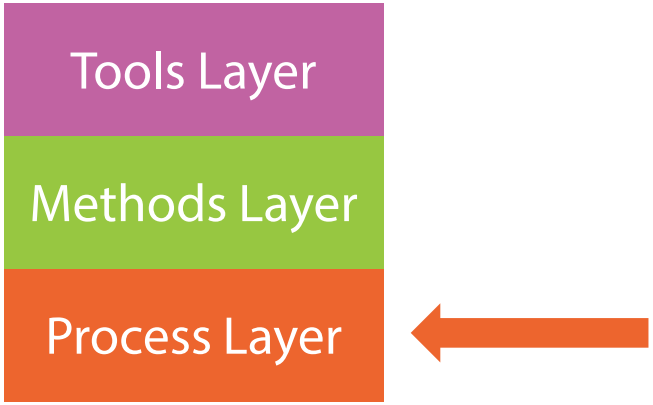
Software Processes

Mohamad Halabi
@mohamadhalabi

pluralsight
hardcore dev and IT training

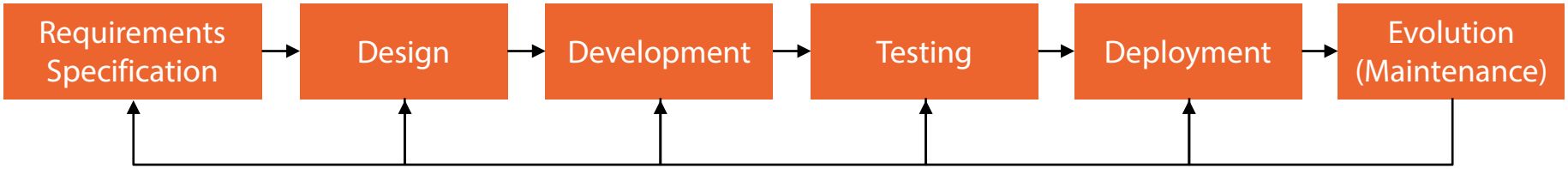
Introduction



- A process consists of generic activities: requirements specification, design, development, testing, deployment, evolution
- A process model indicates activities flow, organization, and artifacts level of detail
- Now let's discuss some of the most common process models

Waterfall (Linear)

- Systematic and sequential approach to the software lifecycle



- Even with feedback loops – changes are difficult to implement
- Customers have to clearly define all requirements at the start
- Customers will wait until the very end to see the first software workable version

Is It Used?

- **When requirements are clear upfront and changes are not expected**
- **However, still the model presents a one-shot workable version of the software**
  - Incremental delivery solves this

Iterative/Incremental

- Created in response to Waterfall model
- The essence of evolutionary models such as Agile and RUP
- What do iterative and incremental mean? How do they differ?

Incremental Delivery

- **Steps of incremental delivery**
  1. Requirements are assigned priority
  2. Increments are defined – each containing portion of the requirements
  3. Each increment then analyzed in detail
  4. Increment then goes through design, development, testing, deployment, and (possibly) evolution
- **Increments are time-boxed (fixed predefined maximum execution time)**
  - If time is not enough, requirements are shifted to later increments
  - If requirements are completed ahead of time, requirements from future increments can be withdrawn and implemented in current increment
- **Finished increments are integrated into the overall system for customers to see**

Iterative Development

- **Incremental delivery is a scheduling method to deliver requirements**
- **Iterative development is a method to refine the work done**
  - Refinement can be done on a specific increment
  - Refinement can also be done on a full process model (ex: Waterfall)
- **Iterative development does not mandate nor is it attached to incremental delivery (although frequently used together)**

Example: Incremental vs. Iterative

Incremental delivery

Increment 1

Increment 2

Full version

Iterative development

Iteration 1

Iteration 2

Final Iteration

Example: Incremental/Iterative Combined

Increment 1

Iteration 1

Increment 1

Iteration 2

Increment 1

Iteration 3

Full version

Increment 2

Iteration 1

Increment 2

Iteration 2

Increment 2

Iteration 3

Combined Iterative/Incremental

- Iterations and increments are essential parts of Agile methodologies (ex: Scrum with its Sprint)
- Unified Process is also built around iterative/incremental delivery

A prototype is an early example, model, or release of a product built to test a concept or process

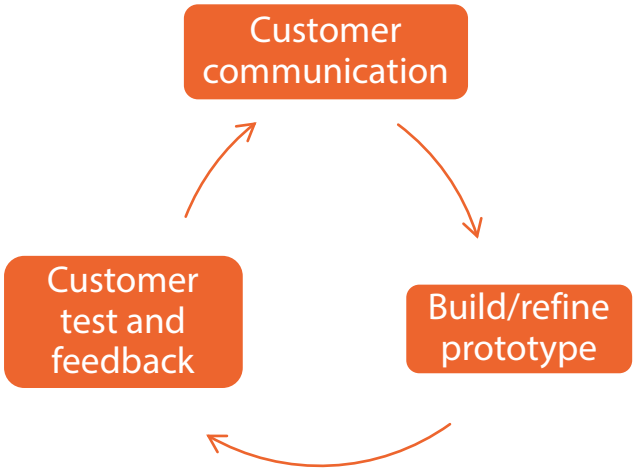
— **Wikipedia**

When Is a Prototype Used?

