EPOKA UNIVERSITY

SWE 202 - Software Modeling and Design

Lecturers: Msc. Ari Gjerazi
Msc. Julia Marku
Email: agjerazi@epoka.edu.al
jmarku@epoka.edu.al

Course Introduction

Unit outline

- Objectives
- Assessment
- Passing the unit
- Lectures, practice classes, the lecturer and consultation
- Recommended reading
- Assignment work
- Web pages

Objectives (1)

- Knowledge on the difficulties of specifying and engineering requirements for producing large software products, leading to
  - an appreciation of the need for requirements engineering methodologies
  - understanding of the distinction between software analysis and design process and programming a software.

Objectives (2)

- An understanding of, and ability to apply, the methods of analysis and design, including:
  - structured analysis and design using Yourdon notation
    - Context Diagram, Event Lists, Data-Flow Diagrams, Entity-Relationship Diagram, State Transition Diagrams, Process Specifications, Data Dictionary, Structure Chart
  - object-oriented analysis and design using UML
    - Use Cases, Class Diagrams, Interaction Diagrams, State Diagrams, Package Diagrams, Activity Diagrams

Objectives (3)

- Knowledge of, and the ability to apply the principles of design patterns and to make use of them in design process for building proper class structures and solid software components. (A dive into Software Architecture)

- Knowledge of , and the ability to apply, principles of user interface design such as affordances, awareness of mental models, visibility, mapping and feedback.

Assessment and Passing

- There are two assessment components:
  - Examinations through:
    - Midterm (25 points)
    - Final Exam (35 points)

  There will be a practical project:
    - A group project with assignments worth 40 points

Assessment and Passing

- You need to participate in at least __of lectures in order to pass.
- You need to achieve 45% in both exams and the assignments and achieve an overall mark of 50%, as a example:
  - You must get at least 25 points out of 55 for the exams
  - You must get at least 15 points out of 45 for the projects
  - You must get at least 55 points out of 100

Lectures

- Lectures will be held at:
  - Please check the timetable…
- Lecture slides for each week will be made available on moodle web site
  - Lecture slides are *not* “lecture notes”. Notes are what *you* write during lectures.
- All lecture material, worksheets and assignment work is examinable
  - *It is your responsibility to ensure that you have copies of all such materials*

Practice Classes

- There will be one practice class (lab) each week
- Students are expected to attend at practice class (lab)
- During a practice class, students are expected to report on their project progress following the formation of the groups and selection of the project theme.
- Practice classes (labs) start in week 2

Recommended Reading

- The following book cover the basic material and you may use it as main book:
  - Software modeling and design_ UML, use cases, patterns, and software architectures, Hassan Gomma (2011)
  - Head First Design Patterns Building Extensible and Maintainable Object-Oriented Software, Eric Freeman, Elisabeth Robson, O'Reilly Media (2021)

  Other secondary books:
  - Software Engineering, A practitioner's approach, 9th Ed., Pressman&Maxim (2020)
  - Software Engineering, A methodical Approach, 2nd Ed., Foster&Towle (2022)
  - System Analysis & Design, An Object-Oriented Approach with UML, 5th Ed, Dennis, Wixon & Tegarden (2015)
  - Software Requirements: Best Practices, 3rd Ed., Wiegers&Beatty (2013)
- A list of further useful books will be provided in the lessons.

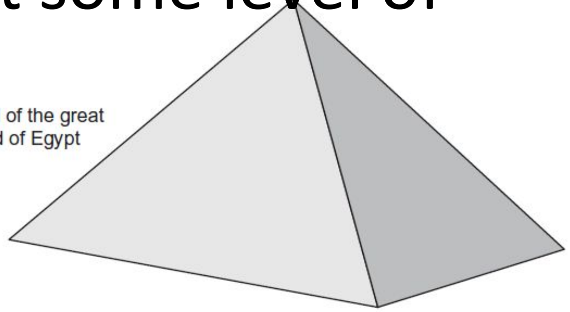
What is Software Engineering?

Group Exercise

- Break into groups of 3 or 4 (i.e. your neighbours, don’t move around the room)
- Take 5 minutes to write down a definition of software engineering - this can be in point form
- After 5 minutes, we will collect definitions from the class

What is Software Modeling?

