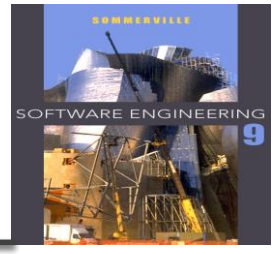
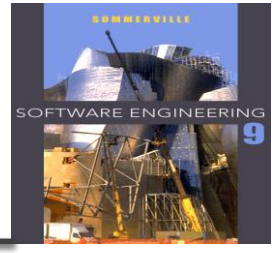


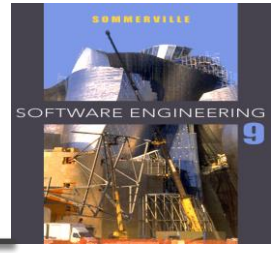
Chapter 1- Introduction and Software Processes

Topics covered



- ✧ Basic definition
- ✧ Role of Management in Software Development
- ✧ Software Products
- ✧ Essential attributes of good software
- ✧ Challenges for Software Engineering Practices
- ✧ Software Engineering Diversity
- ✧ Software Life Cycle
- ✧ Software Process Model and its Types

1.1 Basic Definition



- What is software?

  Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

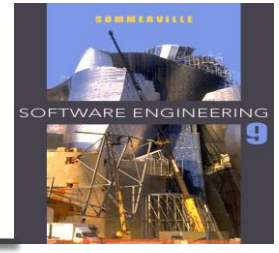
- What is software engineering?

  Software engineering is an engineering discipline that is concerned with all aspects of software production from initial conception to operation and maintenance.

- The IEEE [IEE17] has developed the following definition for software engineering:

  The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

1.2 Role of Management in Software Development



Factor	Role
People	Software development requires good managers. The manager who can understand the psychology of the people and provide good leadership. After having a good manager, project is in safe hands. It is the responsibility of a manager to manage, motivate, encourage, guide and control the people of his/her team.
Product	What is delivered to the customer, is called a product. It may include source code, specification document, manuals, documentation etc. Basically, it is nothing but a set of deliverables only.
Process	Process is the way in which we produce software. It is the collection of activities that leads to (a part of) a product. An efficient process is required to produce good quality products. If the process is weak, the end product will undoubtedly suffer, but an obsessive over reliance on process is also dangerous.
Project	A proper planning is required to monitor the status of development and to control the complexity. Most of the projects are coming late with cost overruns of more than 100%. In order to manage a successful project, we must understand what can go wrong and how to do it wright.

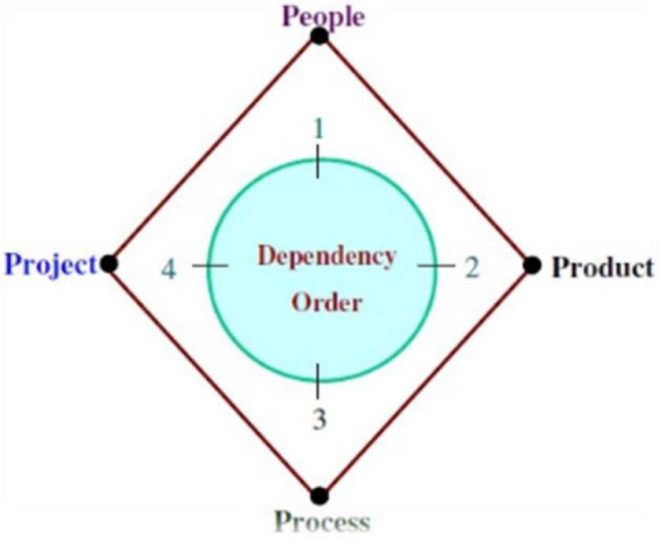
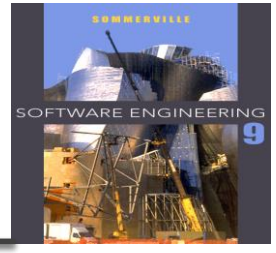


Figure 1.1 4Ps of Software Engineering

1.3 Software products



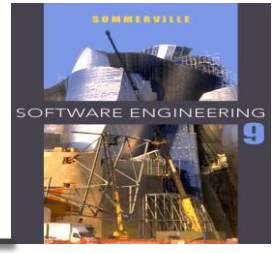
✧ Generic products

- Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
- Examples – PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.

