Chapter 2 – Software Processes

Topics covered

- ✧ Software process models
- ✧ Process activities
- ✧ Planned vs Agile processes
- ✧ Process Models:
  - Waterfall
  - Incremental & Iterative
  - Integration and configuration

The software process

- A structured set of activities required to develop a software system.
- Many different software processes but all involve:
  - Specification – defining what the system should do;
  - Design and implementation – defining the organization of the system and implementing the system;
  - Validation – checking that it does what the customer wants;
  - Evolution – changing the system in response to changing customer needs.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

The software process

- A structured set of activities required to develop a software system.
- Many different software processes but all involve:
  - Specification – defining what the system should do;
  - Design and implementation – defining the organization of the system and implementing the system;
  - Validation – checking that it does what the customer wants;
  - Evolution – changing the system in response to changing customer needs.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

The software process

✧ A structured set of activities required to develop a software system.

✧ Many different software processes but all involve:

- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

✧ A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

The software process

- A structured set of activities required to develop a software system.
- Many different software processes but all involve:
  - Specification – defining what the system should do;
  - Design and implementation – defining the organization of the system and implementing the system;
  - Validation – checking that it does what the customer wants;
  - Evolution – changing the system in response to changing customer needs.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

The software process

- A structured set of activities required to develop a software system.
- Many different software processes but all involve:
  - Specification – defining what the system should do;
  - Design and implementation – defining the organization of the system and implementing the system;
  - Validation – checking that it does what the customer wants;
  - Evolution – changing the system in response to changing customer needs.
- A software process model is an abstract representation of a process. It presents a **description of a process** from some particular perspective.

Software process descriptions

- When we describe and discuss processes, we usually talk about the activities in these processes such as specifying a data model, designing a user interface, etc. and the ordering of these activities.
- Process descriptions may also include:
  - Products, which are the outcomes of a process activity;
  - Roles, which reflect the responsibilities of the people involved in the process;
  - Pre- and post-conditions, which are statements that are true before and after a process activity has been enacted or a product produced.

Software process descriptions

- When we **describe** and discuss processes, we usually talk about the activities in these processes such as specifying a data model, designing a user interface, etc. and the ordering of these activities.
- Process descriptions may also include:
  - Products, which are the outcomes of a process activity;
  - Roles, which reflect the responsibilities of the people involved in the process;
  - Pre- and post-conditions, which are statements that are true before and after a process activity has been enacted or a product produced.

Software process descriptions

- When we describe and discuss processes, we usually talk about the **activities** in these processes **such as specifying a data model, designing a user interface**, etc. and the ordering of these activities.
- Process descriptions may also include:
  - Products, which are the outcomes of a process activity;
  - Roles, which reflect the responsibilities of the people involved in the process;
  - Pre- and post-conditions, which are statements that are true before and after a process activity has been enacted or a product produced.

Software process descriptions

✧ When we describe and discuss processes, we usually talk about the activities in these processes such as specifying a data model, designing a user interface, etc. and the ordering of these activities.

✧ Process descriptions may also include:

- **Products**, which are the outcomes of a process activity;
- **Roles**, which reflect the responsibilities of the people involved in the process;
- **Pre- and post-conditions**, which are statements that are true before and after a process activity has been enacted or a product produced.

