



UNIVERSITY OF SCIENCE
HO CHI MINH CITY

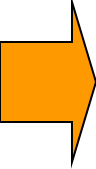
Software Processes

Traditional Methods

Nguyen V. Vu

with some materials adapted from Boehm 2003 and
Sommerville 2007

Outline

- 
- Introduction to Plan-driven and Agile Methods
 - Planning Spectrum
 - Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)

Information Technology Trends

Traditional Development

- Standalone systems
- Stable requirements
- Rqts. determine capabilities
- Control over evolution
- Enough time to keep stable
- Stable jobs
- Repeatability-oriented process, maturity models

Current/Future Trends

- Everything connected (maybe)
- Rapid requirements change
- COTS capabilities determine rqts.
- No control over COTS evolution
- Rapid development
- Outsourced jobs
- Adaptive process models

New Software Development Approaches

- Current and future trends lead to changes and new software development approaches
 - New processes
 - New tools
 - New focus
 - New methods
 - New practices
 - New models
 - ...

Software Development Approaches

- Two approaches to software development
 - Plan-driven (SW-CMM, document-based, heavy process)
 - Agile (XP, tacit knowledge, light process)
- Both have strengths and weaknesses



Key Definitions

- Agile method
 - one which fully adopts the four value propositions and twelve principles in the Agile Manifesto
- Plan – (per Webster)
 - a method for achieving an end (a process plan);
 - an orderly arrangement of parts of an overall design (a product plan)
- Plan-driven
 - a description for disciplined methods (order is often defined in plans)

The Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Popular Plan-driven and Agile Methods

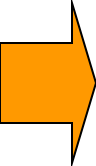
■ Plan-driven methods

- ❑ **Waterfall**
- ❑ **Spiral**
- ❑ **Rational Unified Process**
- ❑ Personal Software Process
- ❑ **CMMI**

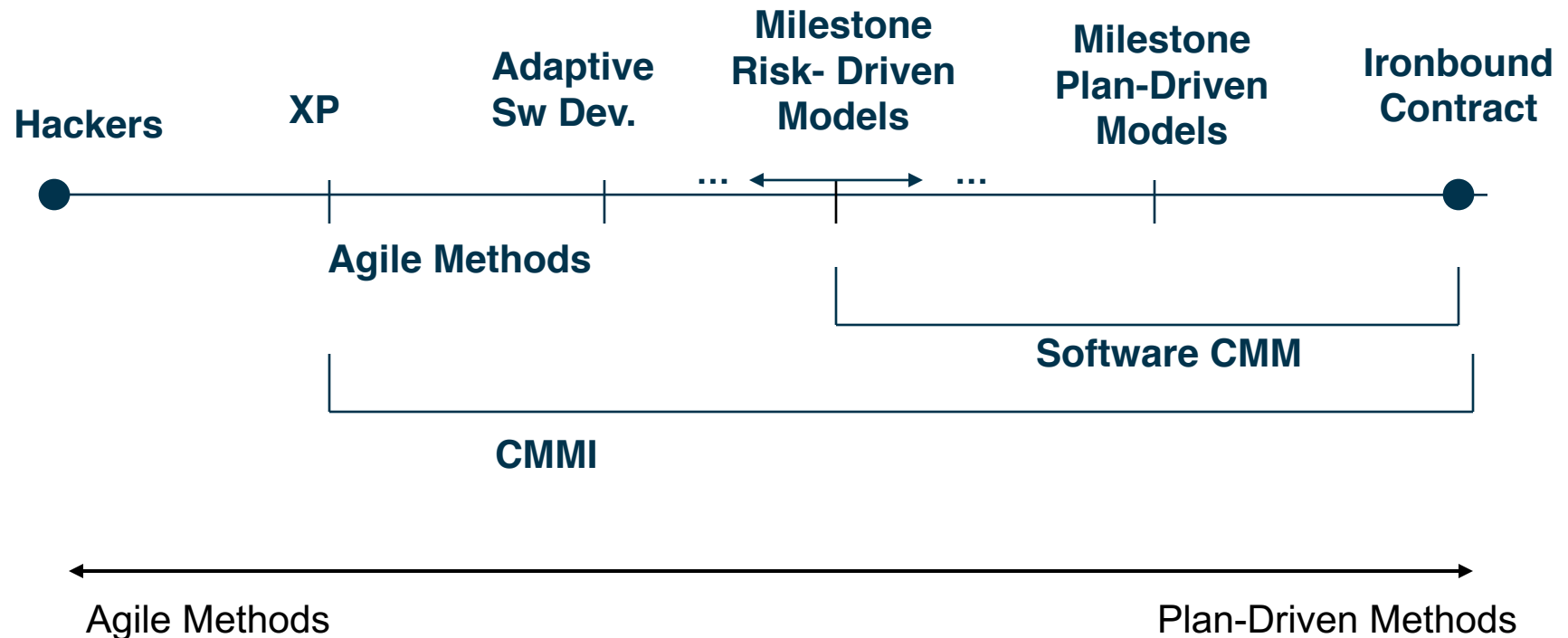
■ Agile methods

- ❑ Extreme Programming
- ❑ Scrum
- ❑ Crystal
- ❑ Feature Driven Development (FDD)

Outline

- 
- Introduction to Plan-driven and Agile Methods
 - Planning Spectrum
 - Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)

The Planning Spectrum



(Boehm 2002)

Outline

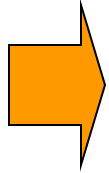
- Introduction to Plan-driven and Agile Methods
- Planning Spectrum
- Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)
 - Strengths and Weaknesses