✧ Customized products

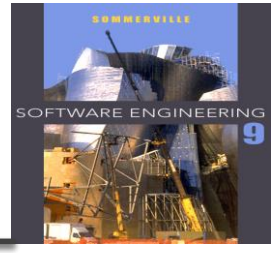
- Software that is commissioned by a specific customer to meet their own needs.
- Examples – embedded control systems, air traffic control software, traffic monitoring systems.

1.4 Essential attributes of good software



Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability and security	Software dependability includes a range of characteristics including reliability, security and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.

1.5 Challenges for Software Engineering Practices



1 ✧ Heterogeneity

- Increasingly, systems are required to operate as distributed systems across networks that include different types of computer and mobile devices.

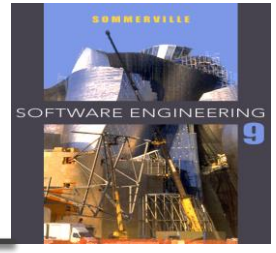
2 ✧ Business and social change

- Business and society are changing incredibly quickly as emerging economies develop and new technologies become available. They need to be able to change their existing software and to rapidly develop new software.

3 ✧ Security and trust

- As software is intertwined with all aspects of our lives, it is essential that we can trust that software.

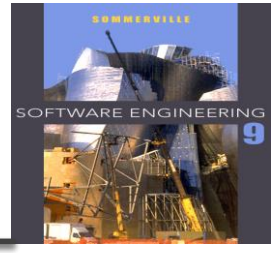
1.6 Software Engineering Diversity



- There are many different types of software system and there is no universal set of software techniques that is applicable to all of these. Rather, a diverse set of software engineering methods and tools has evolved.
- The software engineering methods and tools used depend on the type of application being developed, the requirements of the customer and the background of the development team. There are many different types of application, including:

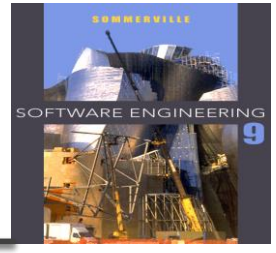
1. **Stand-alone applications**
2. **Interactive transaction-based applications**
3. **Embedded control systems**
4. **Batch processing systems**
5. **Entertainment systems**
6. **Systems for modeling and simulation**
7. **Data collection systems**
8. **Systems of systems**

Software Engineering Diversity : Application types



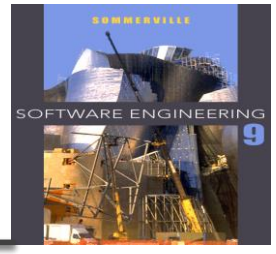
- ✧ Stand-alone applications
  - These are application systems that run on a local computer, such as a PC. They include all necessary functionality and do not need to be connected to a network.
- ✧ Interactive transaction-based applications
  - Applications that execute on a remote computer and are accessed by users from their own PCs or terminals. These include web applications such as e-commerce applications.
- ✧ Embedded control systems
  - These are software control systems that control and manage hardware devices. Numerically, there are probably more embedded systems than any other type of system.
- ✧ Batch processing systems
  - These are business systems that are designed to process data in large batches. They process large numbers of individual inputs to create corresponding outputs.

Software Engineering Diversity : Application types



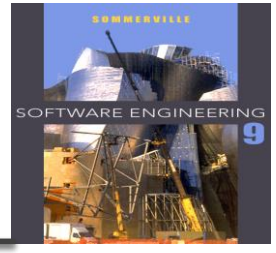
- ✧ Entertainment systems
  - These are systems that are primarily for personal use and which are intended to entertain the user.
- ✧ Systems for modeling and simulation
  - These are systems that are developed by scientists and engineers to model physical processes or situations, which include many, separate, interacting objects.
- ✧ Data collection systems
  - These are systems that collect data from their environment using a set of sensors and send that data to other systems for processing.
- ✧ Systems of systems
  - These are systems that are composed of a number of other software systems.

