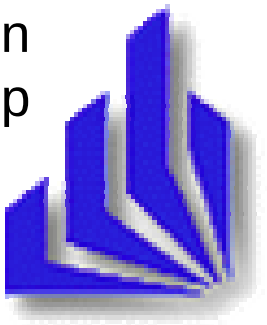
University of Paderborn
Software Engineering Group



Software Engineering for Software-Intensive Systems: III The Development Life Cycle

Assistant Professor Dr. Holger Giese
Software Engineering Group
Room E 3.165
Tel. 60-3321
Email: hg@upb.de

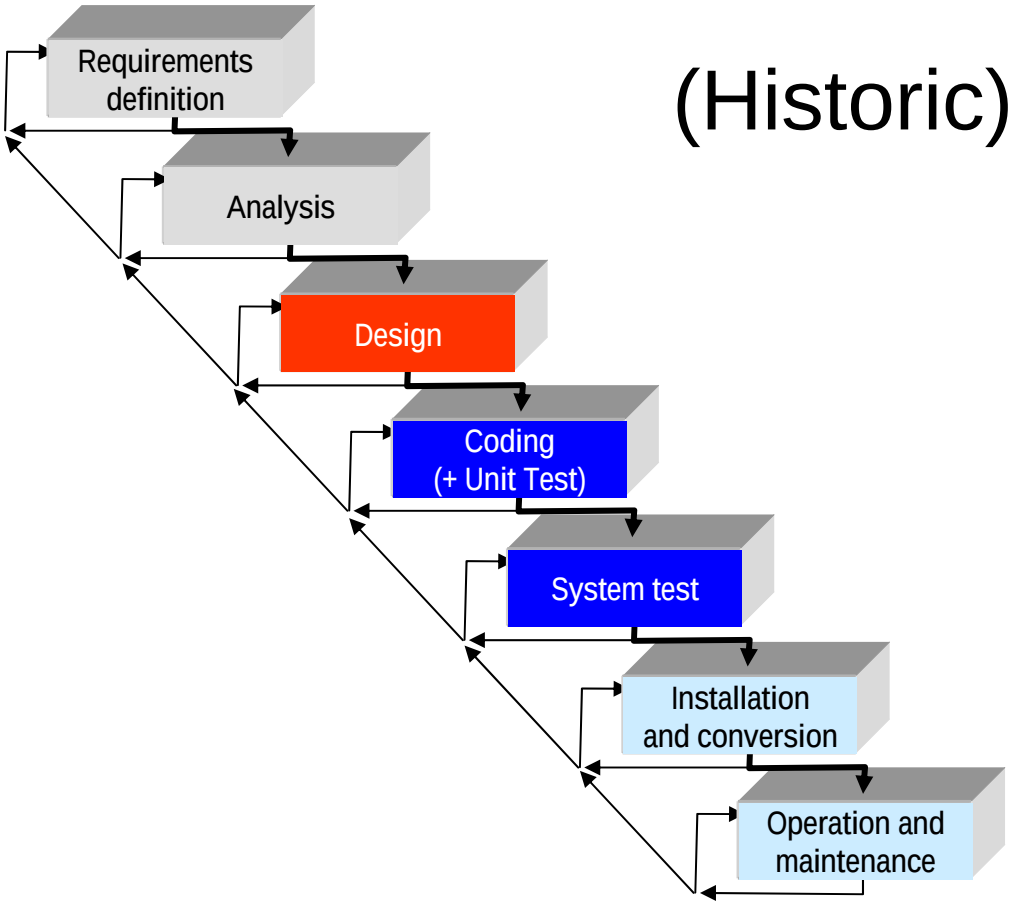
Outline

III The Development Life Cycle

III.1 Software Engineering Life Cycle Models

- **Waterfall Model**
- **Prototyping**
- **V Model**
- **Spiral Model**
- **RUP**

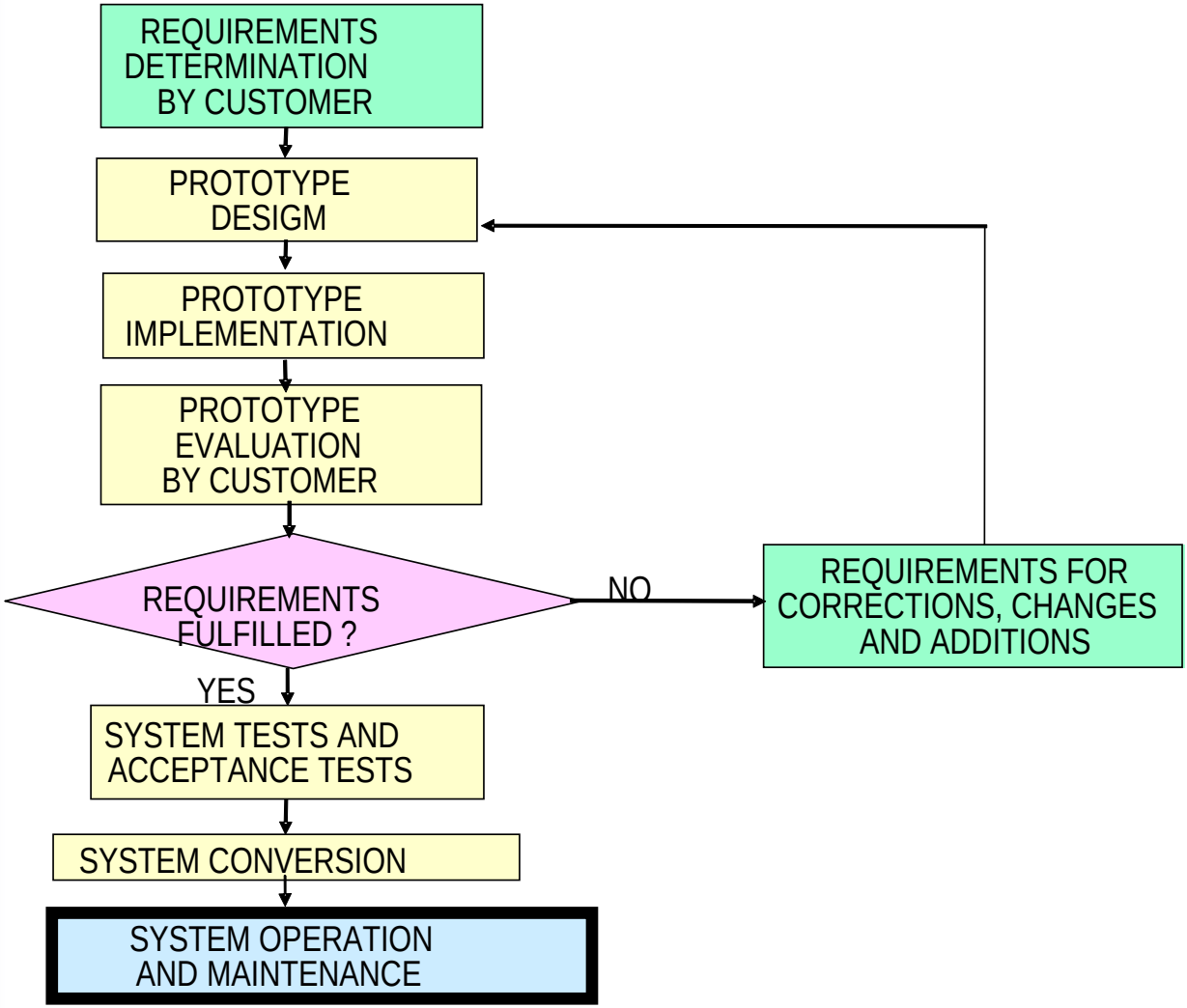
Characteristics of Software Development Methodologies