Plan-Driven Methods

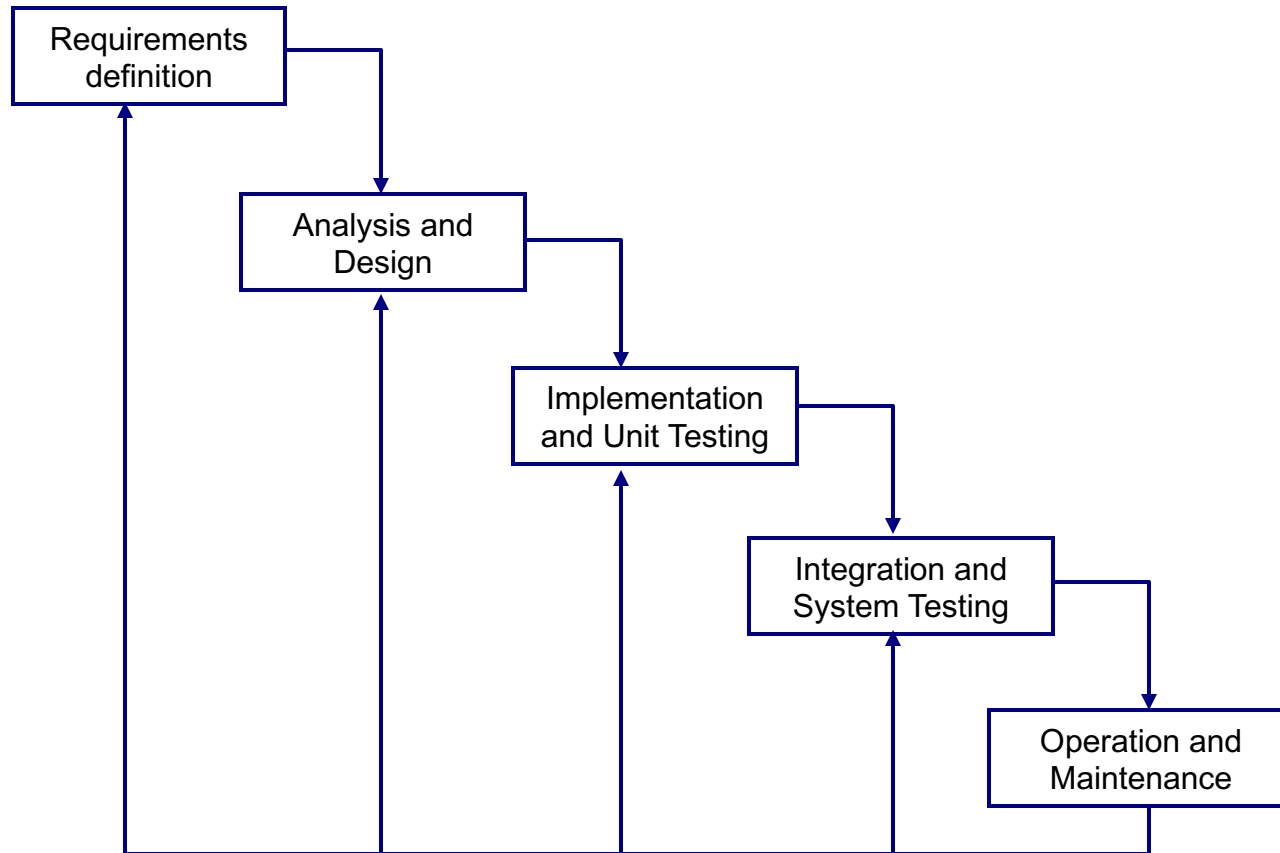
- Structured approaches to software development that are guided or driven by plans
- Also called as disciplined methods
- Often referred to as traditional methods
- Often defined in plans and relied on documentation

Outline

- Introduction to Plan-driven and Agile Methods
- Planning Spectrum
- Plan-driven Methods
 - ❑ Waterfall Model
 - ❑ Spiral Model
 - ❑ Rational Unified Process
 - ❑ Capability Maturity Model Integration (CMMI)