- **To derive customer requirements**
  - Prototypes derive feedback which helps clarify vague, conflicting, and unambiguous requirements
- **Mitigate technical and architectural risks**
  - Complex architectures are tested
  - New technologies are tried

Process Model vs. Technique

- **As a process model**



- **As a technique**
  - Used within a process model
  - RUP and Spiral both utilize prototyping technique
  - Known as “throwaway prototypes”

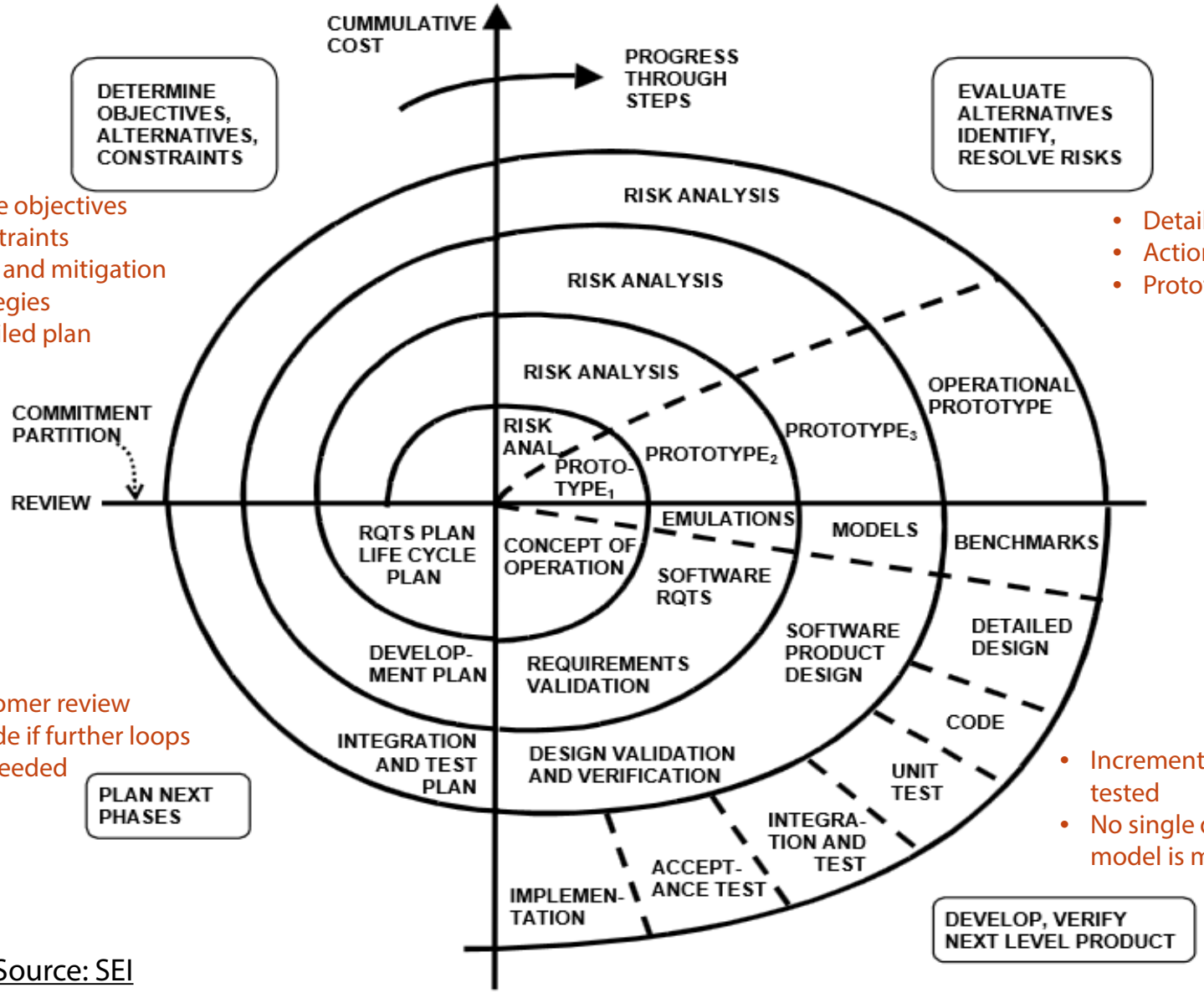
Usage

- **Prototype process models are not that common**
  - Unlikely to have 100% unclear requirements or 100% risky architectural use cases
- **Prototypes as a technique are more common to mitigate requirements and architectural/technical risks**

Spiral

- Spiral model is risk-driven
- Combines incremental/iterative advantages and systematic/controlled Waterfall approach
- Spiral adopts a "*cyclic approach for incrementally growing a system's degree of definition and implementation while decreasing its degree of risk*"…Barry Boehm

Spiral Diagram



- Phase objectives
- Constraints
- Risks and mitigation strategies
- Detailed plan