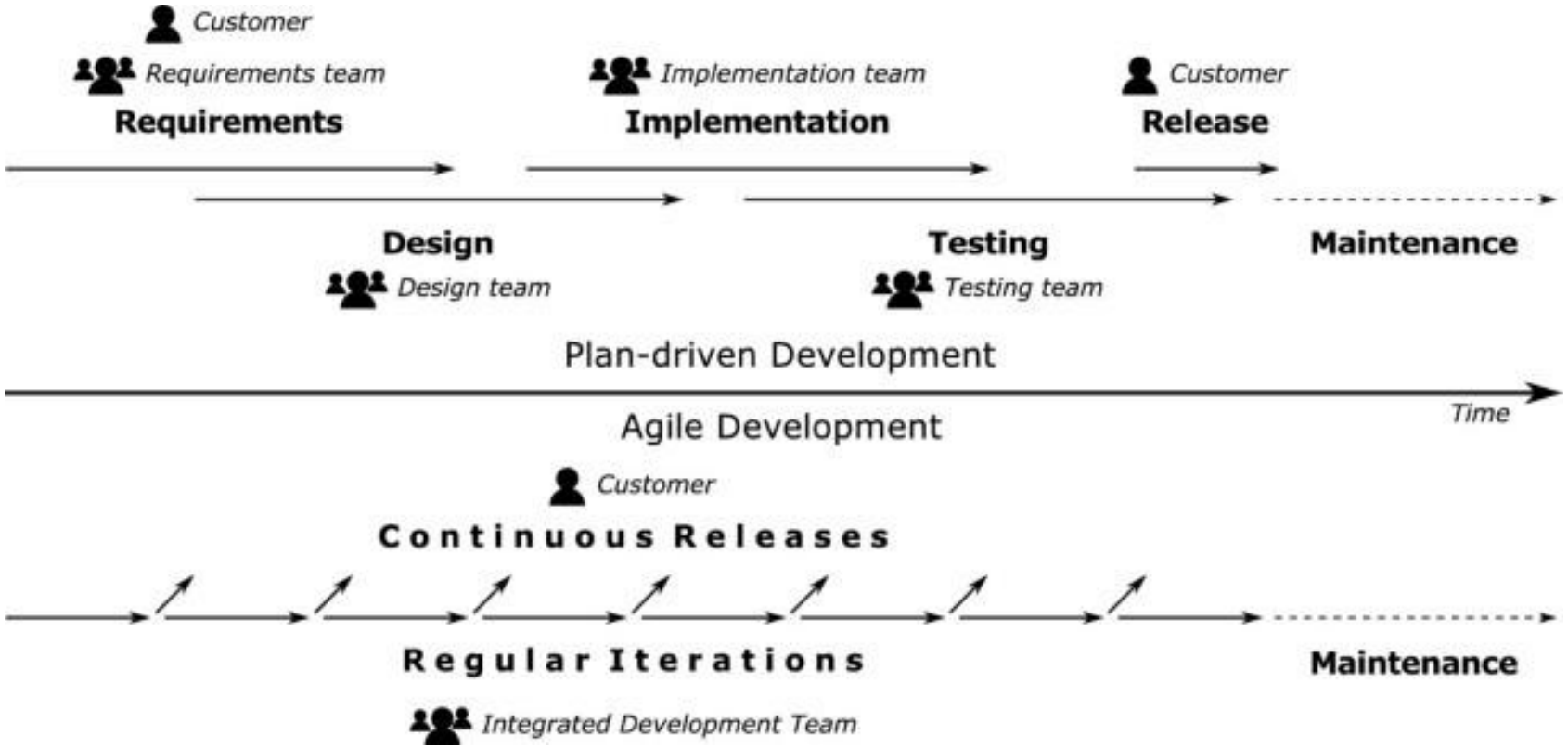
Plan-driven and **agile processes**

- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

Plan-driven and agile processes

- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

Plan-driven and **agile processes**

