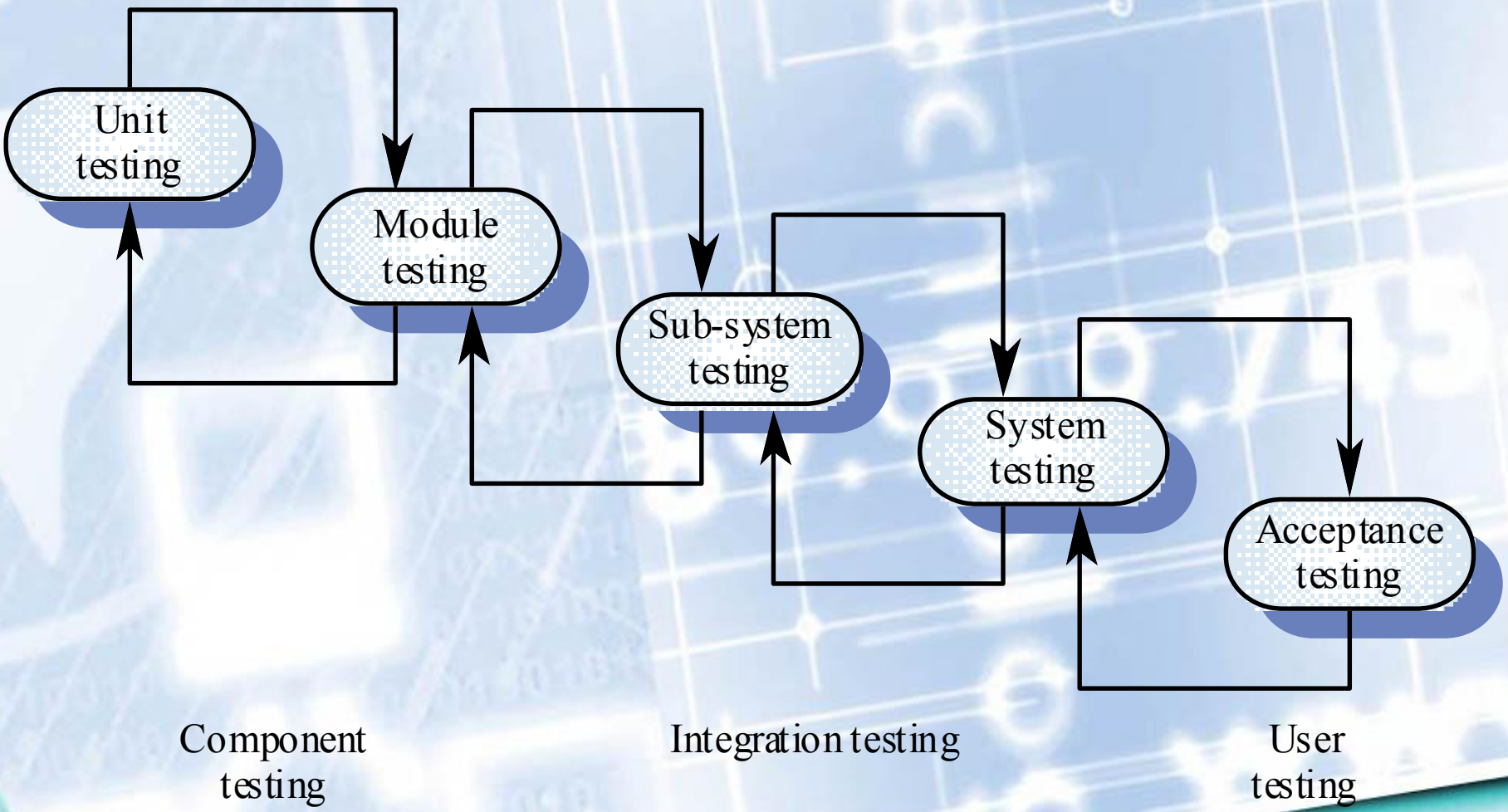
Η διαδικασία του debugging



Επικύρωση/Τεκμηρίωση Λογισμικού
(Software validation)

- Verification = Επαλήθευση, Διακρίβωση, Εξακρίβωση
- Validation = Επικύρωση, Τεκμηρίωση
- Το Verification & το validation, μας δίνουν την δυνατότητα να πιστοποιήσουμε ότι το σύστημα που αναπτύχθηκε είναι σύμφωνο με τις προδιαγραφές που μας δόθηκαν και τις απαιτήσεις του πελάτη.
- Το software validation συμπεριλαμβάνει έλεγχο των διαδικασιών και των διάφορων σταδίων του software engineering όπως επίσης και system testing
- Το system testing αποτελείτε από έλεγχο του συστήματος ο οποίος γίνεται χρησιμοποιώντας διάφορα σενάρια ελέγχου τα οποία έχουν σχεδιαστεί βάση δεδομένων από τις πραγματικές συνθήκες χρήσης του λογισμικού.

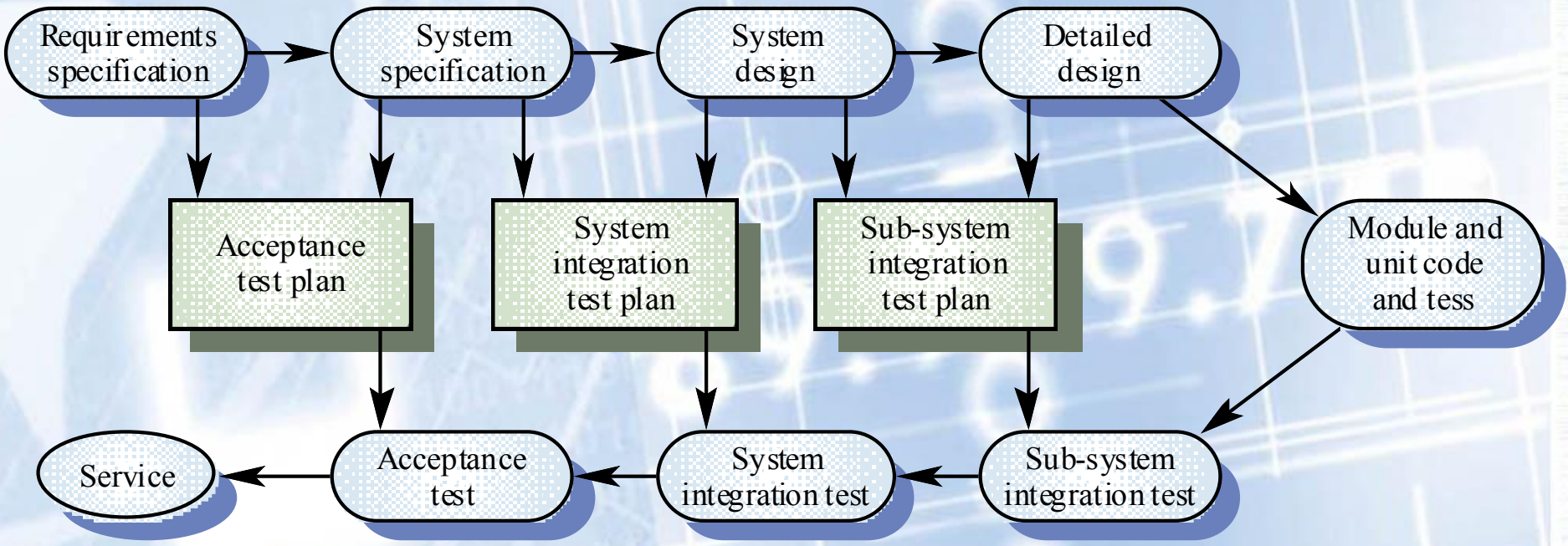
Η διαδικασία του testing



Τα στάδια του Testing

- Testing Μονάδας
  - Ξεχωριστές μονάδες του συστήματος ελέγχονται σε αυτό το στάδιο.
- Testing Ενότητας
  - Ομάδες μονάδων οι οποίες είναι αλληλοεξαρτώμενες ελέγχονται σε αυτό το στάδιο.
- Testing Υπο-συστήματος
  - Διάφορες ενότητες του συστήματος ενοποιούνται σε υπο-σύστημα και ελέγχεται η διασύνδεση τους.
- Testing Συστήματος
  - Τα υπο-συστήματα ενοποιούνται και ελέγχεται το συνολικό σύστημα.
- Testing Αποδοχής
  - Σε αυτό το στάδιο χρησιμοποιούνται πραγματικά δεδομένα και το testing γίνεται από τον πελάτη για να πάρουμε την αποδοχή του.