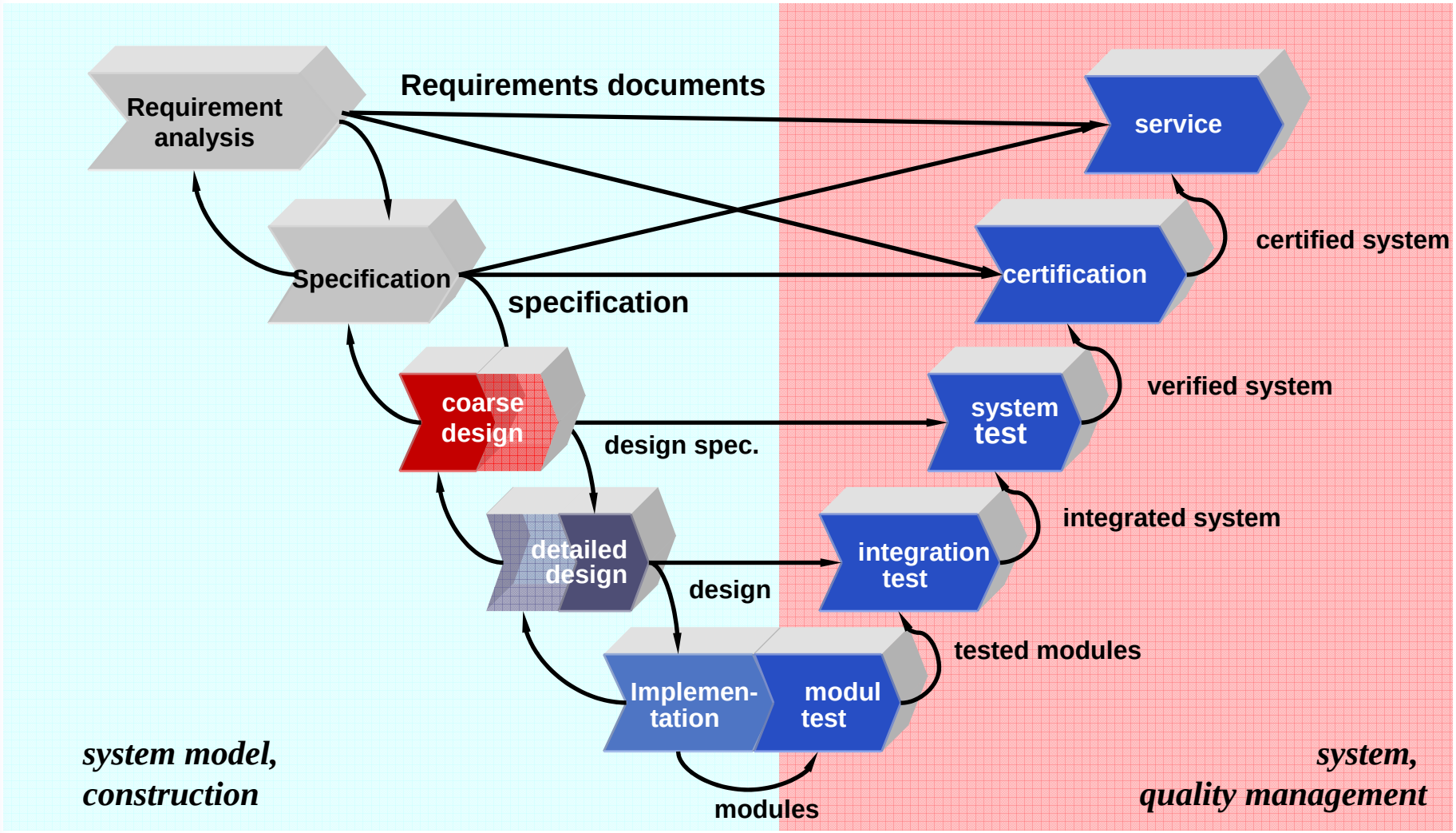
(Historic) Waterfall Model

The Prototyping Process

[Galin2004]

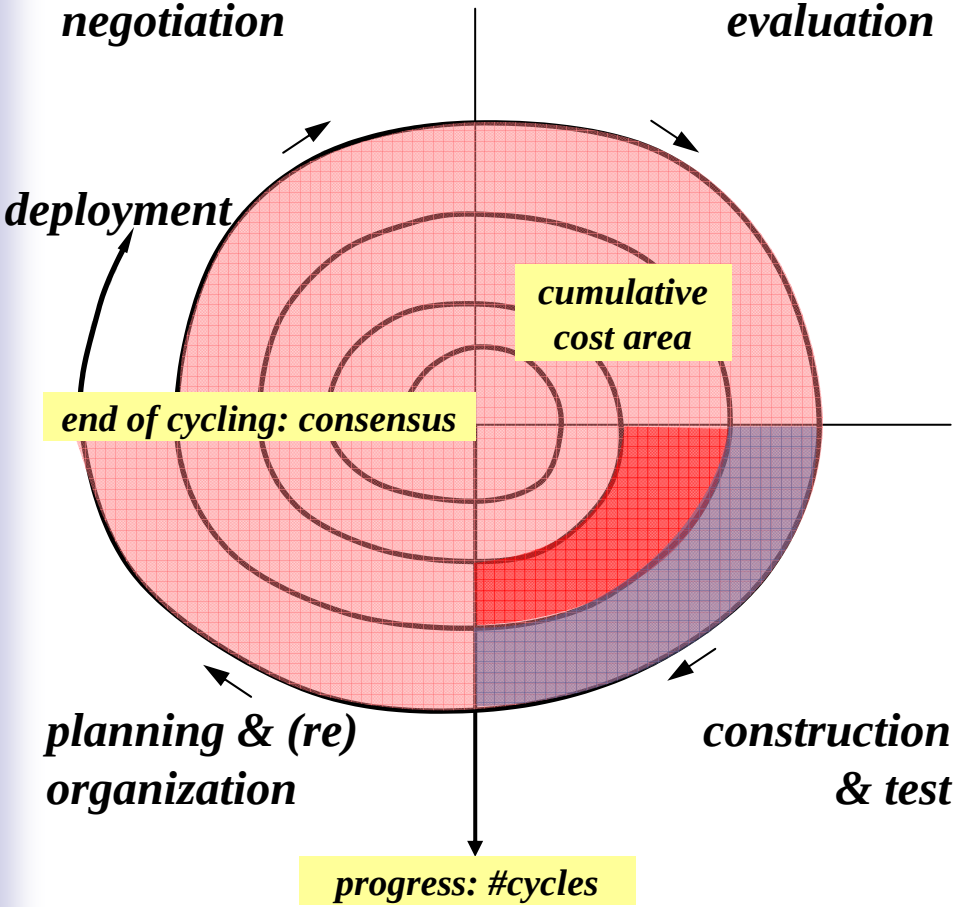


“V” Development Process



Spiral Model Process

[Boehm1988]



Negotiation

- objectives, alternatives, strategies, constraints

Evaluation

- alternatives: „Make-or-Buy“, risk analysis

Construction & Test

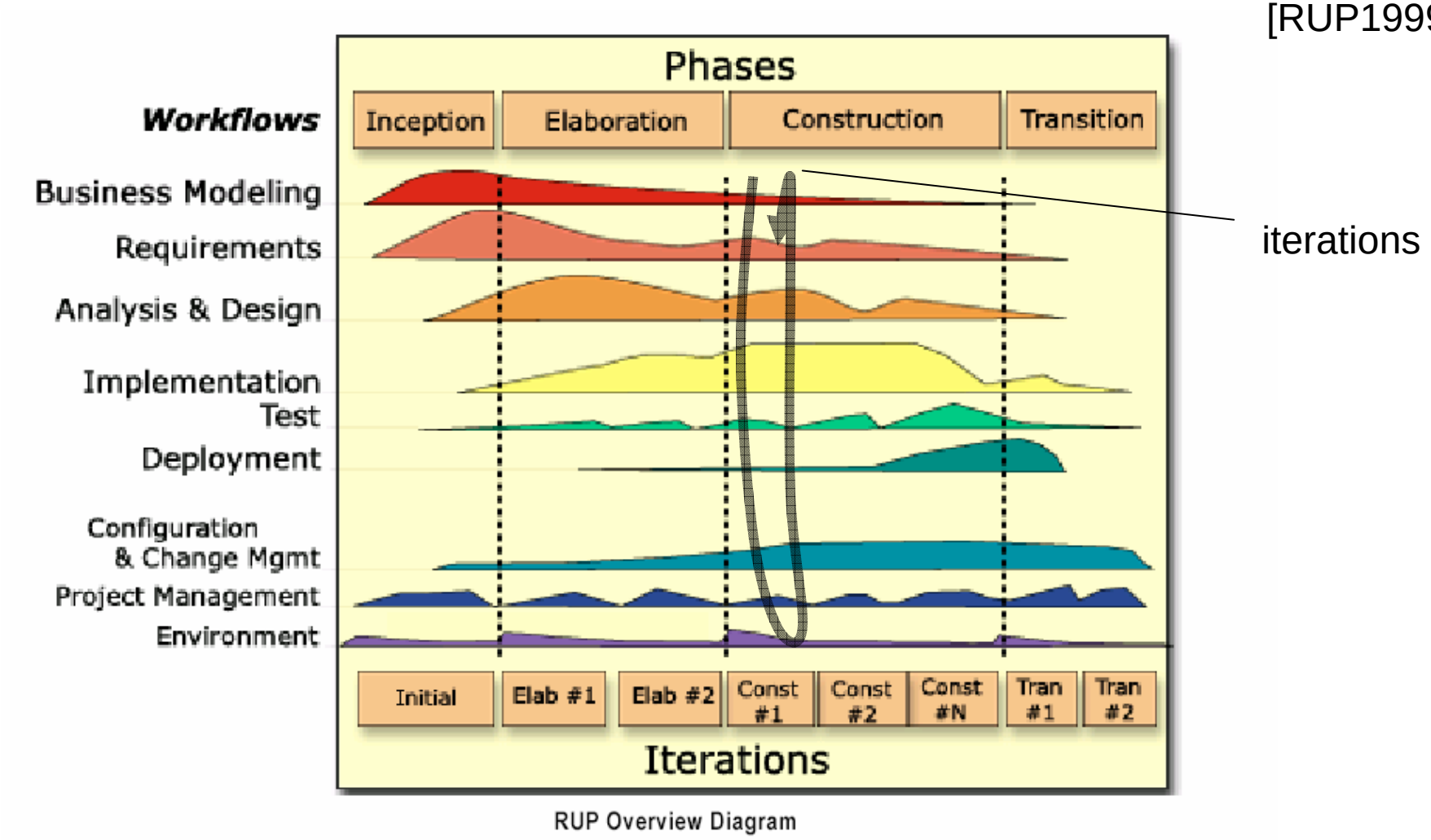
- any SE-Process for partial or full system!