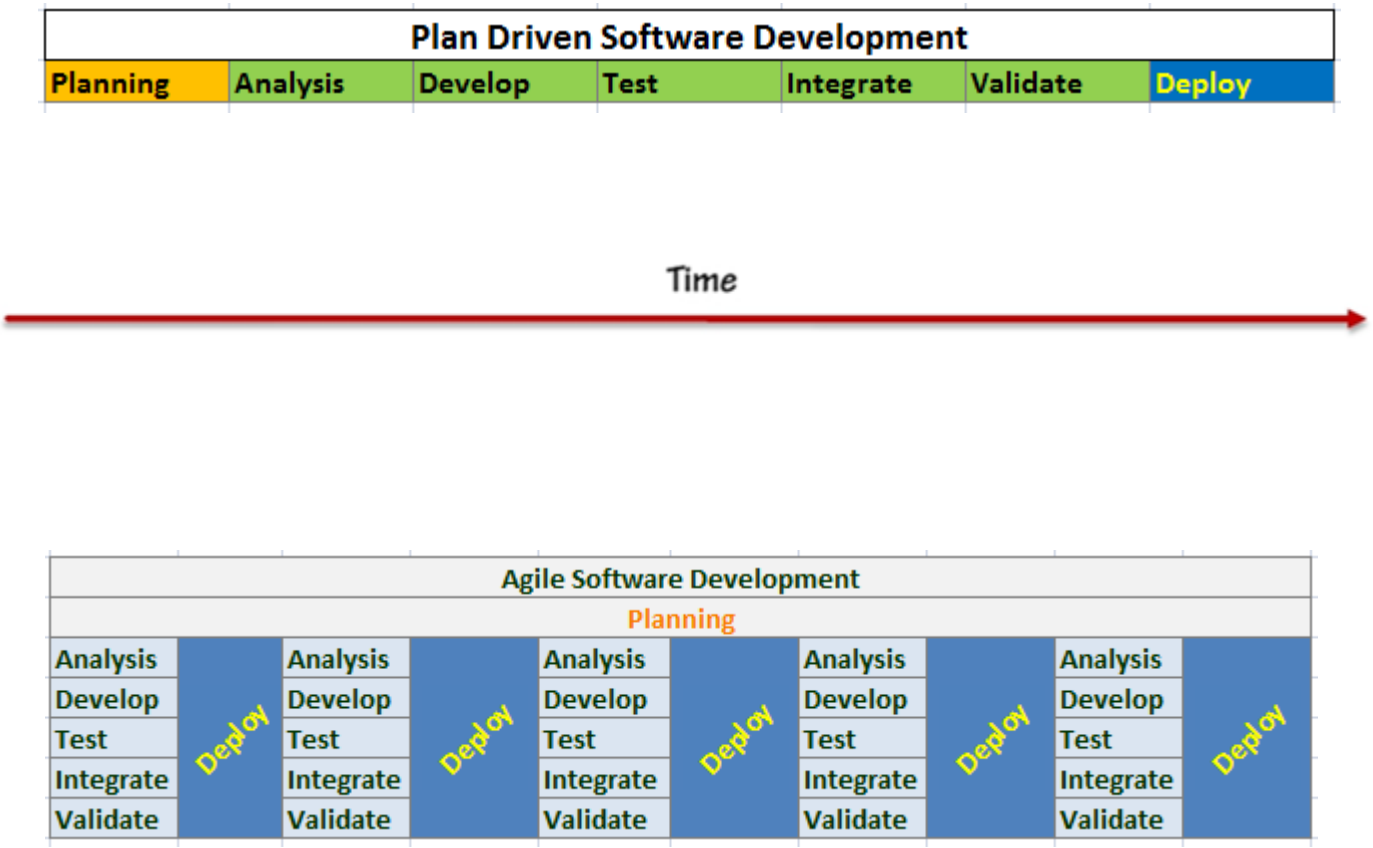


source: https://link.springer.com/article/10.1186/1687-3963-2013-15

Plan-driven and agile processes

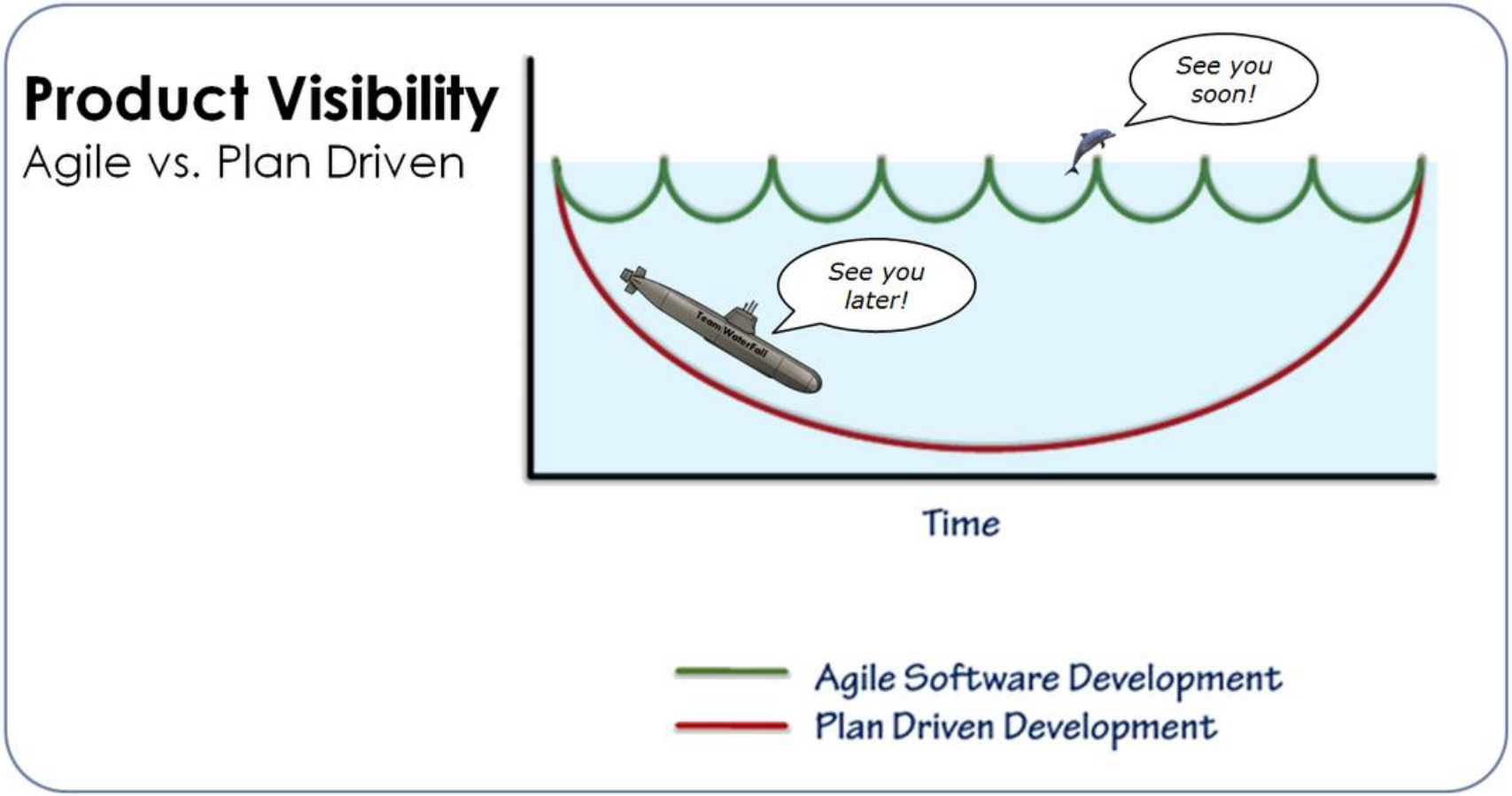
- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