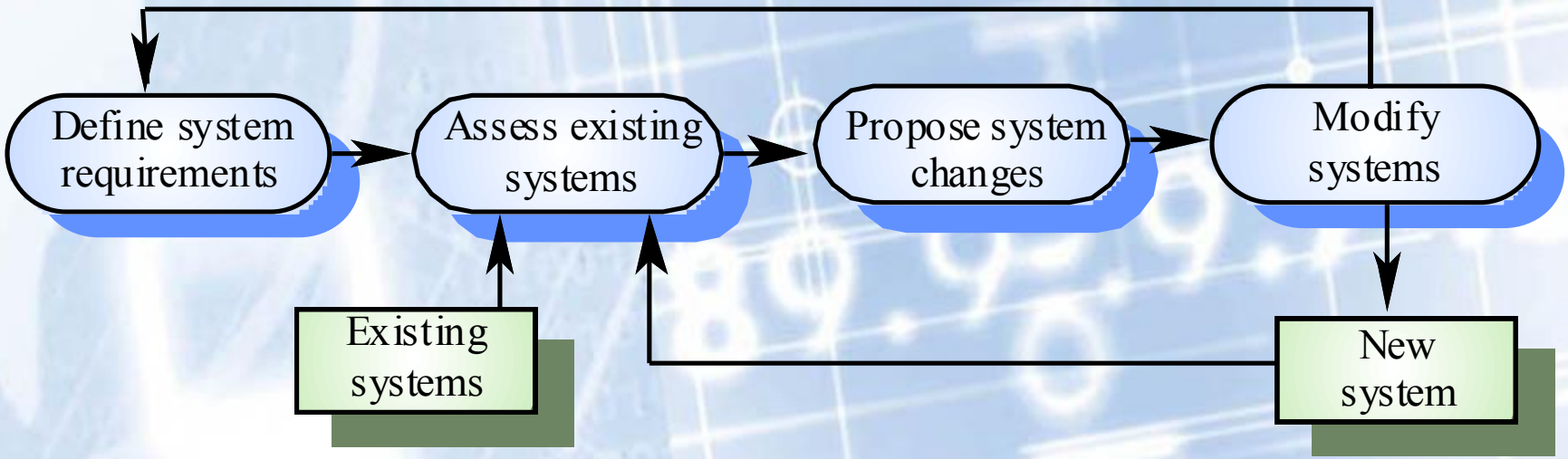
Τα στάδια του Testing



Η Εξέλιξη του Λογισμικού
(Software evolution)

- Το λογισμικό είναι «ζωντανός οργανισμός» που αλλάζει συνεχώς.
- Οι απαιτήσεις αλλάζουν καθώς οι επιχειρησιακές ανάγκες μεγαλώνουν και το λογισμικό θα πρέπει να είναι σε θέση να εξελιχθεί και να συμπεριλάβει τις νέες απαιτήσεις.

Η Εξέλιξη των Συστημάτων
(System evolution)



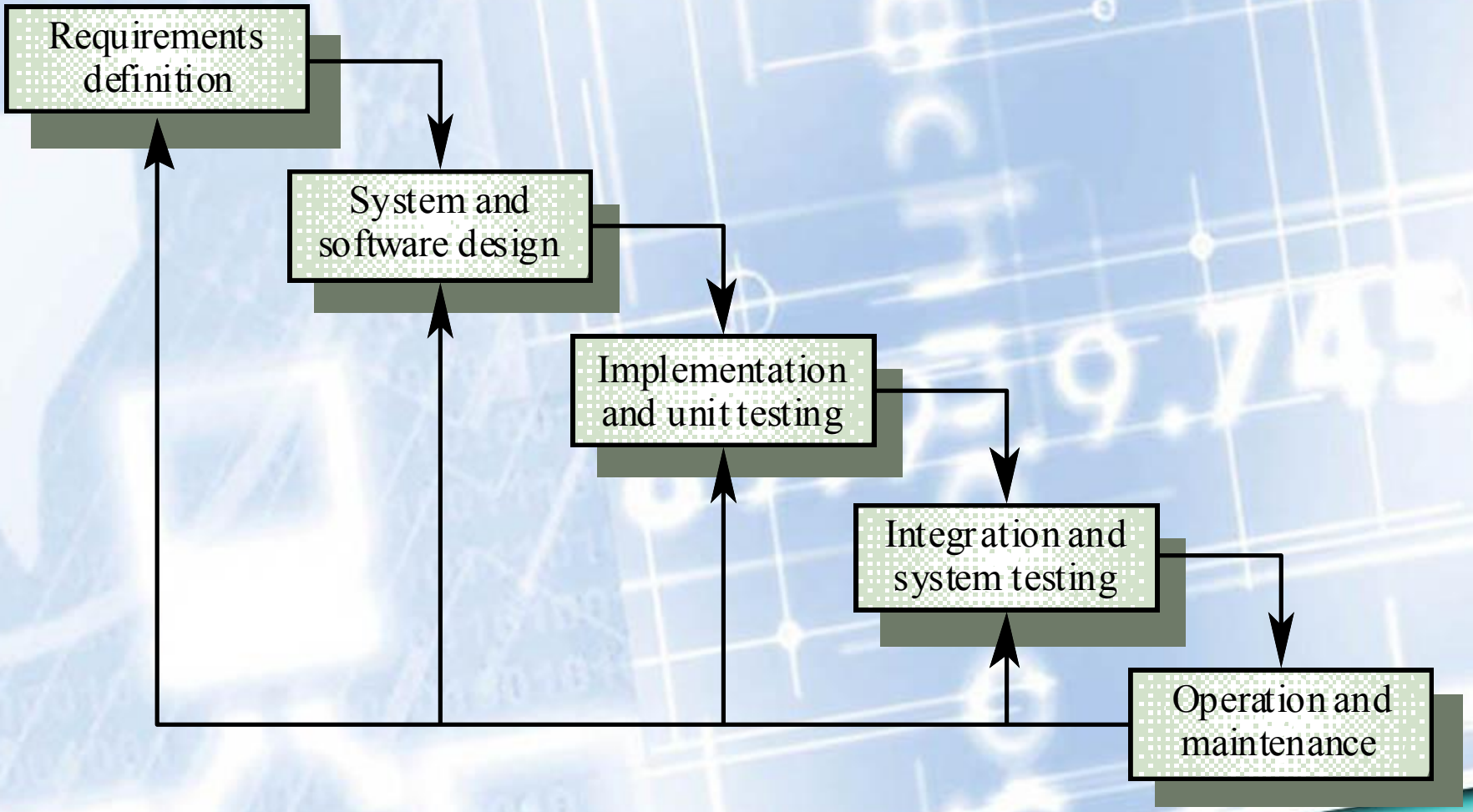
Μοντέλα Διαδικασιών Λογισμικού
(Generic software process models)

