ΗΥ 351: Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
CS 351: Information Systems Analysis and Design

II. (Development) Methodologies
Μεθοδολογίες Ανάπτυξης

…

Lecture : 2
Date : 29-9-2005

Yannis Tzitzikas
University of Crete, Fall 2005

Outline

- Common problems in Information Systems Development
- What is a (Software Development) methodology ?
- The fundamental 4-phase model
  - planning, analysis, design, implementation
- Methodologies
- How we select a methodology?
- Process Improvement Models

Common Problems
in Information Systems Development

The current status in software engineering

The Spandish Group report, 2003:

- only one out of three software projects complete on-time and on-budget.
- 42% of all corporate IS projects were abandoned before completion

- Most errors (54%) are detected after coding and testing.
- Almost half of all errors (45%) are introduced during requirements and design.
- Most errors made during requirements analysis are non-clerical (77%)
- Requirements errors can cost up to 100 times more to fix than implementation errors - if they are not caught early on

Many failed systems were abandoned because analysts tried to build wonderful systems without understanding the organization.

The primarily goal is to **create value for the organization**.

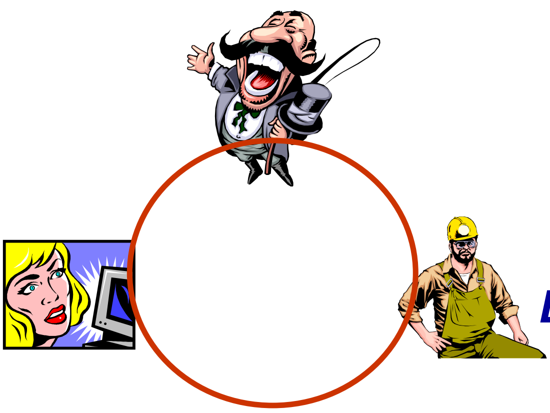
==> Need to do requirements and design right!

Problems in IS Development

What can go wrong?

- It is only by understanding what can go wrong during a system development project that we can hope to avoid failure

client



End user

Developer

Problems from an End User's perspective

- *What system? I haven't seen a new system*

- *It might work, but it's dreadful to use!*

- *It's very pretty but does it do anything useful?*

Problems from a Client's perspective

- *If I'd known the real price, I'd never have agreed*
- *It's no use delivering it now - we needed last April!*
- *Ok, so it works - but the installation was such a mess my staff will never trust it!*
- *I didn't want it in the first place*
- *Everything's changed now - we need a completely new system*

Problems from a Developer's perspective

- *We built what they said they wanted*
- *There wasn't enough time to do it any better*
- *Don't blame me - I've never done object-oriented analysis before*
- *How can I fix it? - I don't know how it's supposed to work*
- *We said it was impossible, but no-one listened*
- *The system's fine - the users are the problem*

Causes of IS project failure (Flynn'98)

Type of Failure	*Reason for failure*	*Comment*
Quality problems	The wrong problem is addressed	System conflicts with business strategy
	Wider influences are neglected	Organization culture is ignored
	Analysis is carried out incorrectly	Poorly skilled team, too small
	Project undertaken for wrong reason	Technology pull or political push
Productivity Problems	Users change their minds	
	External events change the environment	New legislation
	Implementation is not feasible	May not known, until project start
	Poor project control	Inexperienced project manager

Τα αίτια της αποτυχίας

Η αποτυχία πολλές φορές οφείλεται σε κινδύνους (ρίσκα) που δεν λήφθηκαν υπόψη και δεν έγινε σωστό πλάνο αντιμετώπισής τους

Ρίσκο ~ μέτρο της αβεβαιότητας ως προς το αποτέλεσμα

Υψηλό ρίσκο => αύξηση κόστους, πρόκληση καθυστερήσεων

Ρίσκο = f(διαθέσιμης πληροφορίας)

Όσο λιγότερη και χαμηλότερης ποιότητας πληροφορία έχουμε, τόσο μεγαλύτερο το ρίσκο

Associated Risks

Kinds of Risks:

- **Requirements**
  - avoid the big danger, i.e. build the wrong system (the one that does not satisfy customers
- **Technological**
  - the selected technology will work ?
  - Will the various pieces fit together ?
- **Skill**
  - will I get the staff and expertise I need ?
- **Political**
  - are there political forces that can get in the way and seriously affect the project ?

Πληροφοριακά Συστήματα και .. Ηθική (επιπτώσεις πληροφοριακών συστημάτων)

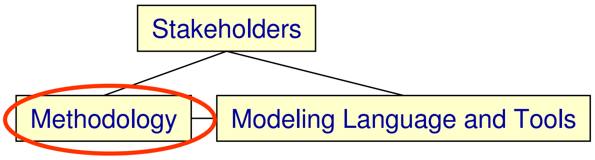
Τα αποτελέσματα της εγκατάστασης ενός ATM έξω από ένα σουπερμάρκετ:

Ομάδα		Αποτέλεσμα εγκατάστασης ATM
• Ταμίες Τραπεζών	→	• Απώλεια θέσεων εργασίας
• Πελάτες Τραπεζών	→	• Καλύτερη εξυπηρέτηση
• Πελάτες Σουπερμάρκετ	→	• Χειρότερη εξυπηρέτηση (συμφόρηση πάρκινγ)
• Μέτοχοι Τράπεζας	→	• Αύξηση μερίσματος
• Μέτοχοι Σουπερμάρκετ	→	• Αύξηση μερίσματος (λόγω του ATM ψωνίζουν επίσης)
• Κάτοικοι περιοχής	→	• Αύξηση ρύπανσης, κυκλοφορίας

Ένα πληροφοριακό σύστημα μπορεί να επηρεάσει τη ζωή πολλών ανθρώπων

What is a (Software Development) Methodology?

What is a (Software Development) Methodology ?



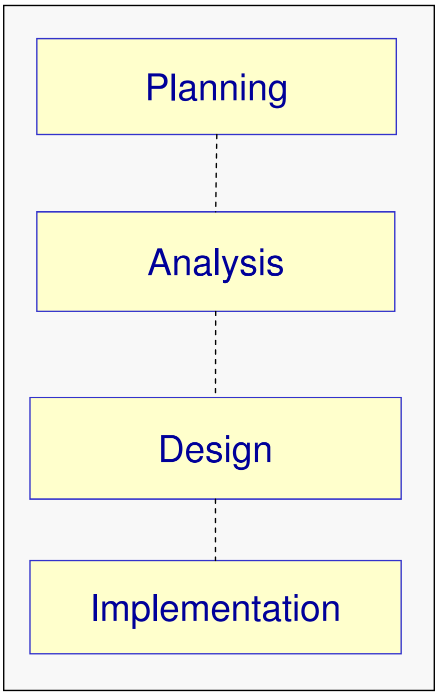
- Defines activities and organizational procedures used in software production and maintenance
- A process model (methodology):
  - states an order of carrying out activities
  - specifies what development artefacts are to be delivered when
  - assigns activities and artefacts to developers
  - offers criteria for monitoring a project’s progress, for measuring the outcomes, and for planning future projects
- Is not susceptible to standardisation

The fundamental 4-phase model

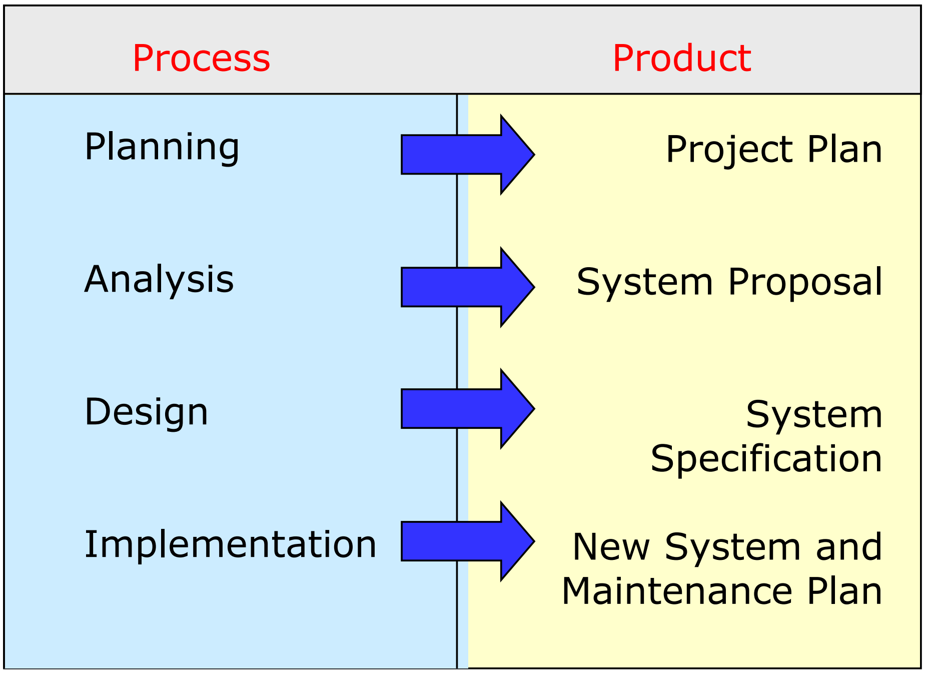
The fundamental 4-phase model (planning, analysis, design, implementation)

In <u>every project</u> and in <u>every development methodology</u> we can identify some fundamental **<u>phases</u>**

- Each phase consists of <u>steps</u>, rely on <u>techniques</u>, and produces <u>deliverables</u>
- Different methodologies specify different order between these
  - they specify how exactly we will work.