Planning

- Review,
- Plan next phases

Rational Unified Process (RUP)

[RUP1999]

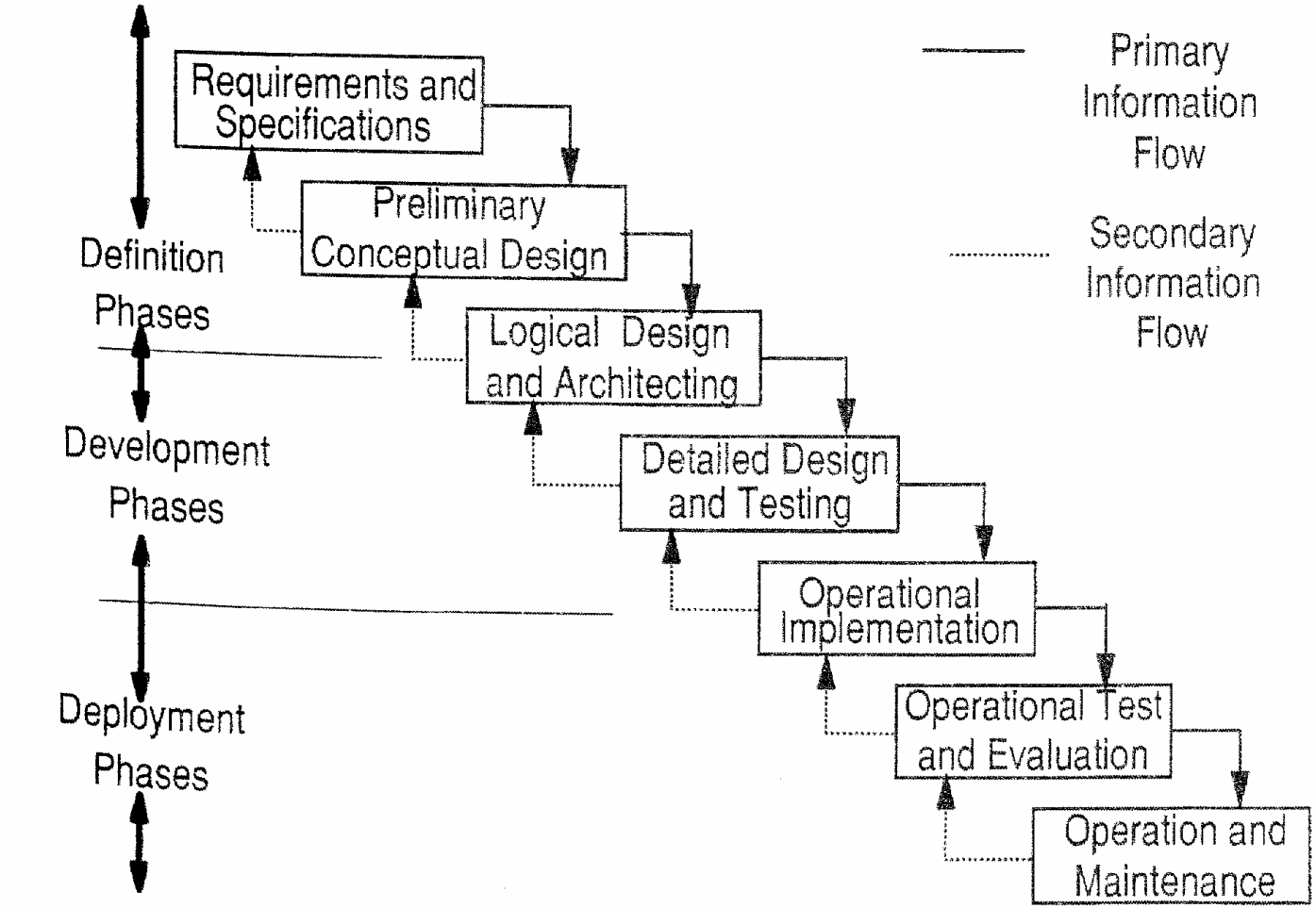


RUP Overview Diagram

III The Development Life Cycle

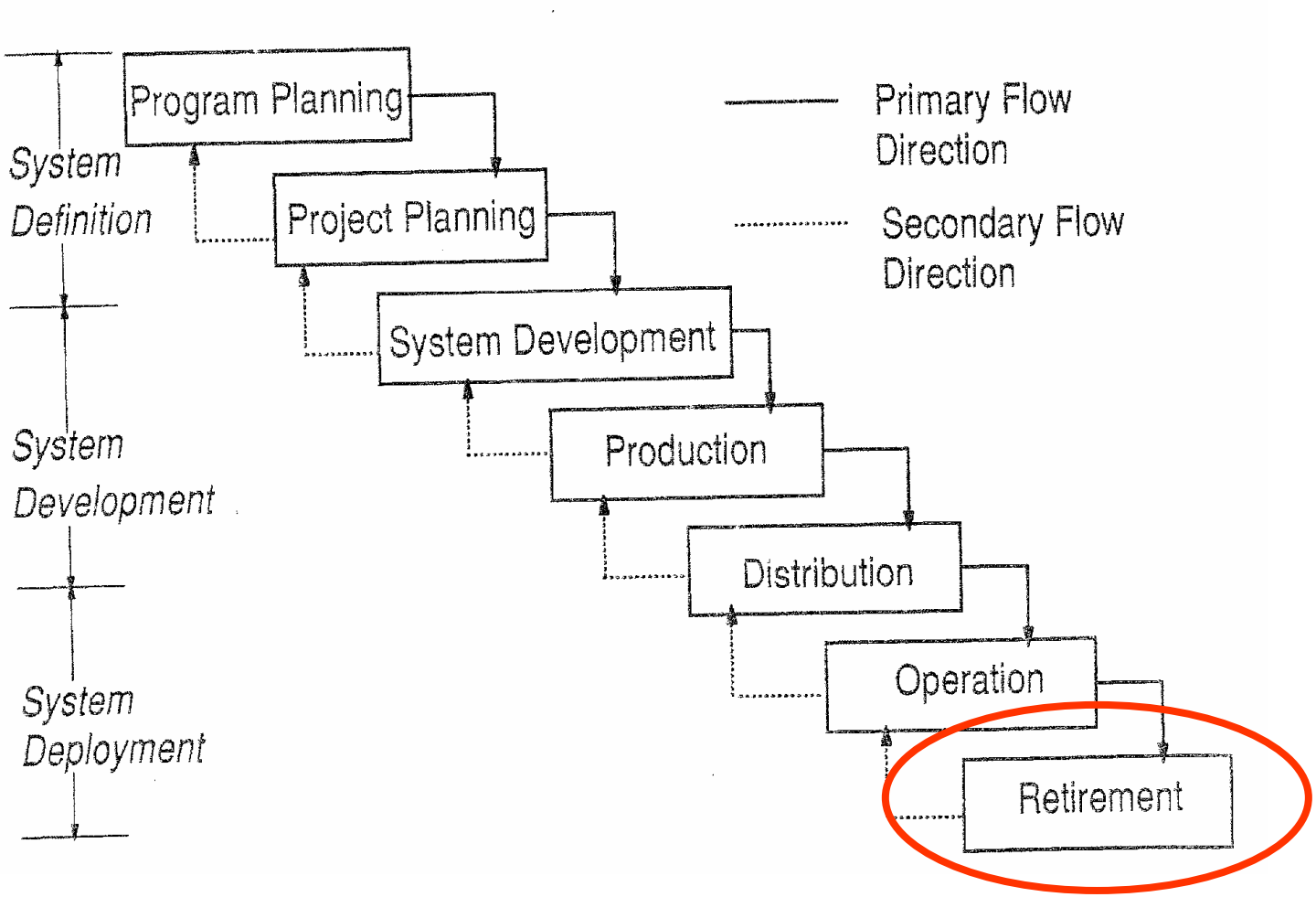
Life Cycle of System Engineering

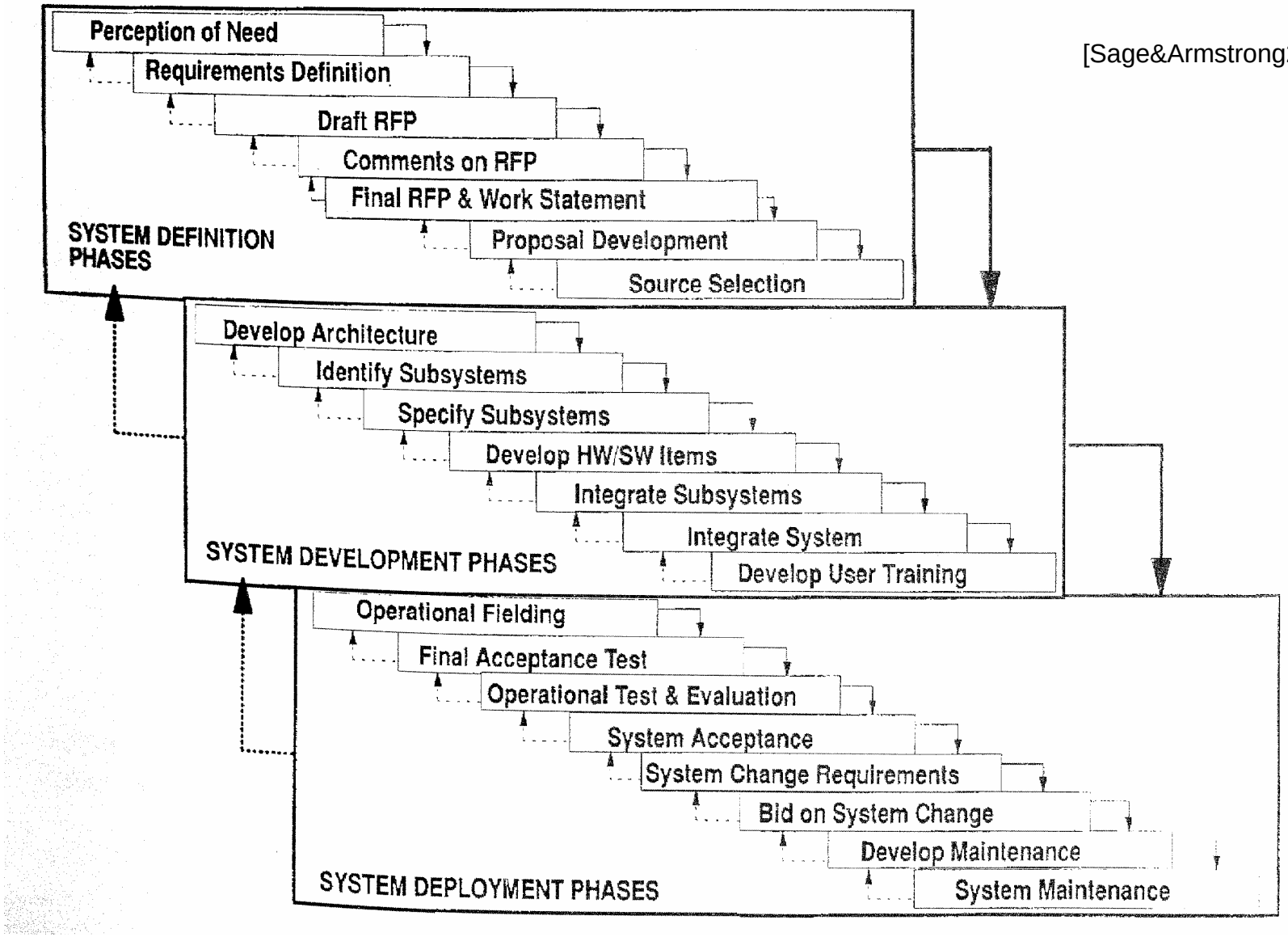
[Sage&Armstrong2000]



