Software Process Models

LECTURE 2

Software Process Models

- The set of *activities* and *associated results* that **produce a software product.**
- **Four fundamental process activities**
  - Software Specification
  - Software Development
  - Software Validation
  - Software Evolution
- **Organized differently by different Software process models having different levels of detail**

1. Software Specifications

- Customers and Software Engineers define the software to be produced and the constraints on its operations. Typical Stages are,
- **Feasibility Study:**
  - Is it possible with the current technologies + within budget?
- **Domain Analysis:**
  - What is the background for the software?
- **Requirements Gathering and Analysis:**
  - What is it that the user wants?
- **Requirements Specification:**
  - Formal documentation on *User* and *System* requirements.
- **Requirements Validation:**
  - Check for realism consistency and completeness , consistency, and completeness.

2. Software Development

- **Consists of Design and Programming**
  - System Analyst design the software and decide how the requirement can be implemented.
  - Programmers write code to translate the high level design into a real code in a chosen programming language

3. Software Validation

- **Software Engineer (or dedicated tester) and Customer**:
  - Check the software to ensure it meets the customers' requirements.
- **Typical Stages:**
  - **Component Testing:** Independent testing of individual components in subsystem.
  - **System Testing:** Testing of integrated components.
  - **Acceptance Testing:** Tested with customer supplied data. Final test before operation.

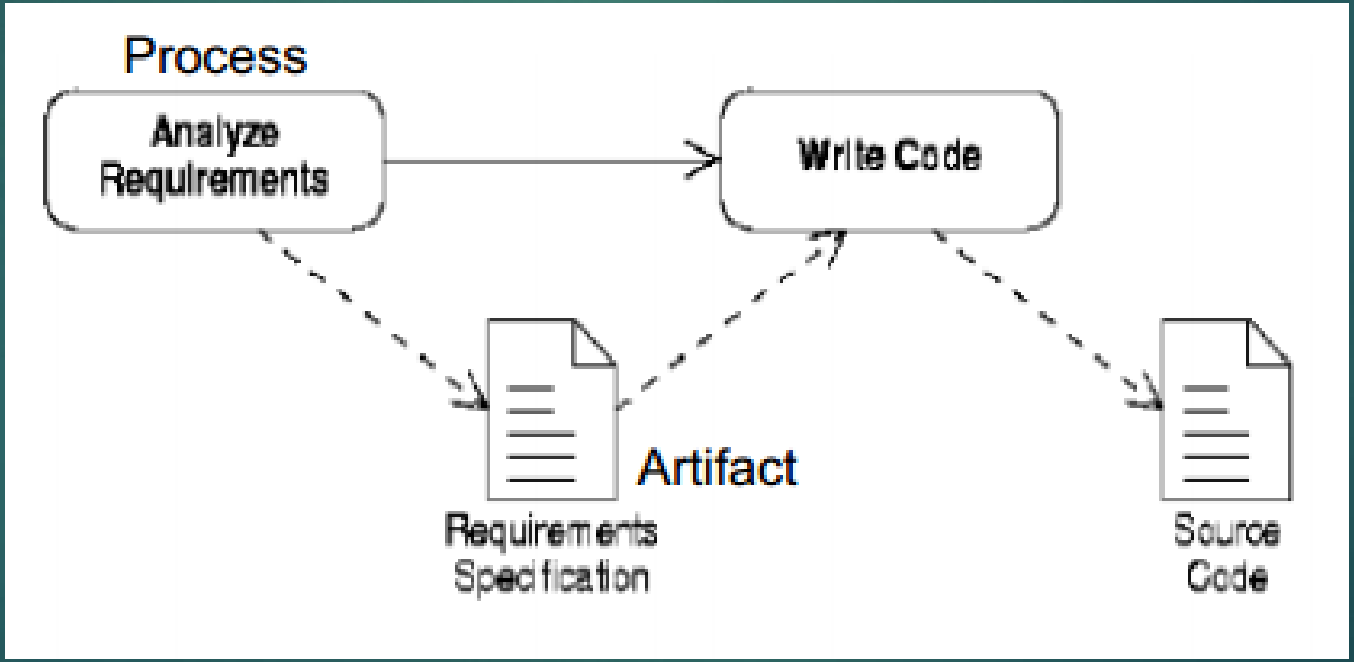
Interactive activity that feedback to previous stages: E.g., an error in component testing triggers re-coding.

4. Software Evolution

- Customers and Software Engineers:
- Define changing requirements.
  - Modify the software system to adapt.
  - Typical Work:
    Update the system for minor new requirements, e.g., changing the telephone number from 7 digits to 8 digits, changing the date representation (the *Millennium Bug*).

Simple Software Process

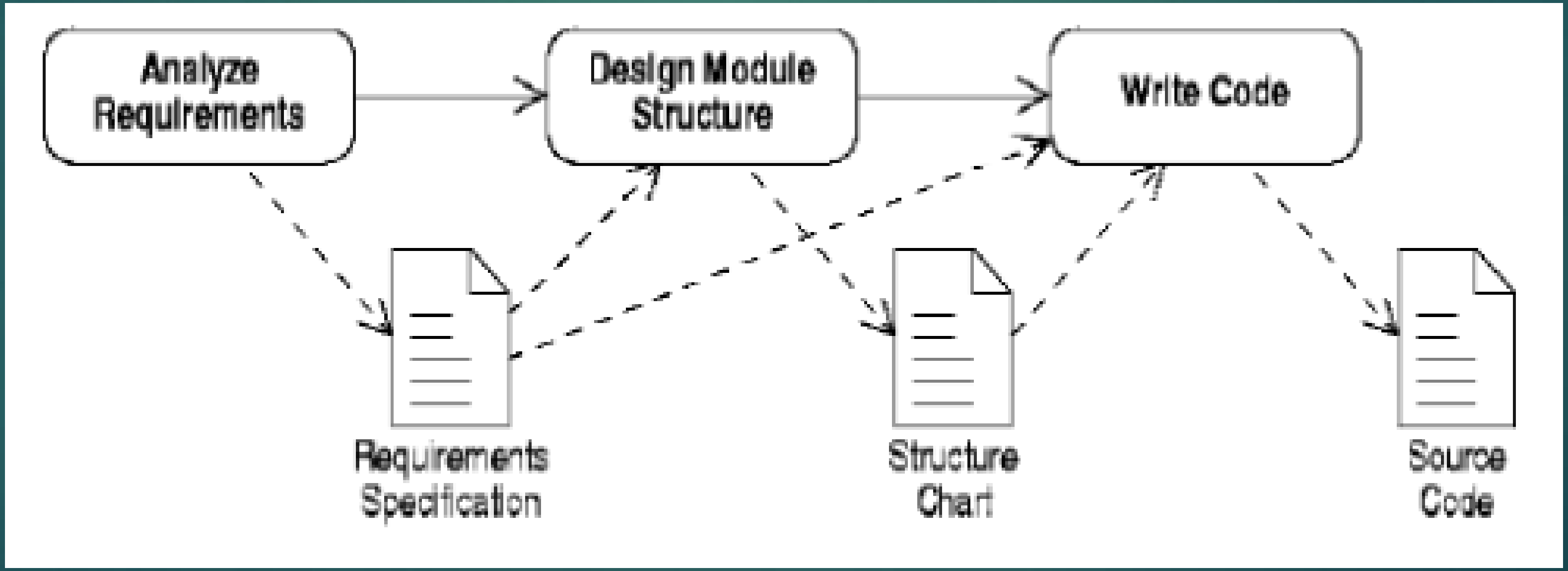
- In the simplest cases, code is written directly from some statements of requirements.



