REKAYASA PERANGKAT LUNAK

PLPG

- **Buku Acuan :**

  Software Engineering: A Practitioner's Approach

  Pengarang : Roger S. Pressman

  Penerbit: Fourth Edition, McGraw-Hill, 1997

JADWAL PERKULIAHAN

No	Materi Pokok	Waktu
1	Introduction to Software Engineering	
2	Project Planning Project Introduction & Team Organization	
3	Software Lifecycle Software Process Models 1	
4	Software Lifecycle Software Process Models 2	
5	Requirement Engineering	
6	Requirements Analysis: Structured Techniques 1	
7	Requirements Analysis: Structured Techniques 2 Software Design Demonstration	
8		

JADWAL PERKULIAHAN

No	Materi Pokok	Waktu
9	Software Design Concepts	
10	Software Architecture	
11	Implementation & Prototyping	
12	Project Presentation & Demonstration 1	
13	Project Presentation & Demonstration 2	
14	Project Presentation & Demonstration 3	
15	Project Presentation & Demonstration 4	
16		

PRE TEST

- Apa yang anda ketahui tentang perangkat lunak ?
- Apa yang anda ketahui tentang rekayasa perangkat lunak ?

Rekayasa Perangkat Lunak

- Definisi Klasik (1969)

  "The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines."

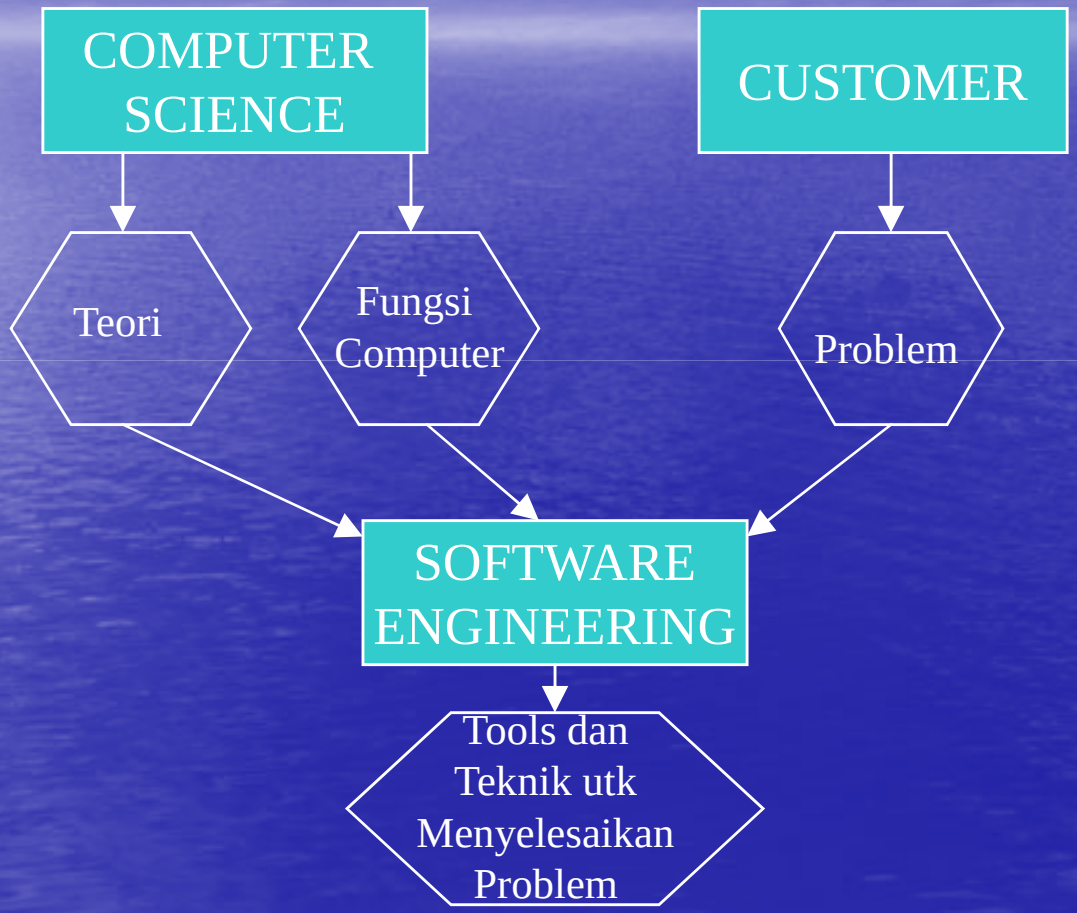
  Penerapan prinsip engineering untuk memperoleh software yang ekonomis, reliable dan bekerja efisien pada komputer

- Definisi IEEE (1993)

  "Software Engineering: (1) The application of a systematic, disciplines, quantifiable approach to the development, operation, and maintenance of software; that is the application of engineering to software. (2) The study of approaches as in (1)."