Waterfall Model - 1

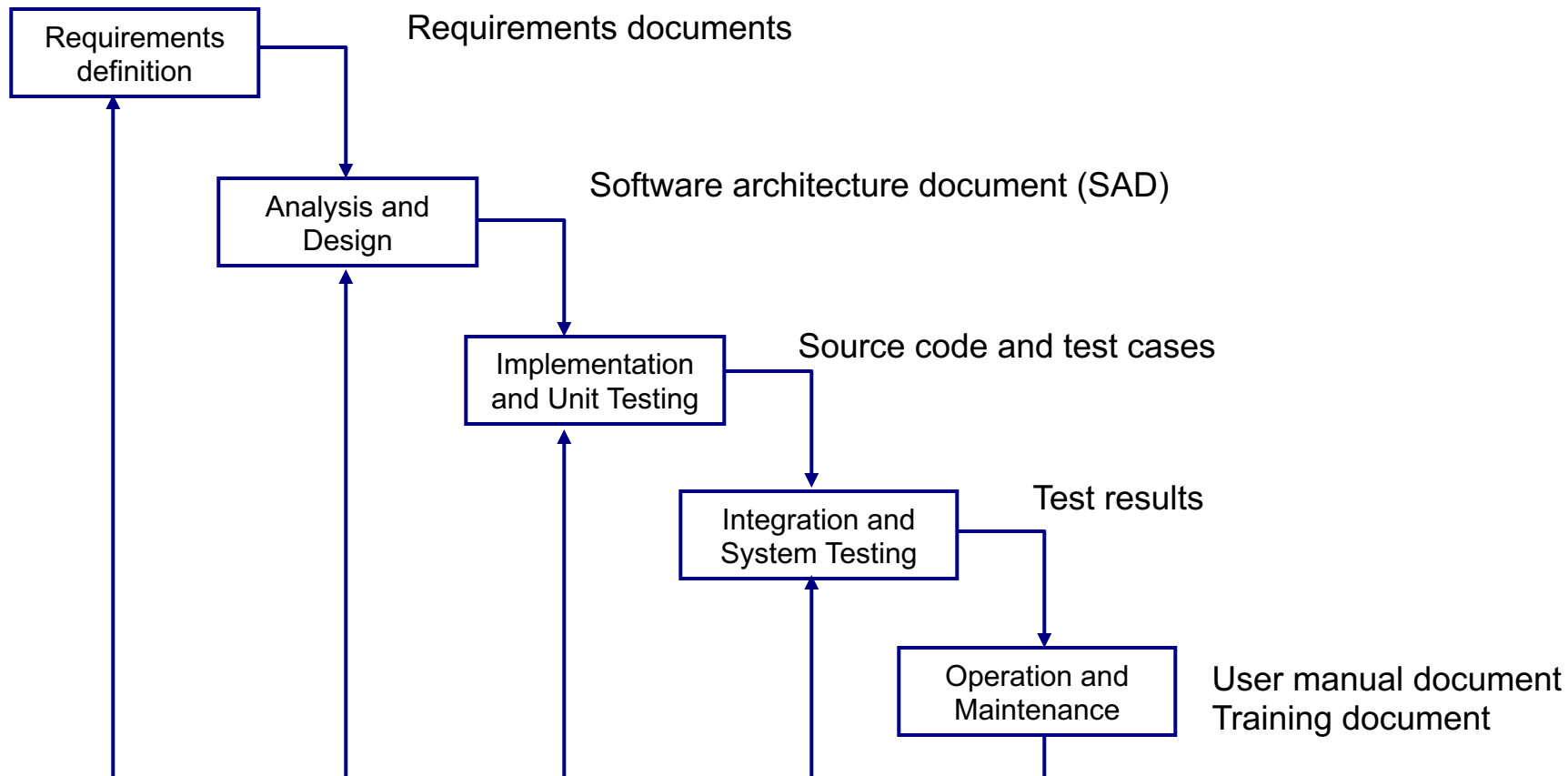


Waterfall Model - 2

- Two stages cannot be performed at the same time
 - Development is not started when requirements are not baselined
 - Baselined requirements are ready to be based on
- Each stage has to be completed before doing the next stage
- A stage is completed if a set of criteria is met
 - Using checklists of criteria check
- Going back to the previous stage is possible

Waterfall Model - 3

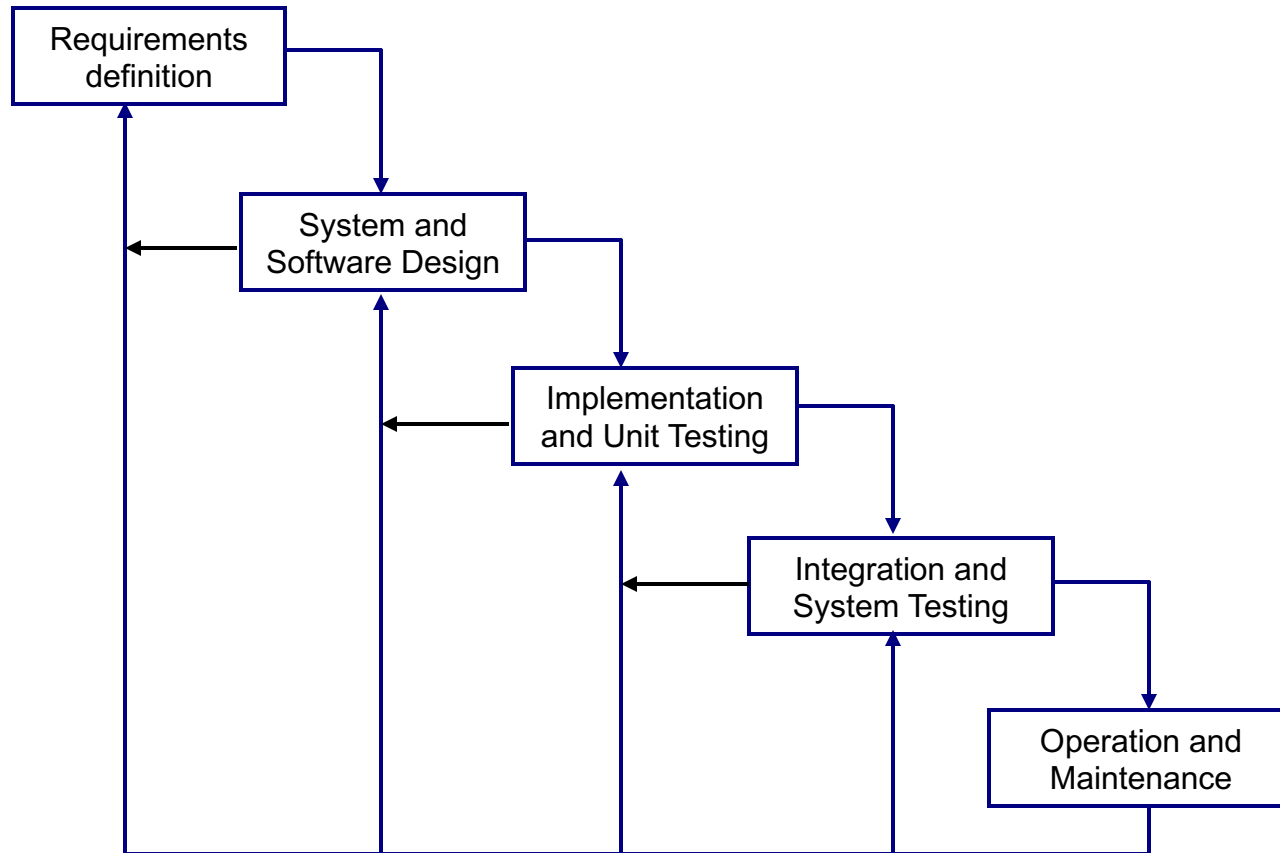
- A set of standard outcomes of each stage is defined



Waterfall Model Problems

- Inflexible partitioning of the project into distinct stages
 - difficult to respond to changing requirements
- Only appropriate when the requirements are well-understood
 - changes will be fairly limited during the design process
 - But, few business systems have stable requirements
- Mostly used for large systems engineering projects