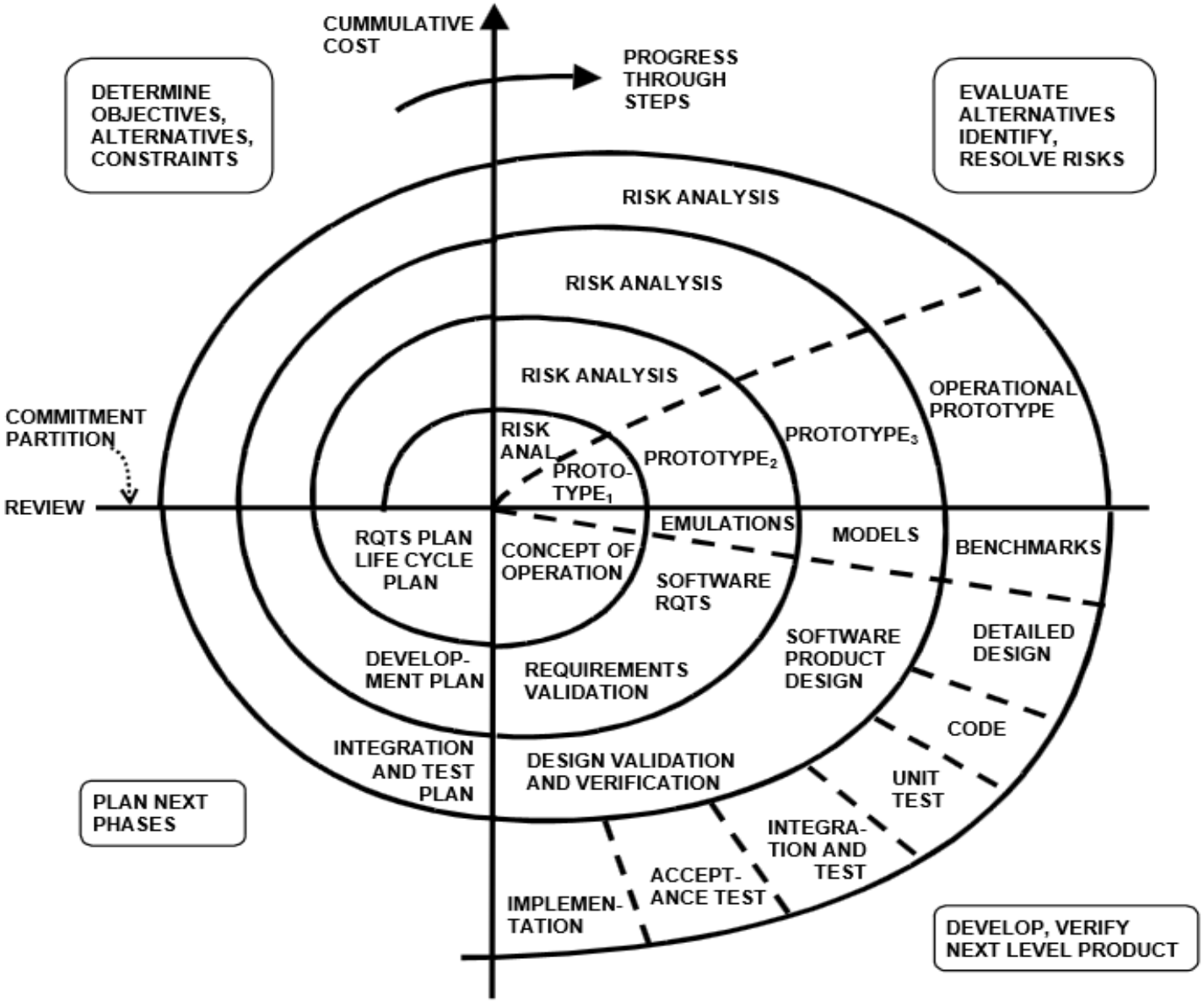
- Detailed analysis for risks
- Action to reduce risks
- Prototypes are built