Μοντέλα Διαδικασιών Λογισμικού

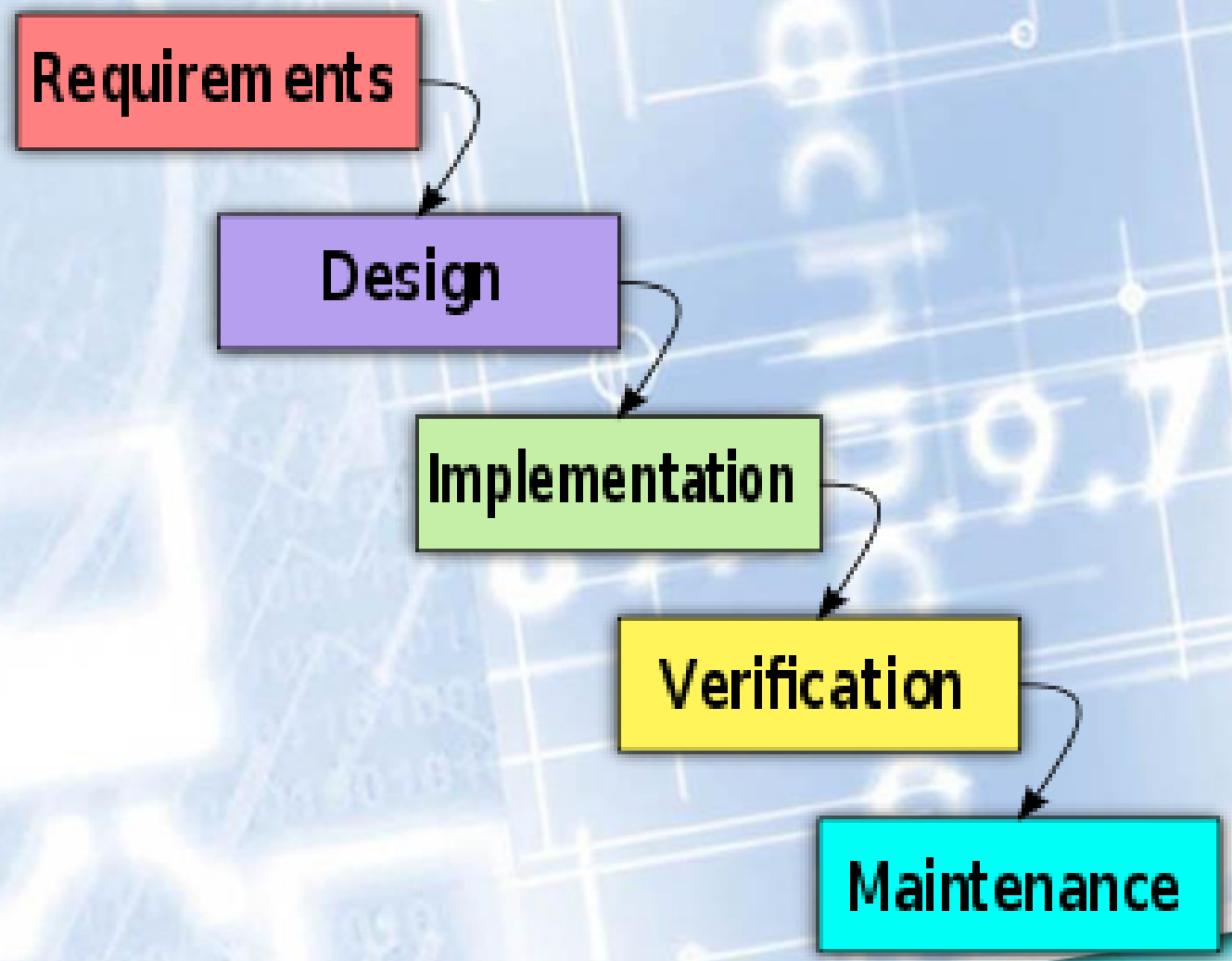
(software process models)

- Waterfall
- Evolutionary development
- Reuse-based development
- Formal System Development
- Incremental Development
- Spiral
- RUP

Το μοντέλο Waterfall



Το μοντέλο Waterfall



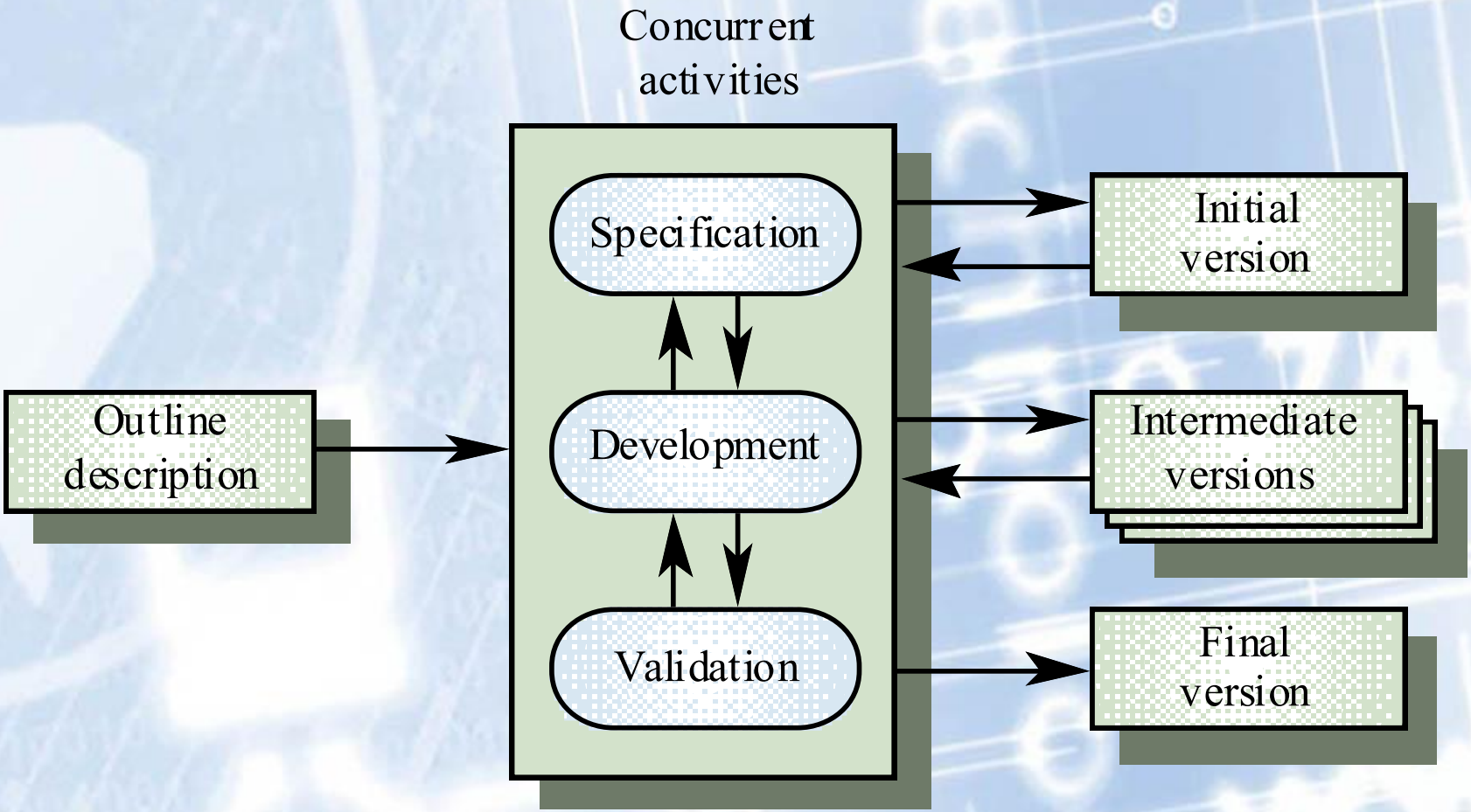
Τα στάδια του μοντέλου Waterfall

- Ανάλυση και προσδιορισμός απαιτήσεων
- Σχεδιασμός συστήματος και λογισμικού
- Υλοποίηση και έλεγχος υποσυστήματος
- Ενοποίηση και έλεγχος συστήματος
- Λειτουργία και συντήρηση
- Το μειονέκτημα αυτού του μοντέλου είναι ότι είναι δύσκολο να γίνουν αλλαγές σε ενδιάμεσα στάδια του μοντέλου.