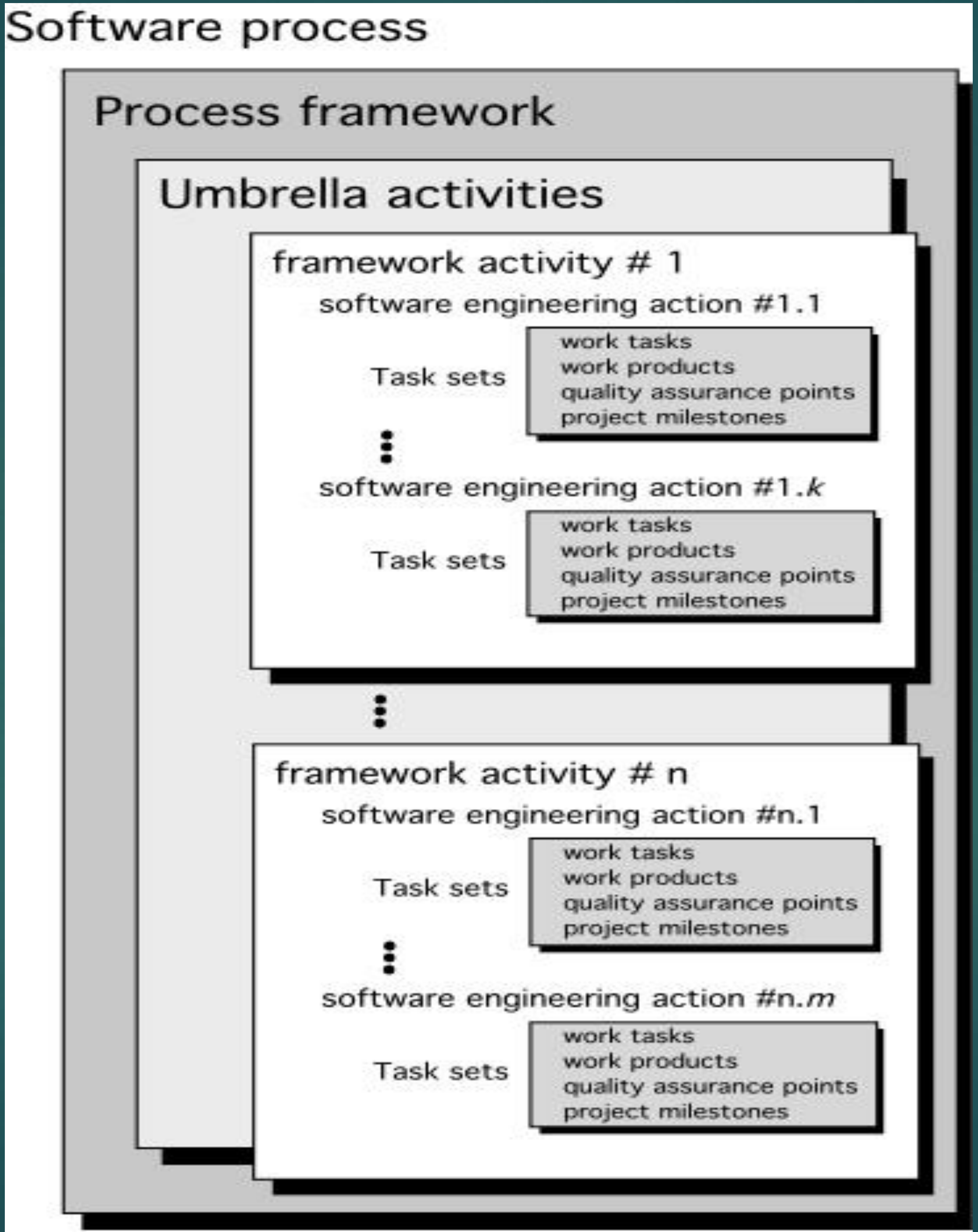
- Two processes:
  - Analyze requirements’
  - ‘Write code’
- Two artifacts:
  - ‘Requirements specification’
  - ‘Source code’
- ‘Requirements specification’ can be written as:
  - an informal outline or
  - a highly detailed description.

A more Complex Software Process

- It is better to design before you code.
- On larger projects, intermediate pieces of documentation are produced.

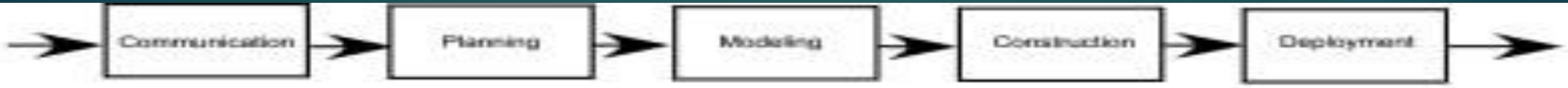


- One new process:
  - ‘Design module structure’ - splitting the program into modules and subroutine
- One new artifact:
  - ‘Structure chart’ – is based on the information contained in the ‘requirements specification’
  - Both the ‘requirements specification’ and the ‘structure chart’ are used when writing the final code.

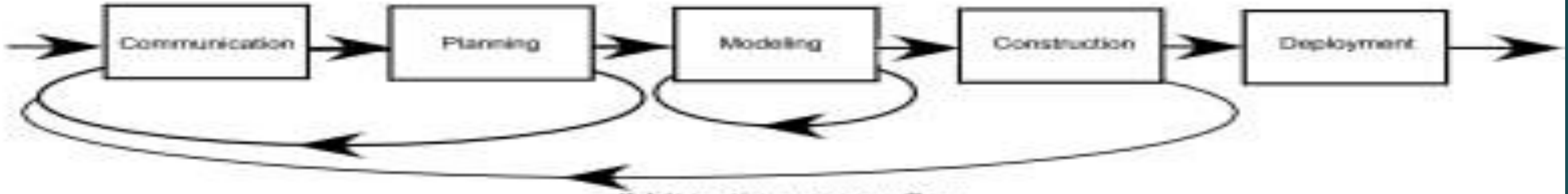

Software process
Process framework
Umbrella activities
framework activity # 1
software engineering action #1.1
Task sets
work tasks
work products
quality assurance points
project milestones
software engineering action #1.*k*
Task sets
work tasks
work products
quality assurance points
project milestones
framework activity # n
software engineering action #n.1
Task sets
work tasks
work products
quality assurance points
project milestones
software engineering action #n.*m*
Task sets
work tasks
work products
quality assurance points
project milestones

Identifying a Task Set

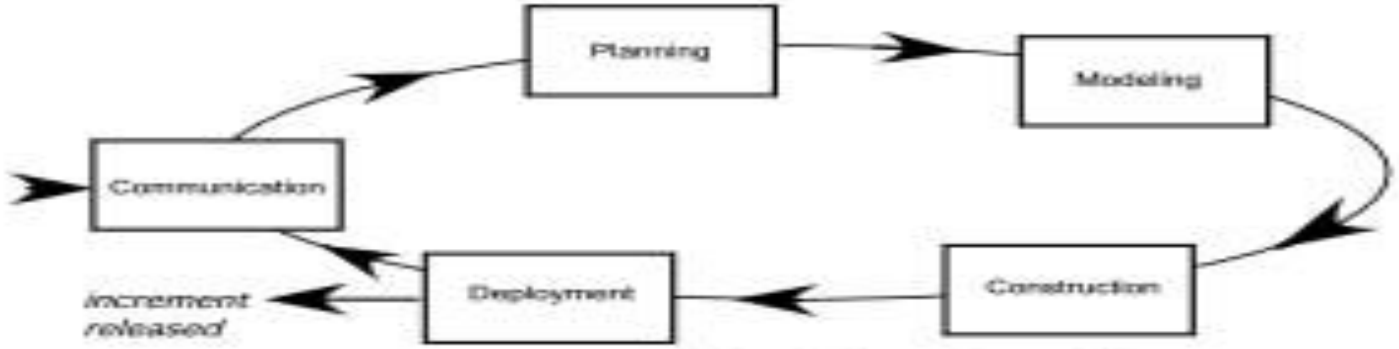
- A task set defines the actual work to be done to accomplish the objectives of a software engineering action.
  - A list of the task to be accomplished
  - A list of the work products to be produced
  - A list of the quality assurance filters to be applied