Alternative View

[Sage&Armstrong2000]





[Sage&Armstrong2000]

III The Development Life Cycle

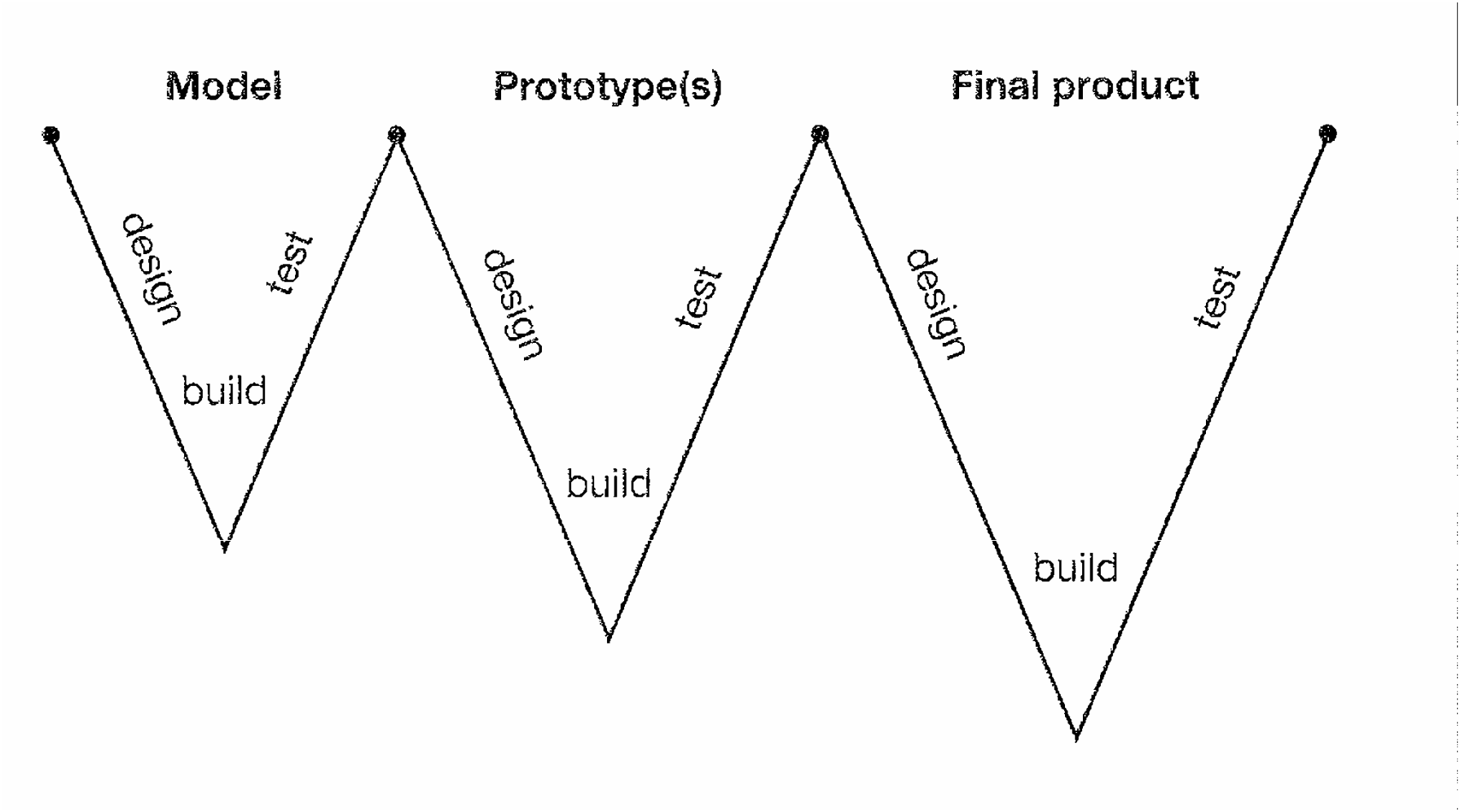
III.3 Embedded System Life Cycle Models

(1) 3V Model

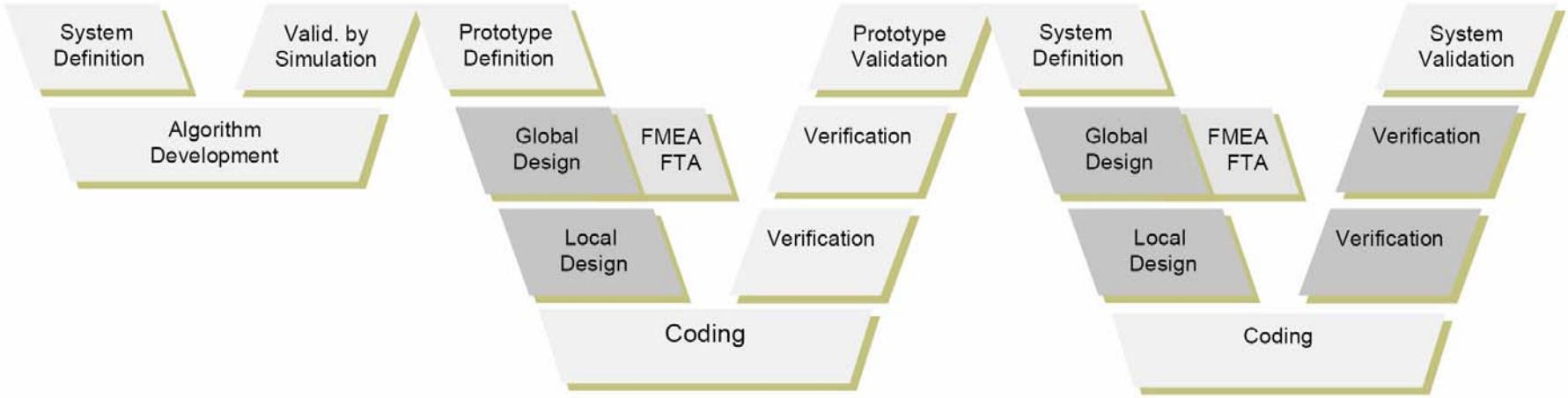
(2) Multiple V Model

(1) 3V Model (1/2)

[Broekman&Notenboom2003]



3V Model (2/2)