- Customer review
- Decide if further loops are needed

- Increment is developed and tested
- No single development model is mandated

Source: SEI

Reading the Spiral Diagram



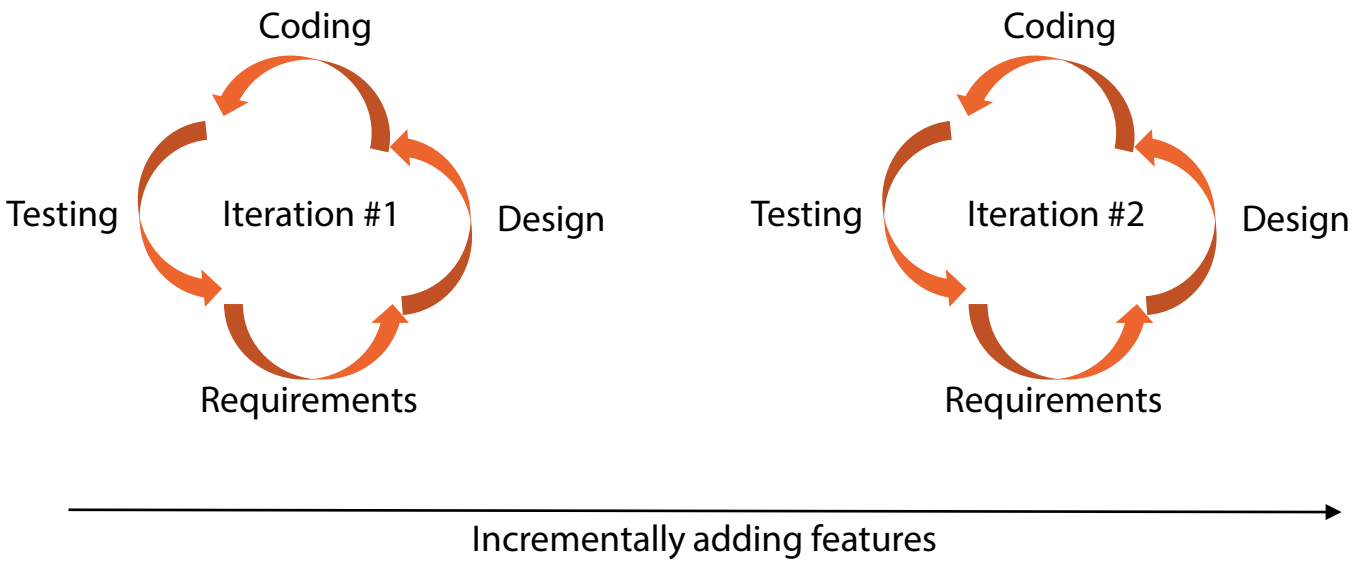
Properties

- **Pros**
  - Suitable for large and complex projects (risk driven)
  - Adopts incremental delivery
  - Uses prototyping technique to mitigate risks
- **Cons**
  - Complex to manage
  - Incurs high cost (risk analysis and prototyping)
- **Therefore not suitable for small to medium size projects and/or where Agility is important**

Agile

- **Based on incremental/iterative delivery**
- **Promise of Agile:**
  - Create reliable software
  - Quickly
  - Eliminate waste and overhead

Agile Activities



- **No emphasis on any specific activity in any given iteration**
  - Focus is on quick delivery of reliable software with waste eliminated
- **RUP (also iterative/incremental) does emphasize specific activity in each phase (discussed next)**

Agile Manifesto

- Created by a group of Agile leaders
- The philosophies that underlie Agile methods
- Manifesto has values and derived principles

Agile Manifesto Values

Value	Explanation
Individuals and interactions over processes and tools	• Strong collaborating team > processes and tools • Coherent team > individual gurus • Put more effort in building coherent teams
Working software over comprehensive documentation	• Documentation is important • Too much documentation is costly and documents become outdated • Generate just enough high-level structure/dynamics documents
Customer collaboration over contract negotiation	• Continuous customer feedback is essential • Customers work closely with team • Feedback derives future iterations

Respond to change over following a plan	• Plans must be flexible to change
	• Changes can and will happen
	• Multi-level planning:
	  • Weekly detailed plans
	  • Iteration plan
	  • Overall plan

Agile Manifesto Values

More valued items	Valued – but less than left items
Individuals and interactions	over processes and tools
Working software	over comprehensive documentation
Customer collaboration	over contract negotiation
Respond to change	over following a plan

- **While there is value in the items on the right, Agile values the items on the left more**
- **Qs: How much planning we do? To what extent we follow a process? And how much analysis and design documentation and models we generate?**
- **A: Do just enough planning and documentation, and follow just enough of a process that helps you achieve the Agile values listed on the left**

Agile Manifesto Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale
4. Business people and developers must work together daily throughout the project
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation

Agile Manifesto Principles

7. Working software is the primary measure of progress
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely
9. Continuous attention to technical excellence and good design enhances agility
10. Simplicity—the art of maximizing the amount of work not done— is essential
11. The best architectures, requirements, and designs emerge from self-organizing teams
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