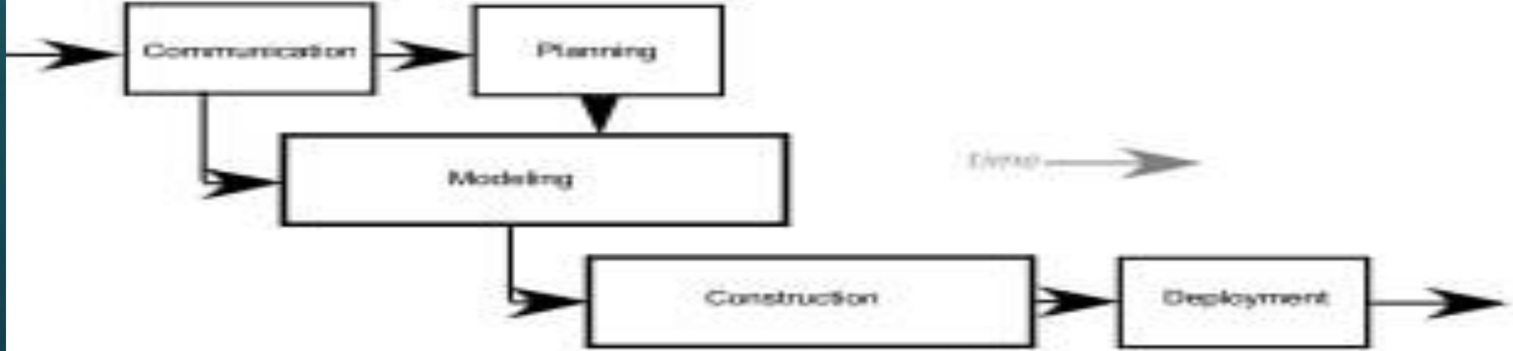
(a) linear process flow



(b) iterative process flow



(c) evolutionary process flow

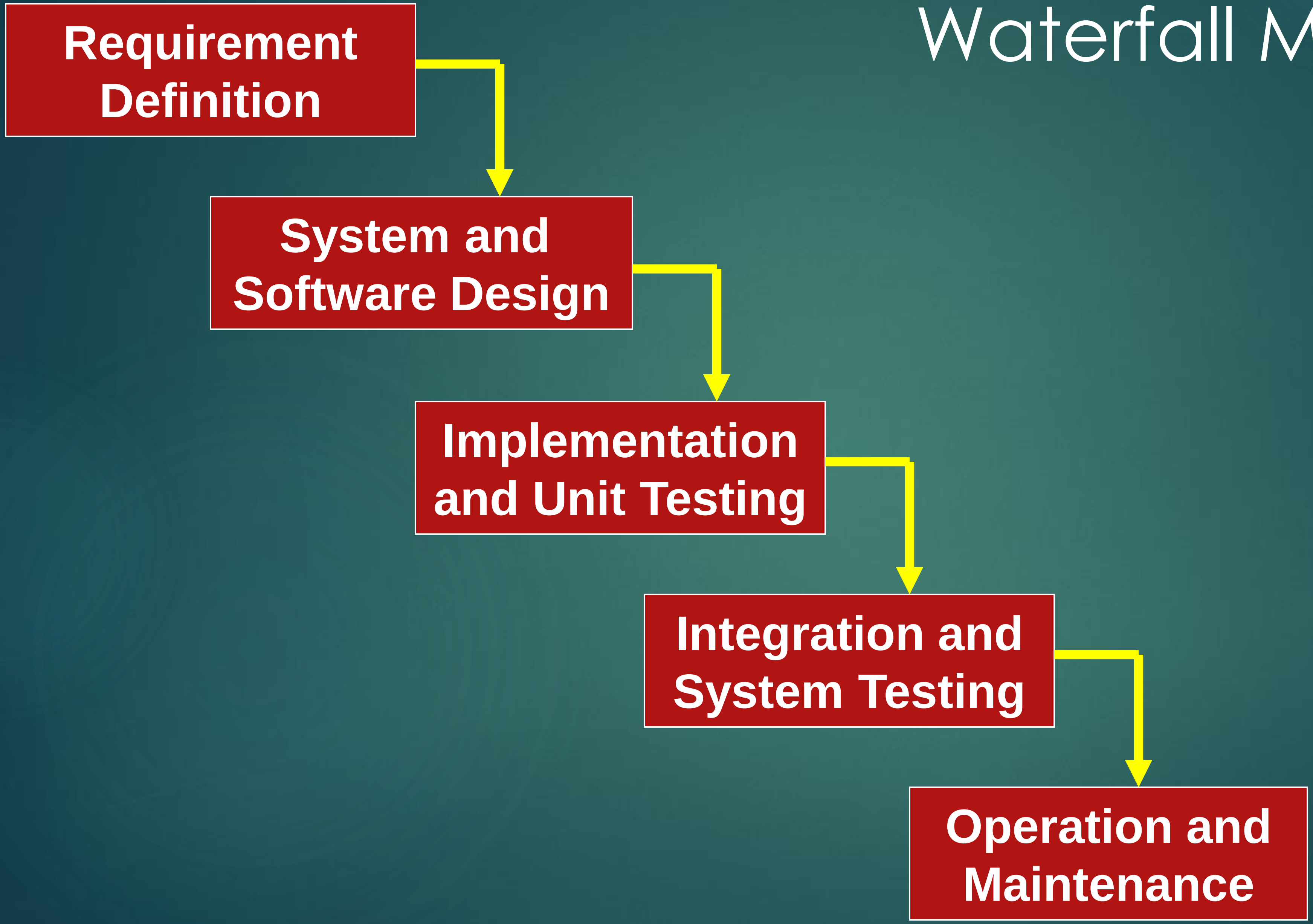


(d) parallel process flow

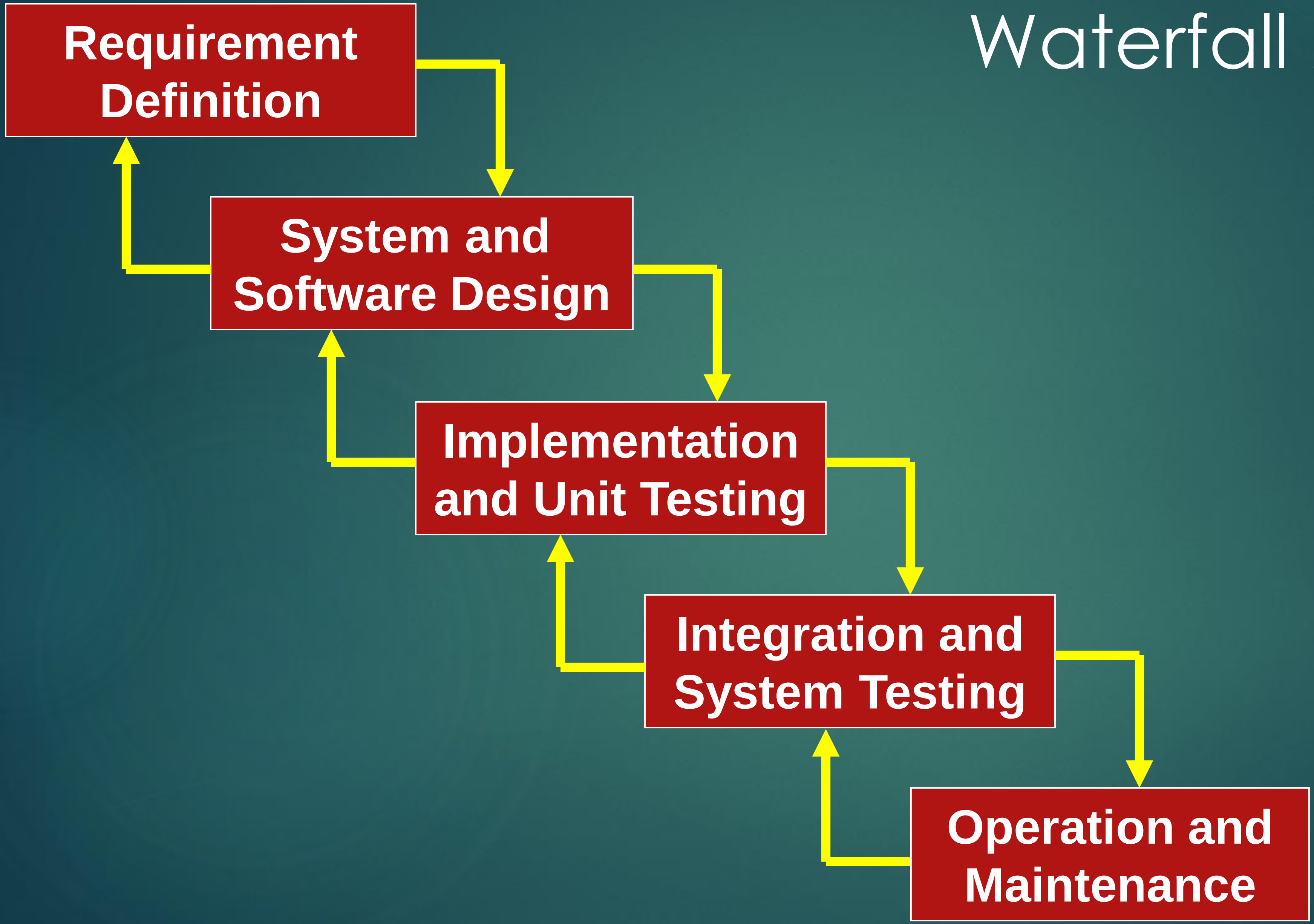
Phases of Software Process Model