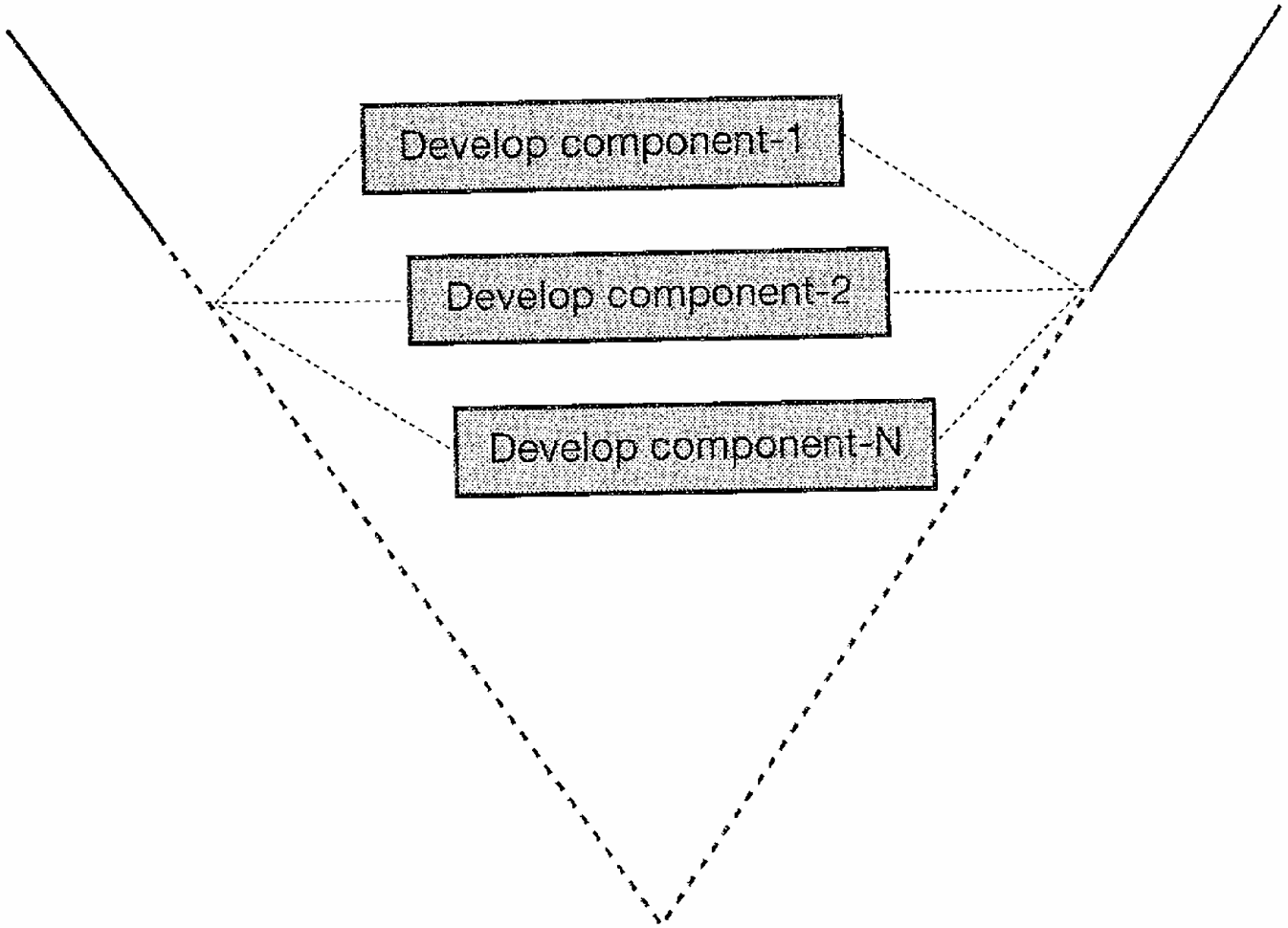


- Model: covers the definition and simulation of the overall system functionality
  - Implementation aspects are not considered
- Prototype: is characterized by rapid prototyping
  - hardware specific parameters become important
  - deployment & message scheduling
  - local design addresses the scheduling of tasks on each node
- Final product: addresses the system development for the final target hardware
  - typical problem: limited performance of the target system

http://www.vmars.tuwien.ac.at/projects/setta
http://www.vmars.tuwien.ac.at/projects/setta/docs/meetings/020121p/final_document.pdf

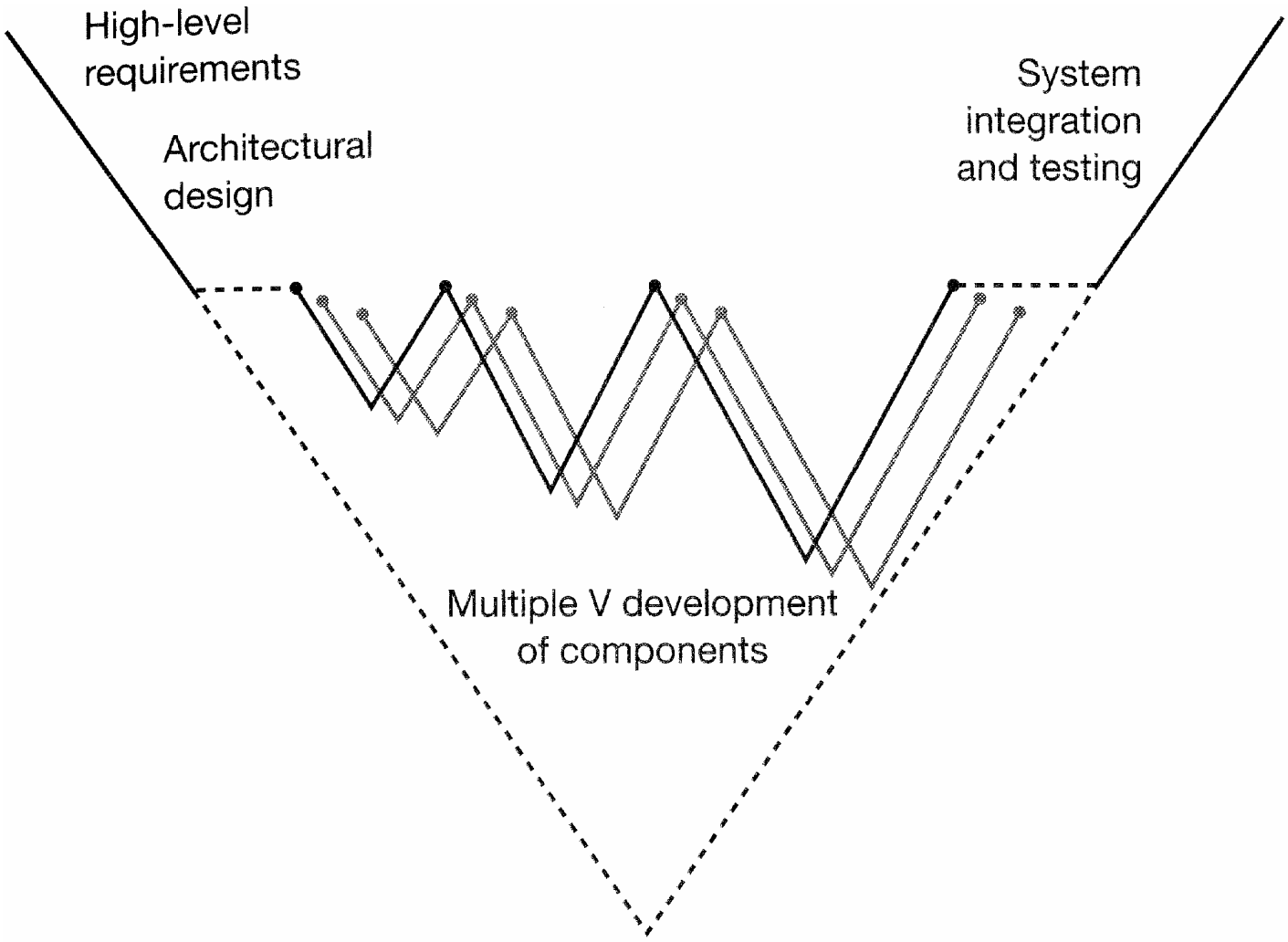
(2) Multiple V Model

[Broekman&Notenboom2003]



Multiple V Model (2/2)

[Broekman&Notenboom2003]