The fundamental 4-phase model (planning, analysis, design, implementation)



Process	Product
Planning	Project Plan
Analysis	System Proposal
Design	System Specification
Implementation	New System and Maintenance Plan

Planning

Planning
Analysis
Design
Implementation

- Why and how the IS should be built
- Two steps
  - 1. At <u>project initiation</u>, the **system's value** to the organization is identified
    - how it will *lower costs* or *increase benefits*?
    - A <u>system request</u> describes in brief the business need, and it explains how a system that supports the need will create business value. The IS department works together with the person or department that generated the request (called <u>project sponsor</u>) to conduct a **<u>feasibility analysis</u>** which examines key aspects of the proposed project:
      - technical feasibility (can we build it?)
      - economical feasibility (will offer business value?)
      - organizational feasibility (if we build it, will it be used?)
    - The system request and feasibility study an <u>approval committee</u> (or steering committee) which decides whether the project should be undertaken.
  - 2. Once the project is approved it enter into <u>project management</u>
    - the project manager creates a workplan, staffs the project and monitors and controls the progress. Deliverable: <u>project plan</u>

Analysis

- Its objective is to answer the questions:
  - who will use the system
  - what the system will do
  - where and when it will be used
- Steps
  - 1. Analysis strategy. Analysis of the current system and its problems and ways to design a new system
  - 2. Requirements Gathering (through interviews, questionnaires). The concept of the new system will then used to describe how the business will operate with the new system
  - 3. The analyses, system concept and models are then combined into one document called system proposal, which is then given to key person in order to decide whether the project should be continued

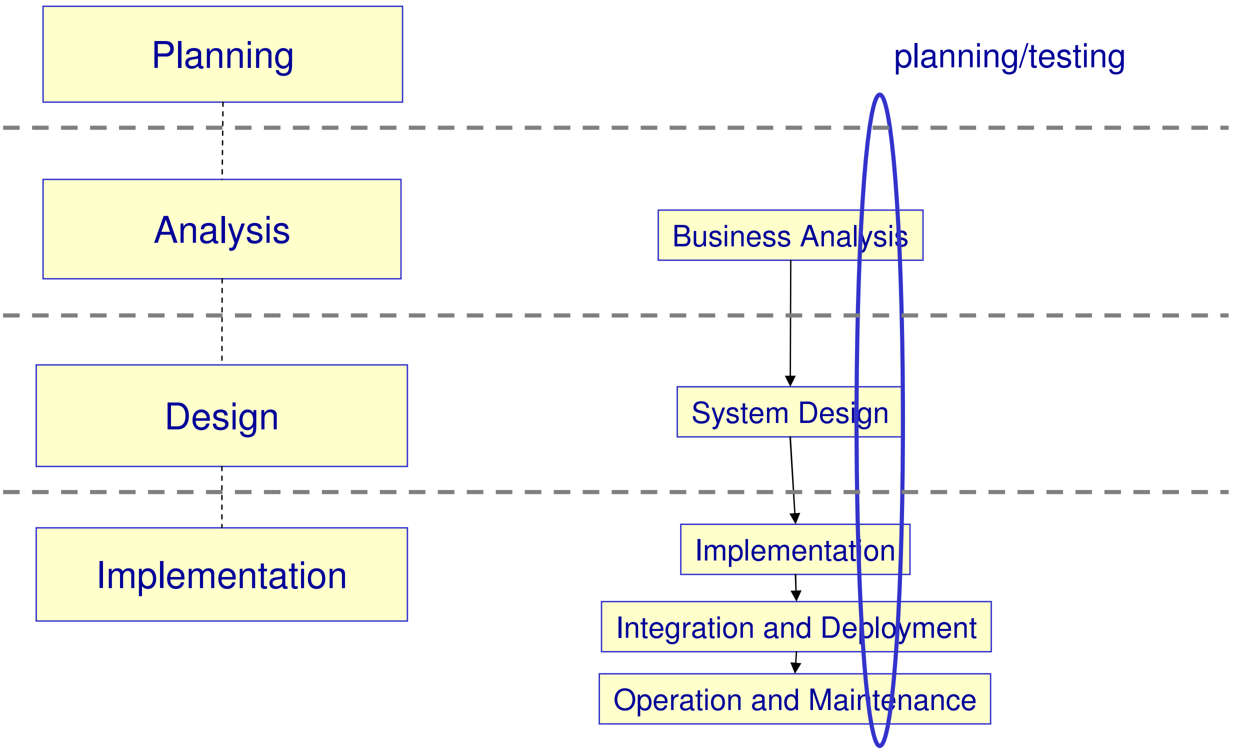
Design

- How the system will operate in terms of
  - hardware, software and network infrastructure, the UI, forms and reports, databases, files, etc
- Steps:
  - 1. Design Strategy. Developed by company itself or outsourced to another firm, or buy an existing software package
  - 2. Architecture Design: hardware, software and network infrastructure, the UI, forms and reports, databases, files, etc
  - 3. Database and file specifications: define exactly what data will be stored and where
  - 4. Program design: programs that need to be written and what each will do.
- Outcome:
  - System Specification that is given to the programming team for implementation

Implementation

- Commonly, this is the longest and most expensive part of the process.
- Steps
  - 1. Construction
  - 2. Installation: The new system replaces the existing (completely, in parallel, phased conversion strategy), training plan.
  - 3. Support Plan: formal and informal post-implementation review, as well as a systematic way for identifying changed needed for the system

The same phases with different names and notations



Planning

Analysis

Design

Implementation

planning/testing

Business Analysis

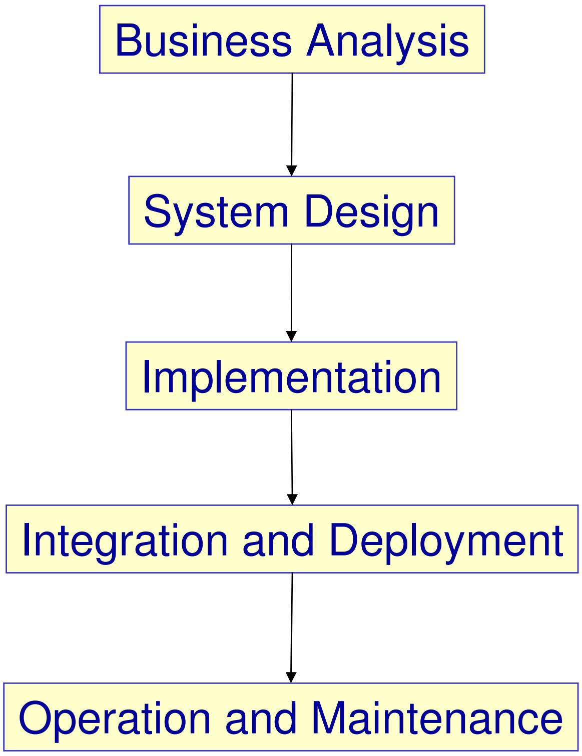
System Design

Implementation

Integration and Deployment

Operation and Maintenance

Lifecycle phases



Business Analysis

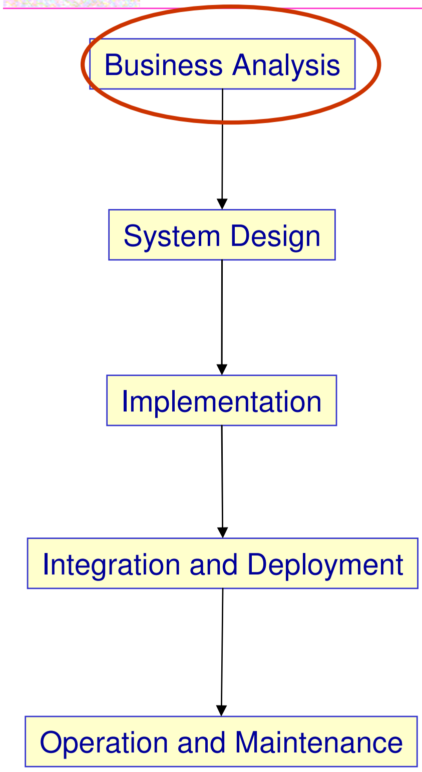
System Design

Implementation

Integration and Deployment

Operation and Maintenance

Business Analysis (or Requirements Analysis)