  RPL : (1) Penerapan secara sistematis, disiplin, pendekatan terukur pada pengembangan, pengoperasian dan pemeliharaan software. (2) Studi terhadap (1)

Penyelesaian Masalah (Problem Solving)



Perkembangan Software

- Generasi Awal
  - Batch orientation
  - Custom software
- Generasi Kedua
  - Multi-user, Real-time
  - Database
  - Product software
- Generasi Ketiga
  - Distributed systems
  - Low cost hardware
- Generasi Keempat
  - Desktop systems
  - Object Oriented Technologies
  - Expert Systems
  - AI, neural networks
  - Parallel computing
  - Network computers

Ongoing Problems (Masalah yang terus menerus ada)

- Kemajuan perangkat keras melebihi kemampuan membuat software
- Kemampuan membangun program baru tidak dapat memenuhi permintaan program-program baru, begitu juga kecepatan membangun program tidak dapat mnegikuti kebutuhan bisnis dan pasar
- Penyebaran penggunaan computer telah membuat kebergantungan masyarakat thdp komputer
- Tantangan untuk membangun software dengan *reliability & quality* yang tinggi
- Kemampuan men-support dan meningkatkan program terancam oleh design yang buruk dan keterbatasan sumberdaya

Coming Up Next...

- Project Overview
- Organisasi Team
- Organization danPerencanaan Project

Breakdown of Effort Cost - Software Cost

Estimation by Capers Jones

No.	Activity	Effort
1	Requirements	3.84%
2	Prototyping	4.50%
3	Architecture	2.25%
4	Project Plans	1.33%
5	Initial Design	3.84%
6	Detailed Design	4.50%
7	Design Reviews	3.02%
8	Coding	13.50%
9	Reuse acquisition	1.13%
10	Package Purchase	1.69%
11	Code Inspection	4.50%
12	Independent V&V	5.42%
13	Configuration Mgt.	0.41%
14	Integration	2.71%
15	User Documentation	9.67%
16	Unit Testing	4.50%
17	Function Testing	4.50%
18	Integration Testing	3.84%
19	System Testing	3.38%
20	Beta Testing	3.02%
21	Acceptance Testing	1.94%
22	Independent Testing	3.38%
23	Quality Assurance	4.50%
24	Installation and Training	1.94%
25	Project Management	6.75%