1.7 Software Life cycle



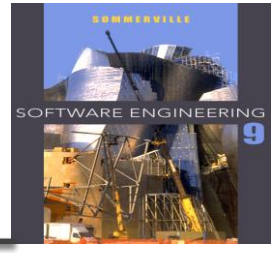
- The goal of Software Engineering is to provide models and processes that lead to the production of well-documented maintainable software in a manner that is predictable.
- In the IEEE standard Glossary of Software Engineering Terminology, the **software life cycle** is: The period of time that starts when a software product is conceived and ends when the product is no longer available for use.
- The software life cycle typically includes a requirement phase, design phase, implementation phase, test phase, installation and check out phase, operation and maintenance phase, and sometimes retirement phase".
- These activities involve the development of software from scratch until it is delivered to customers.

1.8 Software Process Model



- **Software process** is defined as the structured set of activities that are required to develop the 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.

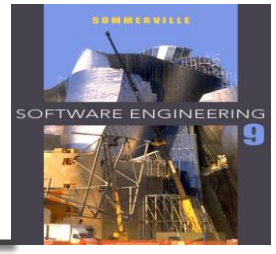
1.9 Types of Software process models



There are many types of software process models that development teams use across different industries. Here are some examples of typical software process models you can use to outline your development process:

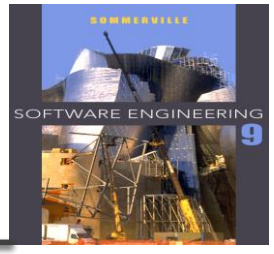
1. The waterfall model
2. Incremental development model
3. Reuse-oriented Model
4. Boehm's Spiral Model
5. Agile methods

1.9.1The waterfall model



- Waterfall model is the earliest software development life cycle (SDLC) approach that was used for software development. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use.
- In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. All phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases.
- In Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model".

The waterfall model



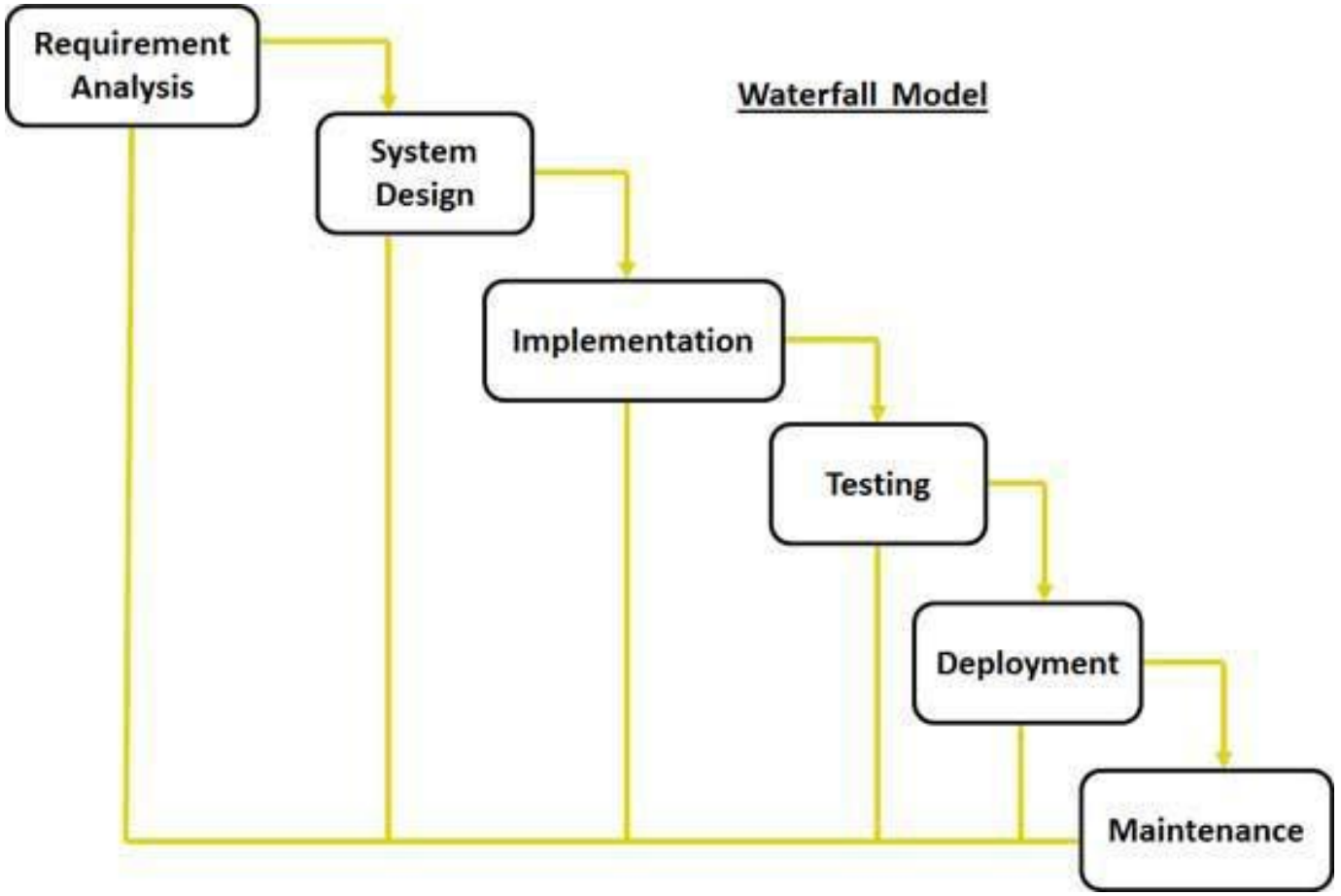
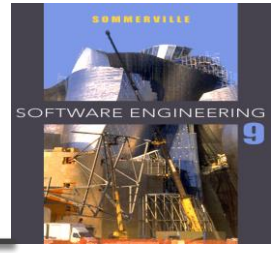