- Requirements phase
- Specification phase
- Design phase
- Implementation phase
- Integration phase
- Maintenance phase
- Retirement

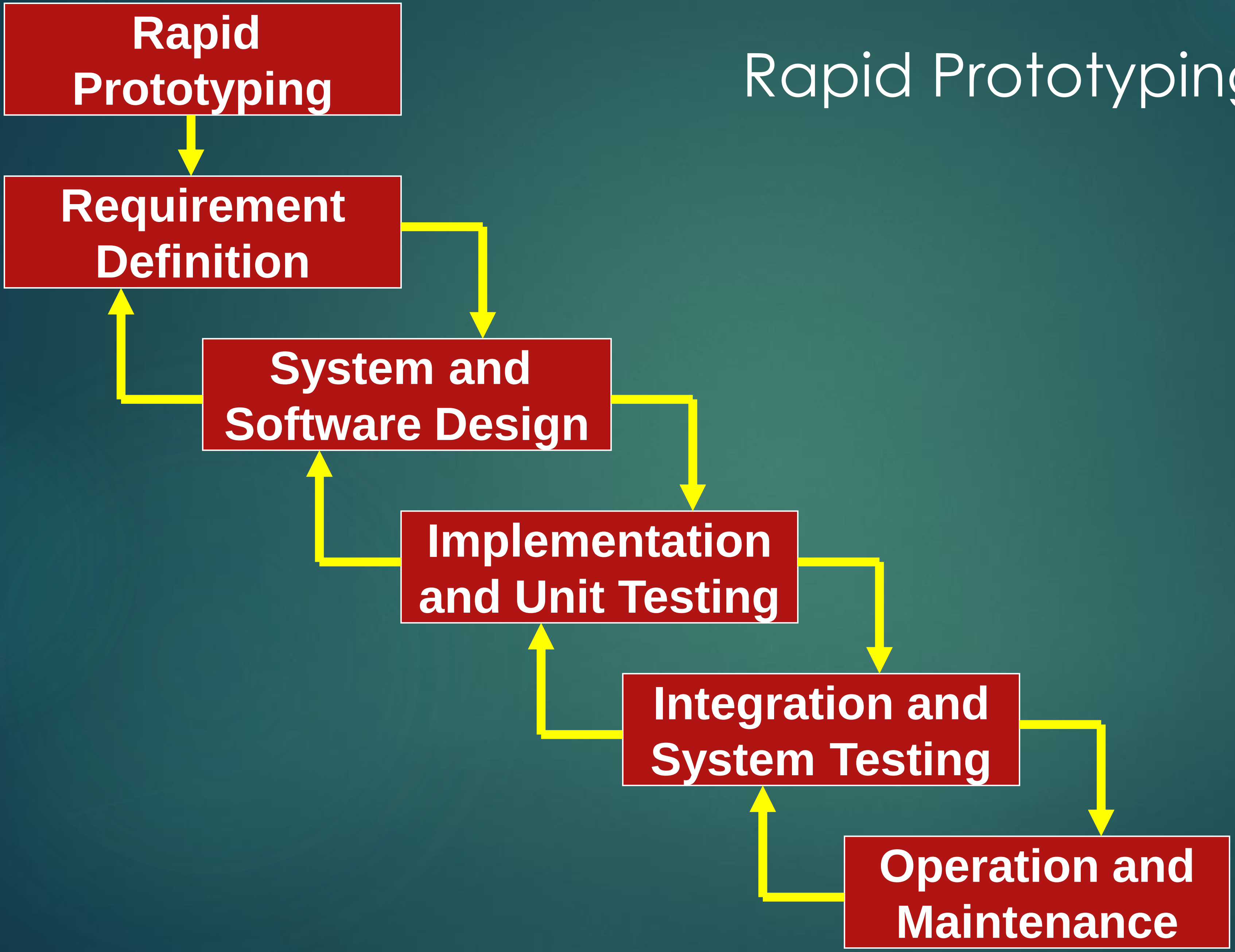
Waterfall Model


Requirement Definition
System and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