Effort Breakdown of ~10000 Projects - Capers Jones

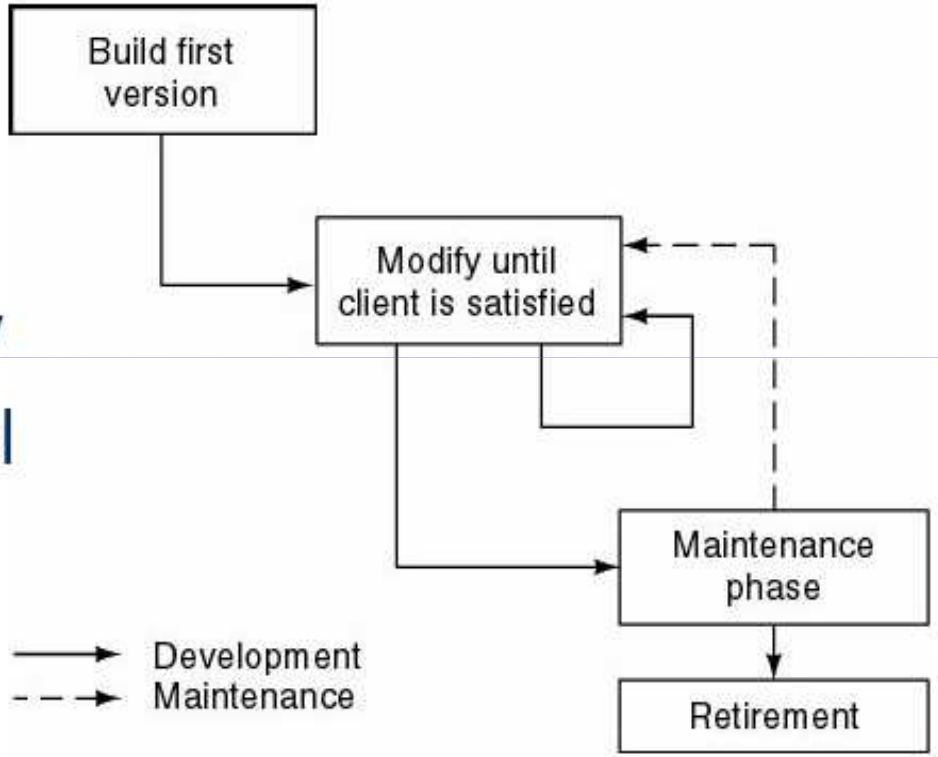
• Project Management	8.08%
• Requirements	14.43%
• Design	11.36%
• Coding	13.50%
• SQA	30.64%
• SCM	13.02%
• Integration	6.54%
• Misc.	~3%

Software Processes

- Build-and-fix model
- Waterfall model
- Rapid prototyping model
- Incremental model
- Extreme programming
- Synchronize-and-stabilize model
- Spiral model
- Object-oriented life-cycle models
- Comparison of life-cycle models

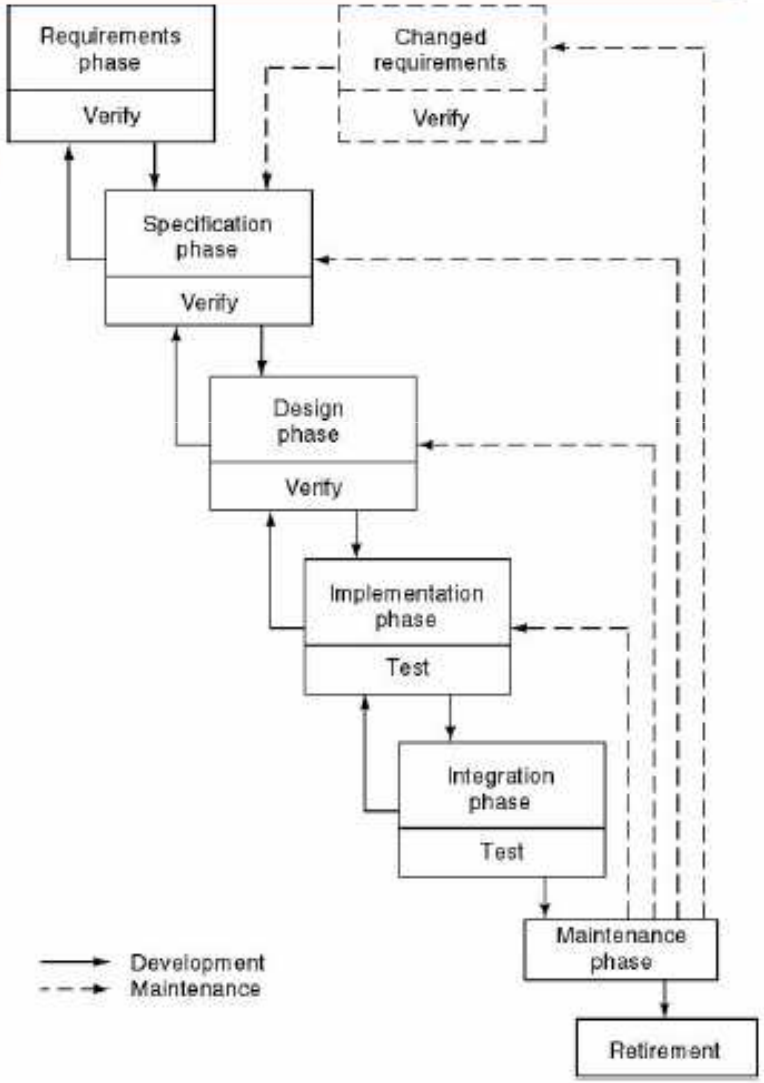
Build and Fix Model

- Problems
  - No specifications
  - No design
- Totally unsatisfactory
- Need life-cycle model
  - "Game plan"
  - Phases
  - Milestones