Plan-driven and agile processes



source: https://link.springer.com/article/10.1186/1687-3963-2013-15

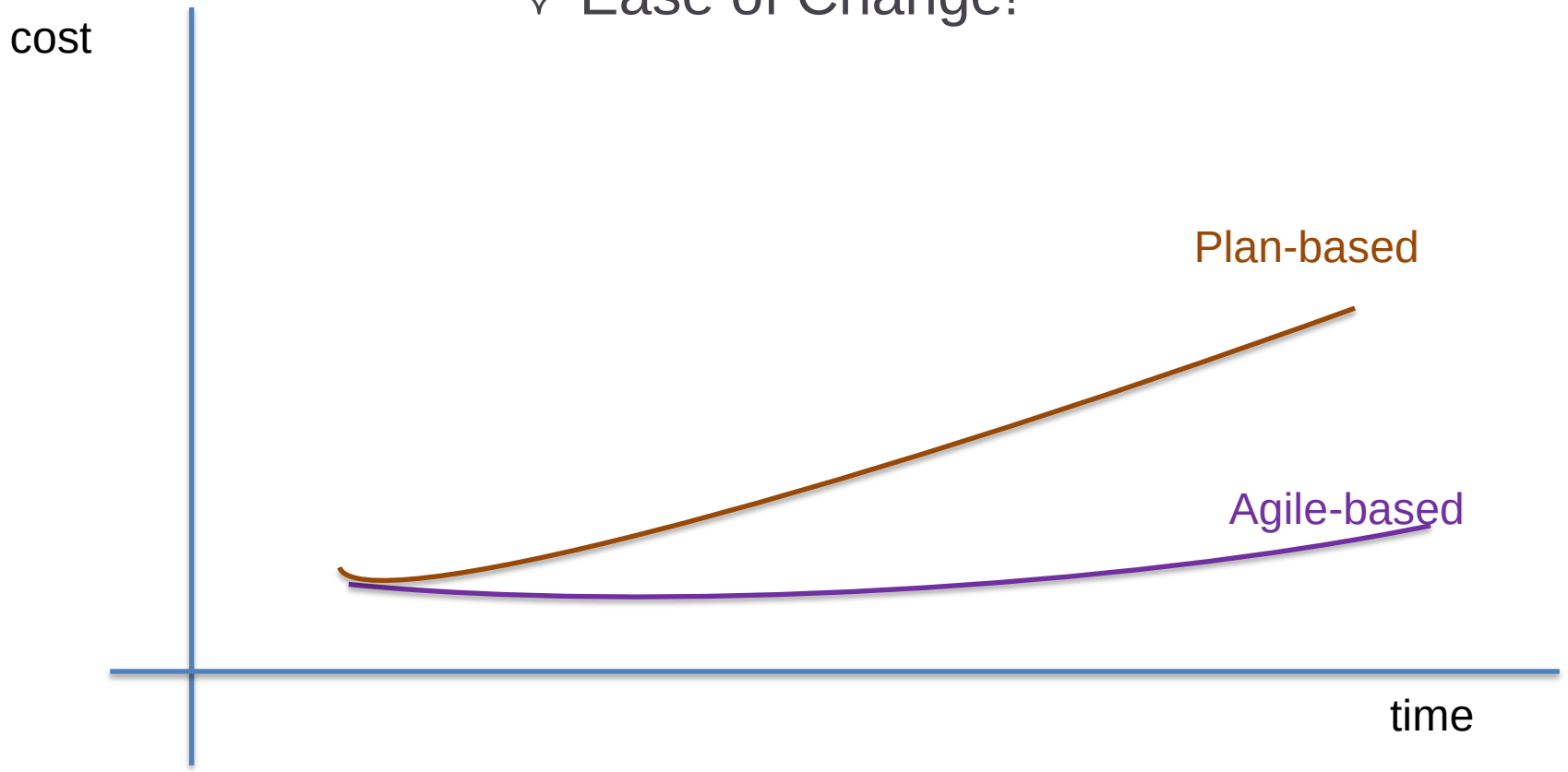
Plan-driven and agile processes



source: https://twitter.com/MarcHemerik/status/657516180442320897?s=20

Plan-driven and **agile processes**

✧ Ease of Change!



Plan-driven and agile processes

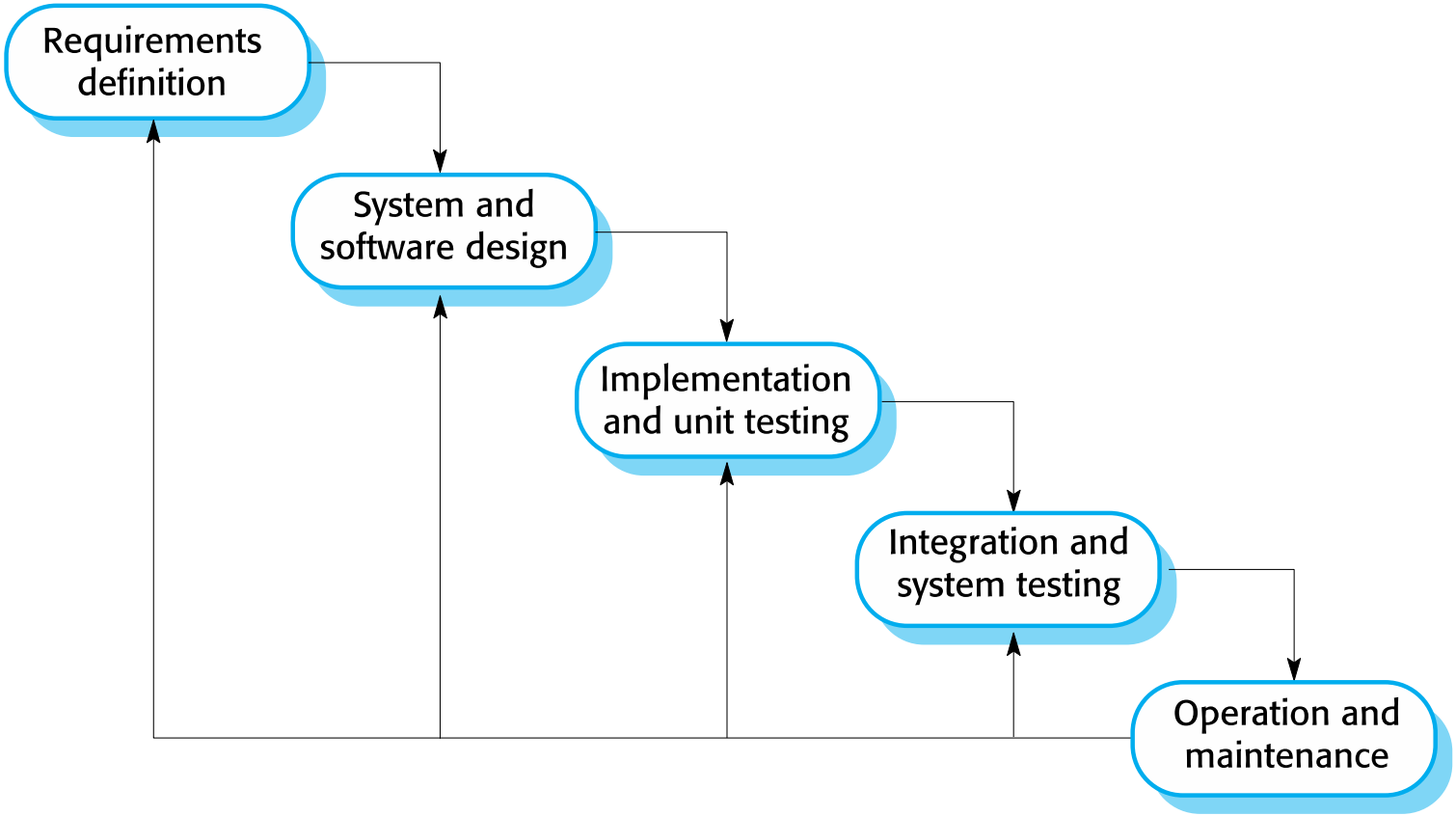
- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.
- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

Software process models

Software process models

- The waterfall model
  - Plan-driven model. Separate and distinct phases of specification and development.
- Incremental development
  - Specification, development and validation are interleaved. May be plan-driven or agile.
- Integration and configuration
  - The system is assembled from existing configurable components. May be plan-driven or agile.
- In practice, most large systems are developed using a process that incorporates elements from all of these models.