Δυσκολίες του μοντέλου Waterfall

- Το έργο δεν διασπάτε εύκολα σε διαφορετικά στάδια
- Αυτό το κάνει δύσκολο να αντεπεξέλθει σε αλλαγές των απαιτήσεων του πελάτη.
- Άρα, το μοντέλο αυτό είναι κατάλληλο σε περιπτώσεις που οι απαιτήσεις του πελάτη είναι πλήρως καταγεγραμμένες.
- Χρησιμοποιείται κυρίως για μεγάλα έργα στα οποία το σύστημα αναπτύσσεται σε περισσότερες της μίας τοποθεσίες

Μοντέλο Evolutionary development



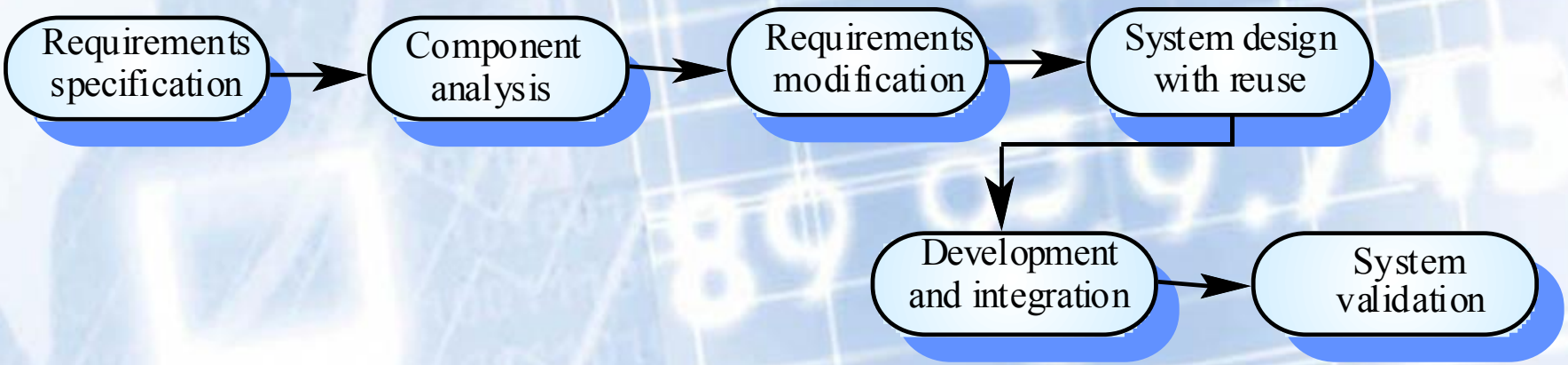
Evolutionary development

- Διερευνητική Υλοποίηση
  - Ο στόχος είναι μέσα από την συνεργασία με τους πελάτες να υλοποιηθεί ένας αρχικός προσδιορισμός των προδιαγραφών βάση των οποίων θα φτιαχτεί το τελικό σύστημα. Για τον λόγο αυτό οι απαιτήσεις θα πρέπει να καταγραφούν με προσοχή και λεπτομέρεια.
- Prototyping
  - Ο στόχος είναι να κατανοηθούν πλήρως οι απαιτήσεις του συστήματος. Συνήθως ξεκινάμε με απαιτήσεις όχι καλά προσδιορισμένες.

Evolutionary development

- Δυσκολίες
  - Δεν παρέχει ξεκάθαρο διαχωρισμό διαδικασιών.
  - Τα συστήματα είναι πολλές φορές λάθος δομημένα.
  - Ειδικές Ικανότητες χρειάζονται πολλές φορές για να χρησιμοποιηθούν οι γλώσσες προγραμματισμού για rapid prototyping
- Εφαρμογές
  - Για μικρο-μεσαία συστήματα
  - Για κομμάτια μεγάλων συστημάτων (π.χ. the user interface)
  - Για συστήματα με μικρή διάρκεια ζωής

Μοντέλο Reuse-oriented



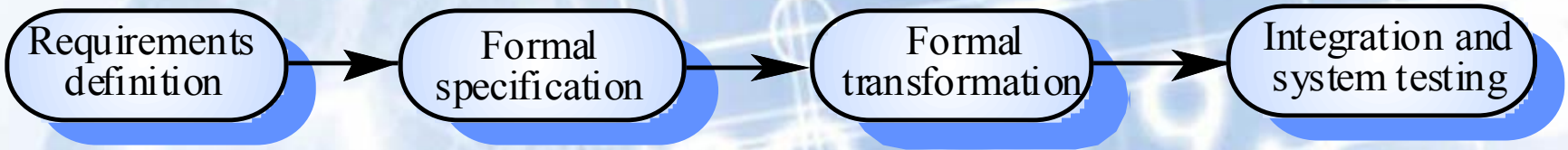
Reuse-oriented or Component-based software engineering

- Βασίζεται στην εκ νέου χρησιμοποίηση υπαρχόντων στοιχείων τα οποία ενώνονται μεταξύ τους για την υλοποίηση του νέου συστήματος. Τα στοιχεία αυτά μπορεί να είναι κομμάτια κώδικα ή έτοιμα προγράμματα που πωλούνται στην αγορά.
- Στάδια
  - Ανάλυση των στοιχείων
  - Μετατροπή των Απαιτήσεων
  - Σχεδιασμός συστήματος
  - Υλοποίηση και ενοποίηση
- Αυτή η προσέγγιση απόκτα όλο και περισσότερη σοβαρότητα και χρησιμότητα καθώς έχουν διαμορφωθεί διαδικασίες τυποποίησης του κώδικα αλλά ακόμη δεν έχει χρησιμοποιηθεί στην υλοποίηση πολλών συστημάτων.