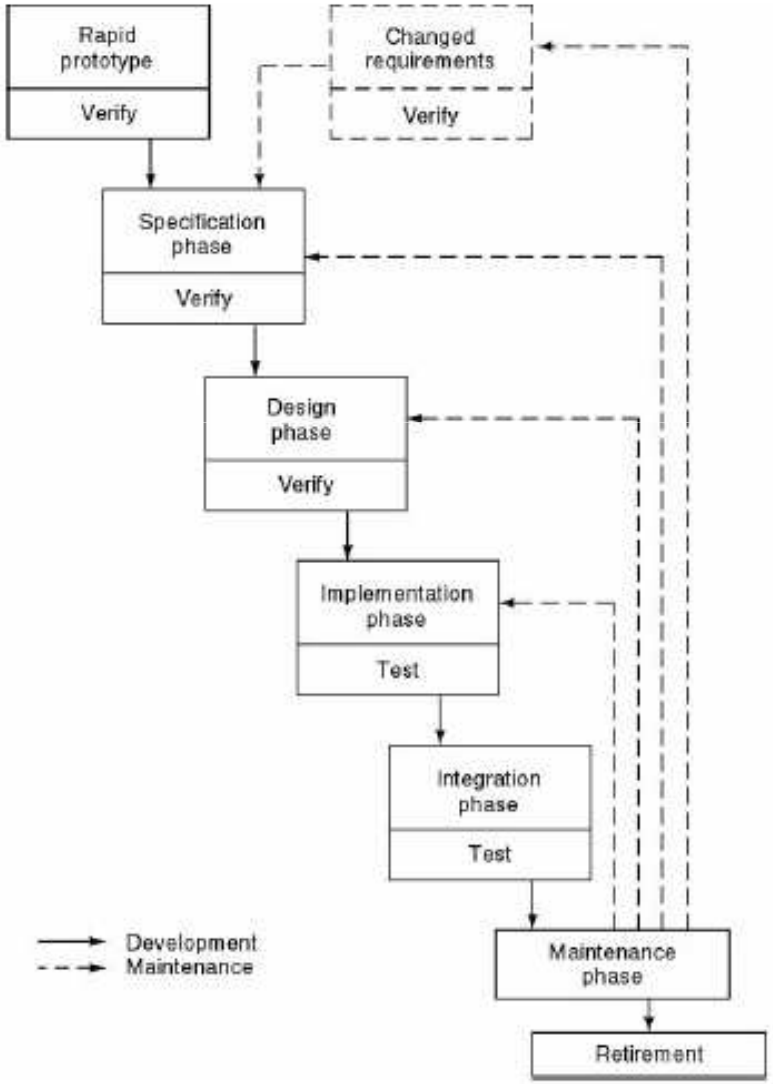
Waterfall Model (contd)

- Characterized by
  - With or without feedback loops
  - Documentation-driven
- Advantages
  - Documentation
  - Maintenance easier
- Disadvantages
  - Client feedback