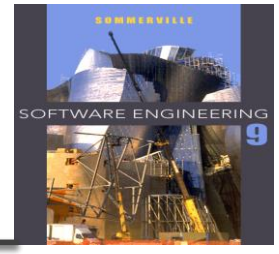
Figure 1.2: Waterfall Model

Waterfall model phases



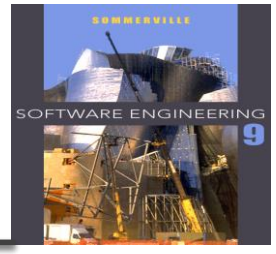
Phase	Description
Requirement Gathering and analysis	All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
System Design	The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
Implementation	With inputs from system design, the system is first developed in small programs called units, which are integrated (وحدات متكاملة) in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
Integration and Testing	All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
Deployment of system	Once the functional and non-functional testing is done, the product is deployed in the customer environment (تثبيت البرنامج للاستخدام) or released into the market.
Maintenance	There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

1.9.2 Incremental Development Model



- Incremental model uses the linear sequential approach with the iterative nature of prototyping. Incremental development is based on the idea of developing an initial implementation, delivering it to user for feedback and improving it through several versions of product releases until an acceptable system is developed.
- In incremental model the whole requirement is divided into various builds.
  - During each iteration, the development module goes through the requirements, design, implementation and testing phase.
  - Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.
- The key to use of incremental development model successfully is rigorous validation of requirements, and verification of each version of the software against the requirements.