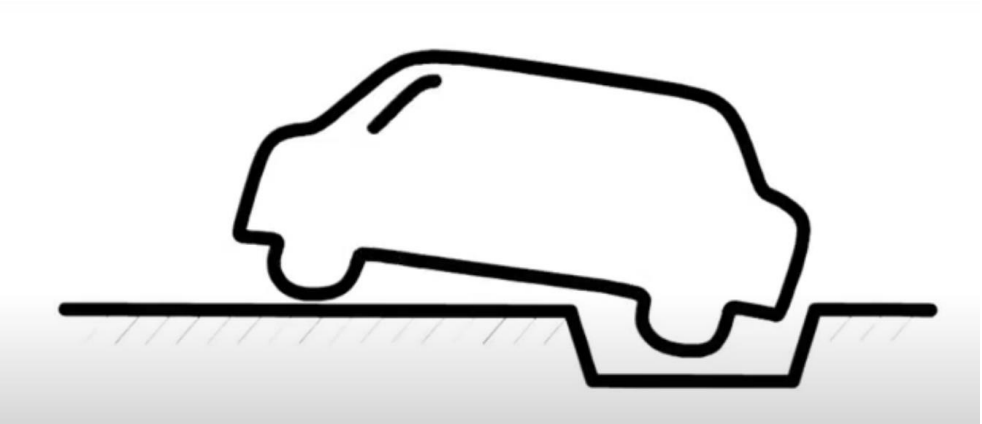
The waterfall model



Waterfall model phases

- There are separate identified phases in the waterfall model:
  - Requirements analysis and definition
  - System and software design
  - Implementation and unit testing
  - Integration and system testing
  - Operation and maintenance
- Example
  - Building a house!
    - Construction
  - Creating a website

Pothole example

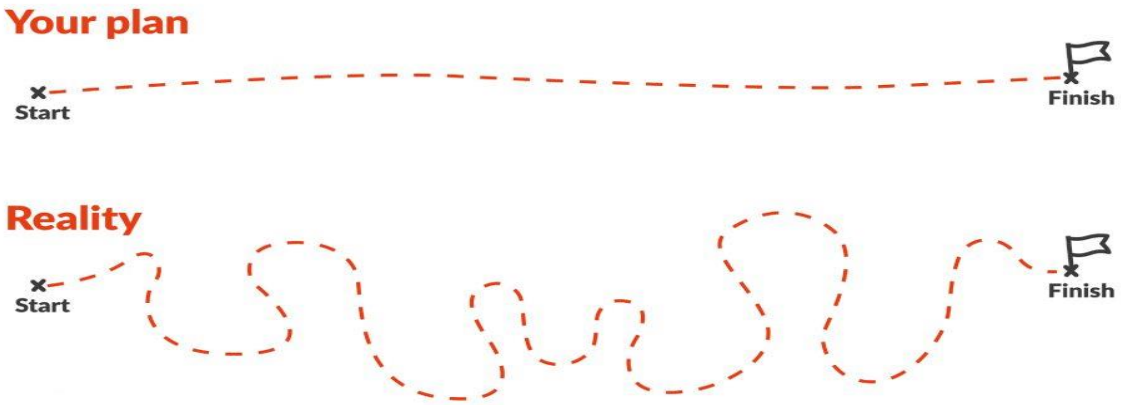


Waterfall model problems

- Inflexible partitioning of the project into distinct phases.
  - Difficult to respond to changing customer requirements
- Feedbacks come very late and changes can be expensive