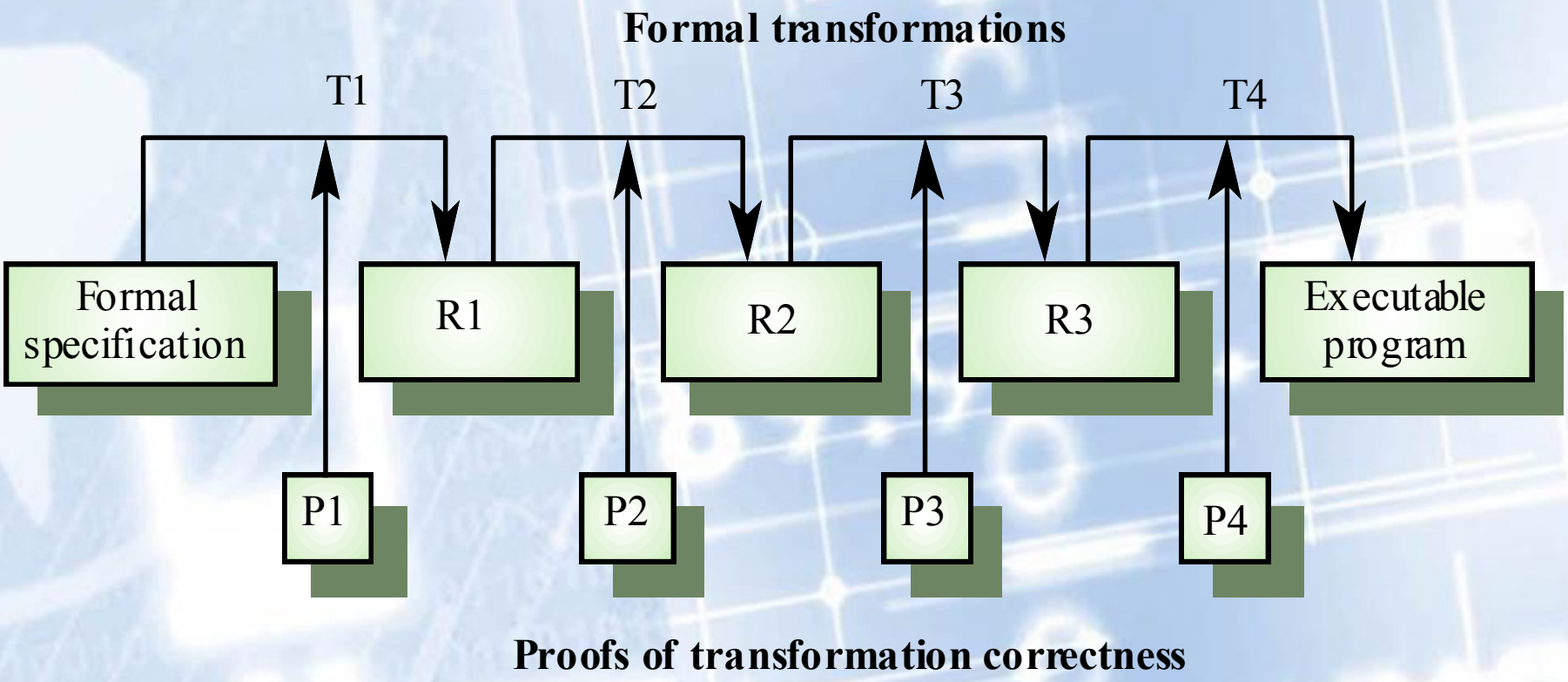
Μοντέλο Formal systems development

- Βασίζεται στην μετατροπή μιας μαθηματικής περιγραφής, μέσω διαφορετικών αναπαραστάσεων, σε εκτελέσιμο πρόγραμμα
- Οι Μετατροπές - Μετασχηματισμοί διαφυλάσσουν εξ' ορισμού την ορθότητα και έτσι είναι εύκολο να δείξουμε ότι το πρόγραμμα είναι σύμφωνο με τις προδιαγραφές του.
- Ενσωματωμένο στην «Cleanroom» προσέγγιση για την ανάπτυξη λογισμικού

Formal systems development



Formal transformations



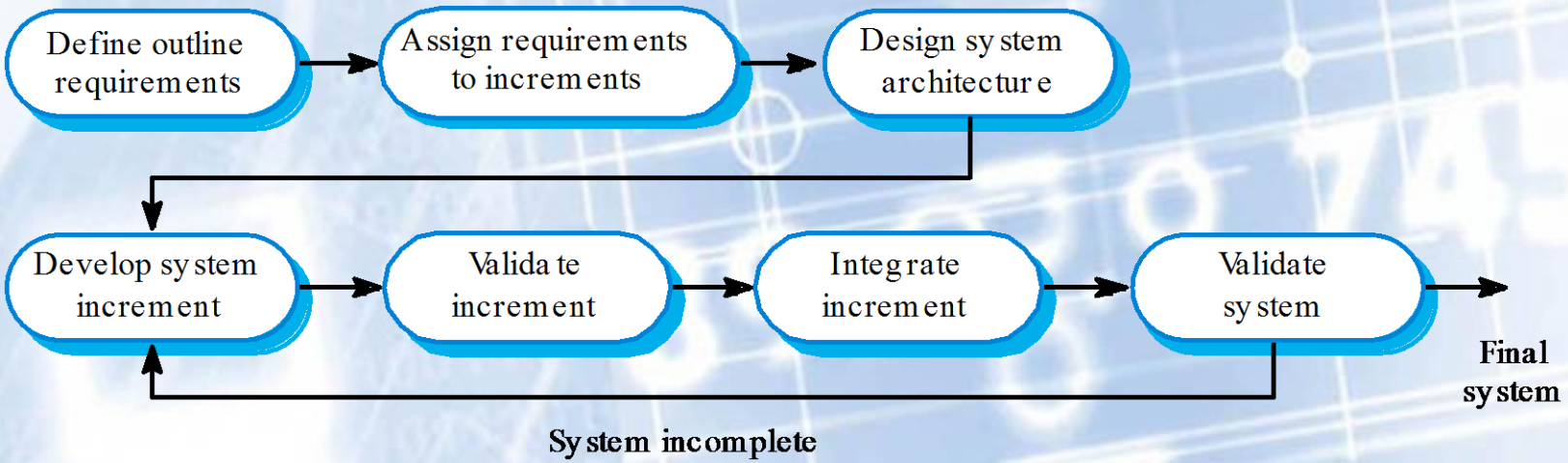
Formal systems development

- Προβλήματα
  - Χρειάζεται εξειδικευμένες δεξιότητες και εκπαίδευση για να εφαρμοστεί η τεχνική
  - Δύσκολο να καθορίσετε επίσημα ορισμένες πτυχές του συστήματος, όπως η διεπαφή χρήστη
- Εφαρμογή
  - Σε «Critical system» ειδικά σε εκείνα όπου η ασφάλεια πρέπει να ελεγχτεί πριν το σύστημα τεθεί σε λειτουργία

Επανάληψη Διαδικασιών
(Process iteration)

- Οι απαιτήσεις του συστήματος αλλάζουν/εξελίσσονται πάντα στην διάρκεια ενός έργου και έτσι δεν είναι δυνατόν να αποφευχθεί η επανάληψης κάποιων σταδίων του έργου ειδικά σε μεγάλα και πολύπλοκα έργα.
- Όλα τα παραπάνω μοντέλα έχουν πλεονεκτήματα και μειονεκτήματα. Για τα περισσότερα συστήματα μεγάλου μεγέθους θα πρέπει να χρησιμοποιηθούν διαφορετικά μοντέλα για τα διάφορα μέρη του λογισμικού. Για να μπορέσει να γίνει αυτό θα πρέπει να χρησιμοποιηθούν υβριδικά μοντέλα (hybrid models).
- Η διαδικασία της επανάληψης μπορεί να χρησιμοποιηθεί σε όλα τα παραπάνω μοντέλα.
- Υπάρχουν δύο σχετικές προσεγγίσεις για την διαχείριση των επαναλαμβανόμενων αλλαγών των απαιτήσεων
  - **Μοντέλο Αυξητικής Ανάπτυξης (Incremental Development)**
  - **Μοντέλο Σπειροειδούς Ανάπτυξης (Spiral Development)**

Αυξητική Ανάπτυξη
(Incremental Development)



Αυξητική Ανάπτυξη
(Incremental Development)