Incremental Development Model



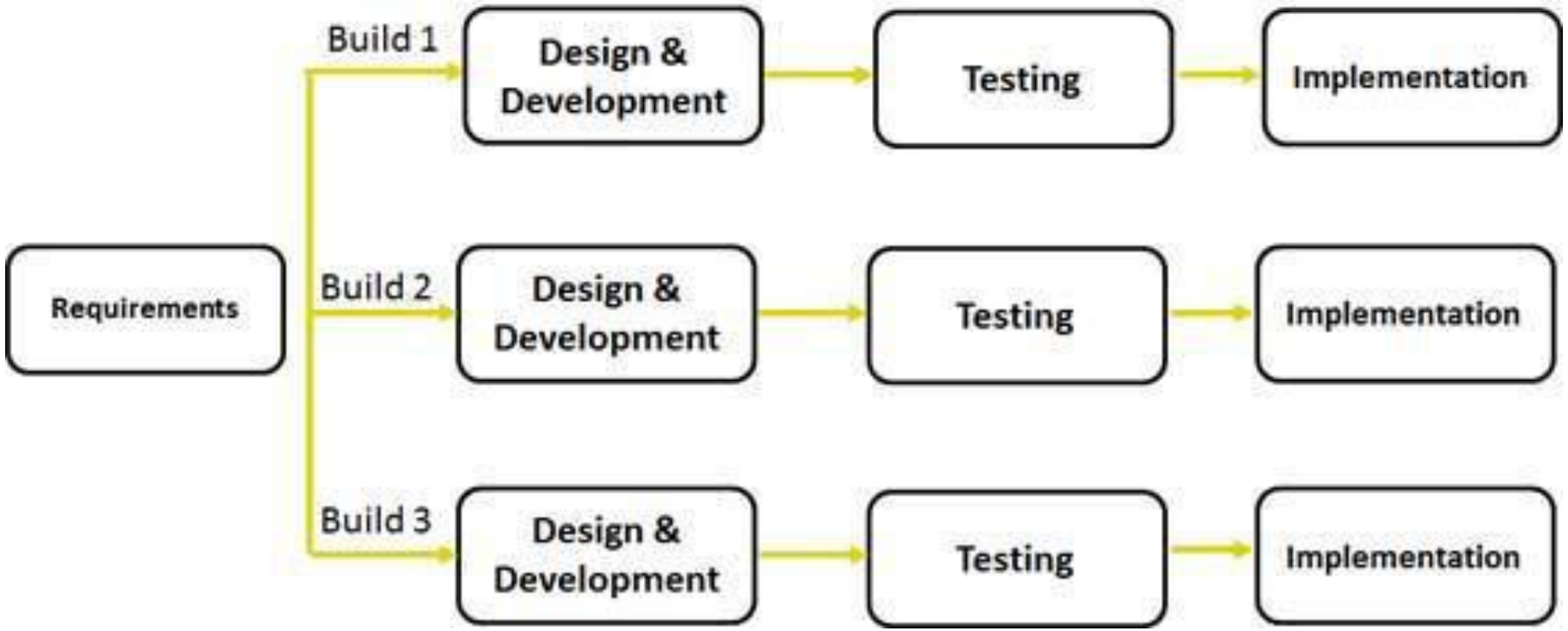
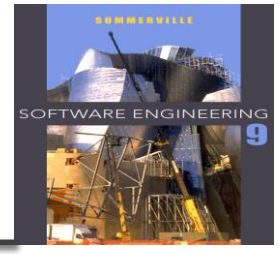


Figure 1.3: Incremental Development Model

1.9.3 Reuse-oriented Model



- The **reuse-oriented development** is where they reuse programming or software in previous projects or existing software components.
- Reuse-oriented development is based on systematic reuse where systems are integrated from existing components.
- This development can reduce the overall cost of software development and it can also save time because each phase of the process builds on the previous phase which has already been refined.