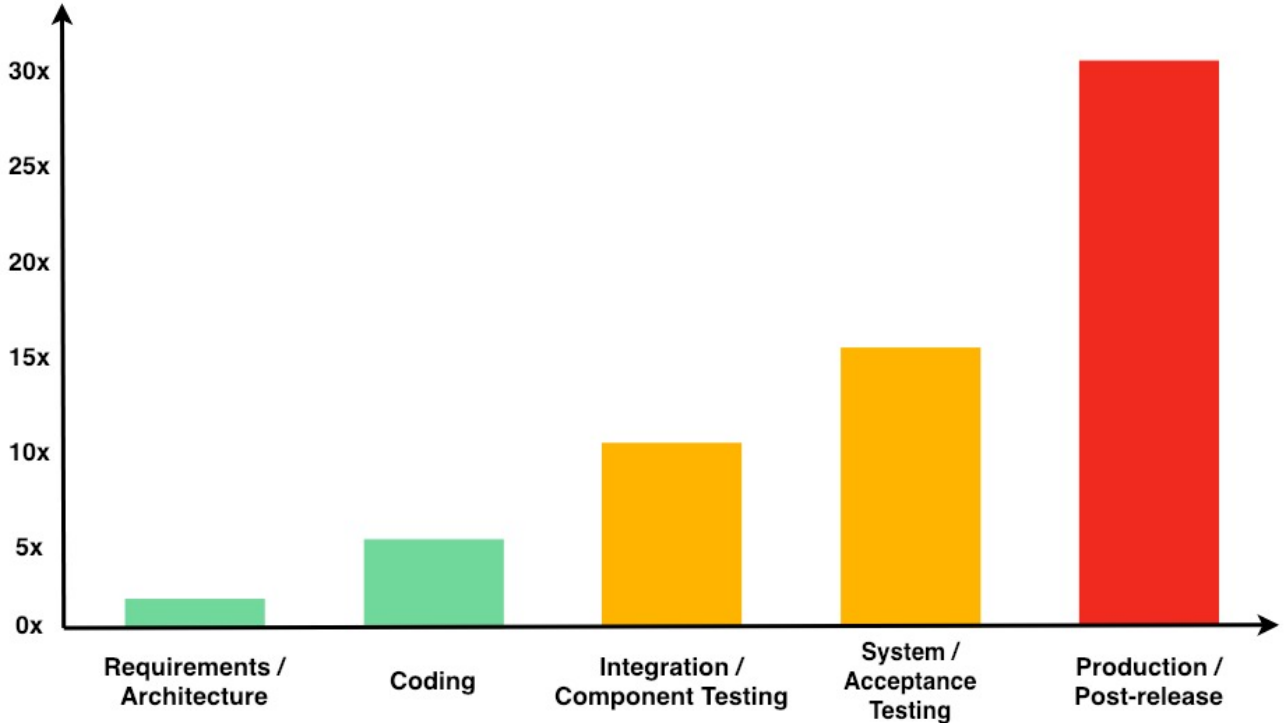
Inflexible project partitioning

- Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing **customer requirements**.
  - Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.
  - Few business systems have stable requirements.



Expensive change requests

- Feedbacks come very late and changes can be expensive



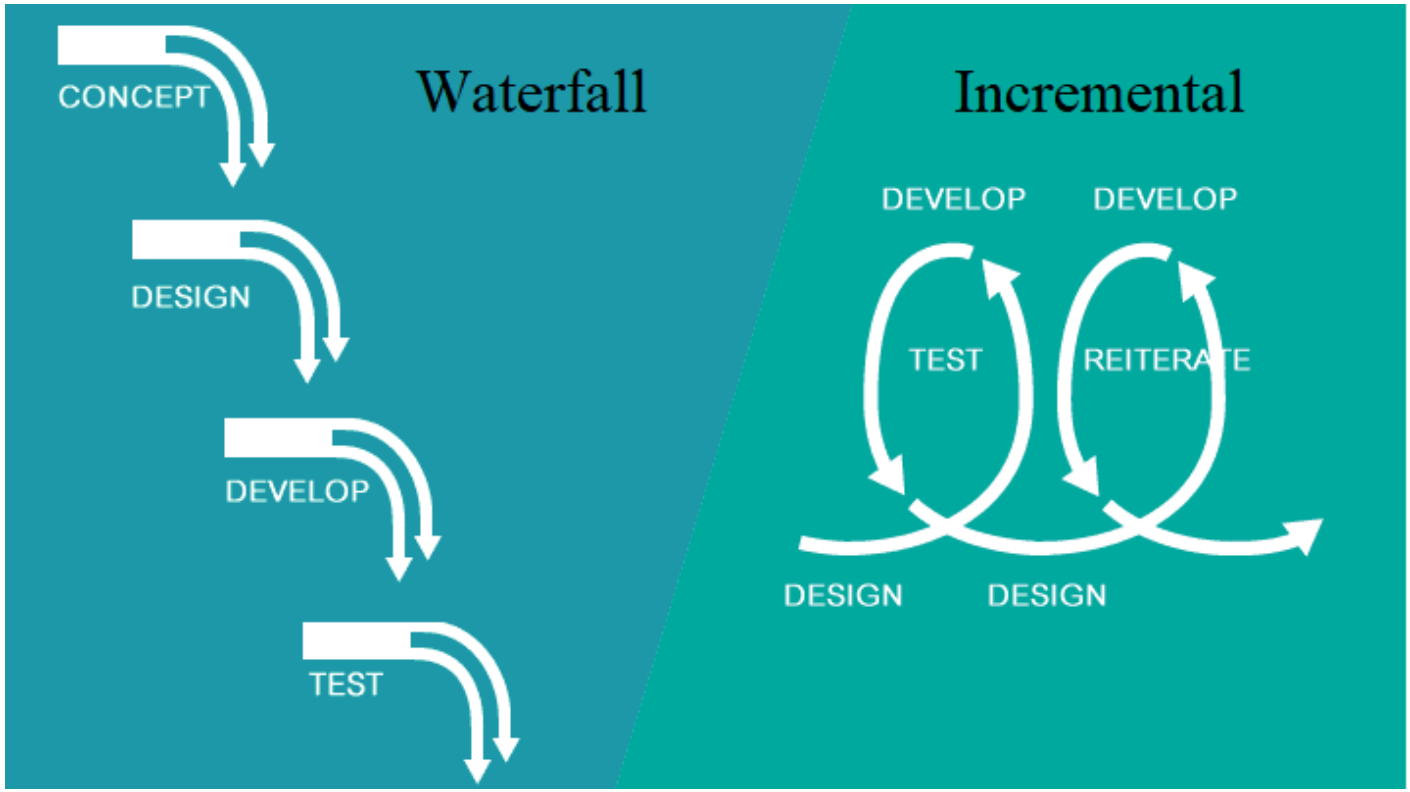
Waterfall model summary

- ✧ Cons:
  - Inflexible partitioning of the project into distinct phases.
  - Difficult to respond to changing customer requirements
  - Feedbacks come very late and changes can be expensive
- ✧ Pros:
  - Easily manageable process
  - The most effective method, if requirements are known
  - Extensive documentation
- ✧ Is it used in:
  - Projects where the requirements are well understood.
    - Clearly definable projects
  - Safety critical projects that need extensive documents