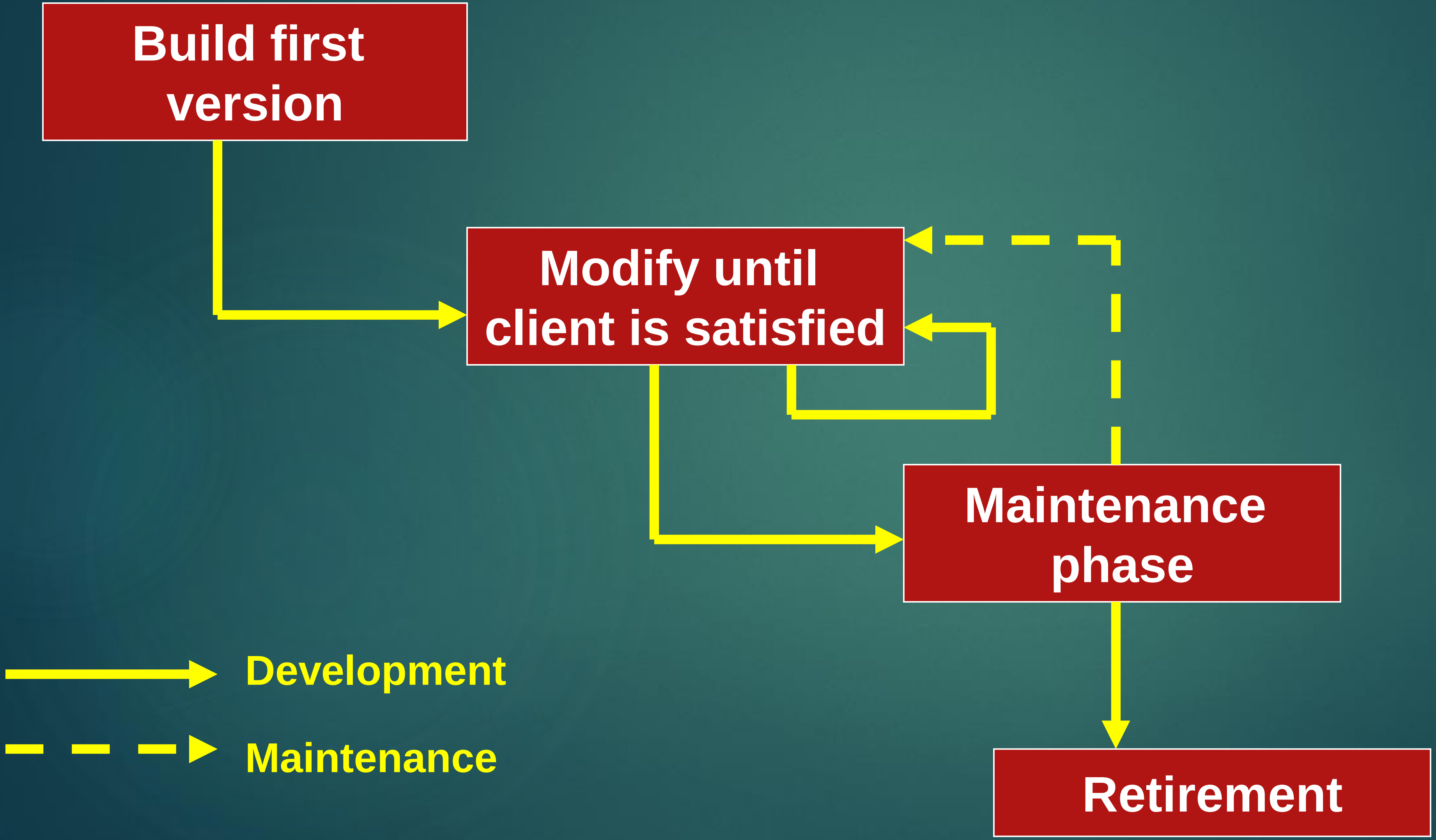
Waterfall Model


Requirement Definition
System and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

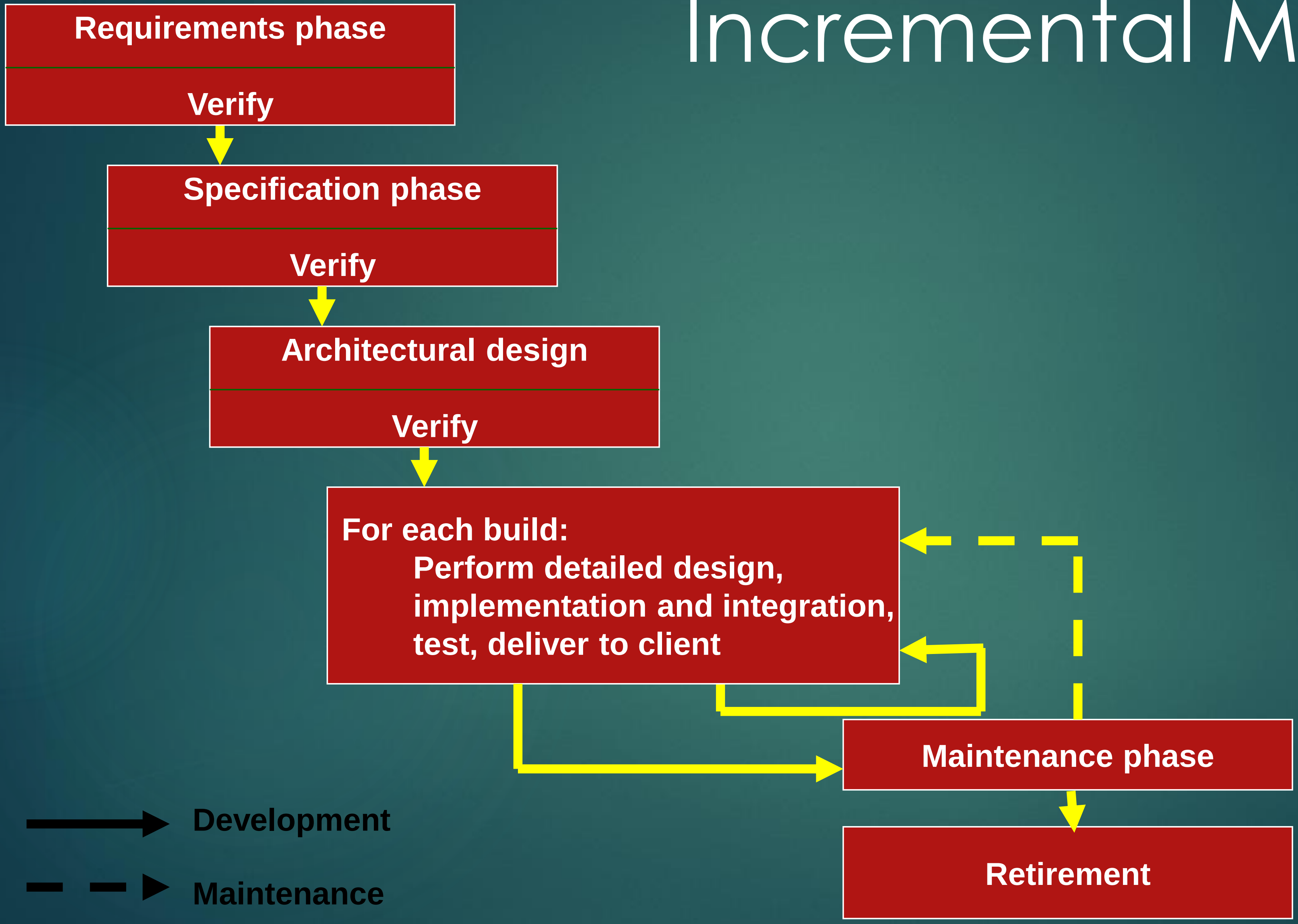
Rapid Prototyping Model


Rapid Prototyping
Requirement Definition
System and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