Business Analysis

System Design

Implementation

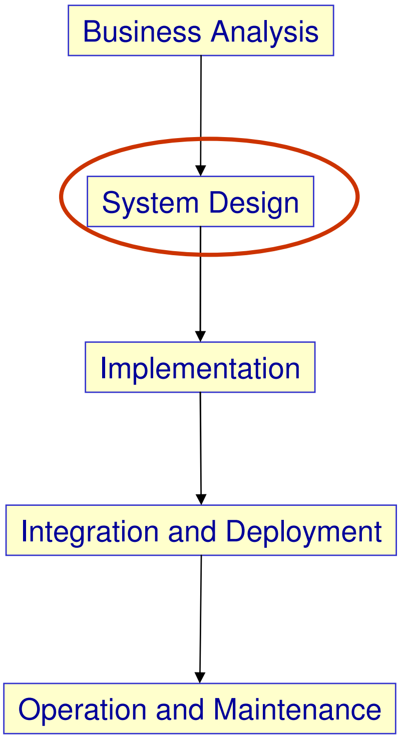
Integration and Deployment

Operation and Maintenance

- determine and specify the customer requirements
- functional and non-functional requirements

- By UML:
  - use case diagrams
  - class diagrams

System Design



Business Analysis

System Design

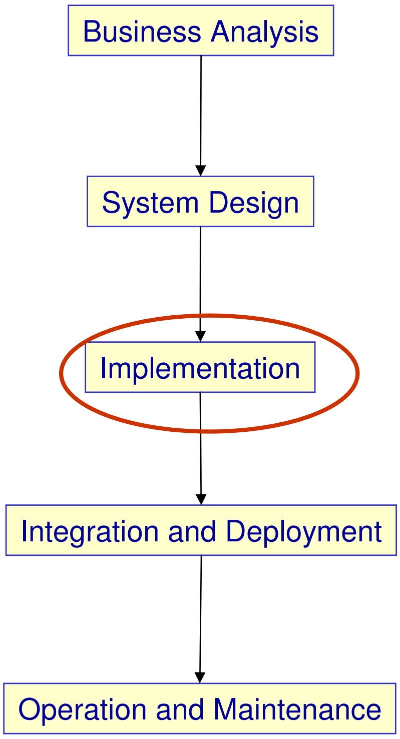
Implementation

Integration and Deployment

Operation and Maintenance

- architectural design
- detailed design

Implementation



Business Analysis

System Design

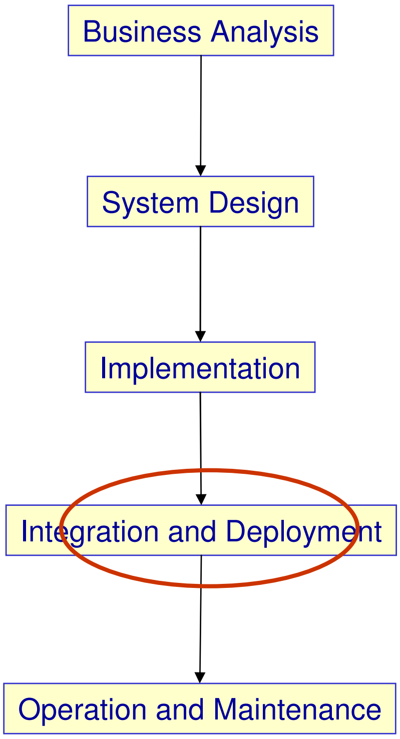
Implementation

Integration and Deployment

Operation and Maintenance

- coding
- round-trip engineering

Integration and Deployment



Business Analysis

System Design

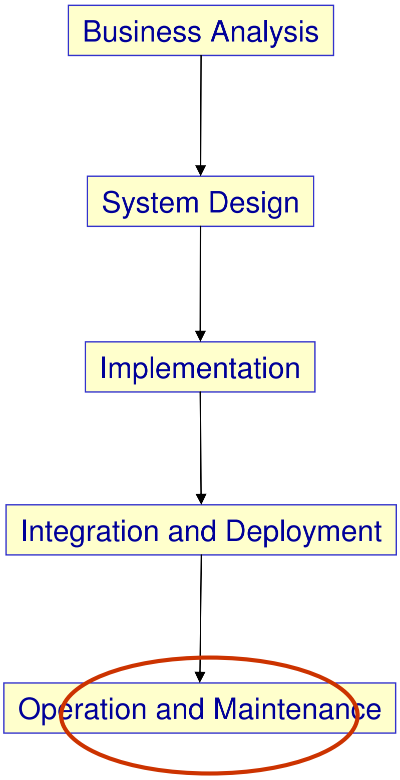
Implementation

Integration and Deployment

Operation and Maintenance

- **Module integration** can take more time and effort than any of the earlier lifecycle phases, including implementation
  - `The whole is more than the sum of the parts' (Aristotle)
- **Deployment**
  - must be carefully managed and allow, if at all possible, fallback to the old solution, if problems encountered

Operation and Maintenance



Business Analysis

System Design

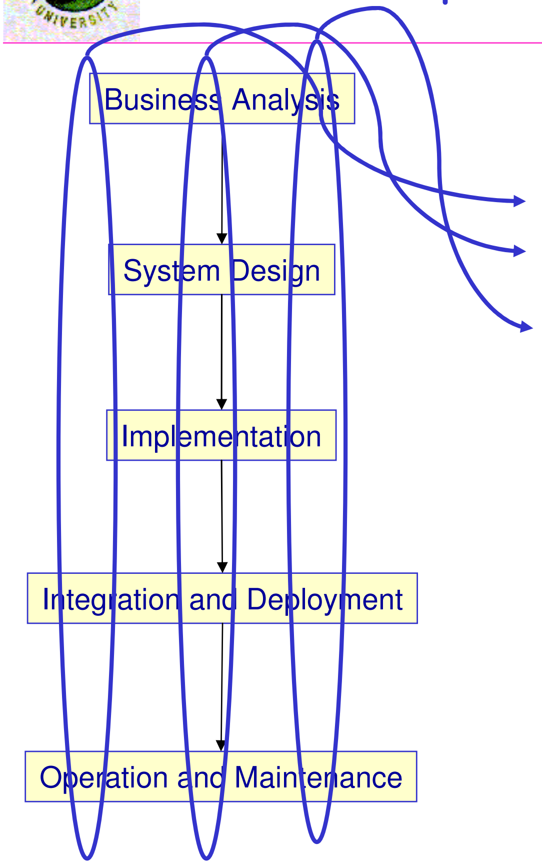
Implementation

Integration and Deployment

Operation and Maintenance

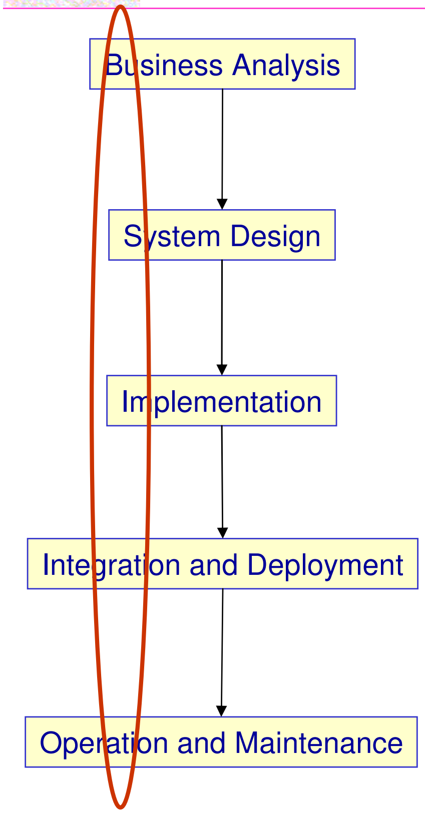
- Operation signifies change over the existing business solution, whether in software or not
- Maintenance is not only an inherent part of the software lifecycle - it accounts for most of it as far as IT personnel time and effort is concerned
  - Housekeeping
  - Adaptive maintenance
  - Perfective maintenance