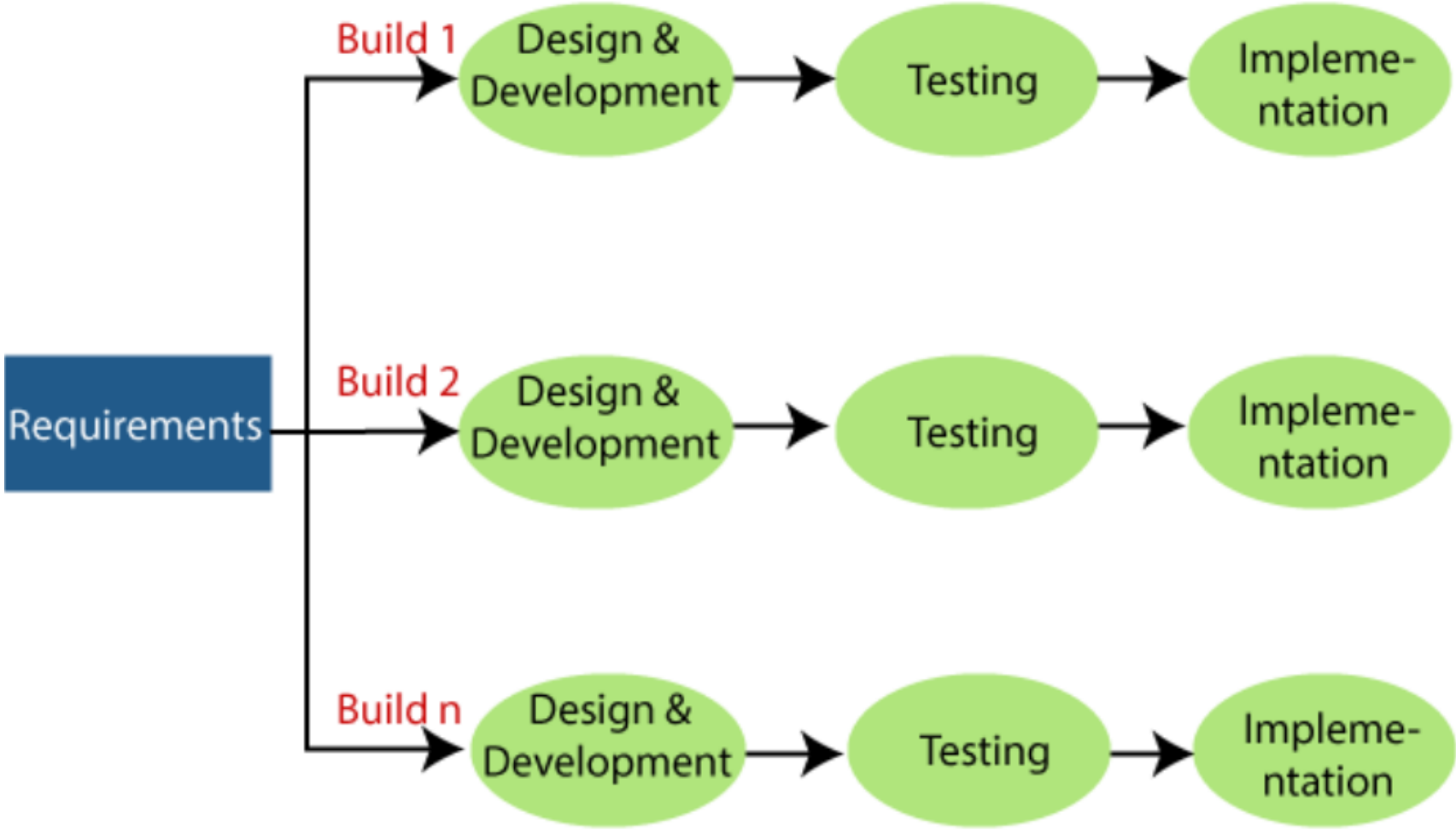
One more thing!

- The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.
  - In those circumstances, the plan-driven nature of the waterfall model helps coordinate the work.