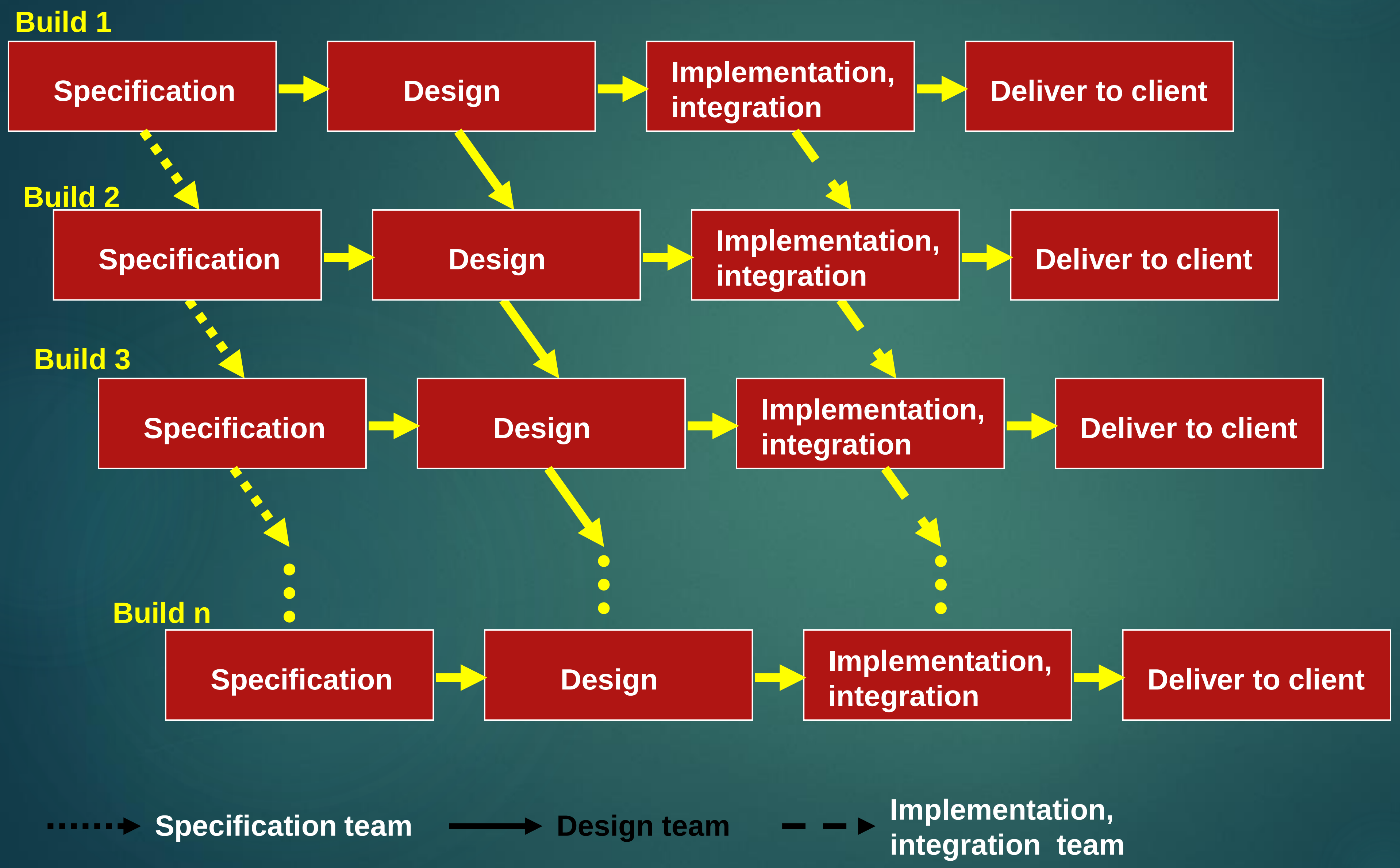
Build and Fix Model



Incremental Model

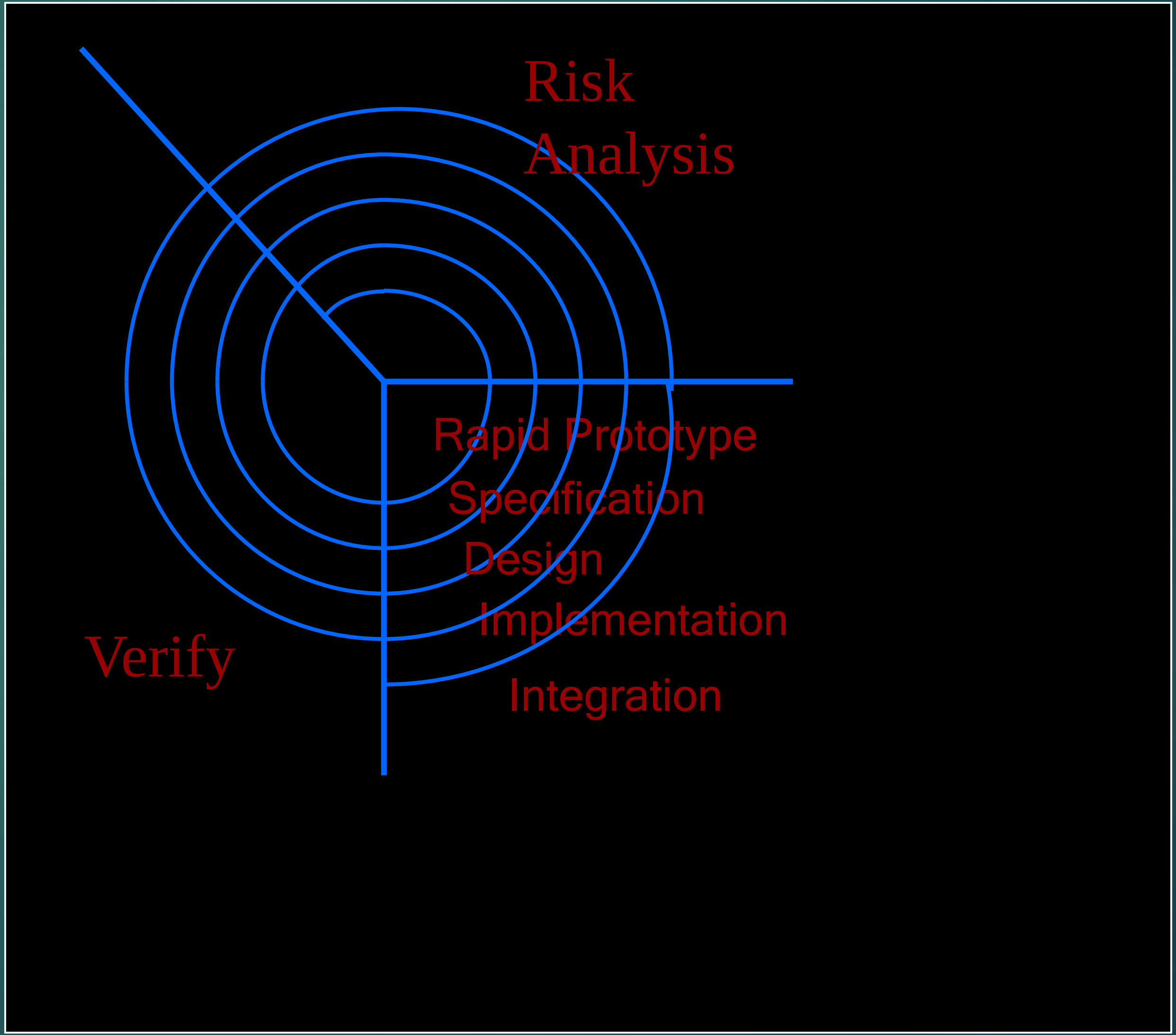


Incremental Model (cont.)