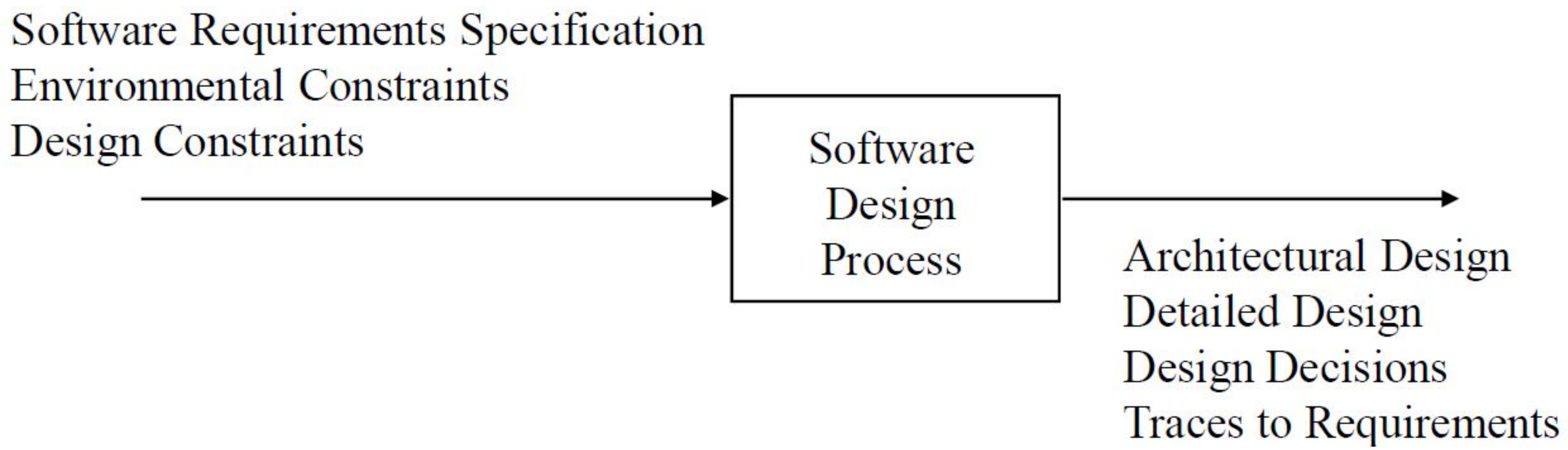
- Modeling Ancient Egypt, Rome, and Greece to provide small-scale plans in art and architecture.
- Modeling is used in Science and Engineering to provide abstractions of a system at some level of precision and detail.


a) A model of the great pyramid of Egypt

b) The great pyramid of Egypt

Software Design

- What is design?
  - noun: mental plan, preliminary sketch or outline
  - verb: to conceive in the mind; to invent
- What is software design?
  - As a product
    - Output of design process
  - As a process
    - Approach to doing design
- Context of Software Design



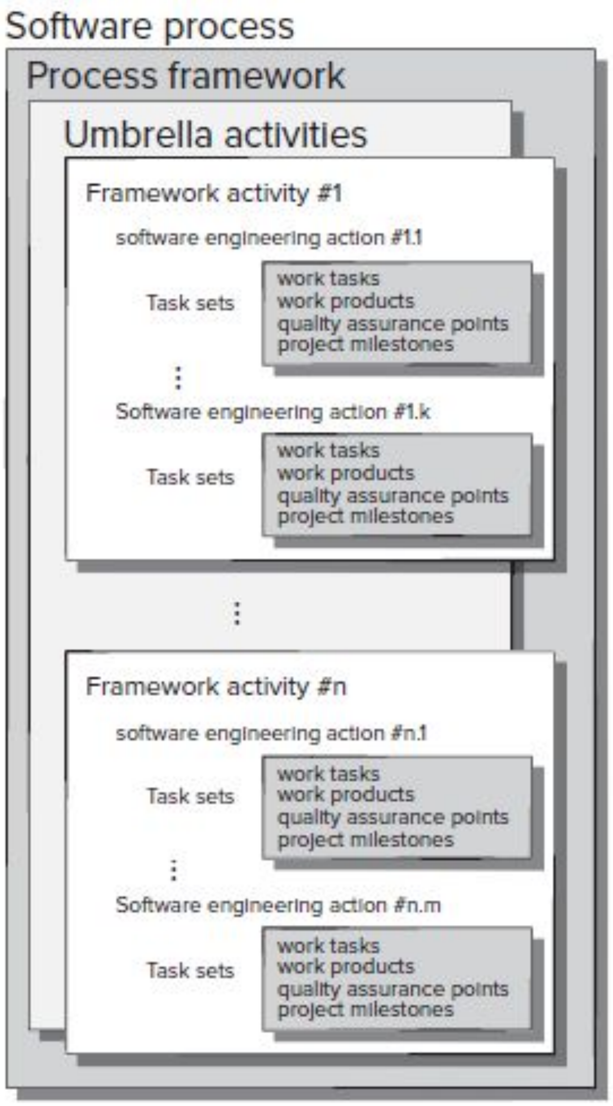
Software Design Process

- Software life cycle (a.k.a. software process)
  - Phased approach to software development
- Software life cycle (a.k.a. process) models
  - Waterfall – limitations of Waterfall Model
  - Incremental - evolutionary prototyping
  - Exploratory - throwaway prototyping
  - Spiral model – risk driven process model