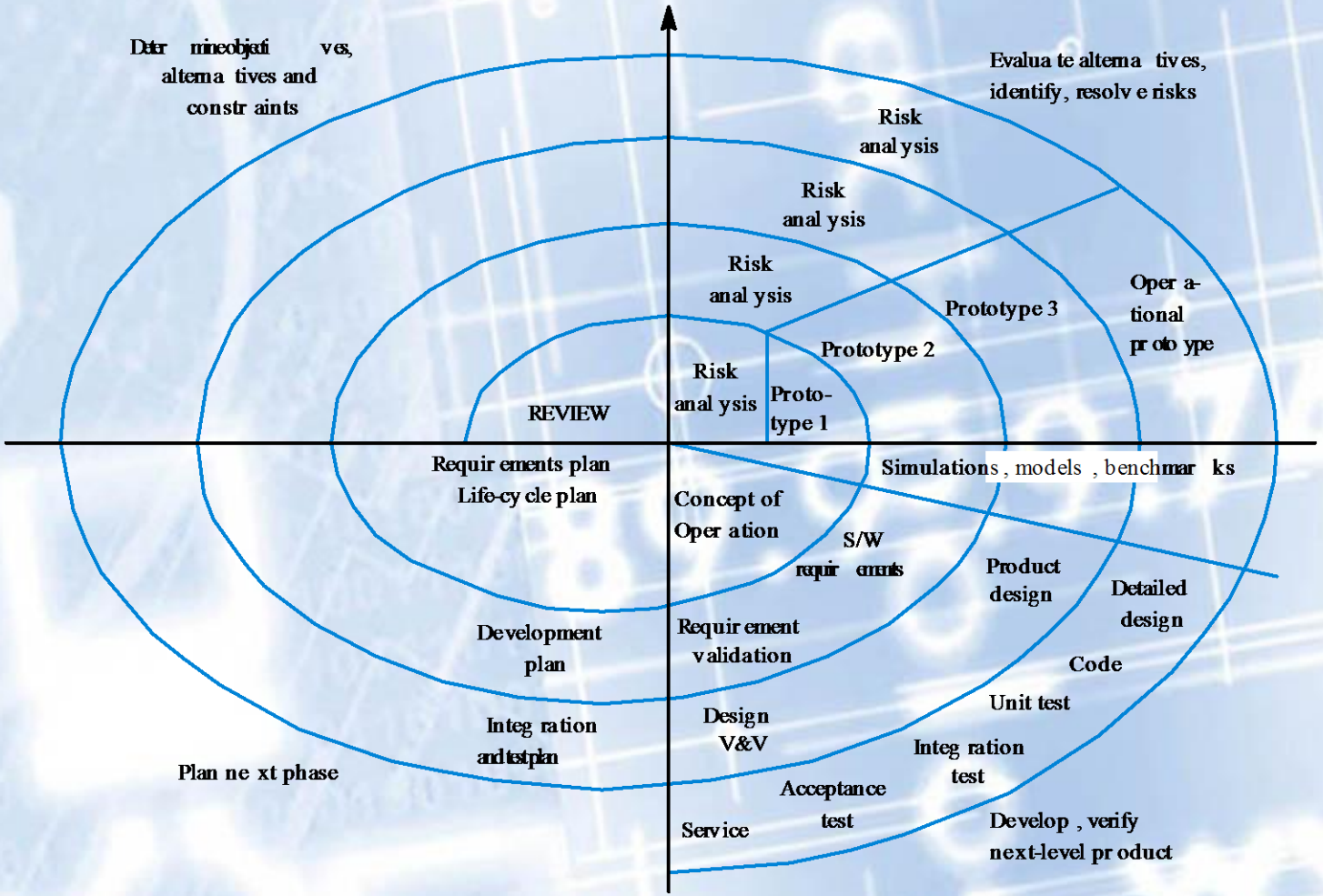
- Αντί να υλοποιηθεί και να παραδοθεί όλο το σύστημα ολοκληρωμένο στο τέλος σπάει η διαδικασία υλοποίησης και παράδοσης σε πολλά μέρη. Έτσι το σύστημα παραδίδεται σταδιακά. Το κάθε κομμάτι που παραδίδεται αντιπροσωπεύει μέρος της λειτουργικότητας του συνολικού συστήματος
- Οι απαιτήσεις χρήστη διαβαθμίζονται ανάλογα με σοβαρότητα και την αμεσότητα και έτσι αυτές με μεγαλύτερη προτεραιότητα υλοποιούνται και παραδίδονται πρώτα.
- Μετά την έναρξη υλοποίησης της κάθε ενότητας η εξέλιξη των απαιτήσεων που εμπλέκονται με αυτήν την ενότητα παγώνει ενώ η εξέλιξη των υπολοίπων συνεχίζεται

Πλεονεκτήματα Αυξητικής Ανάπτυξης

(Incremental Development)

- Τα σχόλια του πελάτη συλλέγονται σταδιακά με την παράδοση των επιμέρους κομματιών και έτσι η λειτουργικότητα του συστήματος επιτυγχάνεται νωρίτερα.
- Τα πρώτα κομμάτια που παραδίδονται λειτουργούν ως πρωτότυπα και βοηθούν στην διευκρίνηση υπαρχόντων απαιτήσεων και στον προσδιορισμό νέων που έχουν παραληφθεί
- Μειώνεται το ρίσκο αποτυχίας του έργου
- Τα σπουδαιότερα κομμάτια του έργου που έχουν μεγαλύτερη προτεραιότητα και άρα παραδίδονται νωρίς ελέγχονται εκτενέστερα.

Σπειροειδής Ανάπτυξη
(Spiral Development)



Σπειροειδής Ανάπτυξη
(Spiral Development)

- Η διαδικασία ανάπτυξης λογισμικού απεικονίζεται σε σπειροειδή μορφή αντί μιας σειριακής ακολουθίας ενεργειών με οπισθοδρόμηση
- Κάθε περιστροφή της σπείρας αντιπροσωπεύει ένα στάδιο της διαδικασίας ανάπτυξης
- Δεν υπάρχουν σταθερά στάδια όπως προσδιορισμός απαιτήσεων ή σχεδιασμός. Οι βρόγχοι επιλέγονται ανάλογα με το τι χρειάζεται την δεδομένη στιγμή.
- Τα ρίσκα αξιολογούνται με σαφήνεια και δίνονται λύσης καθ' όλη την διάρκεια της σπειροειδούς ανάπτυξης

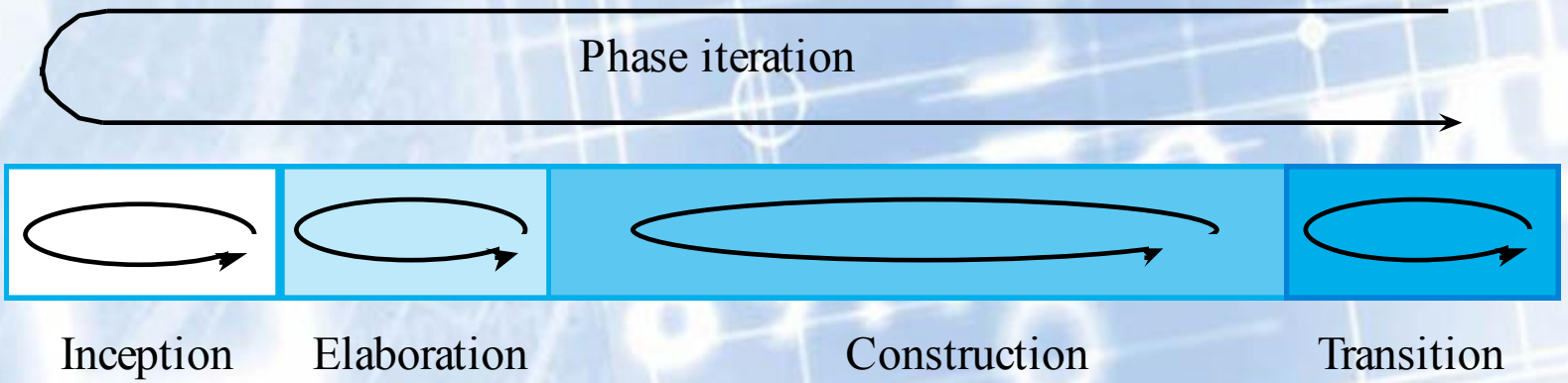
Τομείς Σπειροειδούς Μοντέλου
(Spiral Model Sectors)

- Προσδιορισμός Στόχων
  - Συγκεκριμένοι στόχοι προσδιορίζονται για κάθε στάδιο
- Αξιολόγηση και Ελαχιστοποίηση Ρίσκου
  - Καθώς τα ρίσκα αξιολογούνται και γίνεται προσπάθεια ελαχιστοποίησης τους μένουν πίσω οι υπόλοιπες δραστηριότητες
- Υλοποίηση και Τεκμηρίωση
  - Το μοντέλο υλοποίησης που επιλέγεται μπορεί να είναι οποιοδήποτε από τα γενικά μοντέλα
- Προγραμματισμός
  - Ελέγχεται το έργο και μετά προγραμματίζεται το επόμενο στάδιο της σπείρας.