Activities Spanning the Lifecycle



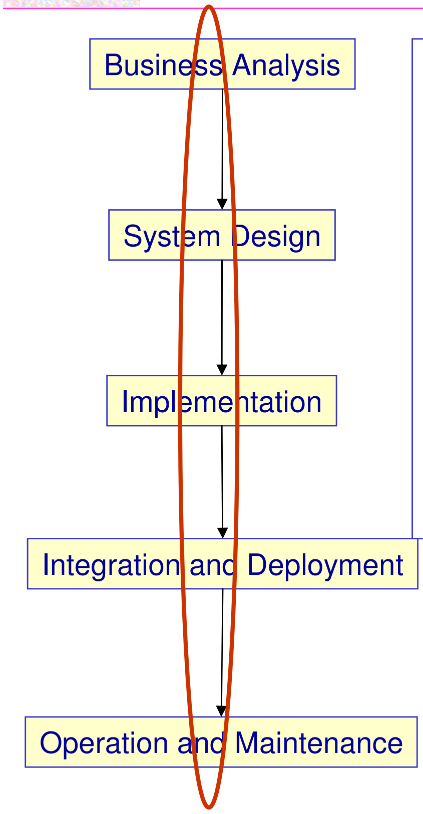
- Project planning
- Metrics
- Testing

Project Planning



- If you can’t plan it, you can’t do it
- Activity of estimating the project’s deliverables, costs, time, risks, milestones, and resource requirements
- Includes the selection of development methods, processes, tools, standards, team organization
- A moving target
- Typical constraints are time and money

Metrics



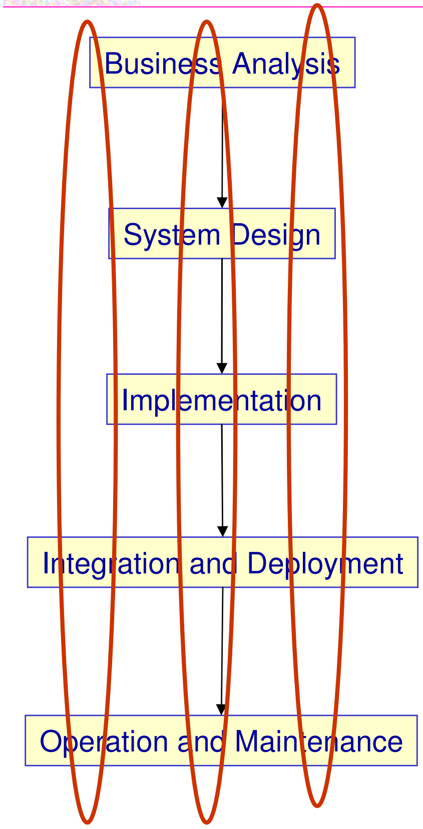
- Measuring development time and effort
- Without measuring the past, the organization is not able to plan accurately for the future
- Metrics are usually discusses in the context of software quality and complexity
- Equally important application of metrics is measuring the development models (development products) at different phases of the lifecycle => to access the effectiveness of the process and to improve the quality of work at various lifecycle phases

So, …

What is a methodology ?

How the previous phases relate to methodologies?

What is a methodology ?
How the previous phases relate to methodologies?



Business Analysis

System Design

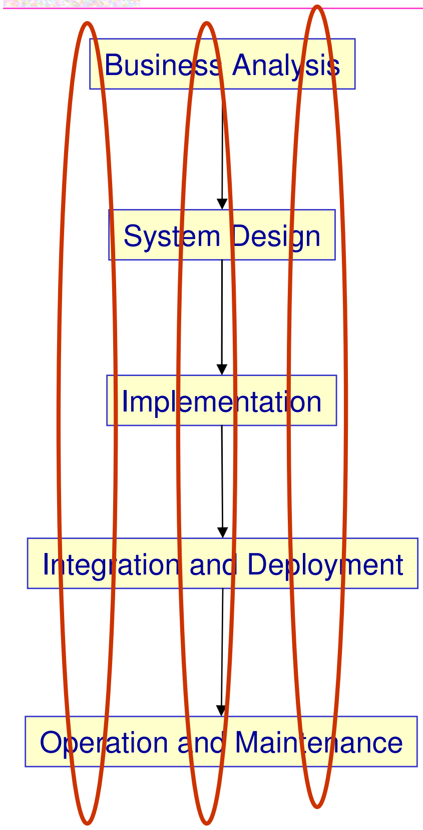
Implementation

Integration and Deployment

Operation and Maintenance

- How exactly we will work ?
  - How many steps?
  - In what order ?
  - What deliverables and when ?

Methodologies (or Development Models, or Lifecycle models)



Business Analysis

System Design

Implementation

Integration and Deployment

Operation and Maintenance

They tell us how exactly we will work

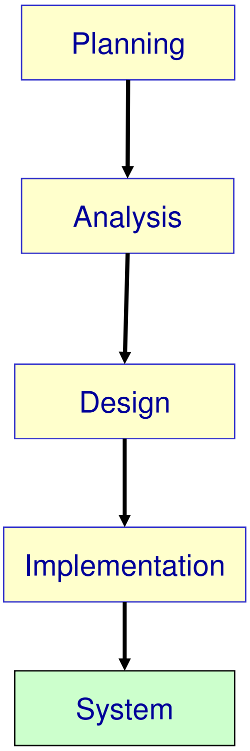
- Define an approach to software production
- Have an associated process

Methodologies

Categorizing Methodologies

- *no-methodology* (code-and-fix)
- *structured design*
    - Waterfall, Parallel
- *evolutionary / rapid application development (RAD)*
    - Phased, Prototyping, Throwaway prototyping
        - RUP (Rational Unified Process)
- *Agile Development*
    - **XP (eXtreme Programming)**
- *By reuse*
- Others:
    - Transformational
    - MDA (Model Driven Architecture)

Structured Design Methodologies
The Waterfall model


Planning

Analysis

Design

Implementation

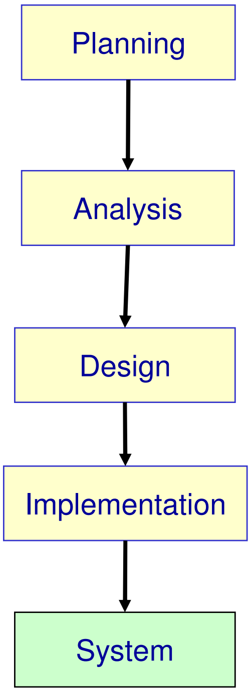
System

- Characteristics
  - linear sequence of phases
  - long reports

Βασικές αρχές μοντέλου Καταρράκτη:

- Ακολουθία σαφώς καθορισμένων βημάτων
- Κάθε βήμα καταλήγει στην δημιουργία προϊόντος (έγγραφο ή κώδικας)
- Κάθε προϊόν αποτελεί τη βάση για το επόμενο βήμα
- η ορθότητα κάθε προϊόντος μπορεί να ελεγχθεί

Structured Design Methodologies
The Waterfall model


Planning

Analysis

Design

Implementation

System

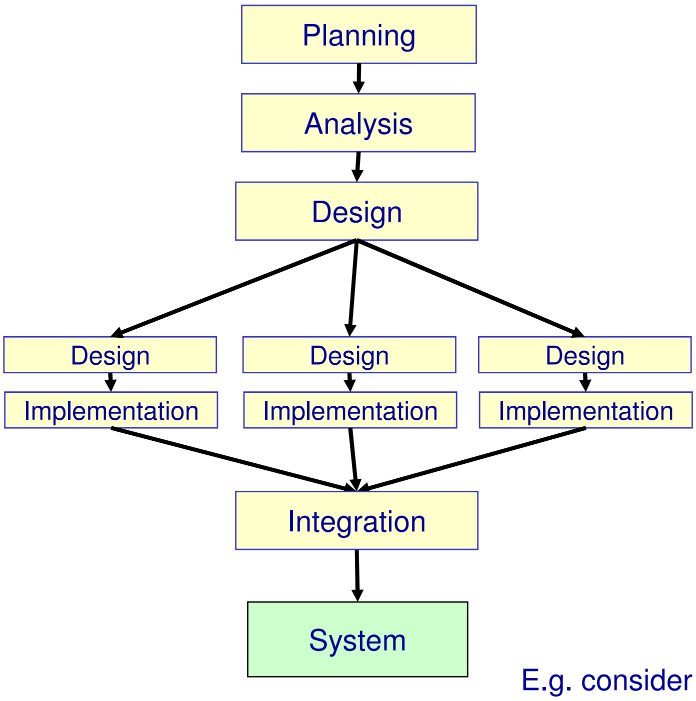
Πλεονεκτήματα

- Διαχωρισμός του έργου σε απλούστερες φάσεις
- Κάθε φάση παράγει ένα σαφώς καθορισμένο παραδοτέο