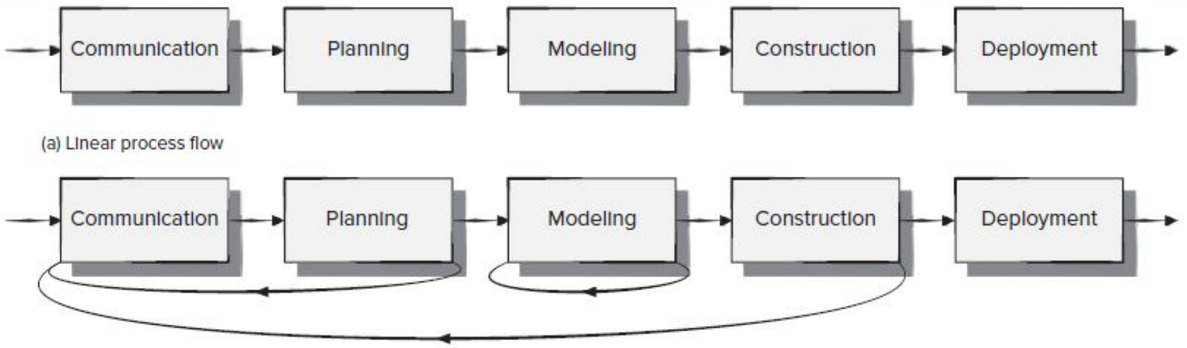
Software Life Cycle Model

- Requirements Analysis and Specification
  - Analysis of user's problem
  - Specification of "what" system shall provide user
- Architectural Design
  - Specification of "how" system shall be structured into components
  - Specification of interfaces between components
- Detailed Design
  - Internal design of individual components
    - Design of logic and data structures
- Coding
  - Map component design to code
- Software Testing
  - Unit, Integration, System and Acceptance
- Software Maintenance
  - Modifications (Fixes, Improvements, Changes, Redesigns)