The Rational Unified Process (RUP)

- Είναι ένα νέο μοντέλο διαδικασίας ανάπτυξης λογισμικού το οποίο δημιουργήθηκε από την UML και άλλες σχετικές διαδικασίες.
- Περιγράφεται από 3 προοπτικές
  - Μια δυναμική προοπτική που απεικονίζει τα στάδια σε σχέση με τον χρόνο
  - Μια στατική προοπτική η οποία δείχνει δραστηριότητες διαδικασιών
  - Μια πρακτική προοπτική η οποία προτείνει σωστές πρακτικές στα θέματα που προκείπτουν

The Rational Unified Process (RUP)



Στάδια της RUP

- Επιθεώρηση
  - Θεσπίζει το επιχειρησιακό κομμάτι του συστήματος
- Επεξεργασία
  - Κατανόηση του τομέα του προβλήματος και της αρχιτεκτονικής του συστήματος
- Δόμηση
  - Σχεδιασμός, υλοποίηση και έλεγχος συστήματος
- Μετάβαση
  - Θέτουμε το σύστημα σε λειτουργία στο περιβάλλον χρήσης του.

Σωστές πρακτικές για την RUP

- Ανάπτυξη λογισμικού με επαναληπτικό τρόπο
- Διαχείριση Απαιτήσεων
- Χρήση αρχιτεκτονικών component-based
- Ο μοντελισμός γίνεται με οπτικά μέσα
- Εξακρίβωση της ποιότητας λογισμικού
- Έλεγχος αλλαγών λογισμικού

Αυτοματοποίηση των Διαδικασιών του Software Engineering - CASE

Αυτοματοποίηση των Διαδικασιών του Software Engineering - CASE

- CASE = **C**omputer **A**ided **S**oftware **E**ngineering
- CASE είναι λογισμικό το οποίο υποστηρίζει/βοηθάει την ανάπτυξη και την εξέλιξη λογισμικών συστημάτων.
- Αυτοματισμός των Εργασιών με CASE tools
  - Graphical editors για την ανάπτυξη των μοντέλων του συστήματος
  - Data dictionary για την διαχείριση του σχεδιασμού
  - Graphical UI builder για την ανάπτυξη της αλληλεπίδρασης συστήματος χρήστη.
  - Debuggers για τον εντοπισμό και την διόρθωση λαθών.
  - Automated translators για την διαχείριση των εκδόσεων του λογισμικού.

Η τεχνολογία Case

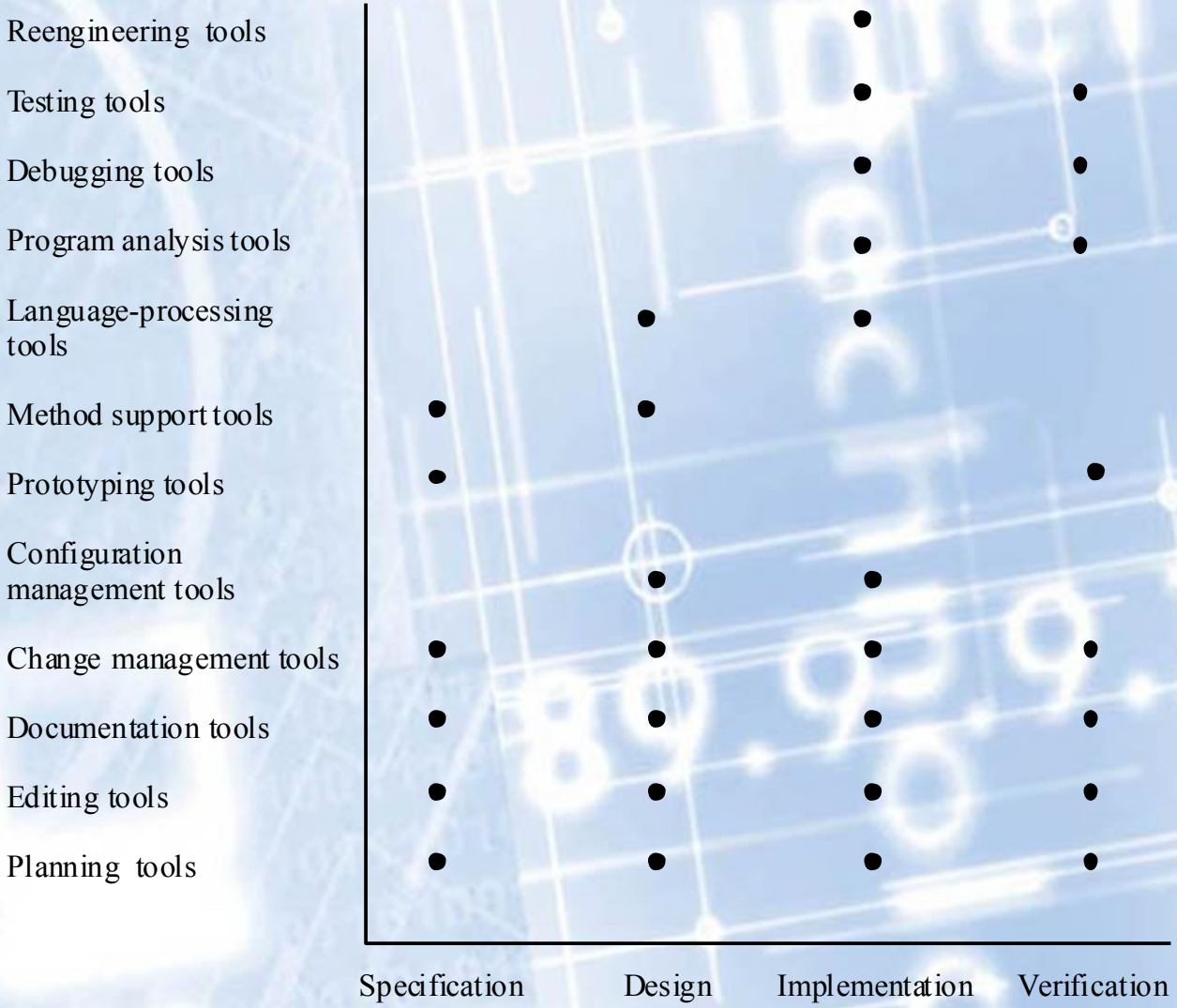
- Η τεχνολογία Case έχει επιφέρει μεγάλες αλλαγές στον τομέα των διαδικασιών λογισμικού αλλά δεν είναι τόσο μεγάλες όσο θεωρήθηκε ότι θα είναι όταν πρωτοεμφανίστηκαν.
  - Το Software engineering απαιτεί δημιουργική σκέψη και αυτό δεν μπορεί εύκολα να αυτοματοποιηθεί.
  - Το Software engineering είναι μια ομαδική δραστηριότητα και ειδικά σε μεγάλα έργα πολύς χρόνος καταναλώνεται για την καλή λειτουργία της ομάδας. Η τεχνολογία CASE δεν είναι σε θέση να υποστηρίξει τέτοιες δραστηριότητες.