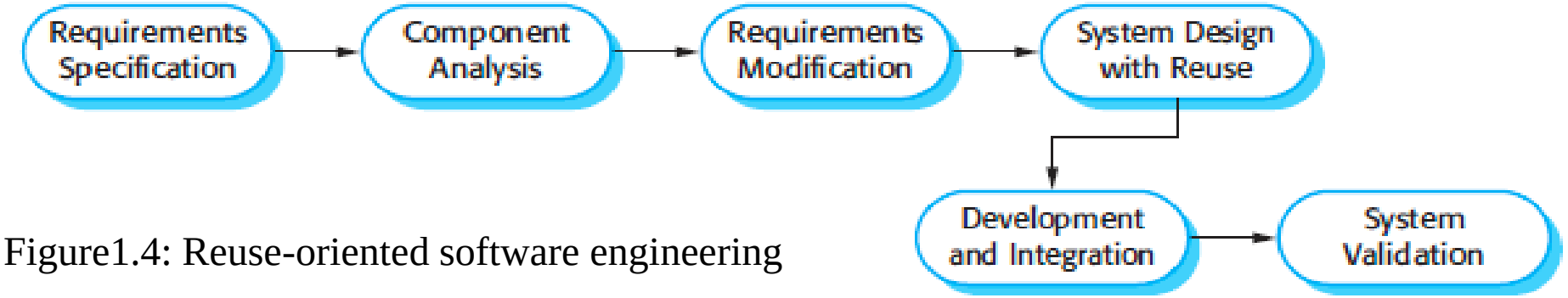
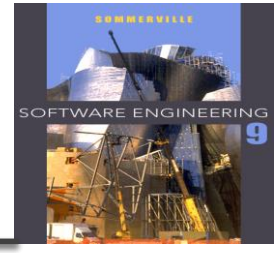


Figure1.4: Reuse-oriented software engineering

Reuse-oriented Model phases



1. **Requirement Specification:** All possible requirements of the system to be developed are captured in this phase.

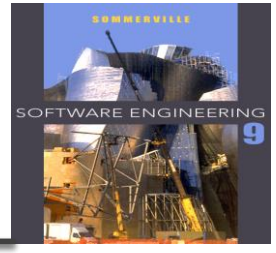
2. **Component Analysis:** According to given requirement, component is selected to implement that requirement specification. That is not possible the selected component provide the complete functionality, but that is possible the component used provide some of the functionality required.

3. **Requirement Modification**: Information about component that is selected during component analysis is used to analysis requirement specification. Requirements are modified according to available components. Requirement modification is critical then component analysis activity is reused to find relative solution.

4. **System design with reuse:** During this stage the design of the system is build. Designer must consider the reused component and organize the framework. If reused component is not available then new software is develop.

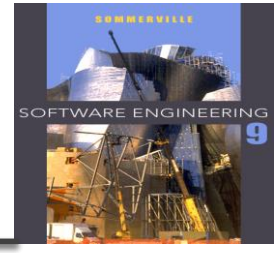
5. **Development and Integration**: Components are integrated to develop new software. Integration in this model is part of development rather than separate activity.