Software Processes

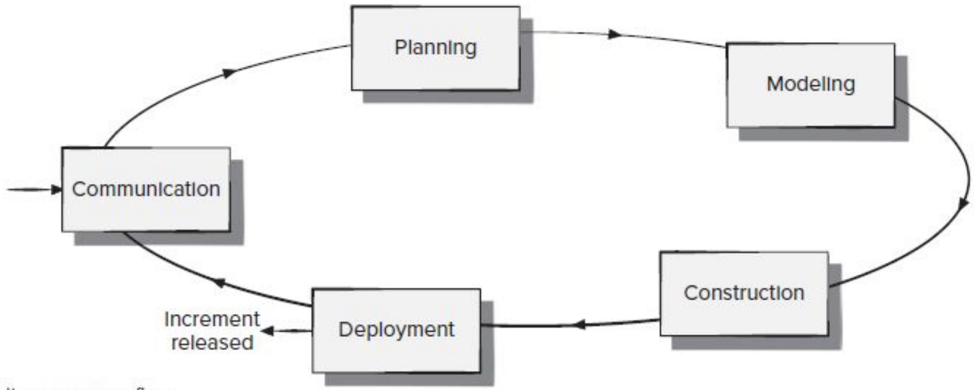


Process Flow

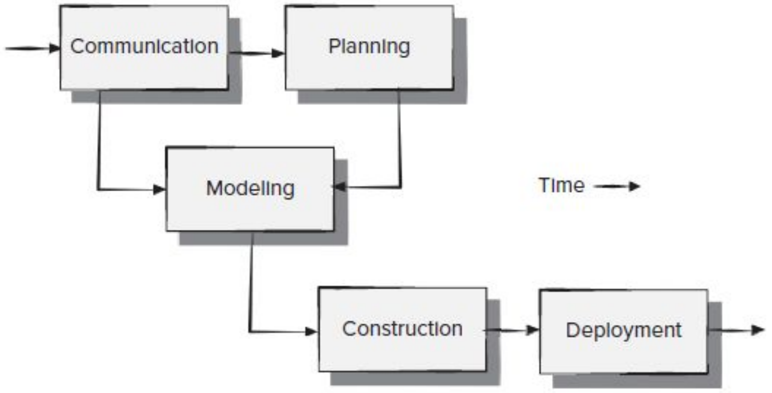


(a) Linear process flow

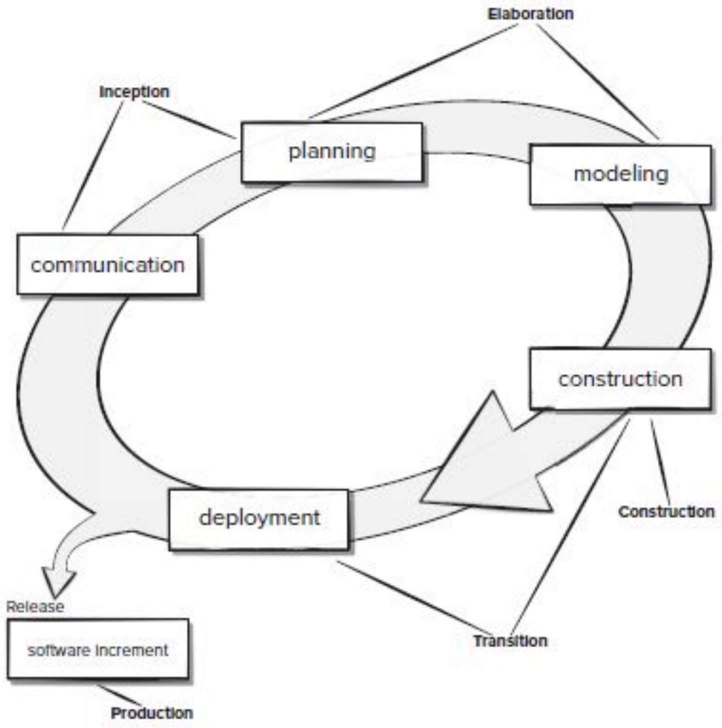
(b) Iterative process flow



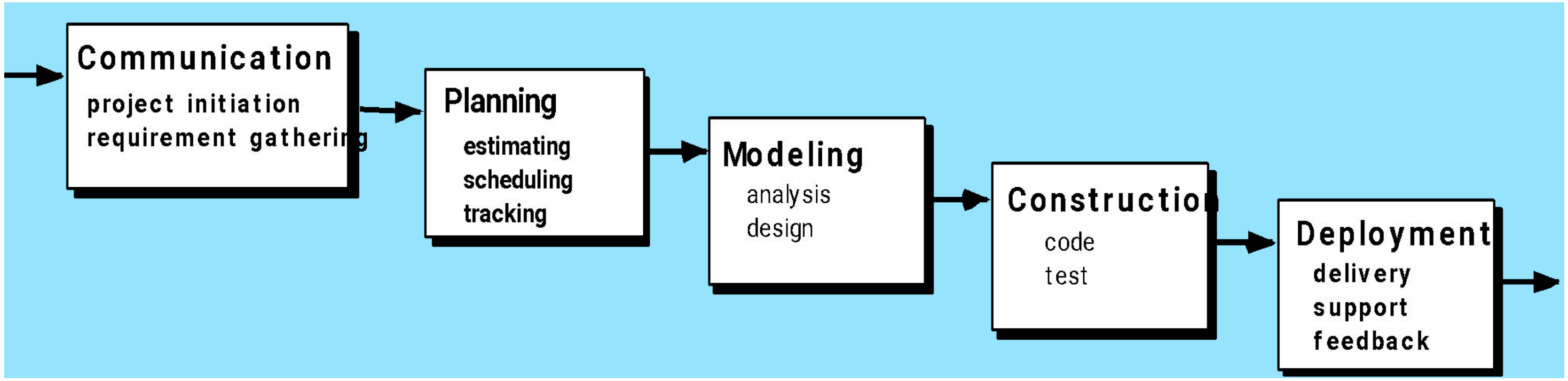
(c) Evolutionary process flow



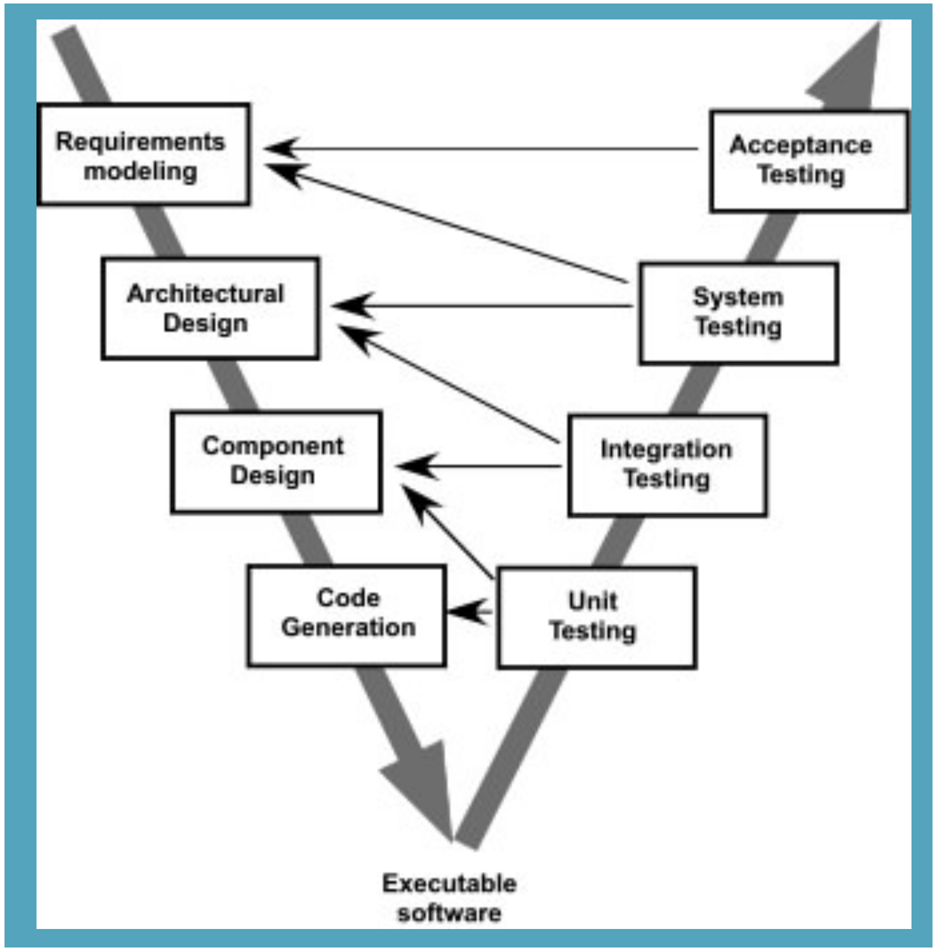