Μειονεκτήματα

- Στην πράξη οι φάσεις αλληλεπικαλύπτονται
- Στην πράξη το μοντέλο δεν είναι γραμμικό: συχνά επιστρέφουμε στην προηγούμενη φάση
- Συχνά, αλλαγές σε κάποιο στάδιο επιβάλλουν την οπισθοχώρηση και πραγματοποίηση αλλαγών σε πολλά από τα προηγούμενα στάδια
- Η σχεδίαση πρέπει να ολοκληρωθεί πριν την έναρξη της υλοποίησης
- *Ο πελάτης βλέπει τι τελικά αγοράζει πολύ αργά*
- Μακροσκελείς αναφορές που δεν βοηθούν την επικοινωνία

Structured Design Methodologies
The Parallel model



Characteristics

- it performs a general design for the entire system and then divides the projects into **subprojects** each of which can be designed and implemented in **parallel**

Advantages

- **reduces** the time needed to build a system

Disadvantages

- sometimes the subprojects are **not completely independent**

E.g. consider a partition to two subprojects:
* one with a long design phase, but short implementation phase
* one with a short design phase, but long implementation phase

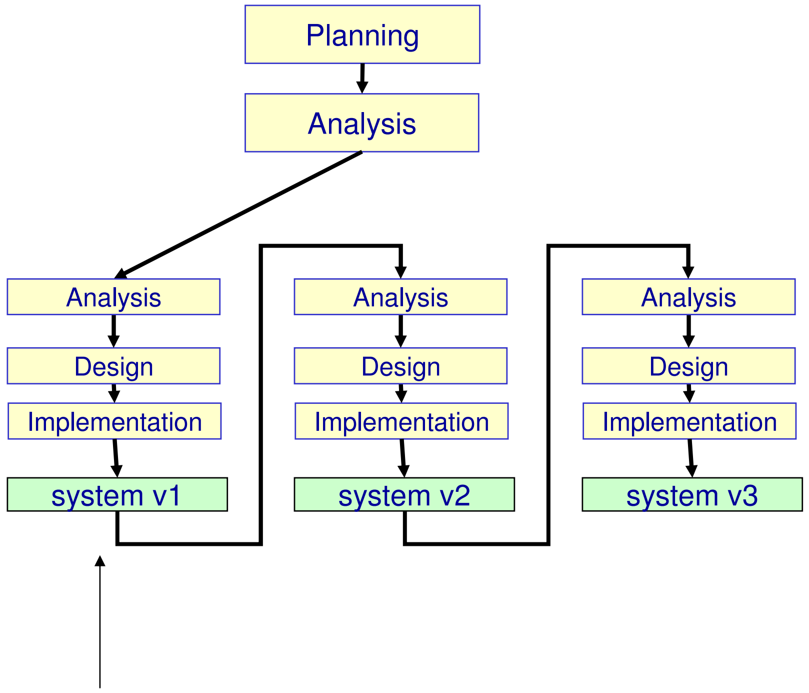
Evolutionary / Rapid Application Development (RAD) Methodologies

Keypoints

- get some parts of the system very quickly
- recommend that analysts use special techniques and tools to speed up the whole process
  - CASE tools
  - joint application design (JAD)
  - visual programming language (VBasic)
  - code generators (produce code from design)

Rapid Application Development (RAD) Methodologies

Phased Development



Most important & clear features

Εξελικτικές Μεθοδολογίες και Πρωτότυπα

Δημιουργία πρωτοτύπων

Στόχος είναι η συνεργασία με τον πελάτη και η δημιουργία ενός τελικού συστήματος ξεκινώντας από μία αρχική περιγραφή. Θα πρέπει να ξεκινά με πολύ καλά κατανοητές απαιτήσεις. Το σύστημα αναπτύσσεται προσθέτοντας σταδιακά νέα χαρακτηριστικά

Το αρχικό πρωτότυπο θα εξελιχθεί στο τελικό προϊόν

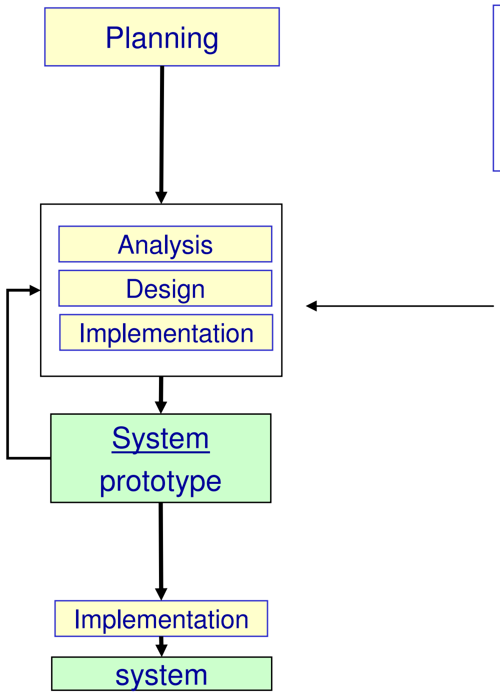
Ανάπτυξη Throwaway πρωτοτύπων

Στόχος είναι η κατανόηση των προδιαγραφών του συστήματος.. Ξεκινά με τις λιγότερο κατανοητές απαιτήσεις.

Το προϊόν πιθανότατα θα δημιουργηθεί με διαφορετικό τρόπο

(throwaway prototype: διαφημιστικά/πρόχειρο πρωτότυπο)

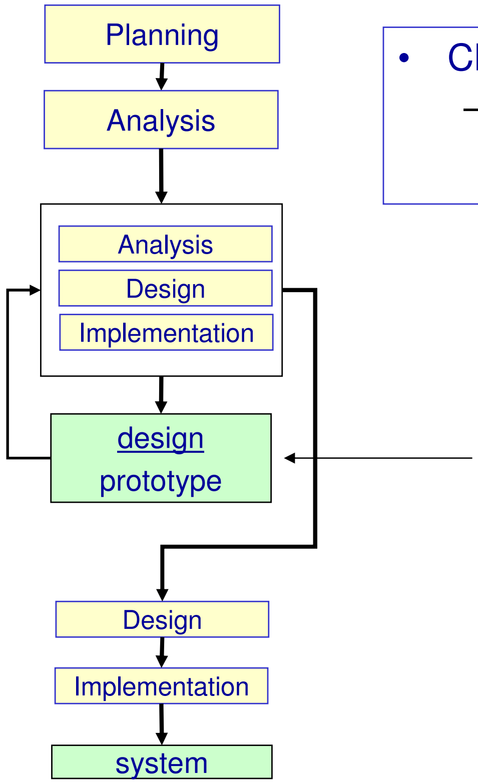
Rapid Application Development (RAD) Methodologies
Prototyping



- Characteristics
  - cyclic process
  - prototype is the central element

Concurrent phases
quick & dirty first prototype

Rapid Application Development (RAD) Methodologies
Throwaway prototyping



- Characteristics
  - prototype is just for making things clear (will not be part of the real system)

E.g. by Vbasic, Web UI,...

Rapid Application Development (RAD) Methodologies

Πλεονεκτήματα

- Αντιμετωπίζει την ασάφεια στις απαιτήσεις
- Παρέχει τη δυνατότητα στον πελάτη να αλλάξει γνώμη πριν υπογράψει
- Μείωση χρόνου ανάπτυξης
- Αρχικά πρωτότυπα χρησιμοποιούνται για εξοικείωση από τους χρήστες
- Μεγαλύτερη πιθανότητα ανάπτυξης φιλικού προς το χρήστη λογισμικού
- Ο πελάτης εμπλέκεται στην ανάπτυξη του προϊόντος
- Αυξανόμενη σταδιακά ικανοποίηση του πελάτη
- Επικοινωνία χρηστών / ομάδος ανάπτυξης

Προβλήματα

- Έλλειψη παρατήρησης στη διαδικασία (αδυναμία πρόβλεψης επαναλήψεων)
- Συστήματα λιτά δομημένα λόγω συχνών αλλαγών
- Εστίαση στη λειτουργικότητα – Λιγότερη έμφαση στις μη λειτουργικές απαιτήσεις
- Ειδικές ικανότητες μπορεί να απαιτηθούν (π.χ. γλώσσες για rapid prototyping)
- Υπερβολικός ενθουσιασμός από τον πελάτη. Ενδεχόμενη υποεκτίμηση του χρόνου ανάπτυξης
- Η δυνατότητα για συνεχείς τροποποιήσεις μειώνουν το βαθμό ικανοποίησης