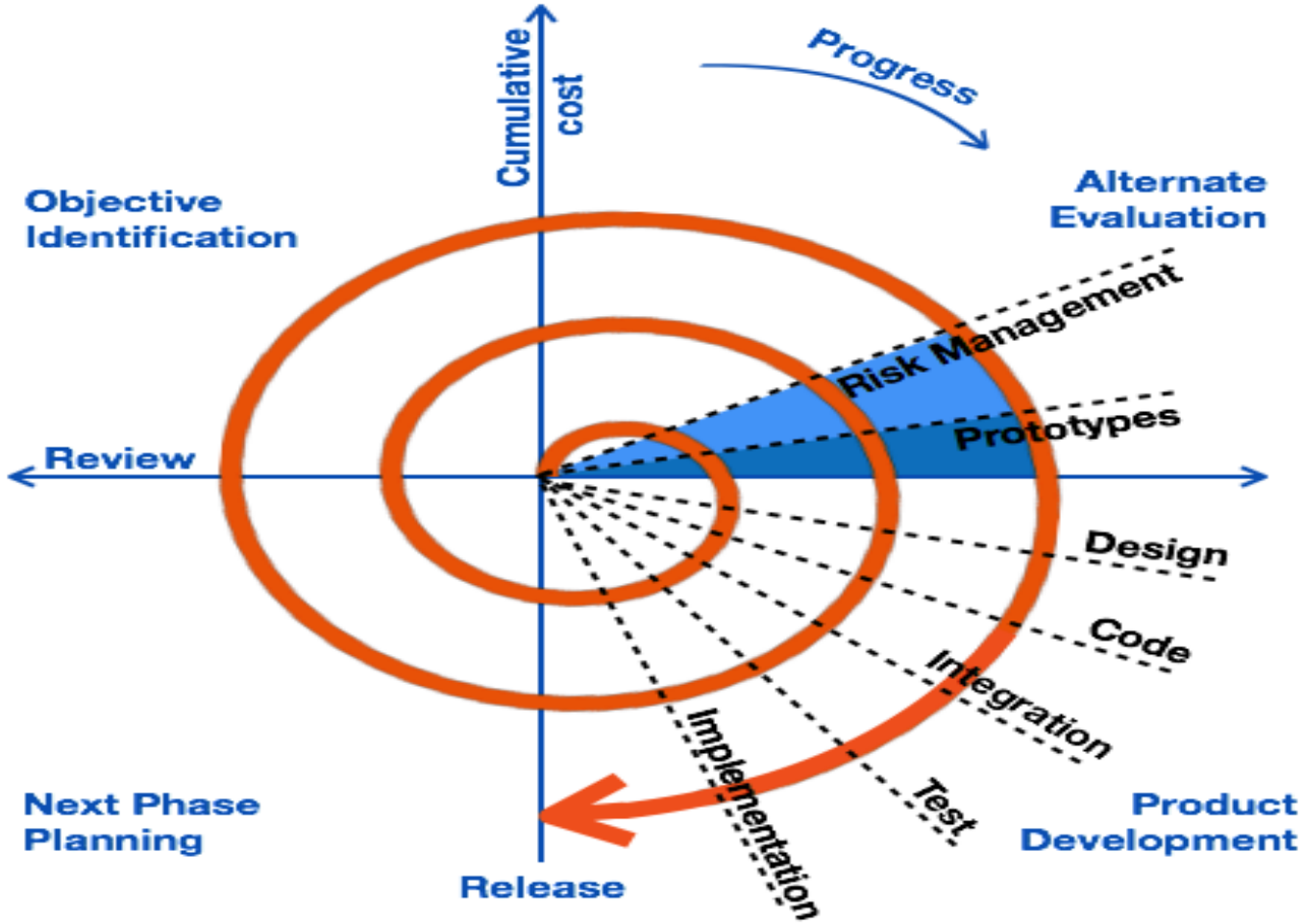
1.9.4 Boehm's Spiral Model



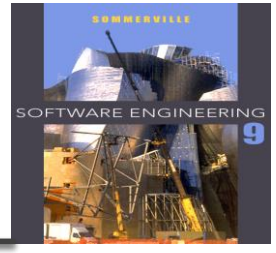
- A risk-driven software process framework (the spiral model) was proposed by Boehm.
- Process is represented as a spiral rather than as a sequence of activities with some backtracking from one activity to another.
- Each loop in the spiral represents a phase in the process.
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process.

Boehm's spiral model of the software process



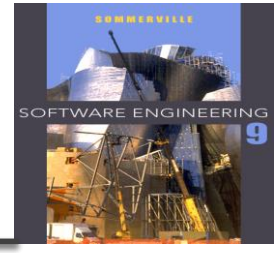


Explanation of Spiral Model Activities



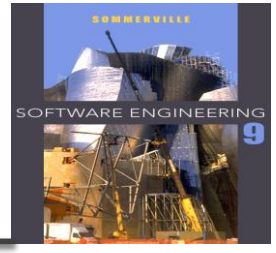
Activity	Description
Objective Identification	Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed. Project risks are identified. Alternative strategies, depending on these risks, may be planned.
Alternate Evaluation	In this quadrant, all the proposed solutions are analyzed and any potential risk is identified, analyzed, and resolved. The Prototype is built for the best possible solution.
Product development	The identified features are developed and verified through testing. At the end of the stage, the next version of the software is available.
Next Phase Planning	The project is reviewed and a decision made whether to continue with a further loop of the spiral. If it is decided to continue, plans are drawn up for the next phase of the project.

1.9.5 Agile methods



- Agile model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.
- Agile Methods break the product into small incremental builds. These builds are provided in iterations. At the end of each iteration a working product is displayed to the customer.
- In Agile development less time is spent on planning and more focus on the features need to be developed. There is feature driven development and the team adapts to the changing product requirements dynamically. The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future.
  - .