(d) Parallel process flow



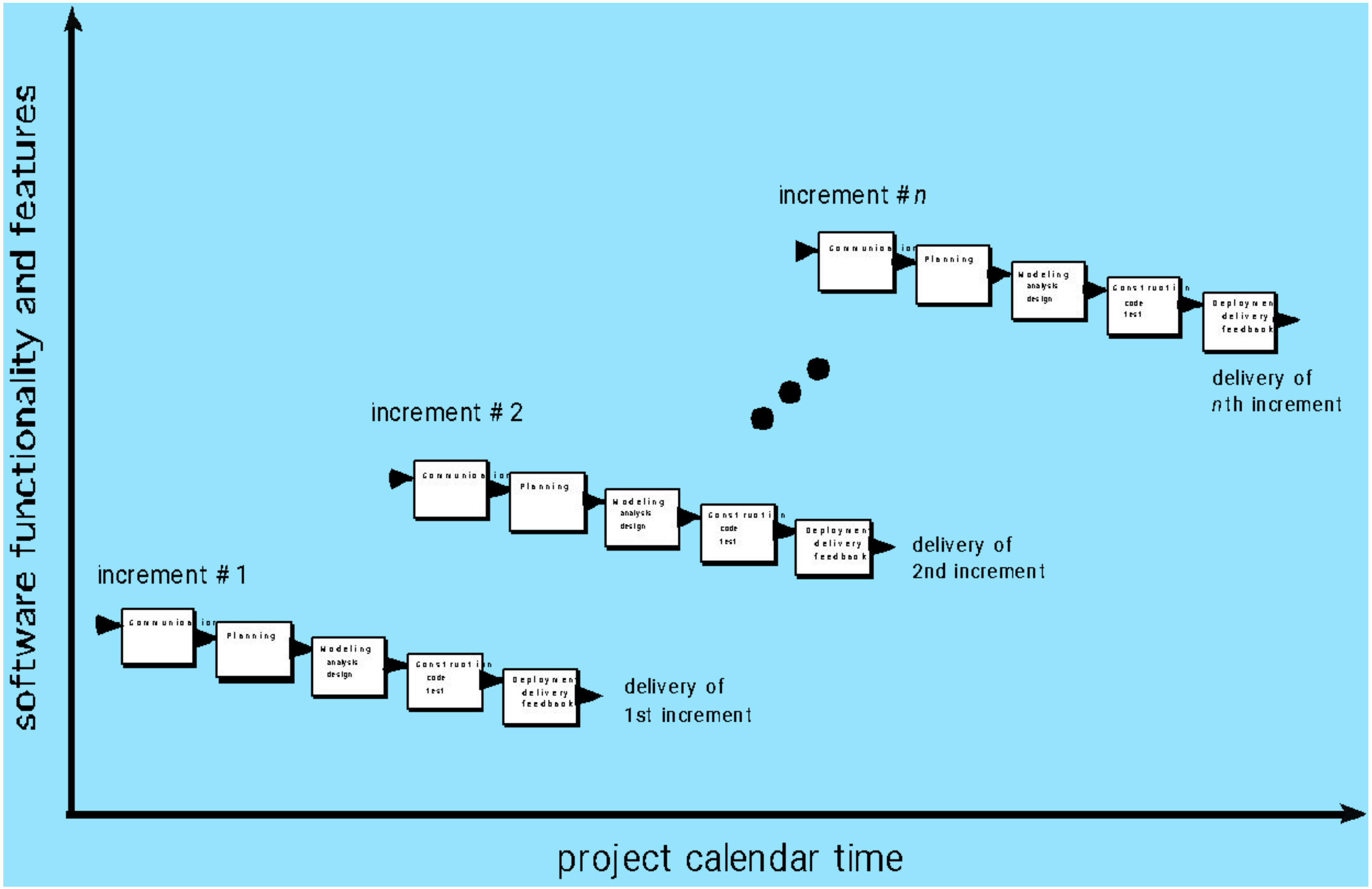
The Waterfall Model



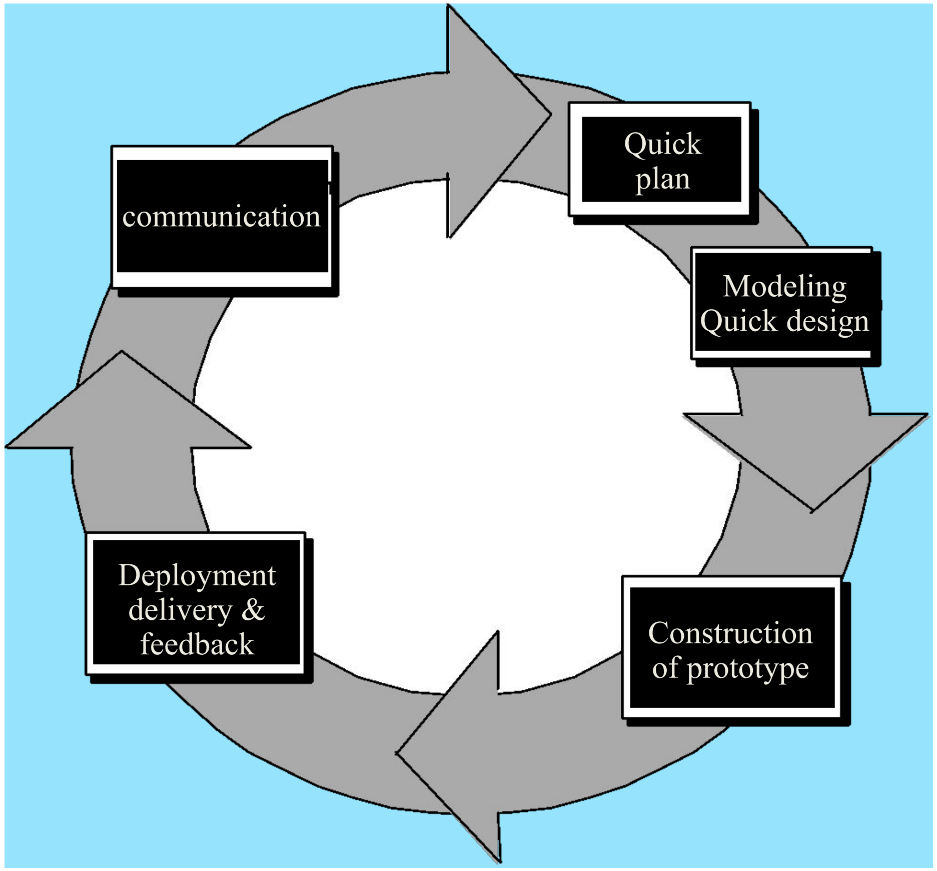
The V-Model


Requirements modeling
Acceptance Testing
Architectural Design
System Testing
Component Design
Integration Testing
Code Generation
Unit Testing
Executable software