Waterfall Model Revised



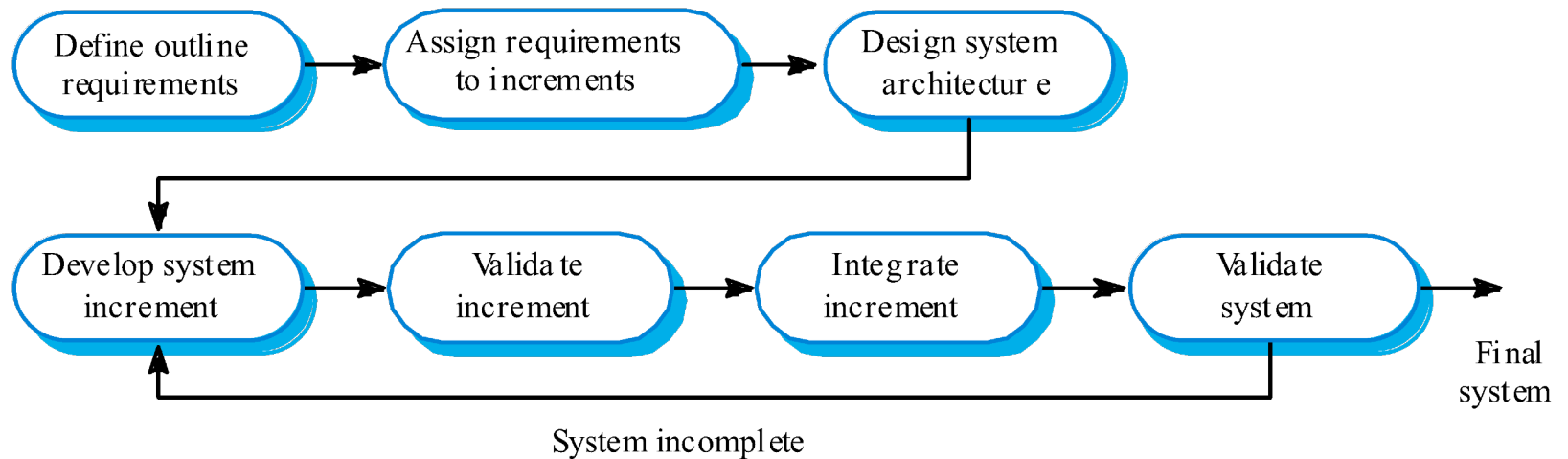
Process iteration

- Project is divided into many iterations
 - Rework can occur in later iterations
- Iteration can be applied to any of the generic process models
- Two (related) approaches
 - Incremental delivery/development
 - Spiral development

Incremental development

- Development and delivery is broken down into increments
 - each increment delivering part of the required functionality
- Requirements are prioritized
 - Highest priority requirements are developed in early increments

Incremental development (cont'd)



Advantages of Incremental Development

- System functionality is available earlier
- Early increments act as a prototype to help elicit requirements
- Highest priority requirements tend to receive most testing
- Lower risk of overall project failure

Outline

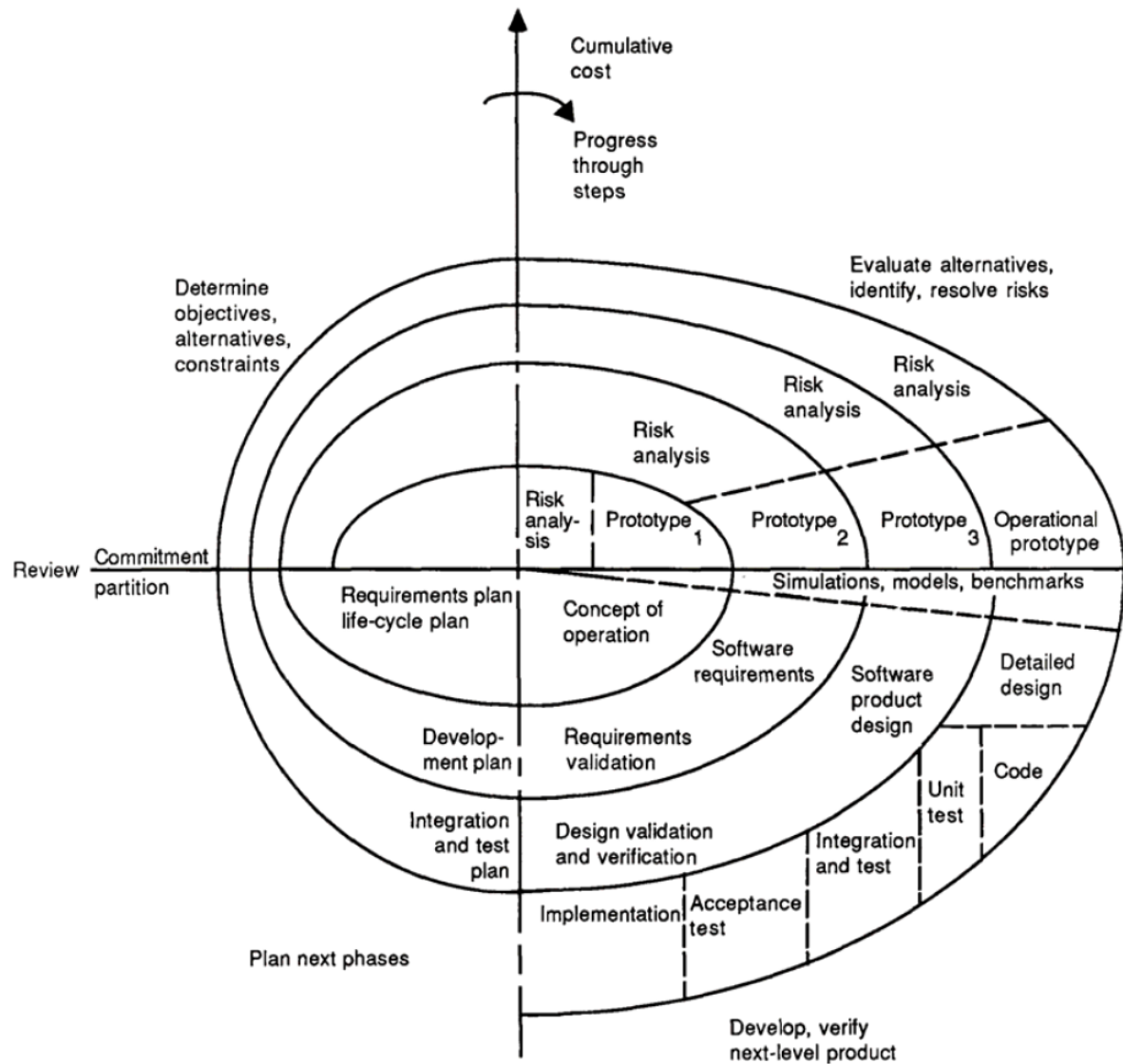
- Introduction to Plan-driven and Agile Methods
- Planning Spectrum
- Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)