Rapid Application Development (RAD) Methodologies

Εφαρμογή

- Για μικρά ή μεσαίου μεγέθους συστήματα (το αρχικό πρωτότυπο πλησιάζει το τελικό προϊόν)
- Για τμήματα μεγάλων συστημάτων (π.χ. η επαφή χρήσης ενός συστήματος εναέριας κυκλοφορίας)
- Για συστήματα με μικρό χρόνο ζωής
- Για συστήματα όπου υπάρχει αδυναμία έκφρασης των απαιτήσεων από πριν

Ακατάλληλη για:

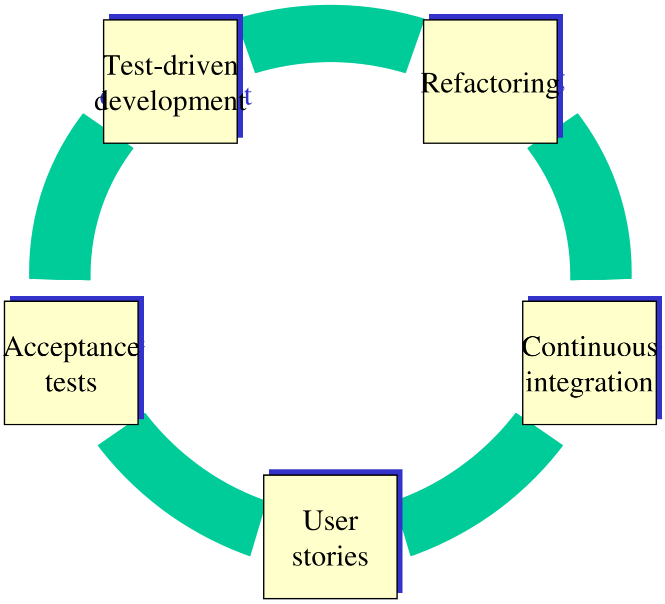
- λογισμικό ενσωματωμένων συστημάτων
- λογισμικό πραγματικού χρόνου
- επιστημονικό λογισμικό

Agile Development Methodologies

- Programming-centric methodologies
- few rules and practises (easy to follow)
- aim at eliminating the modelling and documentation overhead
- emphasise simple and iterative development
- www.agileAlliance.org
- Examples
  - eXtreme Programming (XP)
  - Scrum
  - Dynamic Systems Development Method (DSDM)

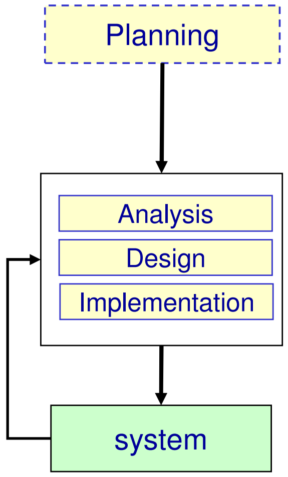
Development Models

Agile software development



- Keypoints
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - customer collaboration over contract negotiation
  - responsibility to change over following a plan
- Best known representatives
  - eXtreme Programming (XP)
  - Aspect Oriented Software Development
  - Feature-driven Development
  - Lean Development

Agile Development Methodologies
eXtreme Programming (XP)


Planning

Analysis

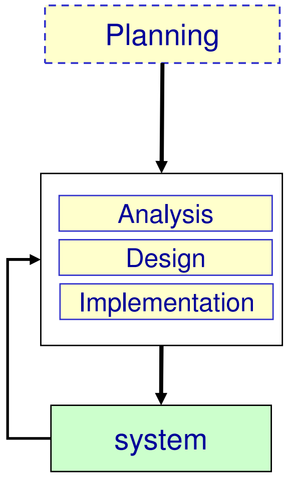
Design

Implementation

system

- The 4 core values of XP
  - communication
  - simplicity
    - The KISS principle (Keep It Simple, Stupid)
  - feedback
  - courage
- Key principles
  - continuous testing
  - simple coding performed by pairs of developers
  - close interactions with end users
- Testing and efficient coding practices
  - code is tested and placed into an integrative testing environment (every day)
- Refactoring

Agile Development Methodologies
eXtreme Programming (XP)