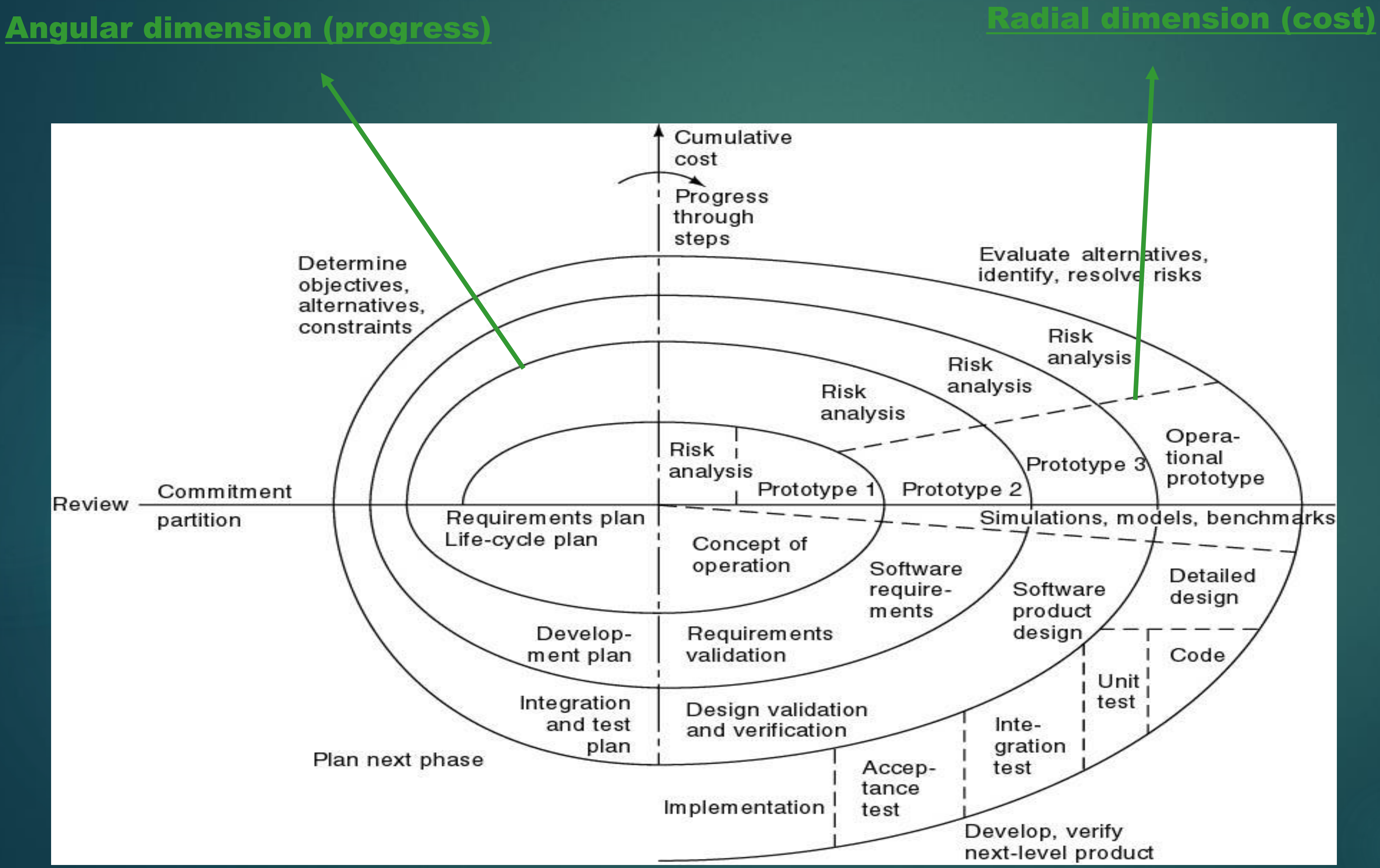


Simplified Spiral Model

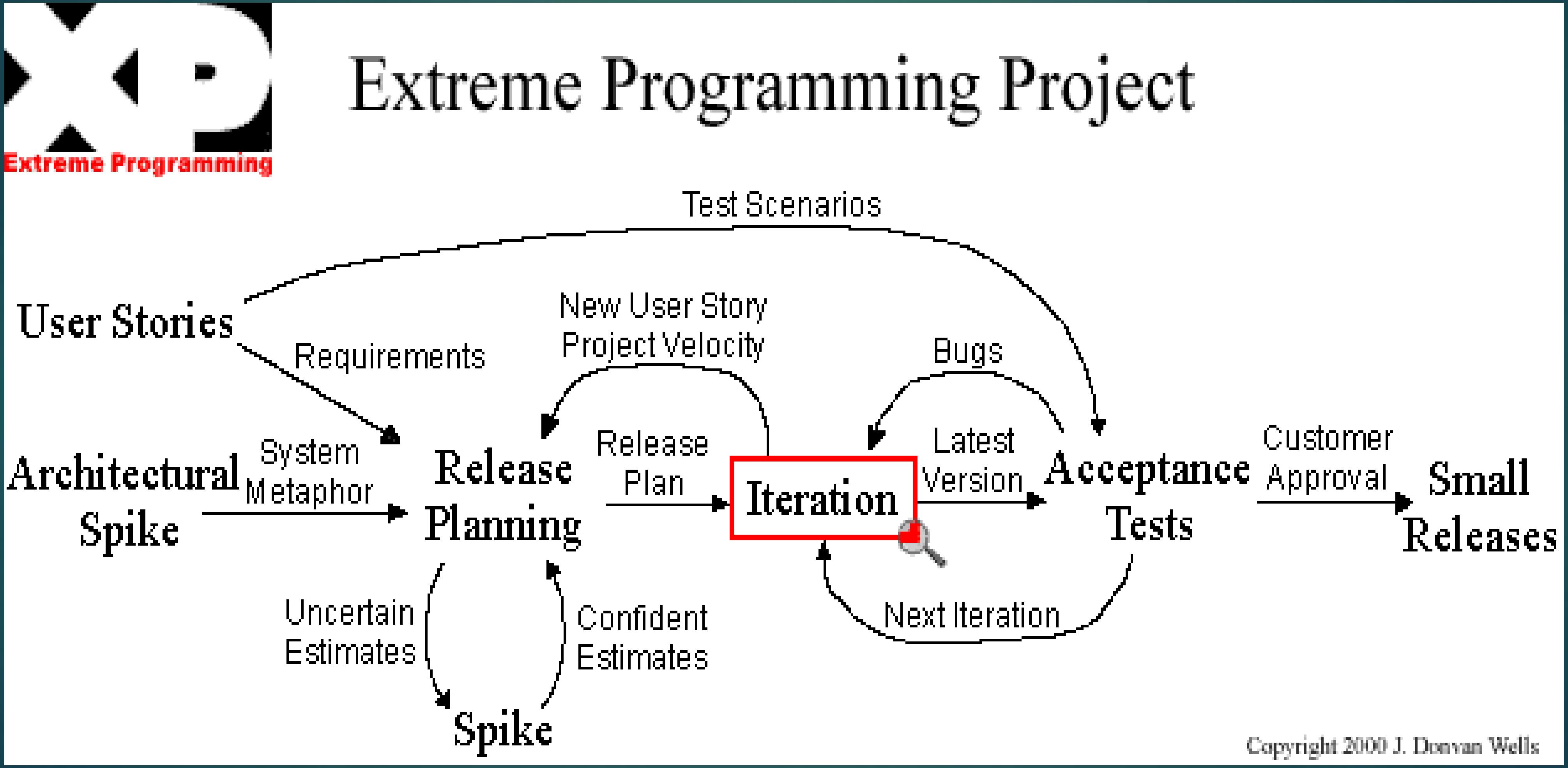
- If risks cannot be resolved, project is immediately terminated