Spiral Development

- Project activities are organized into spirals instead of sequence
- Each spiral represents a phase in the process
- No fixed phases such as specification or design
- Risks are assessed and resolved throughout the process

Spiral Model



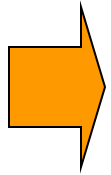
(Boehm 1988)

Spiral Model Sectors

- Objective setting
 - ❑ Specific objectives for the phase are identified
- Risk assessment and reduction
 - ❑ Risks are assessed
 - ❑ Mitigation activities are planned and taken
- Development and validation
 - ❑ Develop and validate system
- Planning
 - ❑ Review previous spiral
 - ❑ Plan for next spiral

Outline

- Introduction to Plan-driven and Agile Methods
- Planning Spectrum
- Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)

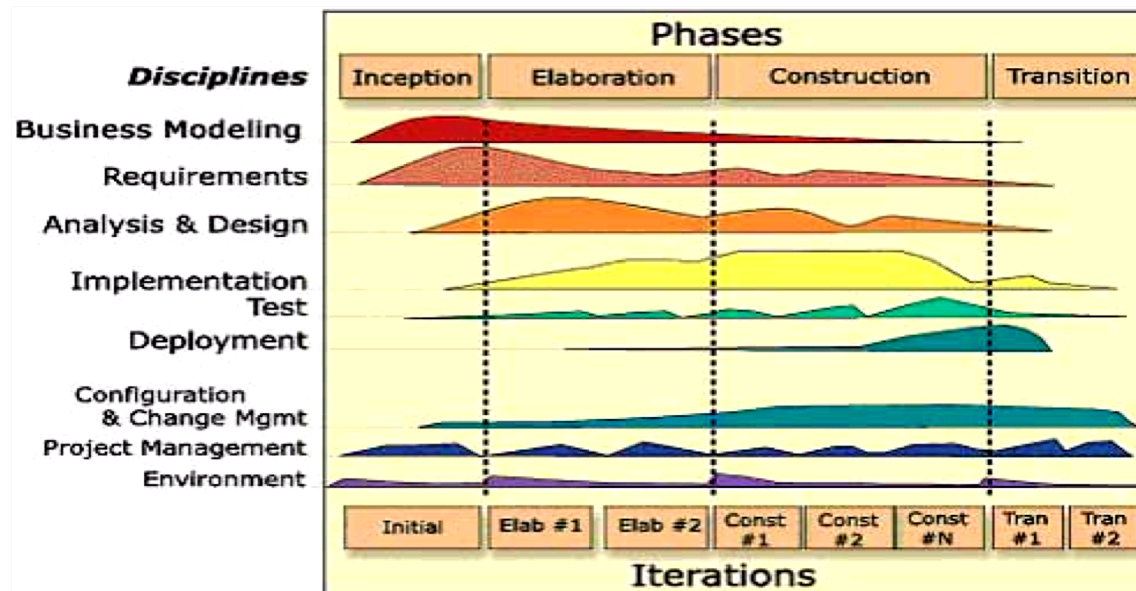


The Rational Unified Process (RUP)

- A popular process model developed and introduced by Rational Software, now IBM Rational
- Normally described from 3 perspectives
 - Dynamic perspective: shows phases over time
 - Static perspective: shows process activities
 - Practical perspective: suggests good practice

RUP

- Activities grouped into workflows
- Four phases Inception, Elaboration, Construction, and Transition
- Each phase divided into one or more iterations



RUP Phases

- Inception
 - ❑ Establish the business case for the system
- Elaboration
 - ❑ Develop an understanding of the problem domain and the system architecture
- Construction
 - ❑ System design, programming and testing
- Transition
 - ❑ Deploy the system in its operating environment

RUP Iteration

- Each phase is divided into one or more iterations
- Each iteration is considered a waterfall cycle
 - including requirements, analysis & design, implementation, testing, and deployment
- Each iteration has entry and exit criteria or check points
 - Iteration is completed if its check points are satisfied
- Typical number of iterations per phase
 - Inception: 1-2 iterations
 - Elaboration: 1-3
 - Construction: 2-3
 - Transition: 2-3

RUP Workflows

- Business modeling
 - The business processes are modeled using business use cases
- Requirements
 - Requirements specification, use-case model
- Analysis and Design
 - Software architecture and design are created
 - Different software views (use-case view, design view, process view, implementation view, deployment view)
- Implementation
 - Software components are implemented

RUP Workflows (cont'd)

- Testing
 - Includes testing activities performed to verify and validate the product against the requirements
- Deployment
 - Activities to release and deploy the product and help users to use the product successfully
- Configuration and Change Management (SCM)
 - Manages changes to the software product
- Project Management
 - Manages software development
- Environment
 - Prepares and ensures software tools, processes, and hardware for the development team

RUP Good Practices

- Develop software iteratively
- Manage requirements
- Use component-based architectures
- Visually model software
- Verify software quality
- Control changes to software
- Use tools
- Use-case driven development
- Modeling and UML