III The Development Life Cycle

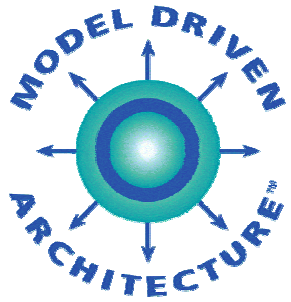
III.4 Advanced Life Cycle Models & MDD

(1) MDA

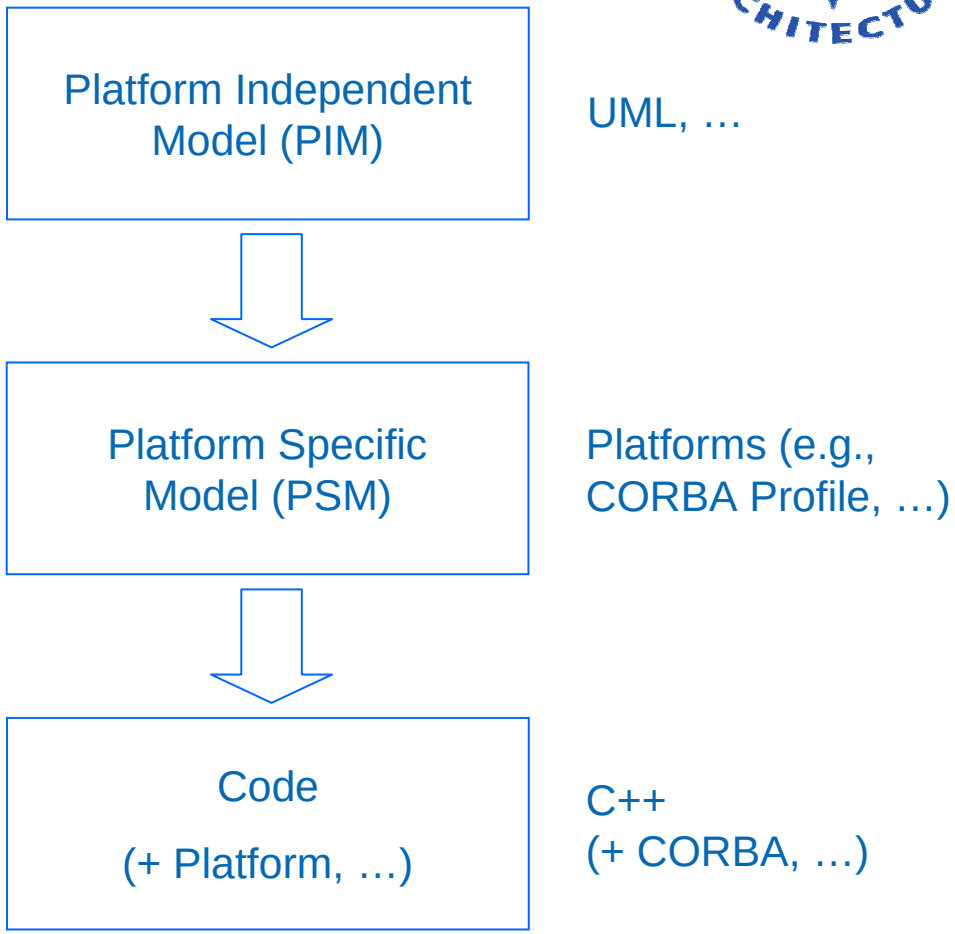
(2) Y-Model

(3) Platform-Based Design

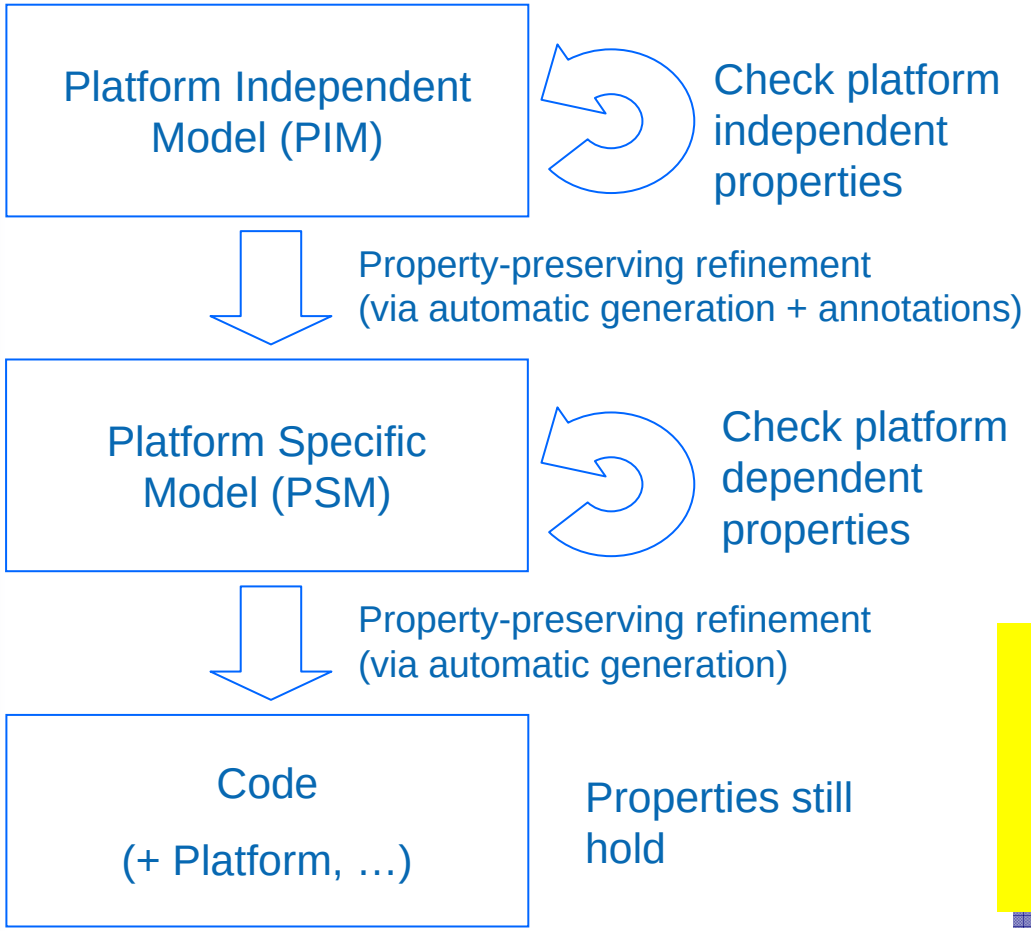
(1) MDA



- An approach to IT system specification that separates the **specification of system functionality** from the specification of the implementation of that functionality on a particular technology platform
- "Design **once**, build it on **any** platform"



Early Problem Detection in MDA



- Models permit to detect some **problems** early on:
  - Reduced defect detection costs
  - Reduced costs for defect removal
- **Traceability** and **portability**

But this is a vision only for software-intensive systems!

(2) Y-Model

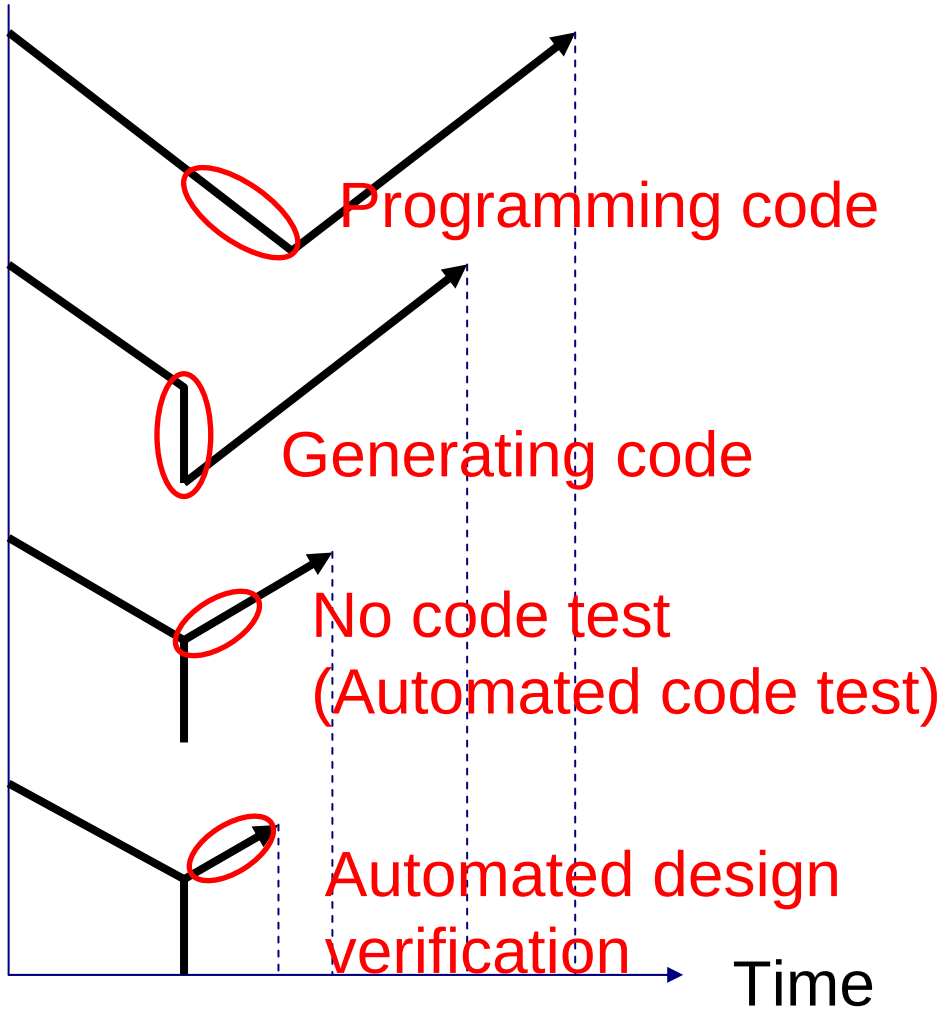
[Camus&Dion2003]
http://www.safeair.org/
http://www.safeair.org/description/D4-2_final_Report_v1.7.pdf

Manual coding

Standard automatic code generator

Qualified code generator

Design verifier



Application Example: Airbus

Tool:

- Safety Critical Applications Development Environment (SCADE)

Application:

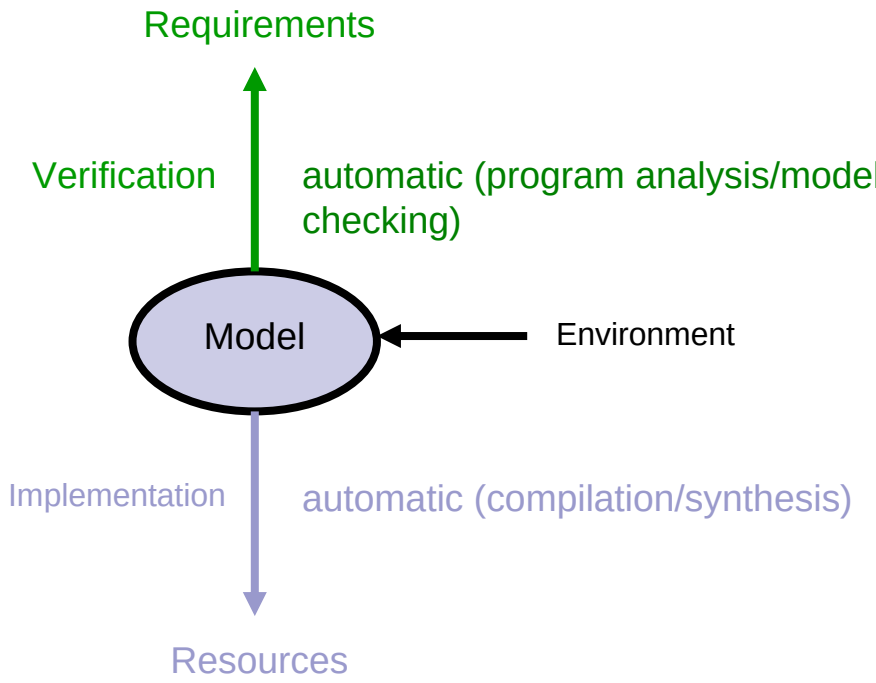
- A340/600 FCSC (Flight Control Secondary Computer):