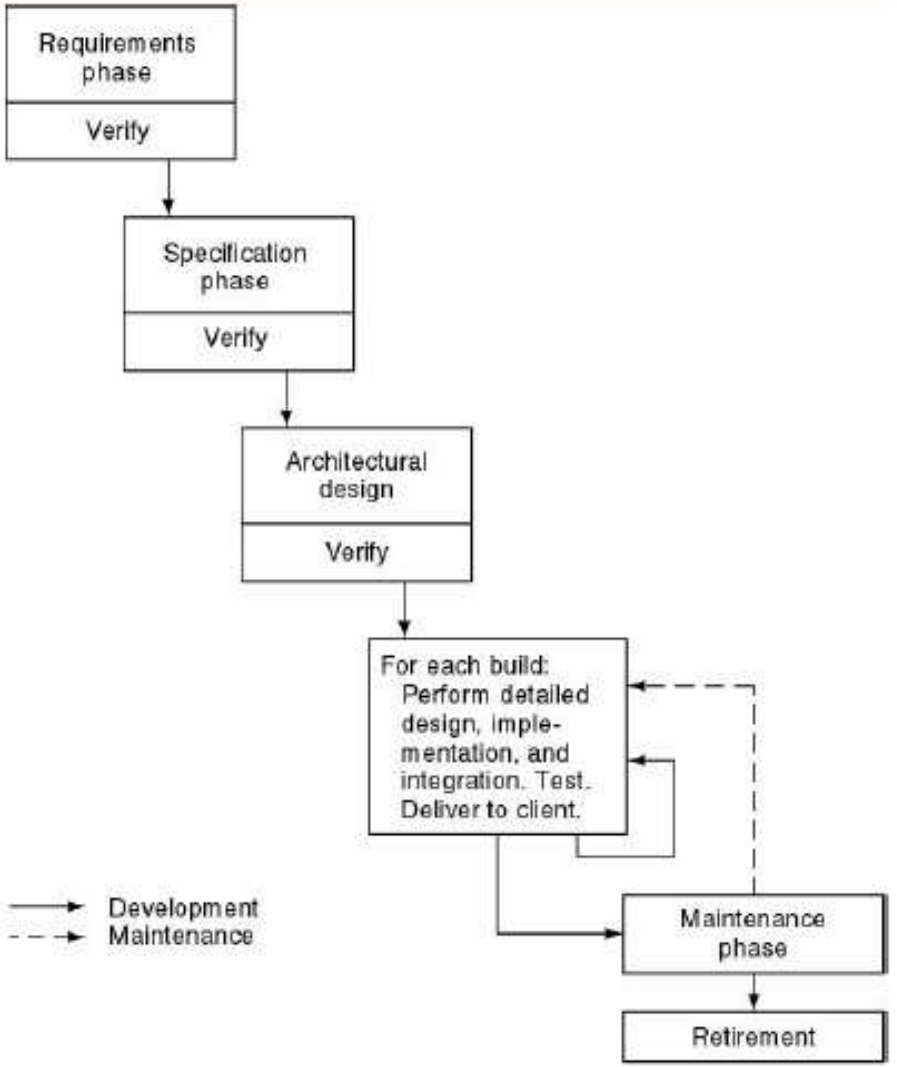
Waterfall and Rapid Prototyping Models

- Waterfall model
  - Many successes
  - Client needs
- Rapid prototyping model
  - Cannot be used for robust applications
- Solution
  - Rapid prototyping for requirements phase
  - Waterfall for rest of life cycle