Planning

Analysis

Design

Implementation

system

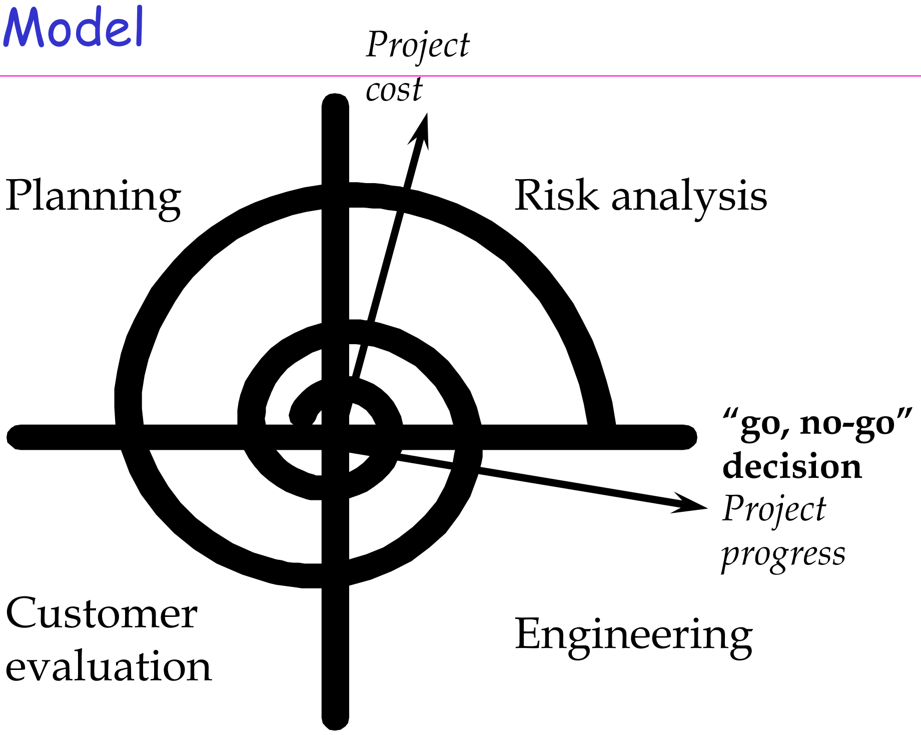
Advantages

- good for small groups
- avoid overheads of documentation, etc

Disadvantages

- lack of analysis and design documentation
- may inappropriate for big projects

Development Models
Spiral Model



Project cost

Planning

Risk analysis

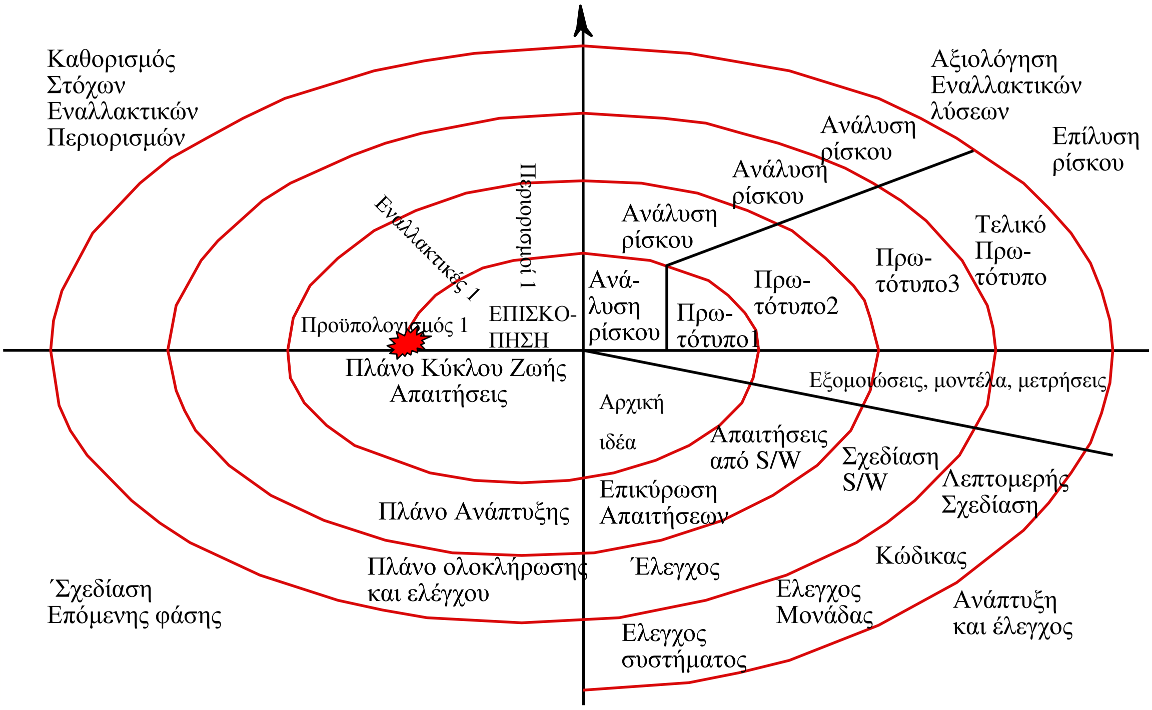
“go, no-go” decision

Project progress

Customer evaluation

Engineering

Development Models
Spiral Model



Καθορισμός
Στόχων
Εναλλακτικών
Περιορισμών

Αξιολόγηση
Εναλλακτικών
λύσεων

Επίλυση
ρίσκου

Ανάλυση
ρίσκου

Ανάλυση
ρίσκου

Ανάλυση
ρίσκου

Περιορισμοί 1

Εναλλακτικές 1

Ανά-
λυση
ρίσκου

Πρω-
τότυπο1

Πρω-
τότυπο2

Πρω-
τότυπο3

Τελικό
Πρω-
τότυπο

Προϋπολογισμός 1

ΕΠΙΣΚΟ-
ΠΗΣΗ

Πλάνο Κύκλου Ζωής
Απαιτήσεις

Εξομοιώσεις, μοντέλα, μετρήσεις

Αρχική

ιδέα

Απαιτήσεις
από S/W

Σχεδίαση
S/W

Λεπτομερής
Σχεδίαση

Πλάνο Ανάπτυξης

Επικύρωση
Απαιτήσεων

Πλάνο ολοκλήρωσης
και ελέγχου

Έλεγχος

Κώδικας

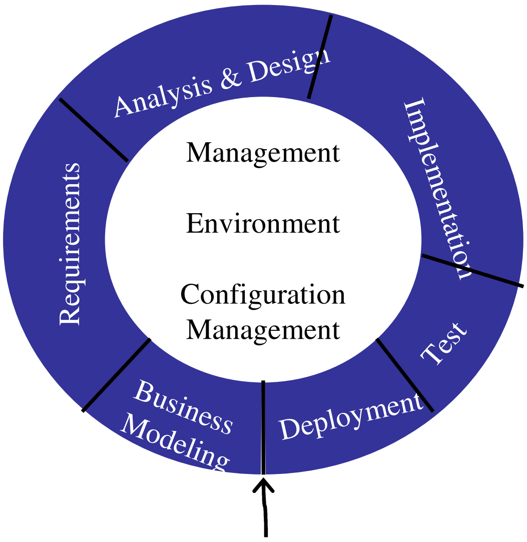
Ελεγχος
Μονάδας

Ανάπτυξη
και έλεγχος

Έχεδίαση
Επόμενης φάσης

Ελεγχος
συστήματος

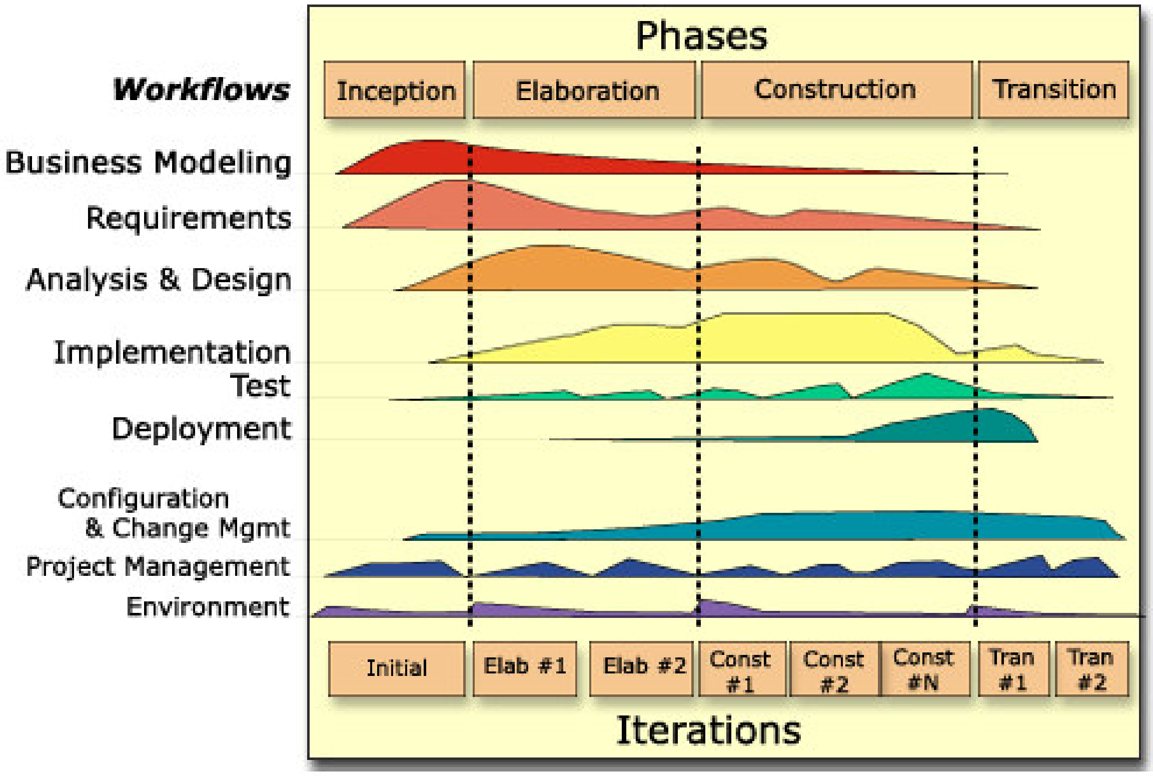
Development Models: RAD/phased
Rational Unified Process (RUP)



Start of the process

- Organises projects in 2D terms
- Horizontal dimension
  - successive phases of each project iteration
    - inception
    - elaboration
    - construction
    - transition
- Vertical dimension
  - represents 7 software development principles and supporting activities of configuration and change mgmt, project mgmnt, and environment

Development Models
Rational Unified Process (RUP)



How we Select a Methodology?

Selecting the Appropriate Development Methodology

- No methodology is best for every case
- Criteria
  - ***Clarity of User Requirements***
  - ***Familiarity with Technology***
  - ***System Complexity***
  - ***System Reliability***
    - ***e.g. a missile control system***
  - ***Short Time Schedules***
  - ***Schedule Visibility***

Selecting the Appropriate Development Methodology

Ability to develop systems with	***Waterfall***	***Parallel***	***Phased***	***Prototyp.***	***Thr. Prot.***	***XP***
with unclear requirements	Poor	Poor	Good	V.Good	V.Good	V.Good
with Unfamiliar Technology	Poor	Poor	Good	Poor	V.Good	Poor
that are Complex	Good	Good	Good	Poor	V.Good	Poor
that are Reliable	Good	Good	Good	Poor	V.Good	Good
with a Short Time Schedule	Poor	Good	V.Good	V.Good	Good	V.Good
with Schedule Visibility	Poor	Poor	V. Good	V. Good	Good	Good

Process Improvement Models

Process Improvement Models

- Every organization engaged in software production wants to improve its development process.
- To improve, the organization has to know the problem of its current process

- Process Improvement Models
  - Capability Maturity Model (CMM)
  - ISO 9000 family of quality standards