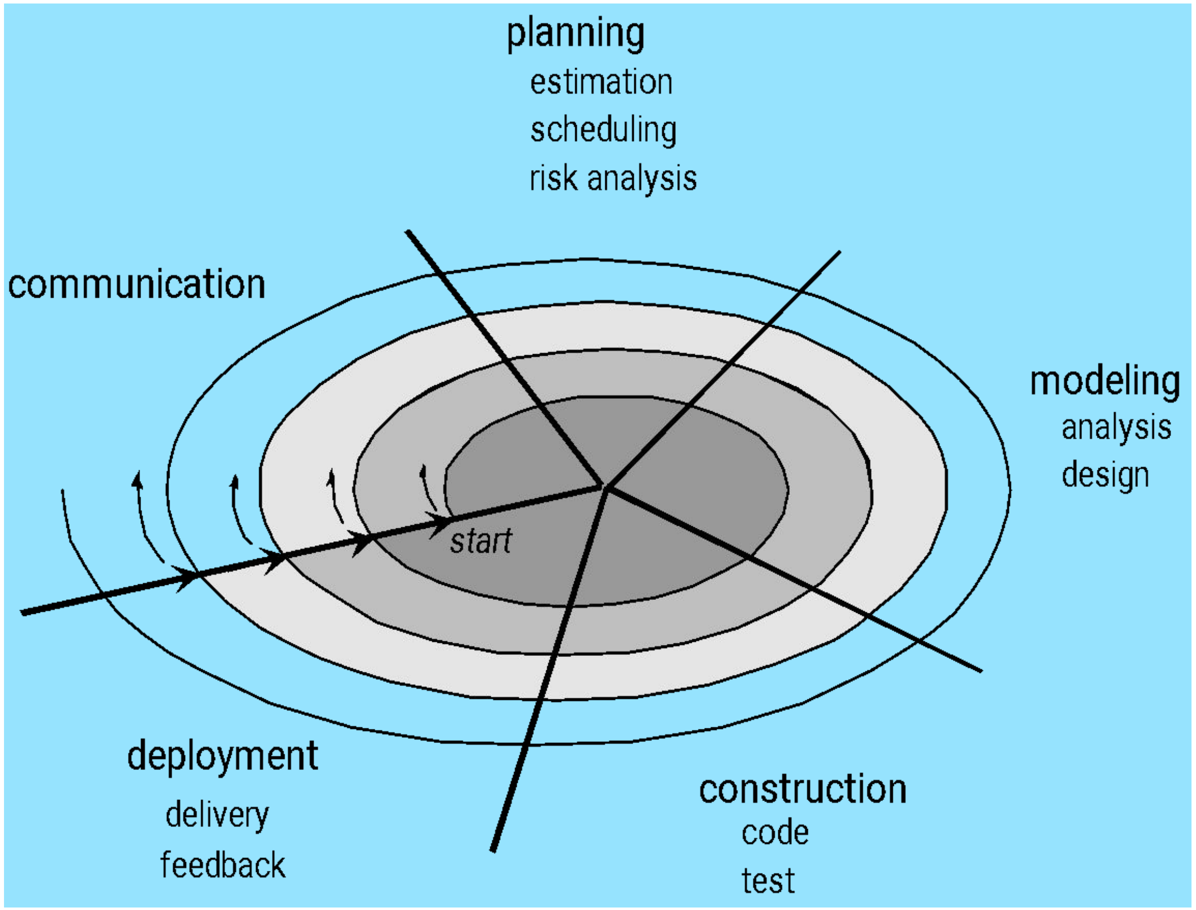
The Incremental Model


software functionality and features
increment # n
Communication
Planning
Modeling analysis design
Construction code test
Deployment delivery feedback
delivery of nth increment
increment # 2
delivery of 2nd increment
increment # 1
delivery of 1st increment
project calendar time