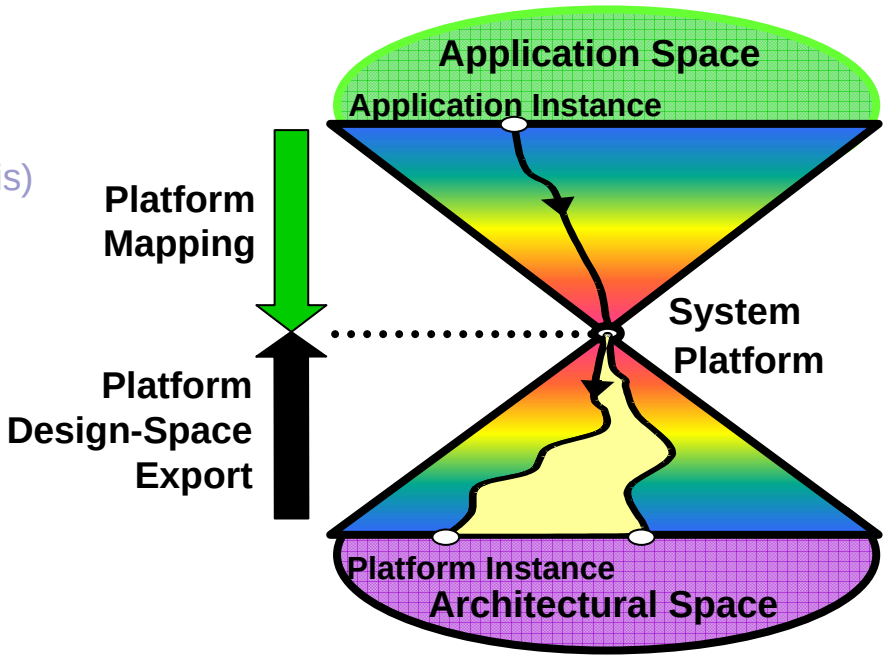
Result:

- 70 % automatically generated code
- 50 % reduction in development cost
- reduction in modification cycle time by factor 3

Source: Esterel Technologies

(3) Platform-Based Design





Idea

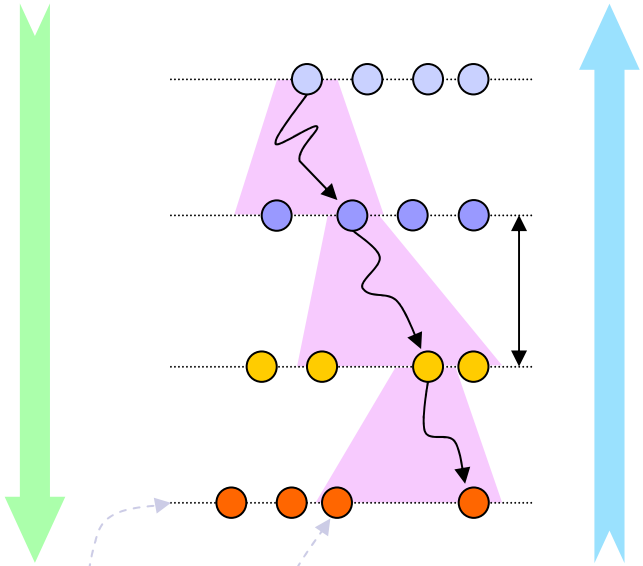
[Sangiovanni-Vincentelli2002]

- **Platform:**
  - a family of architectures satisfying a set of constraints imposed to allow the reuse of hardware and software components.
- **Platform-based design:**
  - meet-in-the-middle approach: In the top-down design flow, designers map an instance of the upper platform to an instance of the lower, and propagate design constraints.
  - exposing key resource limitations
  - hiding inessential implementation details

Platform-Based Design

Top-Down:

Map an instance of the upper platform onto an lower platform considering appropriate constrains.



Bottom-Up:

Find the appropriate platform levels.

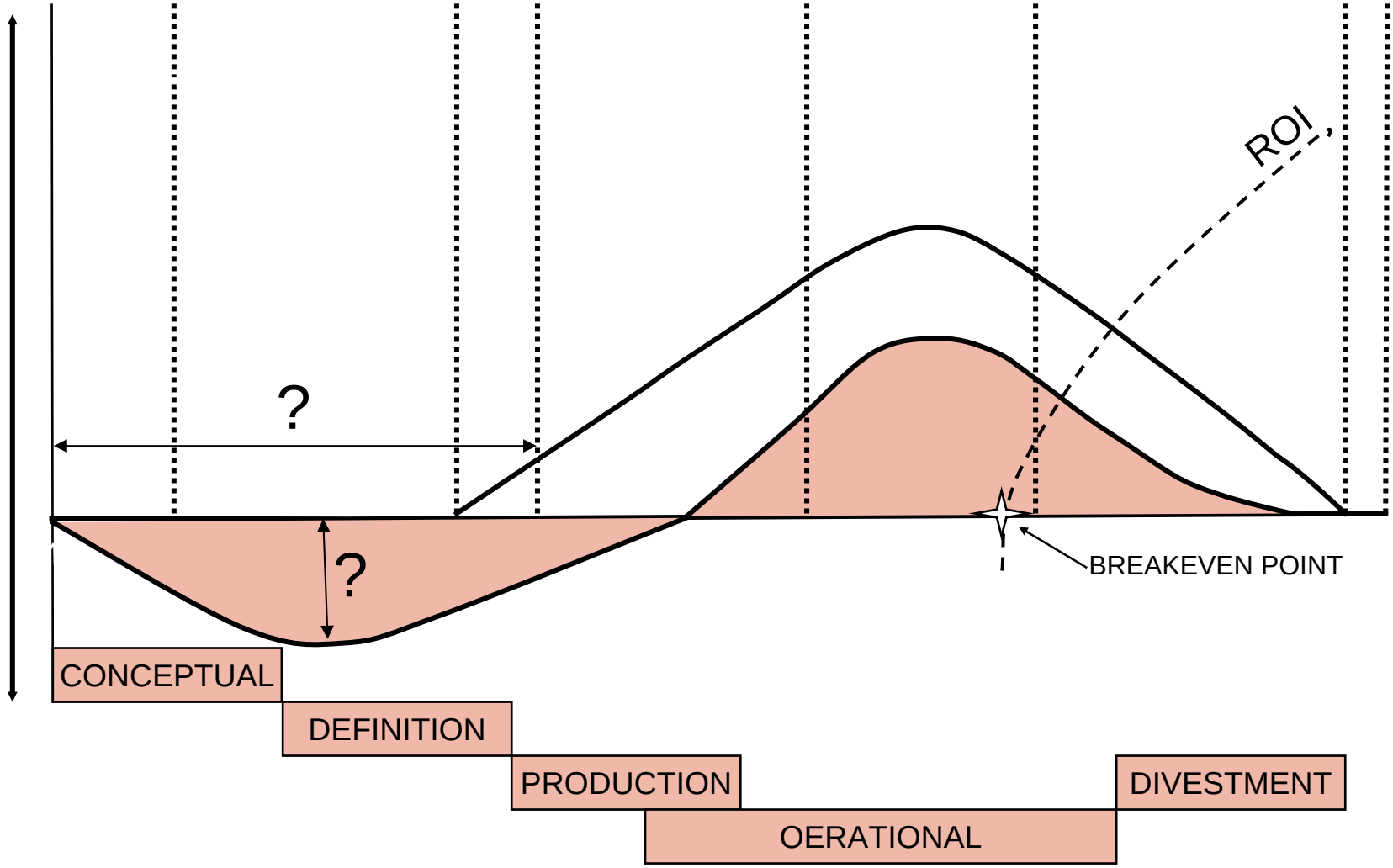
Define platform level parameters

Platform instances

Platform abstraction levels

III The Development Life Cycle

Systems Product Lifecycle



Process Management

Why?

- The quality outcome and timeliness of the system development is highly influenced by the quality of the process used to acquire, develop, and maintain it.

Common Misconceptions

- **I don't need process, I have**
  - really good people
  - advanced technology
  - an experienced manager
- **Process**
  - interferes with creativity
  - equals bureaucracy + regimentation
  - isn't needed when building prototypes
  - is only useful on large projects
  - hinders agility in fast-moving markets
  - costs too much

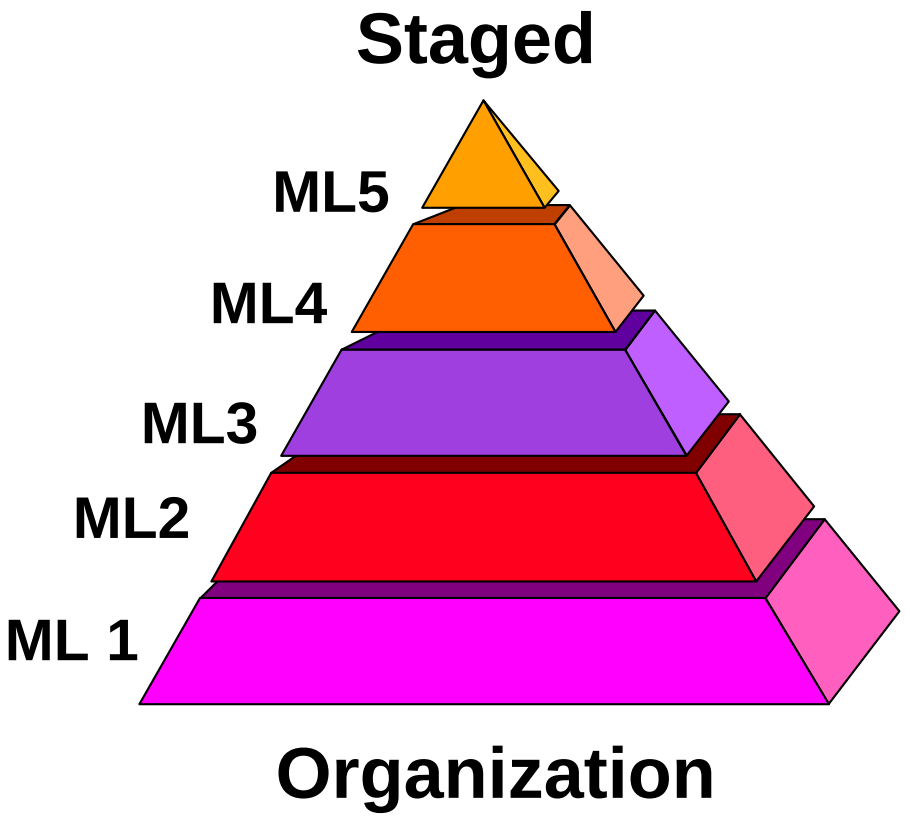
http://www.sei.cmu.edu/cmmi/general/general.html