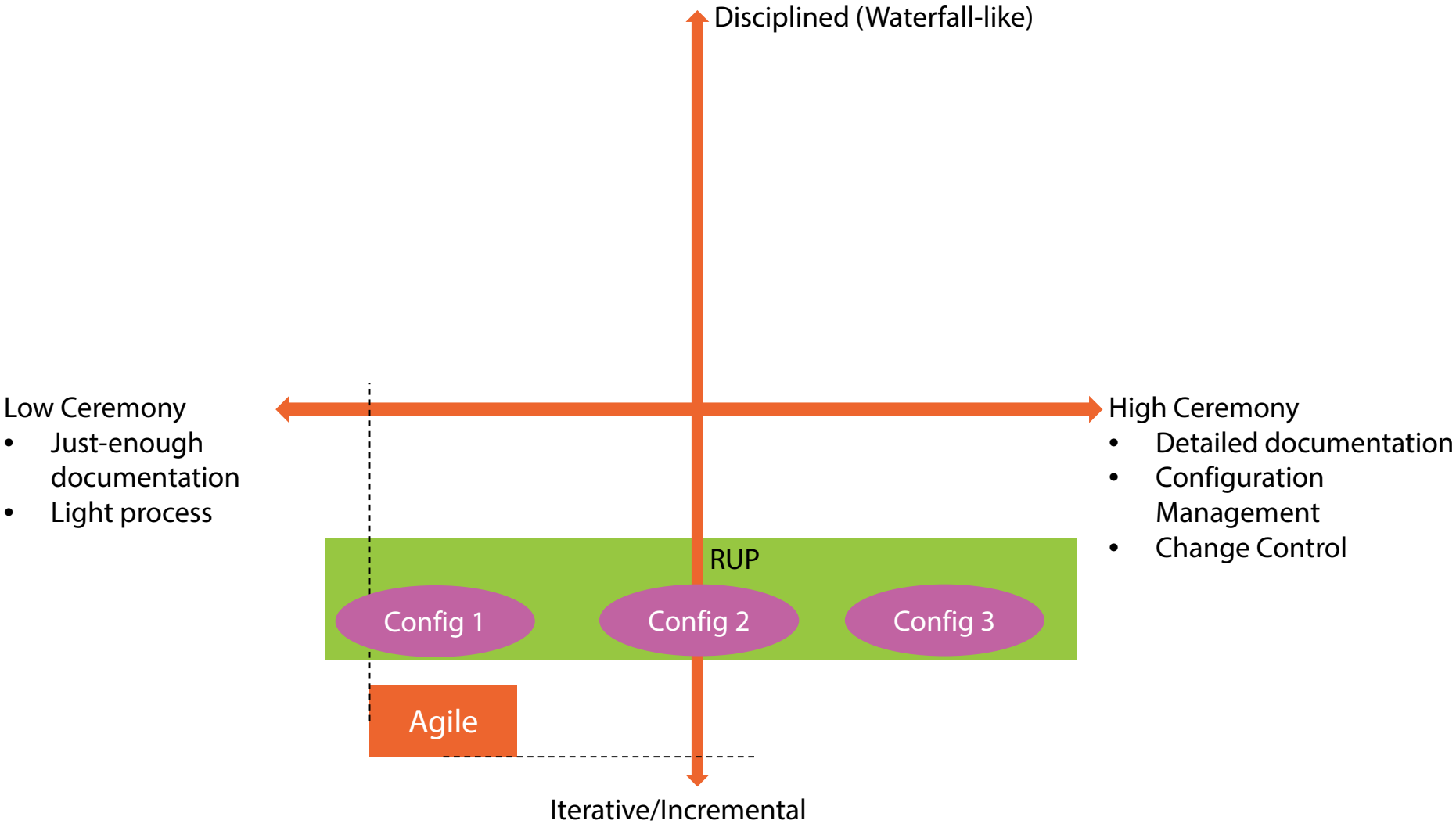
Agile Methodologies

- **Agile methodologies share Manifesto values and principles – although they might differ in activities**
- **Examples:**
  - Scrum
  - XP (Extreme Programming)
  - Lean Software Development
  - Feature-Driven Development
  - Dynamic System Development

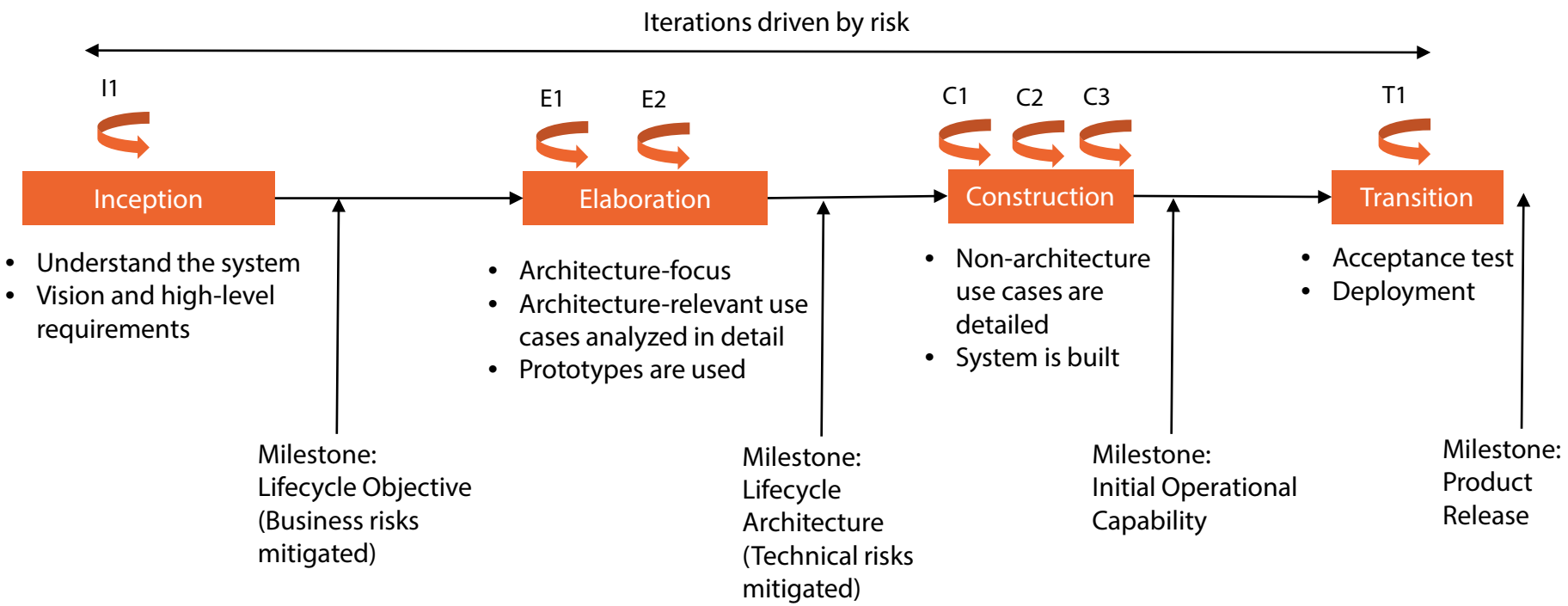
RUP

- **Rational Unified Process (RUP)**
  - Based on incremental/iterative delivery
  - Driven by risk
  - Development approach is use-case driven and architecture centric
- **RUP is also a process framework**
  - Process Configurations are created via process customization and authoring
  - Configurations support:
    - Different team sizes (small, medium, large)
    - Disciplined or less formal development methods