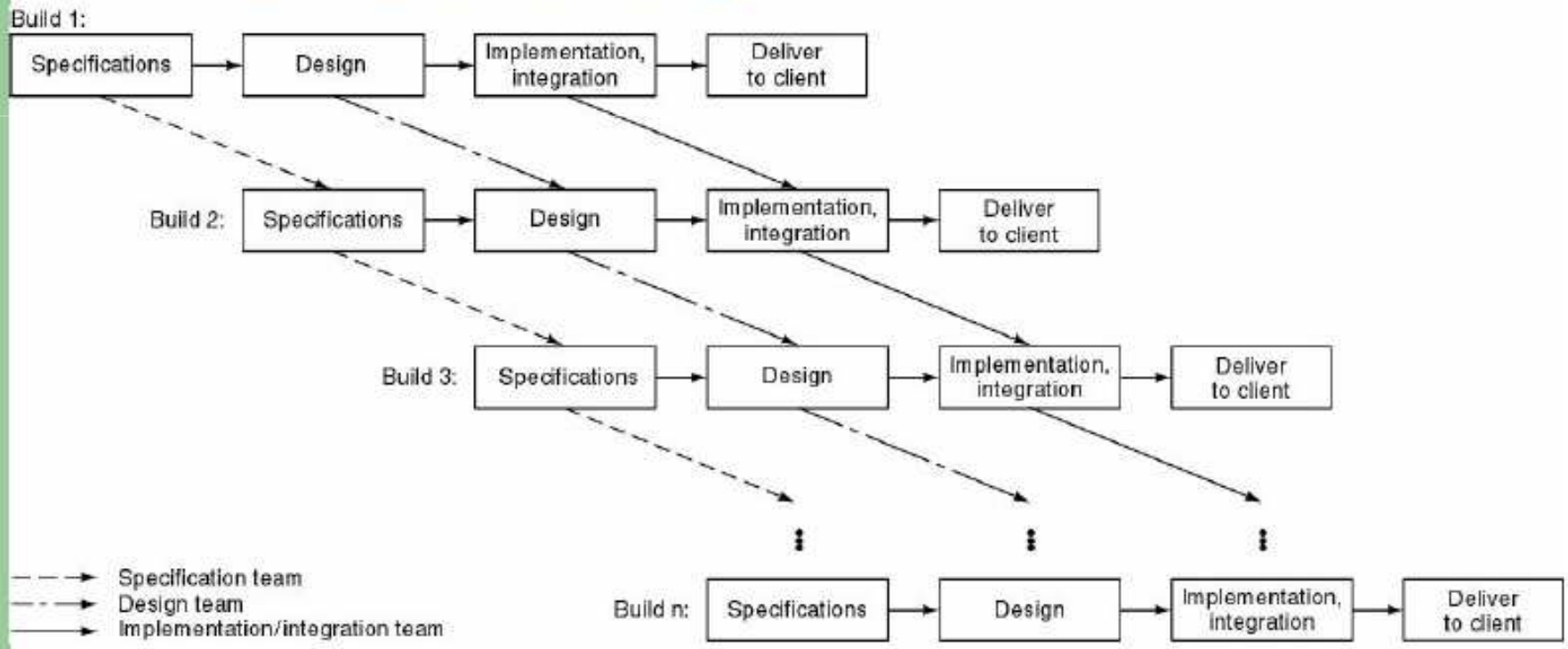
Incremental Model

- Divide project into *builds*



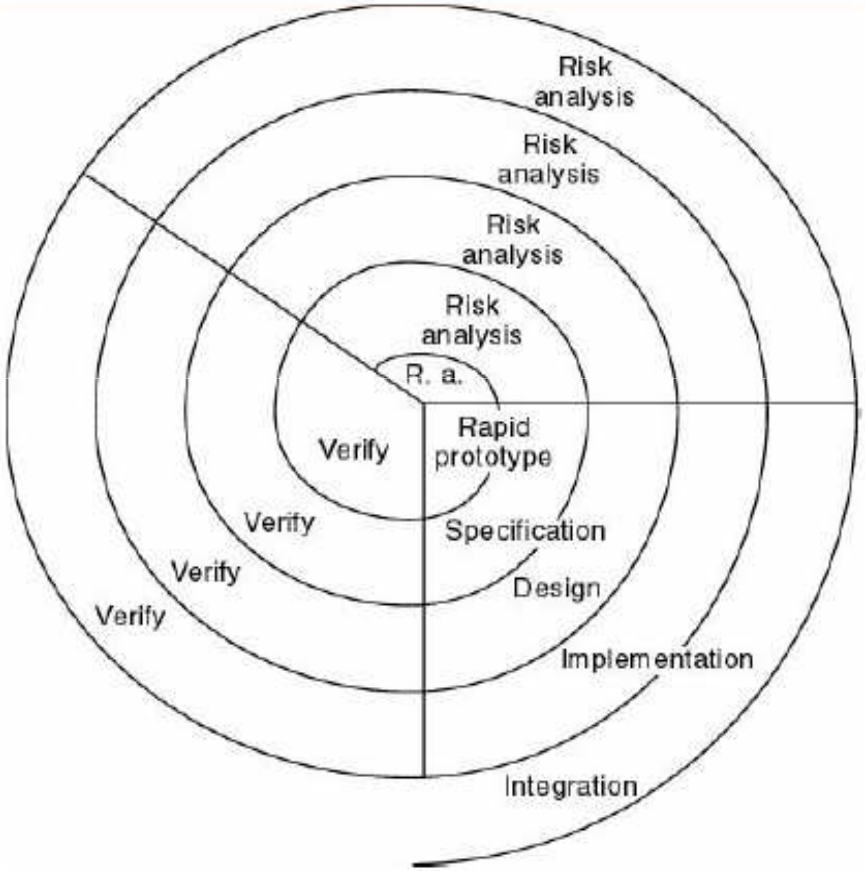
Incremental Model (contd)