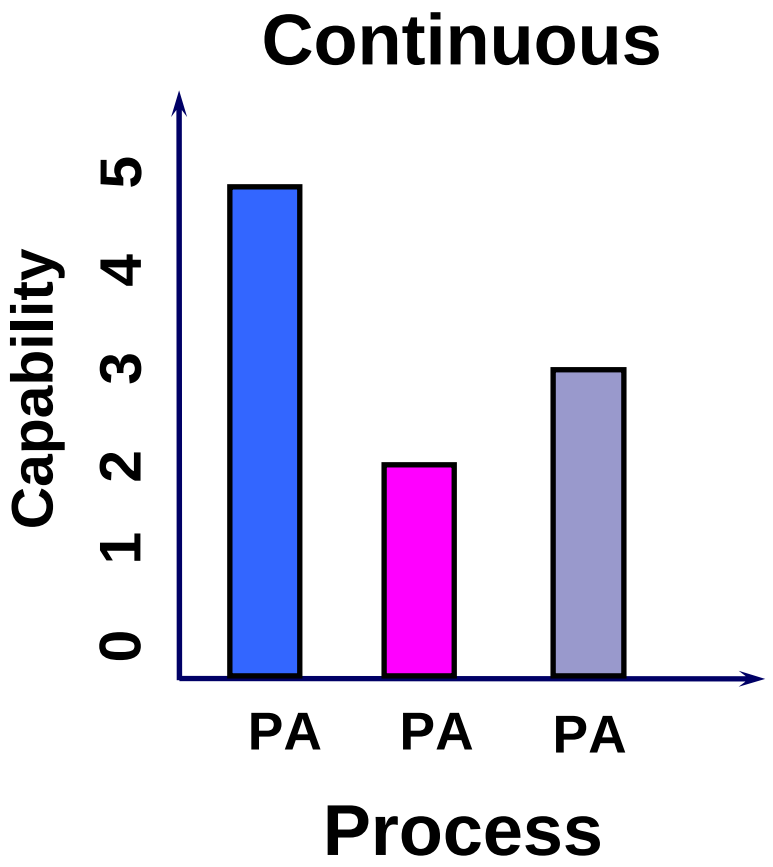
The CMMI Project

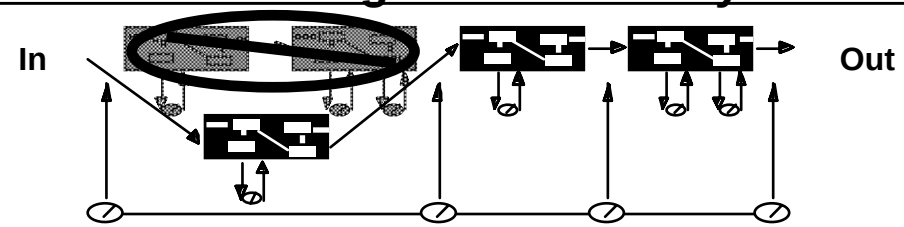
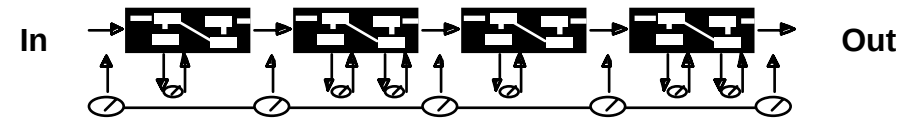
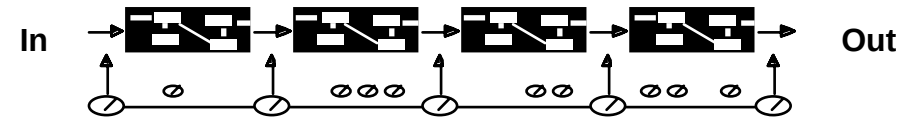
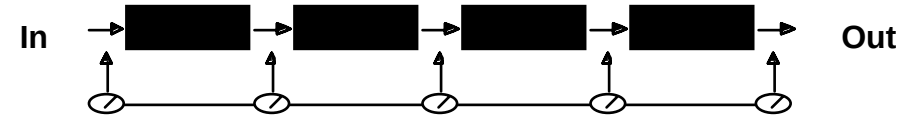
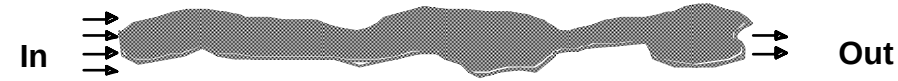
The CMM Integration Project was formed to:

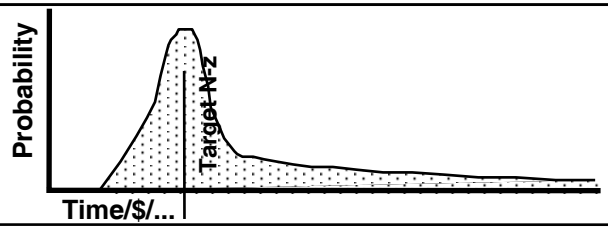
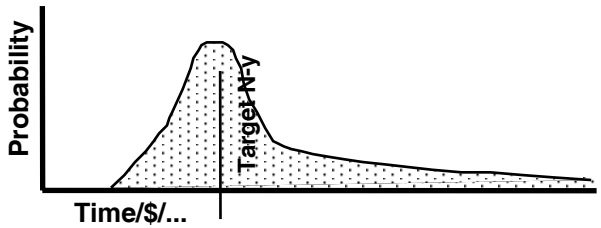
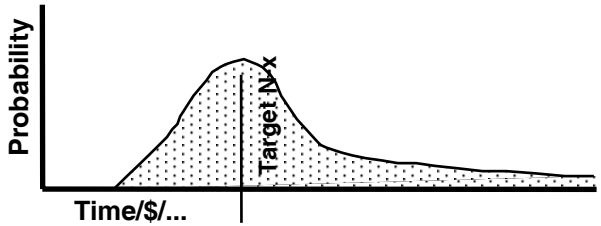
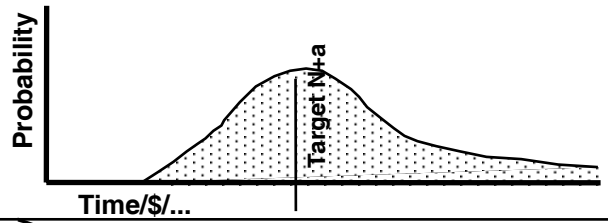
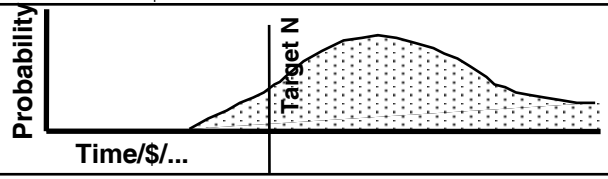
- Establish a framework to integrate current and future models
- Build an initial set of integrated models
- CMMI models that cover both systems engineering and software engineering might best be described as "engineering models." They are intended to cover the enterprise and include all the processes that result in products or services.
- The source models for the CMMI include:
  - Software: CMM for Software v2.0 Draft C,
  - Systems Engineering: EIA – 731 Systems Engineering

CMMI Model Representations





Level	Process Characteristics	Management Visibility
5 Optimizing	Focus is on continuous quantitative improvement	
4 Quantitatively Managed	Process is measured and controlled	
3 Defined	Process is characterized for the organization and is proactive	
2 Managed	Process is characterized for projects and is often reactive	
1 Initial	Process is unpredictable, poorly controlled, and reactive	

Level	Process Characteristics	Predicted Performance
5 Optimizing	Focus is on continuous quantitative improvement	
4 Quantitatively Managed	Process is measured and controlled	
3 Defined	Process is characterized for the organization and is proactive	
2 Managed	Process is characterized for projects and is often reactive	
1 Initial	Process is unpredictable, poorly controlled, and reactive	

III The Development Life Cycle

III.6 Discussion & Summary

- We have nearly **the same life cycle models** in the different disciplines.
- Advanced life cycle models and model-driven approaches try to increase the degree of **automation** and decrease **time-to-market**.
- Especially for organizations which develop large-scale software-intensive systems **process improvement** is crucial.

III.7 Bibliography (Additional ones)

[Boehm1988] Barry W. Boehm. A Spiral Model of Software Development and Enhancement. IEEE Computer, 21(5):61 72, 1988.

[Camus&Dion2003] Jean-Luis Camus and Bernard Dion. Efficient development of airborne software with scade suite. 2003.

[Galin2004] D. Galin, Software Quality Assurance: From theory to implementation. Harlow, England: Pearson Addison Wesley, 2004.

[RUP1999] Ivar Jacobson, Grady Booch, and James Rumbaugh. The Unified Software Development Process. The Addison-Wesley Object Technology Series. Addison-Wesley, January 1999.

[Sage&Armstrong2000] Andrew P. Sage and James E. Armstrong. Introduction to Systems Engineering. Wiley Series in Systems Engineering and Management. Wiley-Interscience, March 2000.