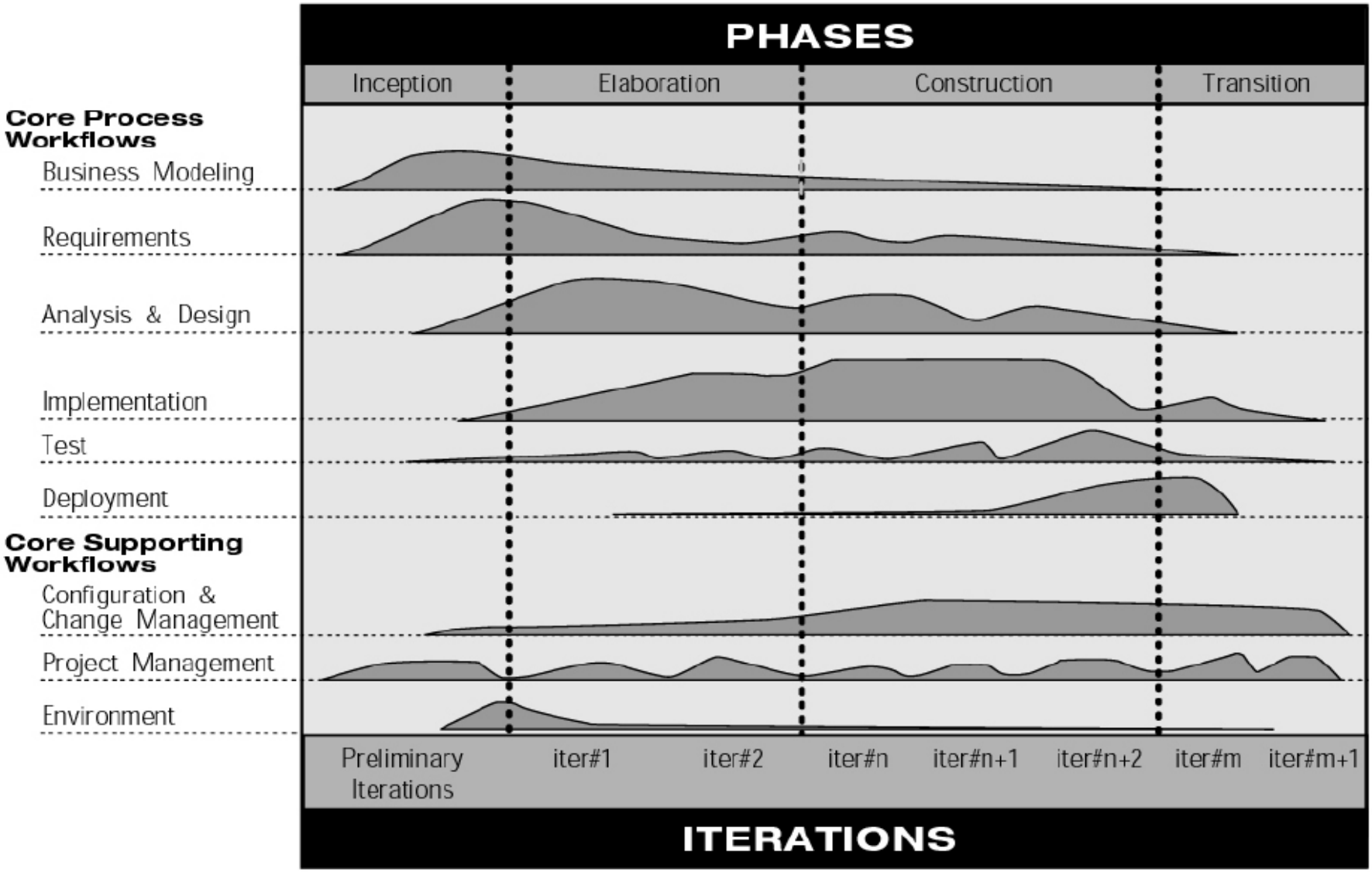
RUP Configurations



Phases and Milestones



Disciplines



Source: IBM

Specialized Models

- **These models are used for certain kinds of projects**
  - They have the characteristics of one of the discussed models
  - …But they have special techniques/methods for a certain type of problem
- **Examples:**
  - Component-Based Development
  - Formal Methods
  - Aspect-Oriented Development

Component-Based Development

- **Development approach based on reuse of software components**
- **Components can be:**
  - Commercial off-the-shelf (COTS) developed by vendors and have well-defined interfaces
  - Custom components (with reusability in mind)
- **Architecture is centered around components integration**
- **Testing is based on exchange of messages between components**

Formal Methods

- **Indicate the use of formal mathematical specifications**
  - Requirements are represented by mathematical equations with precisely defined vocabulary, syntax, and semantics
  - Requirements verification, design, and code generation happen automatically
  - Test cases are automatically derived from specifications
- **Pros**
  - Formal Methods solve the requirements ambiguity problems of non-formal methods
  - Suitable for safety-critical systems
- **Cons**
  - Time consuming and expensive
  - Knowledge is not widely available