Ταξινόμηση του CASE

- Ταξινομώντας την τεχνολογία CASE σε κατηγορίες μας βοηθάει να κατανοήσουμε τους διαφορετικούς τύπους εργαλείων CASE και ποιες δραστηριότητες του software engineering μπορούν να υποστηρίξουν.
- Κατηγοριοποίηση βάση Λειτουργικότητας
  - Τα εργαλεία κατηγοριοποιούνται βάση των λειτουργιών που διαθέτουν.
- Κατηγοριοποίηση βάση Διαδικασιών
  - Τα εργαλεία κατηγοριοποιούνται βάση ποιών και πόσων διαδικασιών του software engineering μπορούν να υποστηρίξουν.
- Κατηγοριοποίηση βάση Ενοποίησης
  - Τα εργαλεία κατηγοριοποιούνται βάση των δυνατοτήτων που έχουν για την ενοποίηση μονάδων, ενοτήτων και υπο-συστημάτων λογισμικού.

Functional tool classification

Tool type	Examples
Planning tools	PERT tools, estimation tools, spreadsheets
Editing tools	Text editors, diagram editors, word processors
Change management tools	Requirements traceability tools, change control systems
Configuration management tools	Version management systems, system building tools
Prototyping tools	Very high-level languages, user interface generators
Method-support tools	Design editors, data dictionaries, code generators
Language-processing tools	Compilers, interpreters
Program analysis tools	Cross reference generators, static analysers, dynamic analysers
Testing tools	Test data generators, file comparators
Debugging tools	Interactive debugging systems
Documentation tools	Page layout programs, image editors
Re-engineering tools	Cross-reference systems, program re-structuring systems

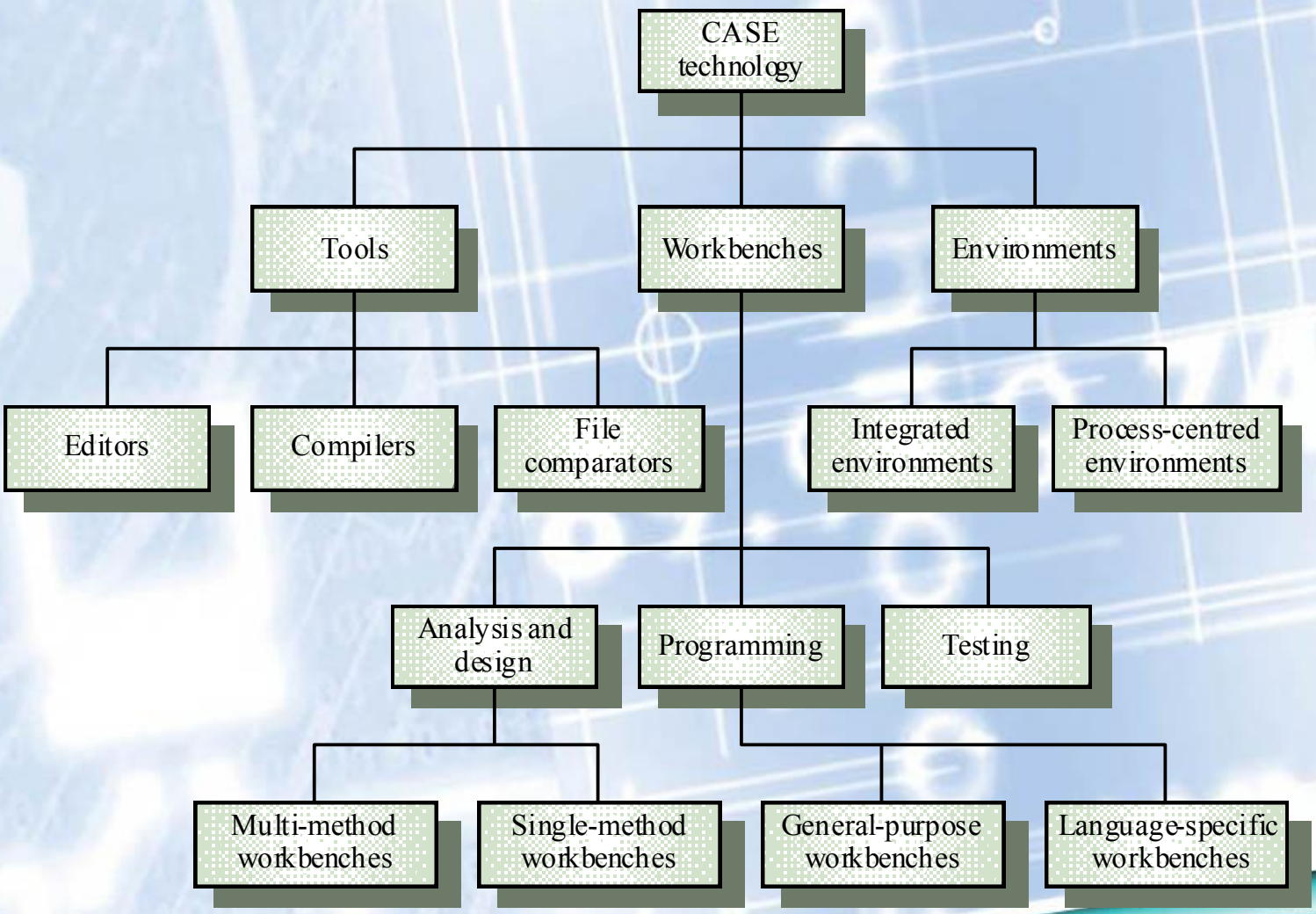


Activity-based classification

Ενοποίηση του CASE

- Εργαλεία
  - Υποστηρίζουν ξεχωριστές εργασίες όπως σχεδιασμός σεναρίων ελέγχου, αντιπαράθεση αρχείων κλπ.
- Workbenches
  - Υποστηρίζουν ένα ολόκληρο στάδιο μιας διαδικασίας όπως καταγραφή απαιτήσεων, σχεδιασμός, κλπ. Συνήθως εμπεριέχουν μια σειρά από εργαλεία.
- Περιβάλλοντα
  - Υποστηρίζουν όλη την διαδικασία του software engineering ή ένα μέρος αυτής. Συνήθως εμπεριέχουν μια σειρά από workbenches

Εργαλεία, Workbenches, Περιβάλλοντα



Βιβλιογραφία - Further Reading

- Tucker A. (ed.), Software process models., 1997, CRC Press
- Stevens R. et.al., Systems Engineering: Coping with Complexity, 1998, Prentice Hall

Υπόμνημα-Λεξικό

Ασκήσεις Θεωρίας

- Προσδιορίστε ποίο μοντέλο διαδικασίας θα επιλέγατε για να υλοποιήσετε το παρακάτω λογισμικό και γιατί:
  - Λογισμικό λογιστηρίου του ΤΕΙ
  - Anti lock σύστημα φρένων αυτοκινήτου
  - Λογισμικό εικονικής πραγματικότητας για ένα παιχνίδι
  - Ιστοσελίδα προαγωγής προϊόντων καταστήματος
- Γιατί λογισμικό που έχει υλοποιηθεί με το μοντέλο evolutionary είναι δύσκολο στην συντήρηση;
- Πως μπορούν να ενσωματωθούν στο μοντέλο spiral τα μοντέλα waterfall και prototyping;
- Περιγράψτε τις βασικές δραστηριότητες της διαδικασίας σχεδιασμού και τα αποτελέσματα αυτές των δραστηριοτήτων.
- Γιατί το λογισμικό δεν μπορεί να είναι στάσιμο; Ποιες είναι οι συνθήκες που το αναγκάζουν να εξελιχθεί;

Την επόμενη φορά

- Απαιτήσεις Λογισμικού

Τέλος για σήμερα - Ερωτήσεις