Outline

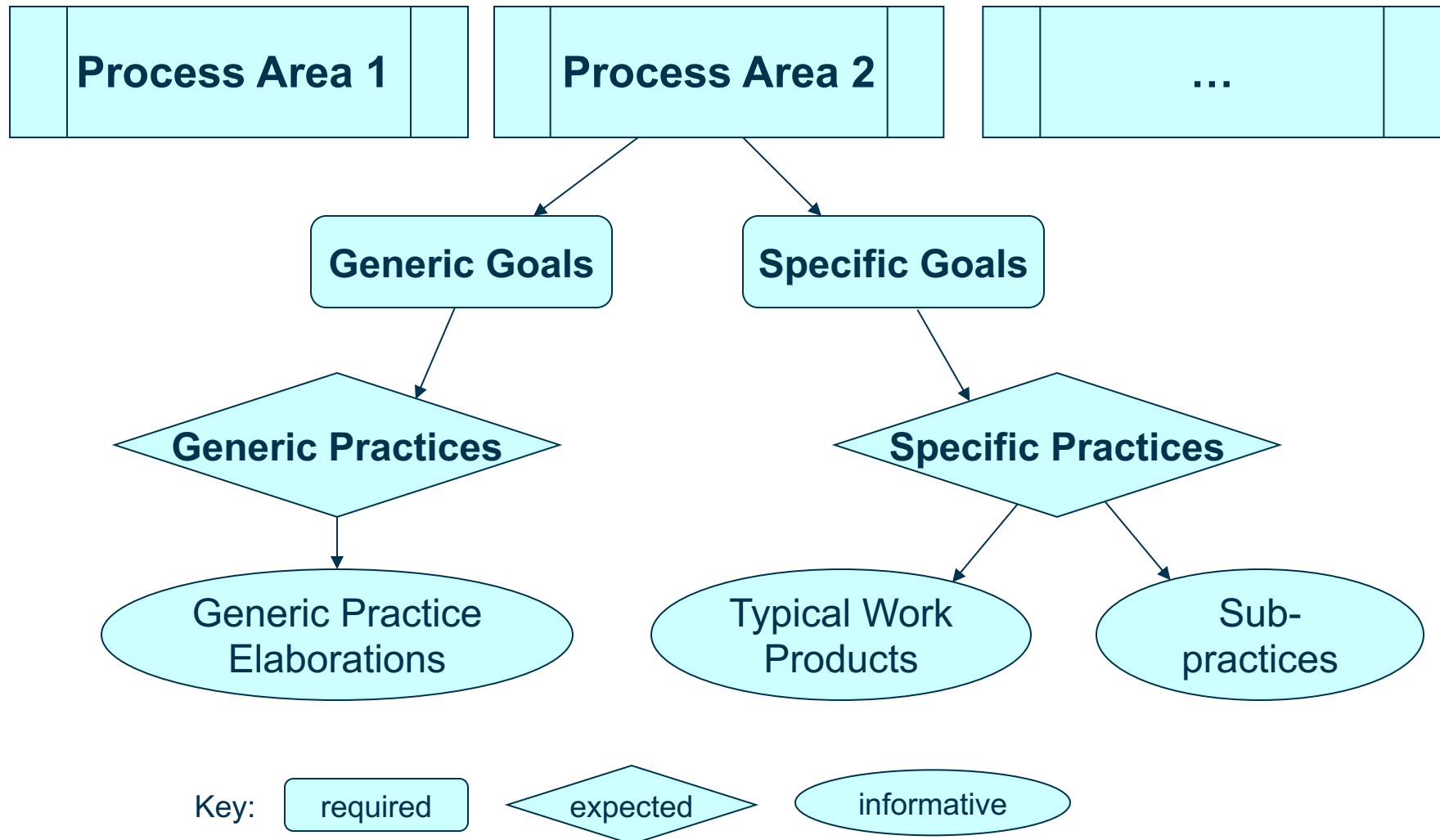
- Introduction to Plan-driven and Agile Methods
- Planning Spectrum
- Plan-driven Methods
 - Waterfall Model
 - Spiral Model
 - Rational Unified Process
 - Capability Maturity Model Integration (CMMI)



CMMI

- CMMI – Capability Maturity Model Integration
- Developed and promoted by CMU's Software Engineering Institute (SEI)
- **CMMI is a framework for software and systems process maturity**
- Allowing to support current and future software development trends, e.g.,
 - ❑ everything connected
 - ❑ rapid requirements change
 - ❑ adaptive process models
 - ❑ requirements, architecture, development are done concurrently
 - ❑ collaborative

CMMI Model Components



CMMI Model Components (cont'd)

■ Process Area

- A cluster of related practices in an area that satisfy a set of **goals**

■ Goal

- **Generic goal**: a required model component that describes the characteristics that must be present
- **Specific goal**: a required model component that describes the **unique** characteristics that must be present to satisfy the process area

CMMI Model Components (cont'd)

■ Practice

- ❑ **Generic practice**: an expected model component that is considered important in achieving the associated **generic** goal
- ❑ **Specific practice**: an expected model component that is considered important in achieving the associated **specific** goal