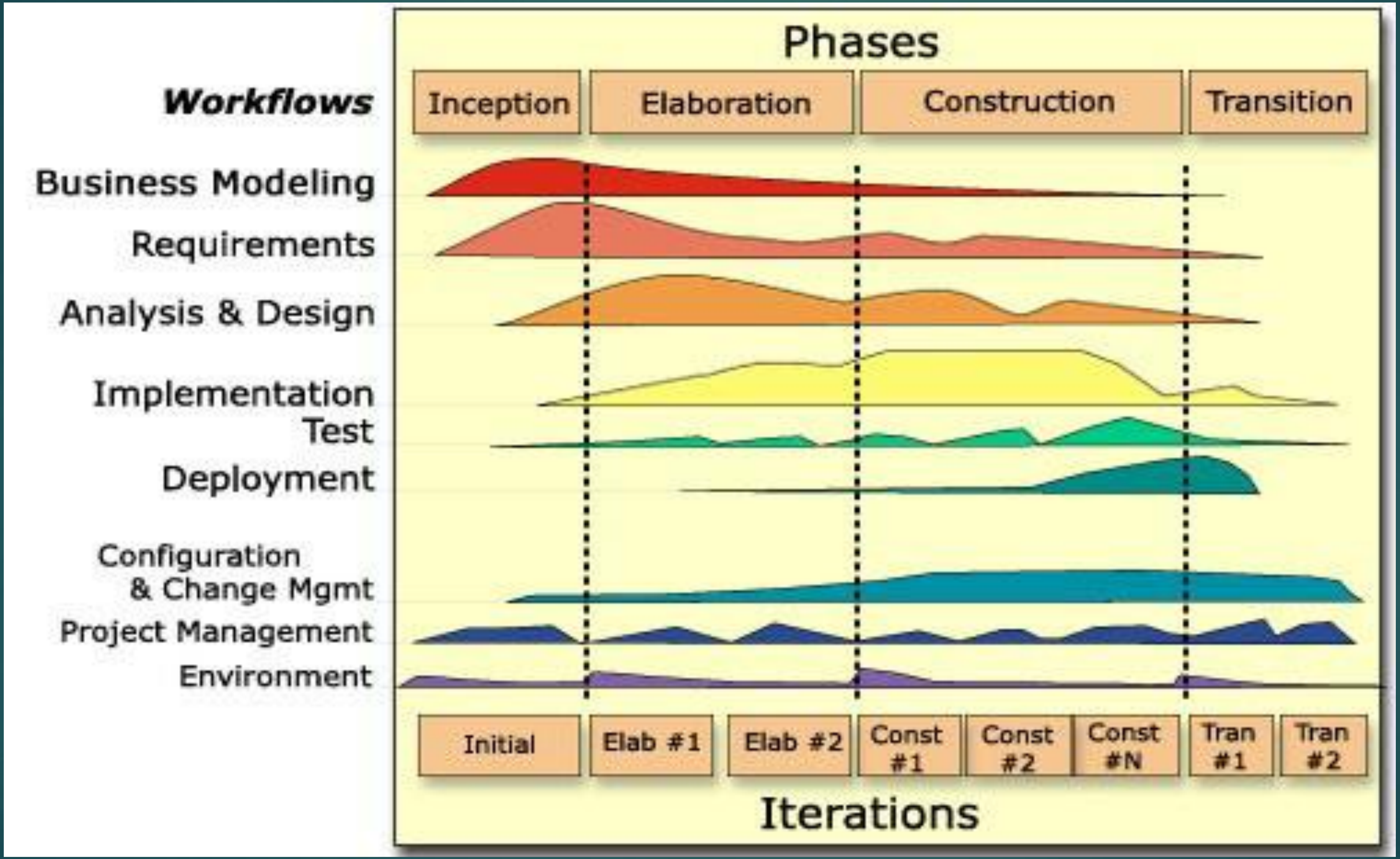
Full Spiral Model



Extreme Programming



Unified Process

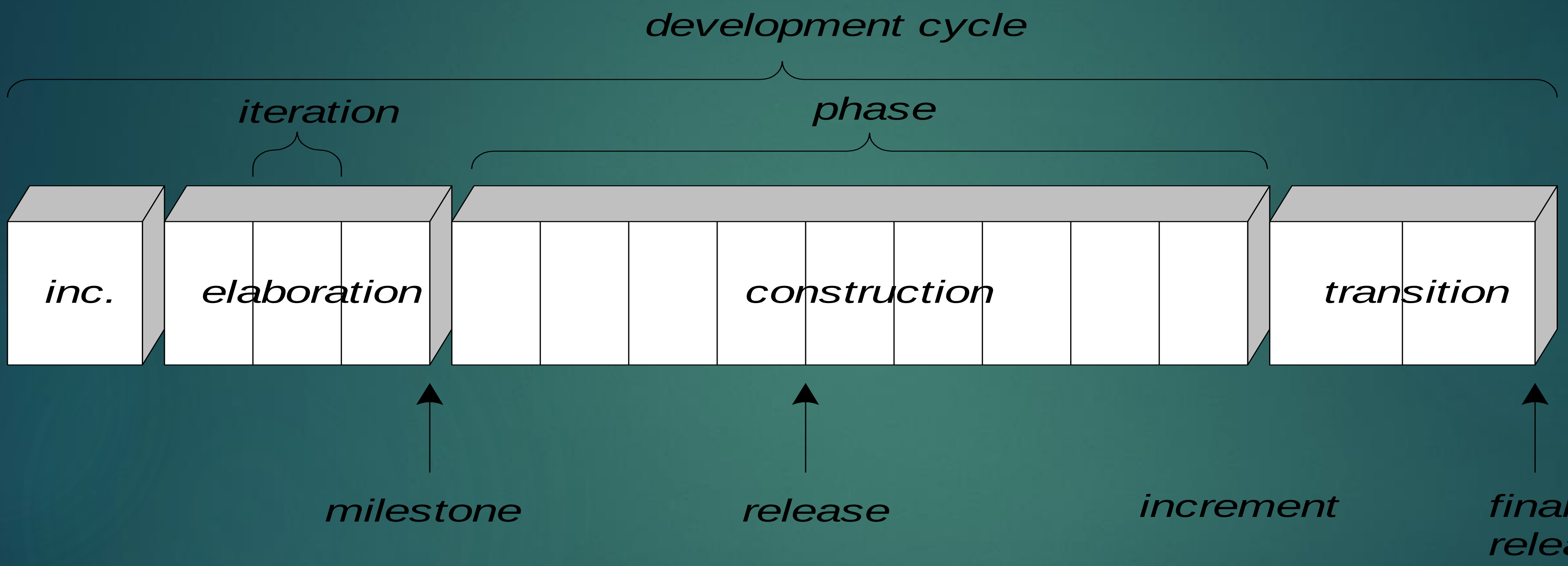


(This image is from P. Krutchen's paper)

What are the UP Phases?

- Inception:
  - Approximate vision, business case, scope, estimates
- Elaboration:
  - Refined vision, core implemented iteratively, attack high risks, most requirements identified
- Construction:
  - Fill in the details through iteration
- Transition:
  - Beta tests and deployment

UP Model



An iteration end-point when some significant decision or evaluation occurs

A stable executable subset of the final product. The end of each iteration is a minor release.

The difference (delta) between the releases of 2 subsequent iterations.

At this point, the system is release for production use

Documentation Phase?

- There is NO documentation phase
- Every phase must be fully documented before starting the next phase
  - Postponed documentation may never be completed
  - The responsible individual may leave
  - The product is constantly changing—we need the documentation to do this
  - The design (for example) will be modified during development, but the original designers may not be available to document it

Requirement Definition	• Rapid prototype, or • Requirements document	• Rapid prototype • Reviews
Functional Specification	• Specification document (specifications) • Software Product Management Plan	• Traceability • FS Review
Design	• Architectural Design • Detailed Design	• Traceability • Review
Coding	• Source code • Test cases	• Traceability • Review • Testing
Integration	• Source code • Test cases	• Integration testing • Acceptance testing
Maintenance	• Change record • Regression test cases	• Regression testing

Traceability matrix

Requirement ID	Use Case ID	UID	Class/ function	Test Case ID