- More risky version—pieces may not fit
  - CABTAB and its dangers

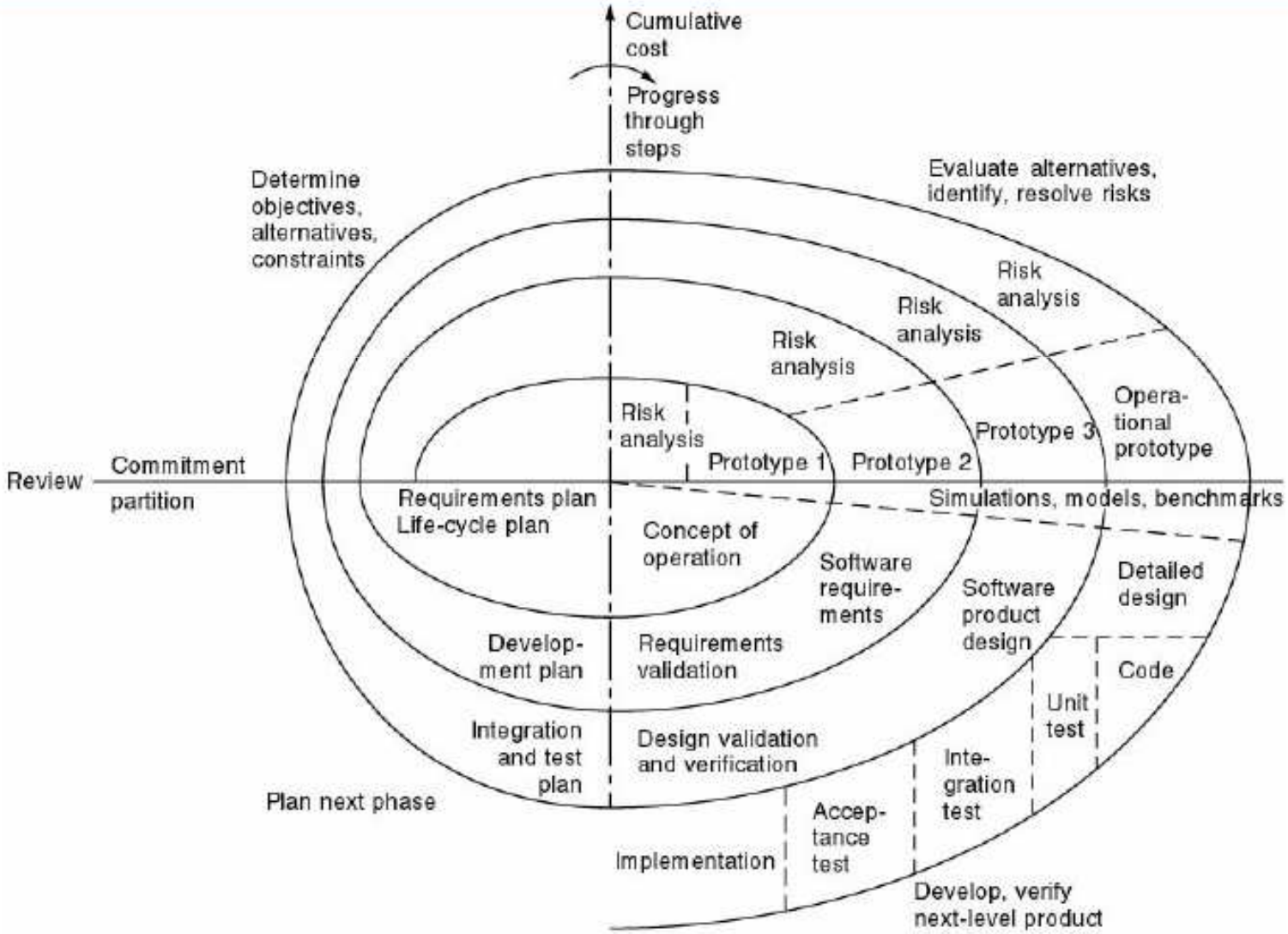


Simplified Spiral Model

- If risks cannot be resolved, project is immediately terminated



Full Spiral Model (contd)



Object-Oriented Life-Cycle Models

- Need for iteration within and between phases
  - Fountain model
  - Recursive/parallel life cycle
  - Round-trip gestalt
  - Unified software development process
- All incorporate some form of
  - Iteration
  - Parallelism
  - Incremental development
- Danger
  - CABTAB