■ Work product

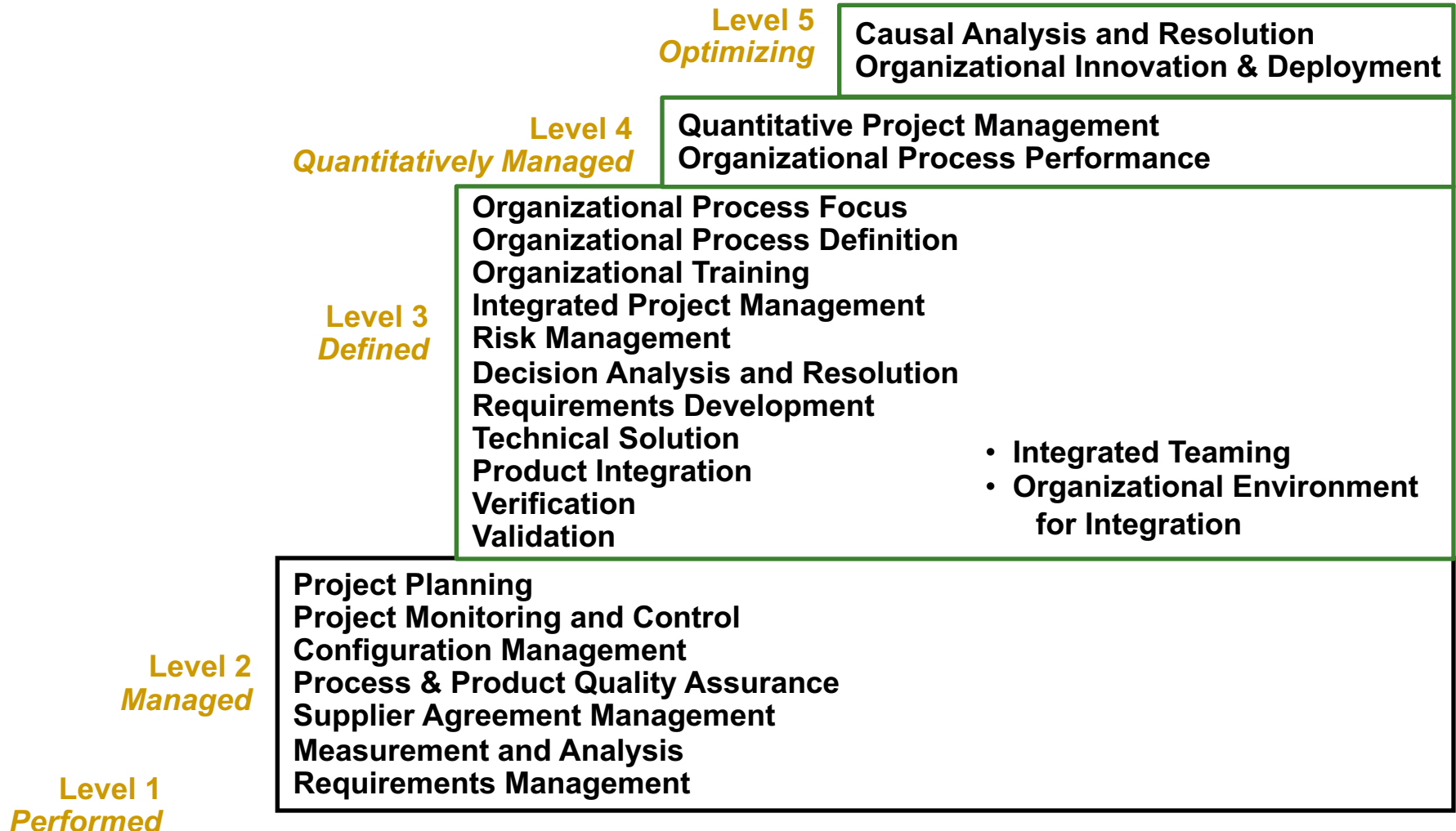
- ❑ Any artifact produced by a process

■ Sub-practice

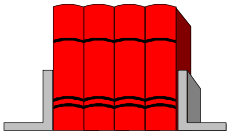
- ❑ Providing guidance for interpreting and implementing specific or generic practices

CMMI Process Areas

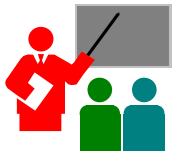
Staged Representation



Common Features (for all Process Areas)



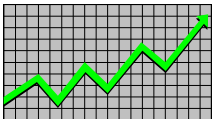
- **Commitment to Perform** includes practices that ensure the process is established and will endure.
 - Organizational policies



- **Ability to Perform** includes practices that establish the necessary conditions for implementing the process completely.
 - Plans, resources, responsibility, and training.



- **Activities Performed** includes practices that directly implement a process area.



- **Directing Implementation** includes measurement practices that are necessary to collect and analyze data related to the process.
 - Configuration management, stakeholder involvement, monitor and control

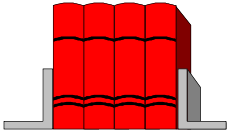


- **Verification** includes practices that ensure compliance with the process that has been established.
 - Senior management reviews, quality audits

Typical Organizational Responsibilities

| | | Commitment to Perform | Ability to Perform | Activities Performed | Directing Implementation | Verification |
|---------|---|-------------------------|-------------------------|----------------------|--------------------------|--|
| Level 5 | Causal Analysis and Resolution (Support) Org Innovation and Deployment (Process) | | | | | |
| Level 4 | Quantitative Project Management (PM) Organizational Process Performance (Process) | | | | | |
| Level 3 | Organizational Process Focus (Process) Organizational Process Definition (Process) Organizational Training (Process) Integrated Project Management (PM) Risk Management (PM) Decision Analysis and Resolution (Support) Requirements Development (Eng) Technical Solution (Eng) Product Integration (Eng) Product Verification (Eng) Validation (Eng) | Organizational policies | Organizational training | | Standard metrics | Senior management review Quality Assurance organization |
| Level 2 | Project Planning (PM) Project Monitoring and Control (PM) Configuration Management (Support) Product and Process Quality Assurance (Support) Supplier Agreement Management (PM) Data Management (Support) Measurement and Analysis (Support) Requirements Management (Eng) | | | | | |

Organizational Assets (for all Process Areas)



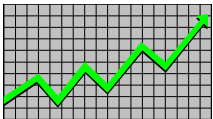
- Organizational policies



- Organizational training office



- Organizational procedures



- Standard metrics
- Estimation process, tools & support



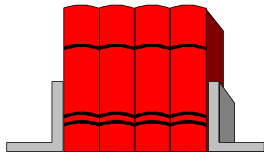
- Senior management review process
- Independent quality assurance organization

Organizational Process Areas - Level 3



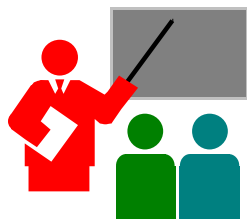
■ Organizational Process Focus

- ❑ Establishes and maintains an understanding of the organization's processes and process assets, build an infrastructure to support their use, and plan and coordinate the organization's process improvement activities



■ Organizational Process Definition

- ❑ Establishes and maintains a usable set of organizational process assets.



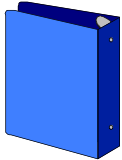
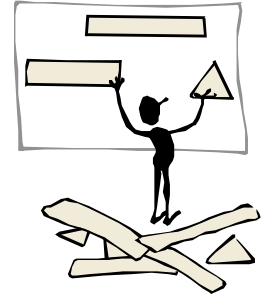
■ Organizational Training

- ❑ Develops the skills and knowledge of people so they can perform their roles effectively and efficiently.

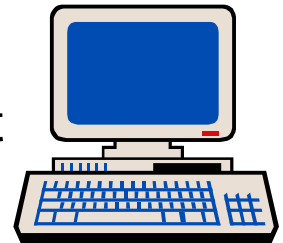
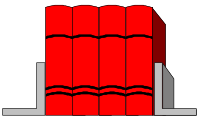
Organization's Process Assets



- Organization's standard process (including the process architecture and process elements)
- Descriptions of life cycles approved for use