Incremental Model



Incremental development



Incremental development

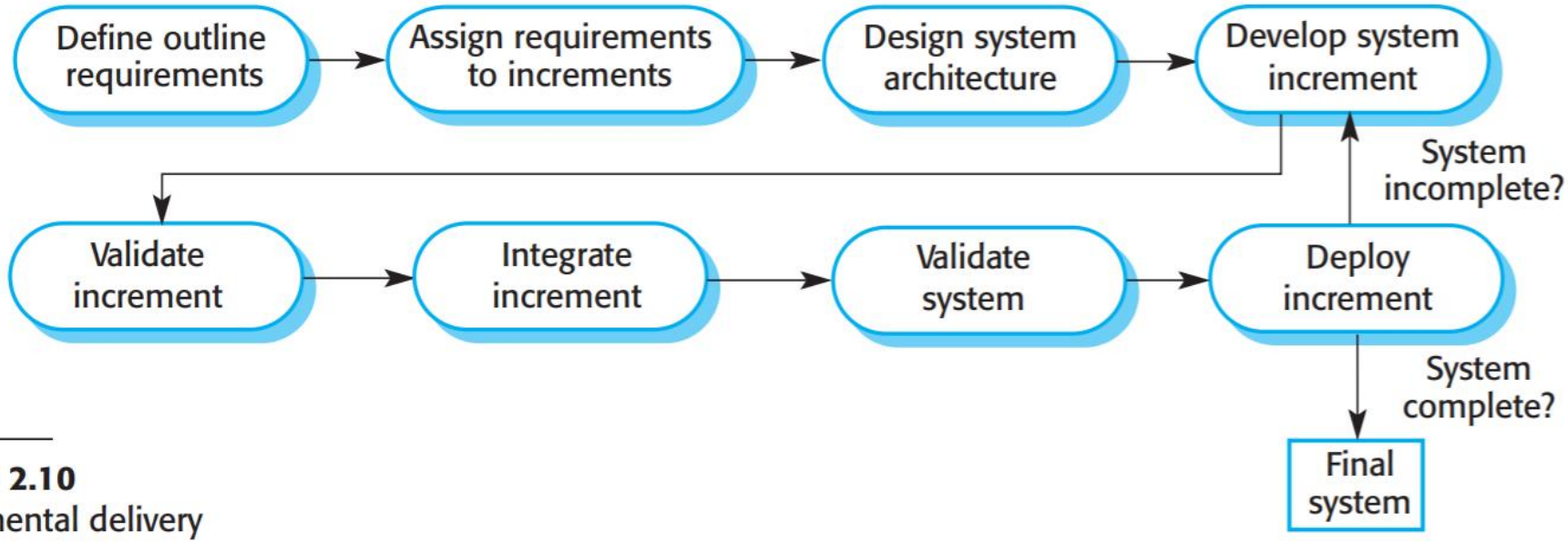


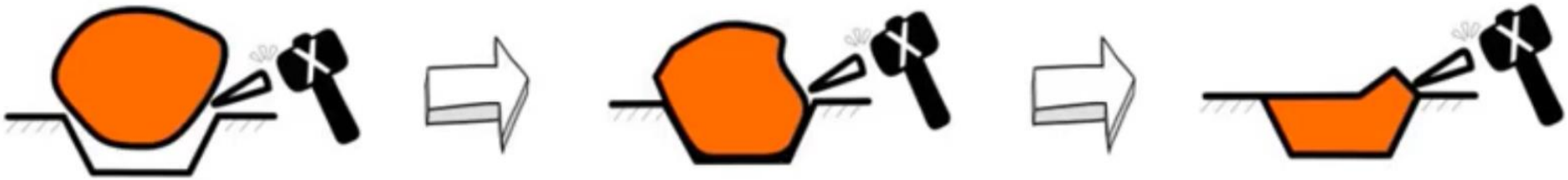
Figure 2.10
Incremental delivery

Pothole example!

✧ Planned incremental

Pothole example!

✧ Iterative approach!



Pothole example!

✧ Agile incremental (Incremental & Iterative approach!)

Incremental development benefits

- More rapid delivery and deployment of useful software to the customer is possible.
  - Customers are able to use and gain value from the software earlier than is possible with a waterfall process.
- Early increments act as a prototype to help
  - Elicit requirements for later increments
  - Get feedback on system performance
  - Final product better matches true customer needs
    - Lower risk of overall project failure
    - Job satisfaction is increased for developers who can see early results of their work
      - Can see how much has been implemented

Incremental development problems

- ✧ The process is not visible.
  - Can be harder to plan and control than waterfall development
  - Managers need regular deliverables to measure progress.
    - If systems are developed quickly, it is not cost-effective to produce documents that reflect every version of the system.
- ✧ Can be more expensive than waterfall development
  - May require more experienced staff
  - Incorporating further software changes becomes increasingly difficult and costly.
- ✧ System architecture must be adaptive to change
  - System structure tends to degrade as new increments are added.

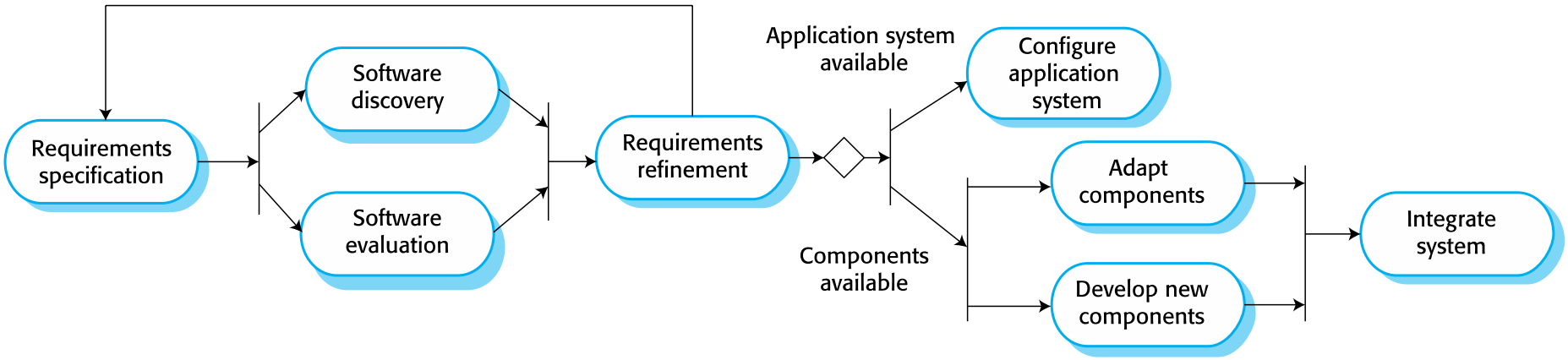
Integration and configuration

- Based on software reuse where systems are integrated from existing components or application systems (sometimes called COTS -Commercial-off-the-shelf) systems).
- Reused elements may be configured to adapt their behaviour and functionality to a user's requirements
- Reuse is now the standard approach for building many types of business system
  - Reuse covered in more depth in Chapter 15.

Types of reusable software

- Stand-alone application systems (sometimes called COTS) that are configured for use in a particular environment.
- Collections of objects that are developed as a package to be integrated with a component framework such as .NET or J2EE.
- Web services that are developed according to service standards and which are available for remote invocation.

Reuse-oriented software engineering



Key process stages

- Requirements specification
- Software discovery and evaluation
- Requirements refinement
- Application system configuration
- Component adaptation and integration

Advantages and disadvantages

- Reduced costs and risks as less software is developed from scratch
- Faster delivery and deployment of system
- But requirements compromises are inevitable so system may not meet real needs of users

Key points

- Software processes are the activities involved in producing a software system. Software process models are abstract representations of these processes.
- General process models describe the organization of software processes.
  - Examples of these general models include the ‘waterfall’ model, incremental development, and reuse-oriented development.
- Requirements engineering is the process of developing a software specification.

Key points

- Design and implementation processes are concerned with transforming a requirements specification into an executable software system.
- Software validation is the process of checking that the system conforms to its specification and that it meets the real needs of the users of the system.
- Software evolution takes place when you change existing software systems to meet new requirements. The software must evolve to remain useful.
- Processes should include activities such as prototyping and incremental delivery to cope with change.