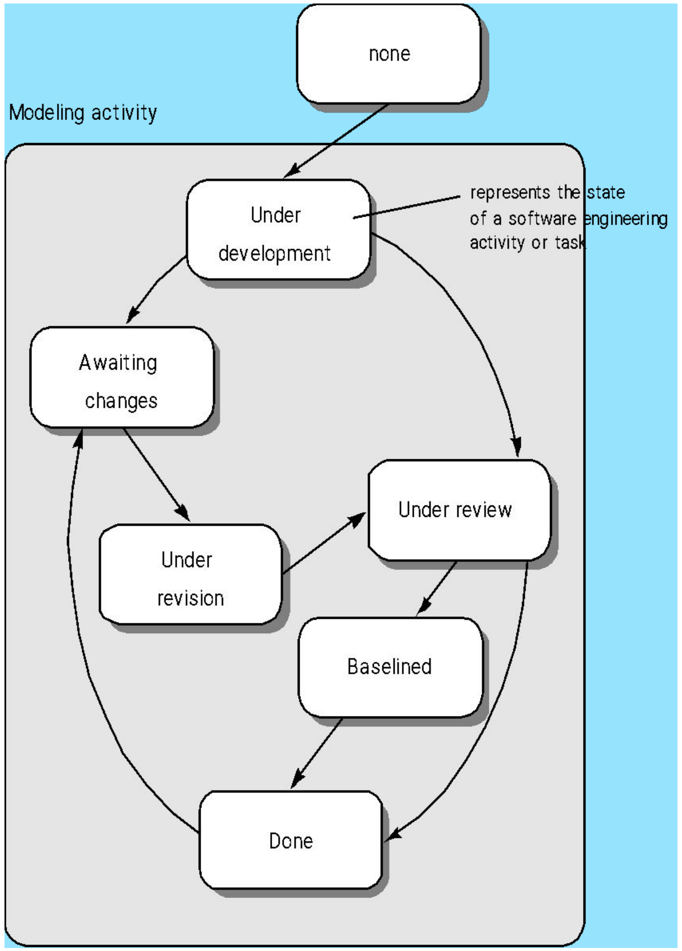
Evolutionary Models: Prototyping



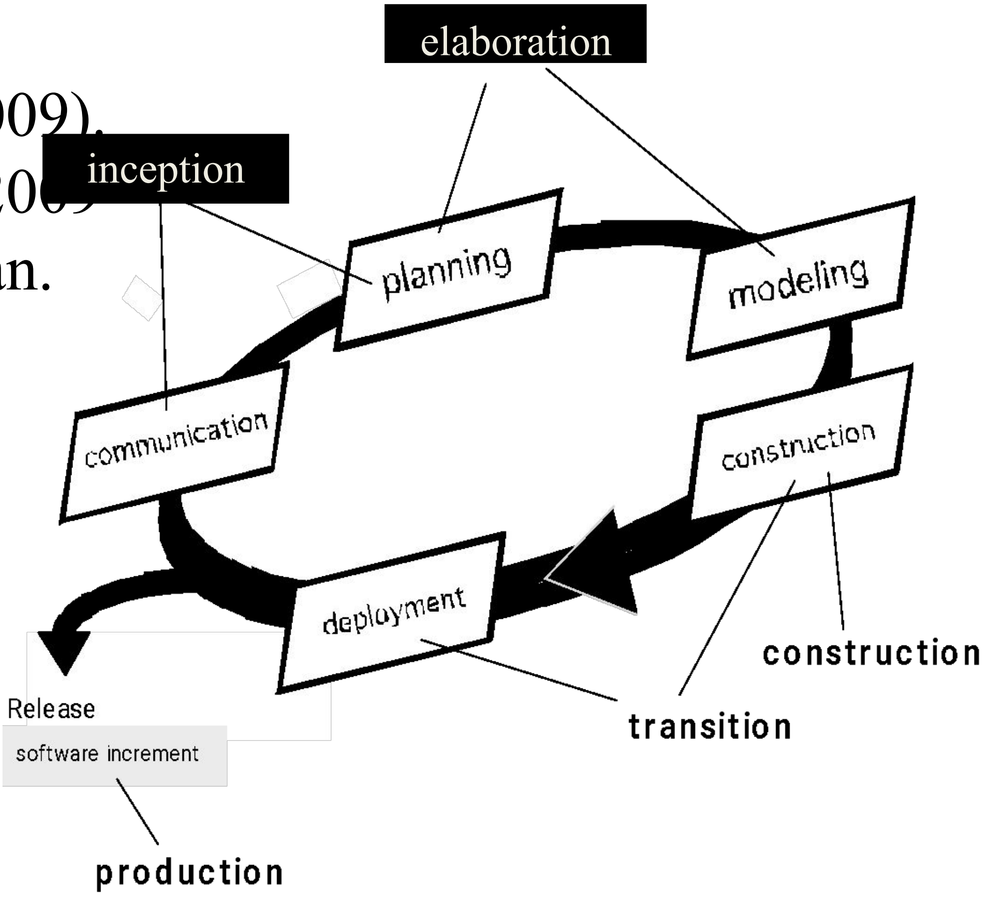
Evolutionary Models: The Spiral



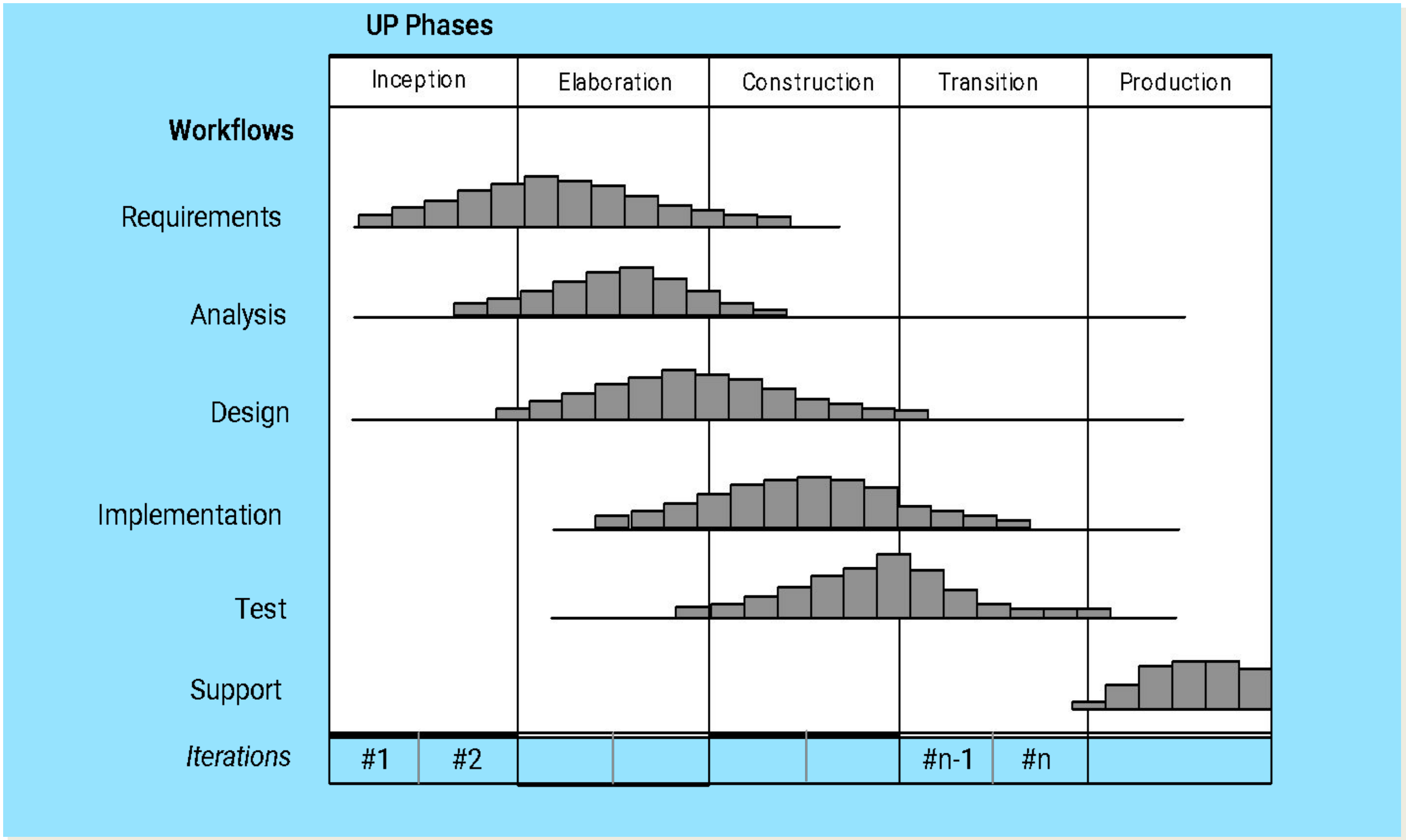
Evolutionary Models: Concurrent



The Unified Process (UP)



UP Phases


UP Phases
Inception
Elaboration
Construction
Transition
Production
Workflows
Requirements
Analysis
Design
Implementation
Test
Support
Iterations
#1
#2
#n-1
#n

UP Work Products

Inception phase

Vision document
Initial use-case model
Initial project glossary
Initial business case
Initial risk assessment.
Project plan,
  phases and iterations.
Business model,
  if necessary.
One or more prototypes

Elaboration phase

Use-case model
Supplementary requirements
  including non-functional
Analysis model
Software architecture
  Description.
Executable architectural
  prototype.
Preliminary design model
Revised risk list
Project plan including
  iteration plan
  adaptedworkflows
  milestones
  technical work products
Preliminary user manual

Construction phase

Design model
Software components
Integrated software
  increment
Test plan and procedure
Test cases
Support documentation
  user manuals
  installation manuals
  description of current
    increment

Transition phase

Delivered software increment
Beta test reports
General user feedback

EPOKA UNIVERSITY

THE END