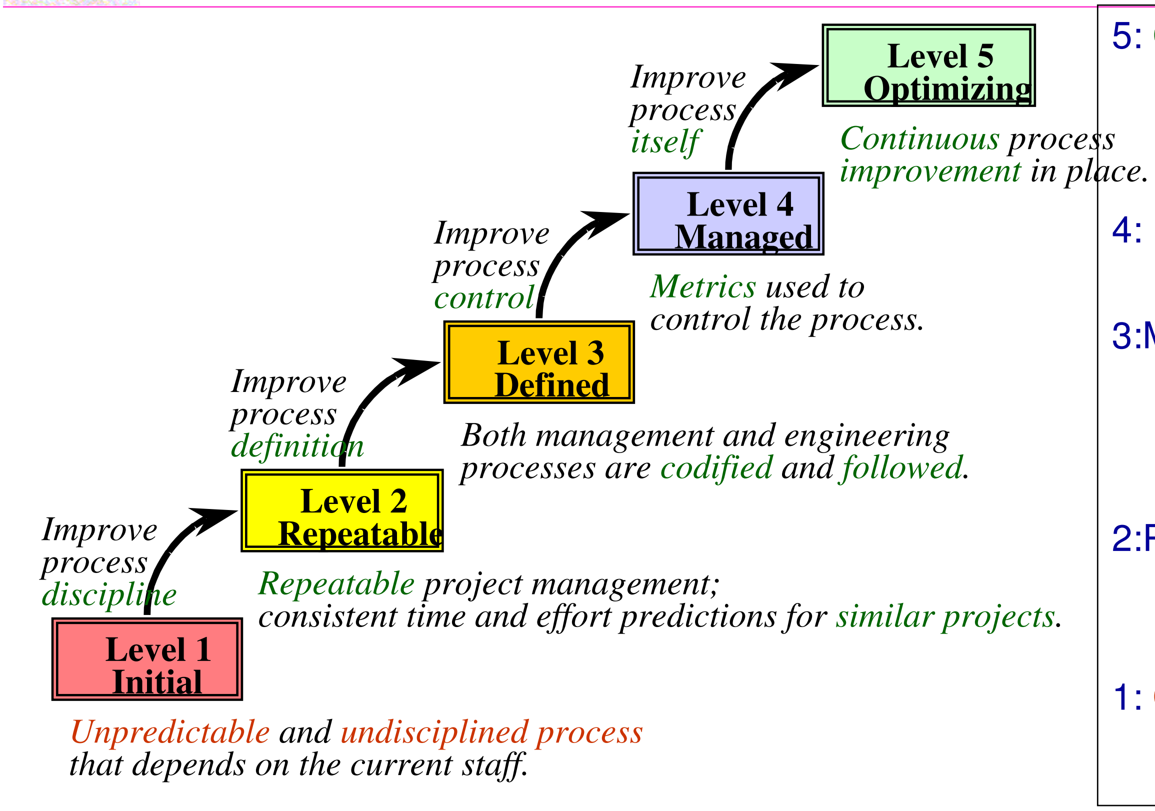
Process Improvement Models
Capability Maturity Model (CMM)

- Capability Maturity Model (CMM)
  - “the stairway to software excellence”
  - A popular method for process assessment and improvement
  - is essentially a questionnaire that an IT organization fills in
    - The questionnaire if followed by a verification and attestation process, which assigns the organization to one of the five CMM levels (the higher the better)

SEI (Software Engineering Institute), U. of Carnegie Mellon

Process Improvement Models

Process maturity levels in CMM



Level 5 Optimizing

Continuous process improvement in place.

Improve process itself

Level 4 Managed

Metrics used to control the process.

Improve process control

Level 3 Defined

Both management and engineering processes are codified and followed.

Improve process definition

Level 2 Repeatable

Repeatable project management; consistent time and effort predictions for similar projects.

Improve process discipline

Level 1 Initial

Unpredictable and undisciplined process that depends on the current staff.

5: **Continuous quality improving**

4: Measurement

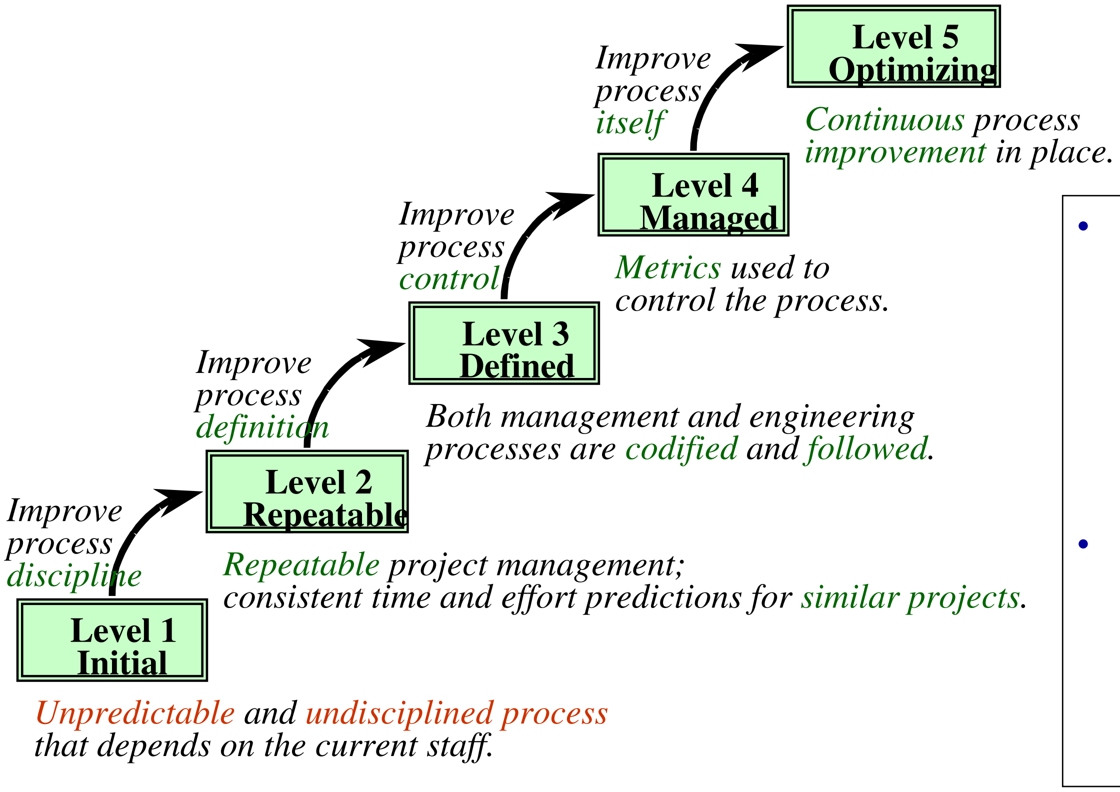
3:Method and tools

2:Project mgmt

1: **Chaos**

Process Improvement Models

Process maturity levels in CMM



Level 5 Optimizing

Continuous process improvement in place.

Improve process itself

Level 4 Managed

Metrics used to control the process.

Improve process control

Level 3 Defined

Both management and engineering processes are codified and followed.

Improve process definition

Level 2 Repeatable

Repeatable project management; consistent time and effort predictions for similar projects.

Improve process discipline

Level 1 Initial

Unpredictable and undisciplined process that depends on the current staff.

- Most organizations are at level 1; some at level 2; very few are known to be at level 5
- experience shows that it takes years to progress one level up the maturity scale

Process Improvement Models
CMM Level 2

- For CMM level 2 an organization must provide positive answers to all these questions (and more)
  - Does the software quality assurance function have a **management reporting channel** separate from the software development project mgmt?
  - Is there a **software configuration control function** for each project that involves software development?
  - Is a formal process used in the management review of each software development prior to main contractual commitments ?
  - Is a **formal procedure** used to produce **software development schedules** ?
  - Are **formal procedures** applied to **estimate software development cost**?
  - Are **statistics** on **software code** and **test errors** gathered ?
  - Does senior management have a mechanism for the **regular review** of the status of software development projects ?
  - Is a mechanism for **controlling changes to the software requirements** ?

CMM level and some statistics (for a project of 200K lines of code)

Organization's **CMM Level**	Project **duration** (months)	Project **person months**	Number of **Defects**	Median **Cost**
1	30	600	61	5.5 M $
2	18	153	12	1 M $
3	15	80	7	0.5 M$

Process Improvement Models

ISO 9000 family of quality standards

- ISO: International Organization for Standardization
- The ISO standards
  - apply to the **quality management** and the process to produce a quality product
  - apply to **any industry** and all types of businesses, including software development
- The main premise
  - if the **process** is right then the process outcome (product or service) will also be right
  - but the ISO standards do not enforce specific processes -> the standards provide models of **what** must be accomplished, **not how** activities must be performed

Reading and References

- **Systems Analysis and Design with UML Version 2.0** (2nd edition) by A. Dennis, B. Haley Wixom, D. Tegarden, Wiley, 2005. CHAPTER 1
- **Requirements Analysis and System Design** (2nd edition) by Leszek A. Maciaszek, Addison Wesley, 2005, CHAPTER 1
- Shari Lawrence Pfleeger. Τεχνολογία Λογισμικού: Θεωρία και Πράξη, 1. Κλειδάριθμος, Αθήνα, 2003, Κεφάλαιο 1,2.
- **Object-Oriented Systems Analysis and Design Using UML** (2nd edition) by S. Bennett, S. McRobb, R. Farmer, McGraw Hil, 2002, Chapter 2