Aspect-Oriented Development

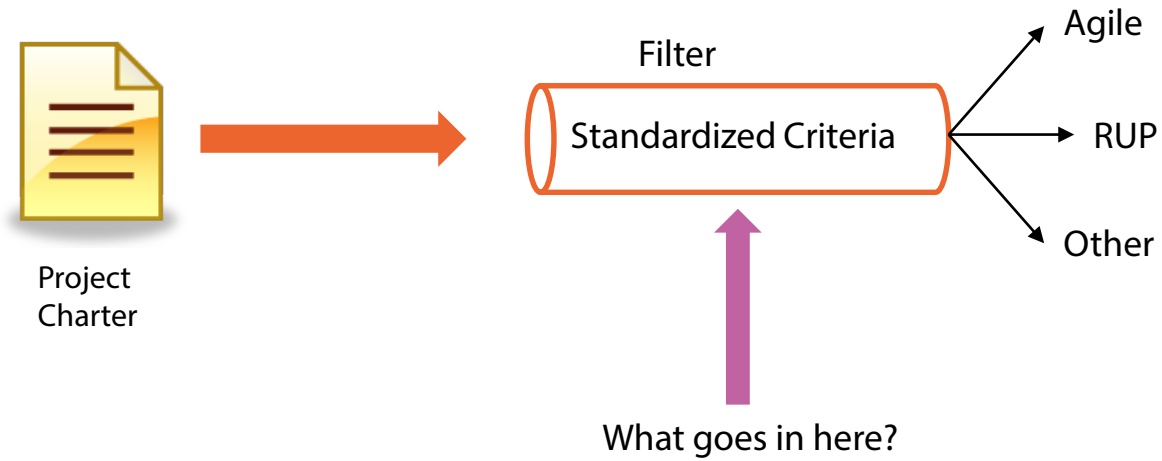
- **A cross-cutting concern is a part or aspect that spans the entire architecture**
  - A functionality that is required at various functions, components, and services
  - Mostly represent non-functional requirements:
    - Security
    - Logging
    - Caching
    - Transaction Management
- **Aspect-Oriented Development defines these concerns as Aspects**
  - Provides a methodology to specify, design, and develop these Aspects
  - Since Aspects are isolated (i.e. developed and maintained separately), they can be composed and reused more easily

Which Process Model to Use?

- **There is no one-size-fits-all solution. No absolute right or wrong!**
- **The right process is whatever process that helps a team deliver:**
  - The right software
  - With the appropriate level of quality
  - Within planned time and budget
- **There are recommendations based on experience and lessons learned**

The Process Filter

- **Teams can select from multiple process models**
  - Each model is suitable for certain scenarios and/or project types



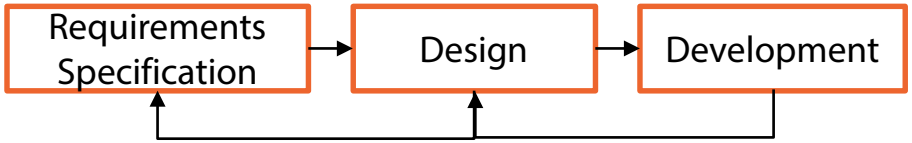
Sample Evaluation Method

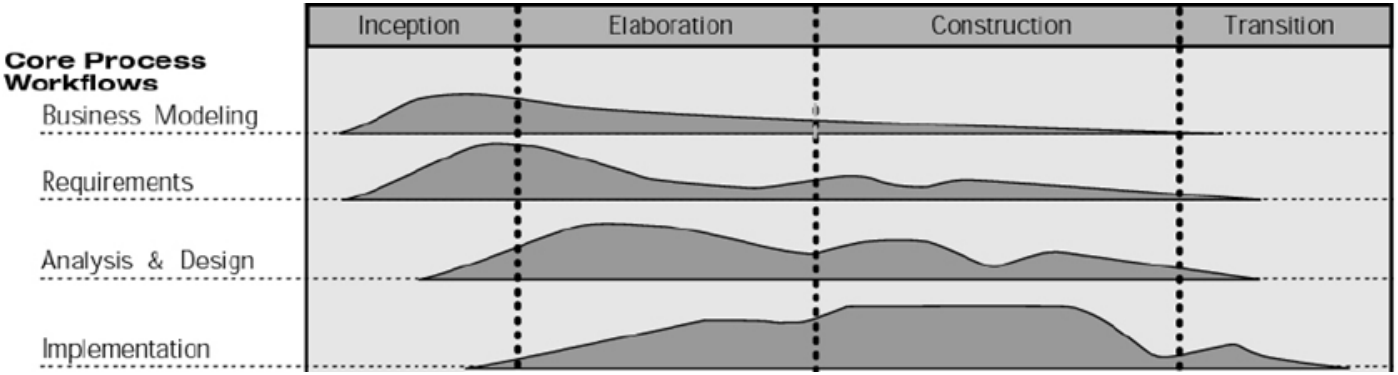
Requirements Changing Rate	Architecture Complexity	Project Size (man days, team size)	
High	High/medium	Medium/large	→ RUP: low ceremony
Low	High/medium	Medium/large	→ RUP: medium ceremony
High	High/medium	Small	→ Agile
High	Low	Small/Medium	→ Agile
High	Low	Large	→ RUP: low ceremony