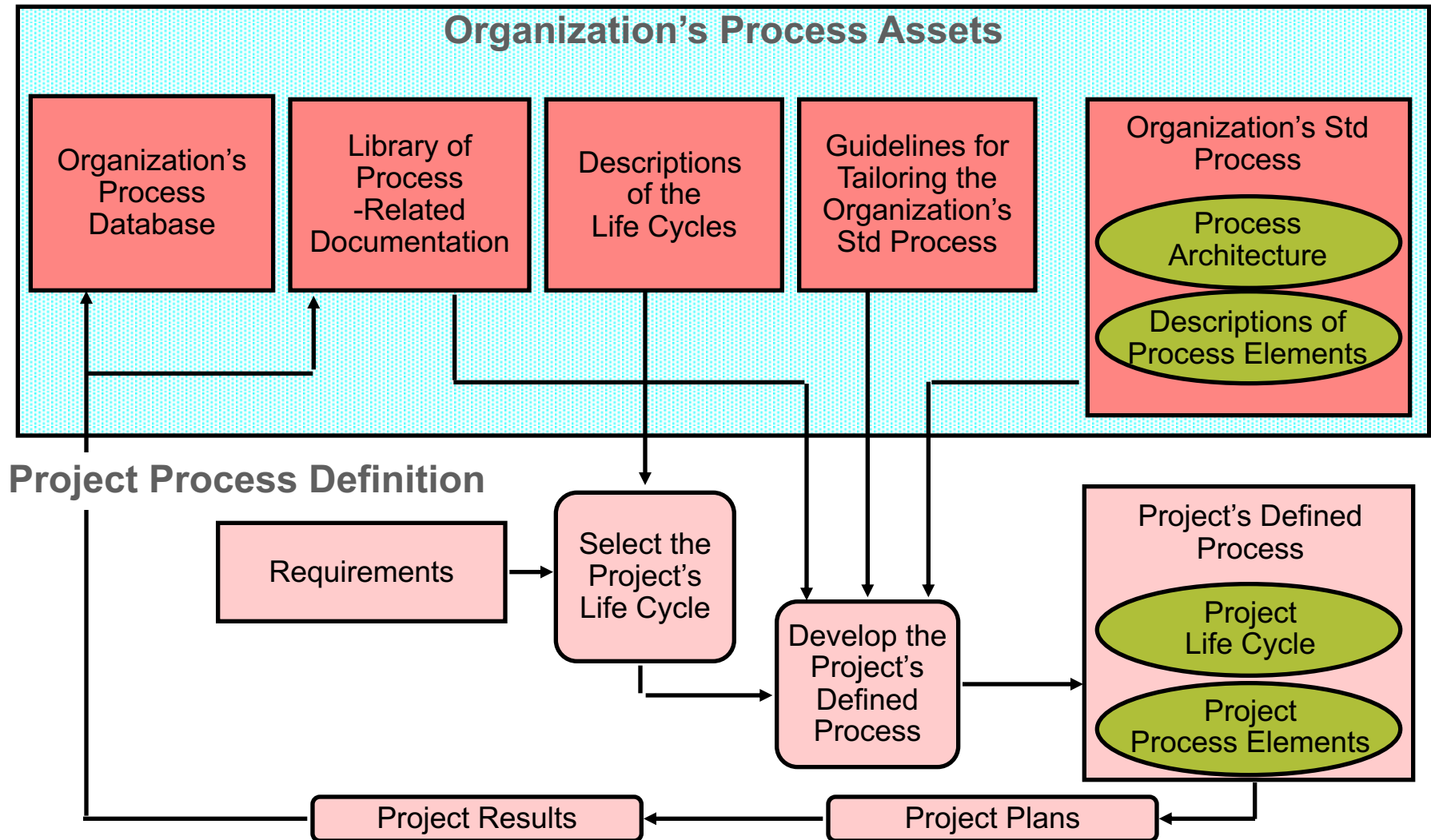


- Guidelines and criteria for tailoring the organization's standard process
- Organization's process database (historical cost and schedule data)
- Library of process-related documentation

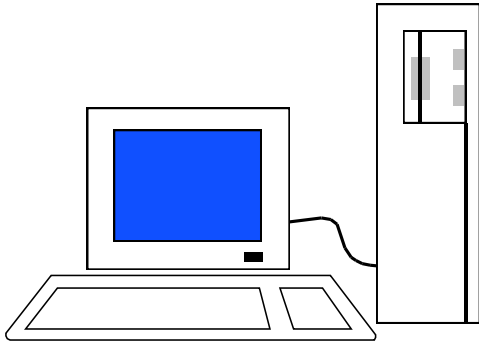


The process assets are available for use by the projects in developing, maintaining, and implementing their defined software process.

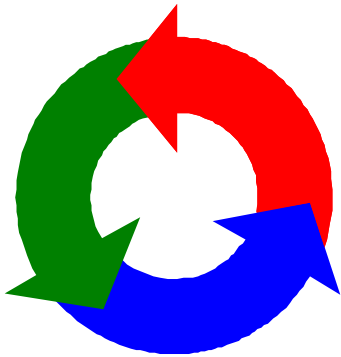
Conceptual Software Process Framework



Organizational Process Areas - Levels 4 & 5



- **Organizational Process Performance**
 - ❑ Establishes and maintains a quantitative understanding of the performance of the organization's set of standard processes
 - ❑ Provides the process performance data, baselines, and models to quantitatively manage the organization's projects.



- **Organizational Innovation and Deployment**
 - ❑ Selects and deploys incremental and innovative improvements that measurably improve the organization's processes and technologies.

Summary

- Two Software Development Approaches
 - Plan-driven
 - Agile
- Each has strengths and weaknesses
- Plan-driven methods
 - Waterfall
 - Spiral
 - Rational Unified Process
 - CMMI

Summary (cont'd)

■ Strengths of plan-driven methods

- ❑ High assurance, suitable for safety-critical software systems
- ❑ Suitable for large software systems
- ❑ Documentation available for training new staff
- ❑ Making sure everyone to work in certain ways through clear processes

■ Weaknesses of plan-driven methods

- ❑ Documentation overhead
- ❑ Outdated and useless documents
- ❑ Inflexible to changes
- ❑ Obstruct innovations
- ❑ Engineers don't like documentation

References

- B. Boehm, “A spiral model of software development and enhancement”, *Computer*, 1988
- B. Boehm, “Get Ready for Agile Methods, with Care”, *IEEE Computer*, 2002
- B. Boehm, R. Turner, “Balancing Agility and Discipline: A Guide for the Perplexed”, *Addison-Wesley*, 2003
- I. Sommerville, “Software Engineering,” 8th Edition, *Addison-Wesley*, 2007