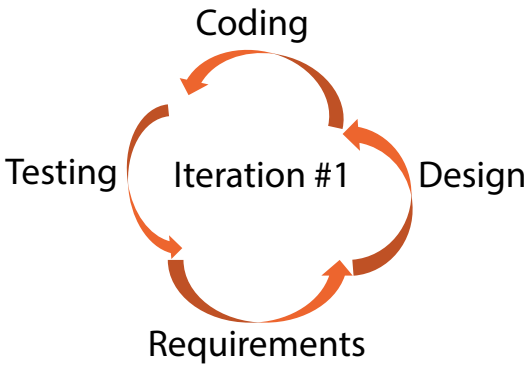
Does the Process Model Choice Affect Future Modules?

- **Recall:**
  - Generic process activities (requirements, design, development, testing, deployment, evolution) are available in all process models
  - Process models dictate activities flow, organization, roles/responsibilities, and artifacts granularity
- **Therefore, the discussion about activities (ex: requirements) and methods (ex: OOAD) in future modules is independent of any specific process model**
  - Order of activities, their flow, roles/responsibilities, and artifacts granularity will differ; concepts will not

Example: Requirements Engineering





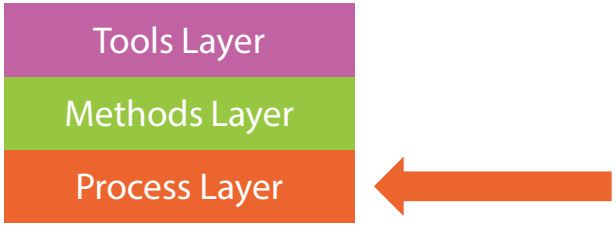


- **Takeaway: future modules are independent on any process model**
  - Concepts are the same; process model choice will affect flow and emphasis

More Resources

- **Search "Agile" in Pluralsight library**
- ***The Rational Unified Process : An Introduction* by Philippe Kruchten**

Summary

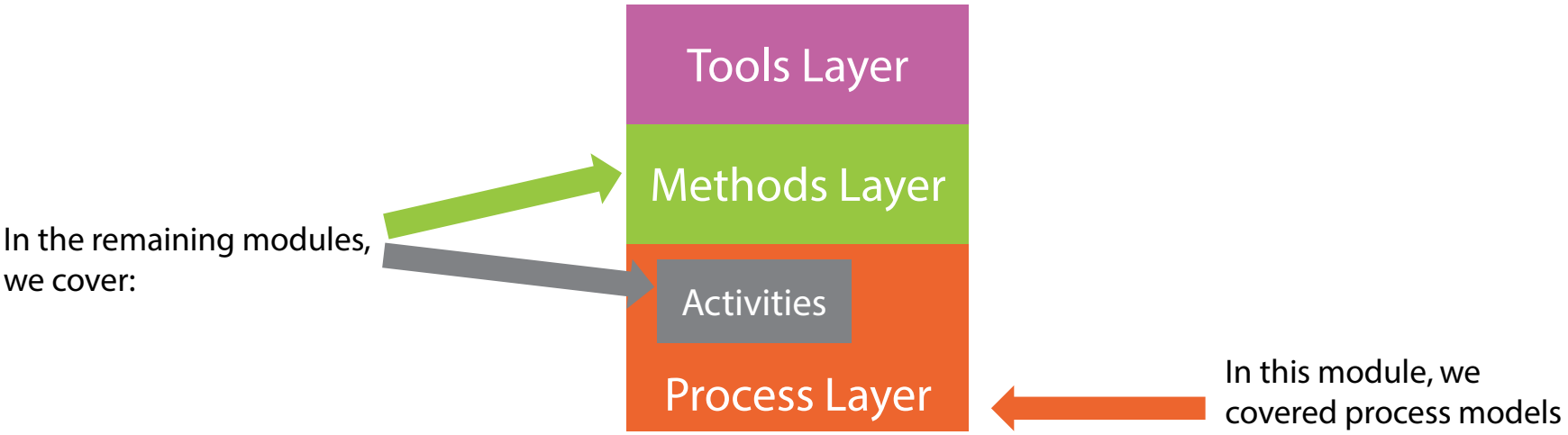


- **Waterfall: systematic and sequential approach**
- **Iterative/Incremental: staged delivery and refined development**
- **Prototyping: early release to test concepts and/or mitigate risks**
- **Spiral: driven by risk analysis**
  - Each iteration is a Spiral loop and represents a phase
  - Despite its advantages, it's complex and incurs high cost

Summary

- **Agile: quick delivery with minimum overhead**
  - Agile Manifesto defines a set of values and principles
  - Values team work, customer collaboration, continuous delivery, and change response
- **RUP: risk/use-case driven and architecture-centric**
  - It's also a process framework
  - Consists of four phases (inception, elaboration, construction, transition)
- **Specialized models: techniques suitable for certain problems**
  - Component-Based Development
  - Formal Methods
  - Aspect-Oriented Development

What's Next?



- **Next module: Requirements Engineering**