Agile methods



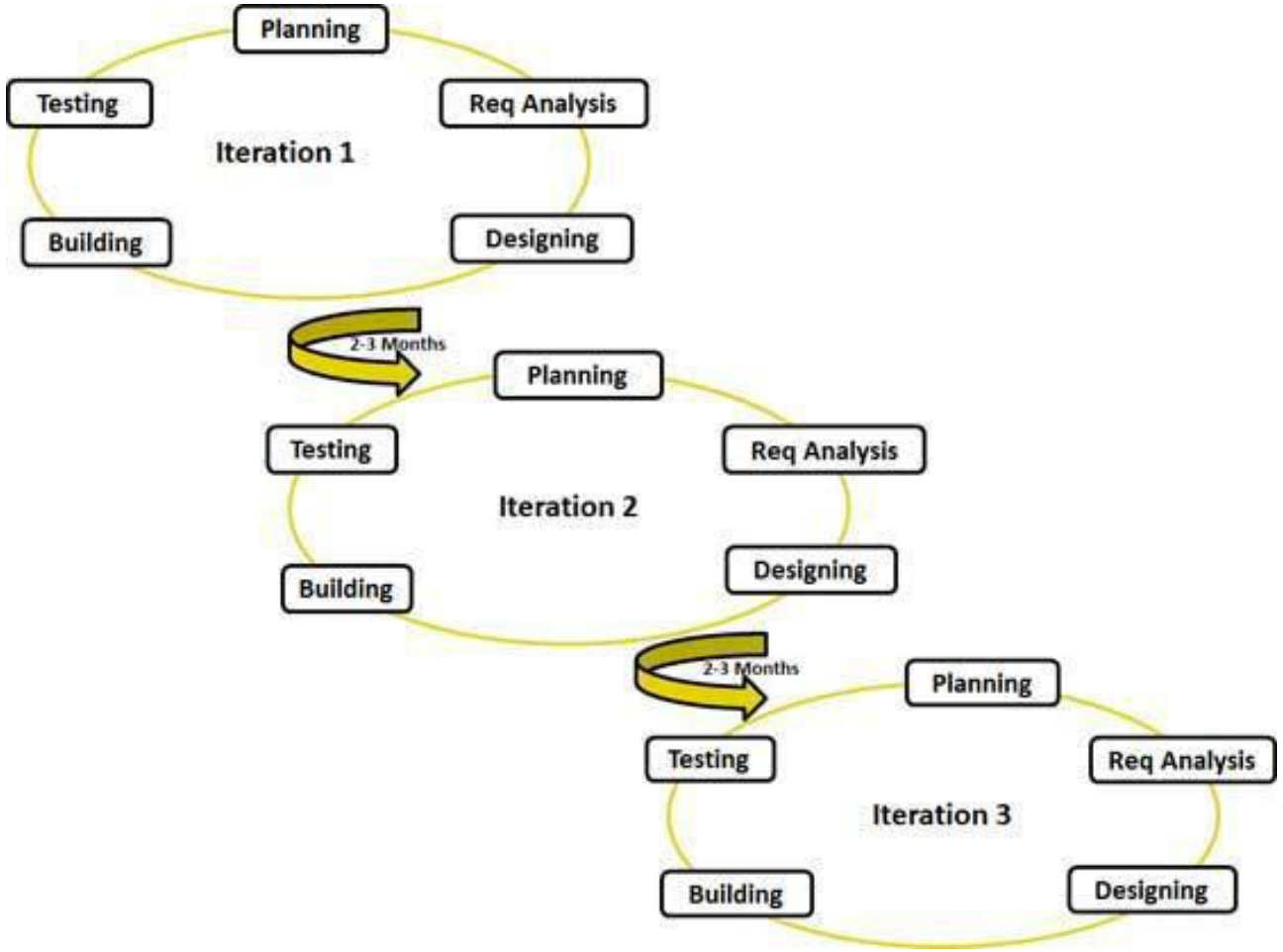
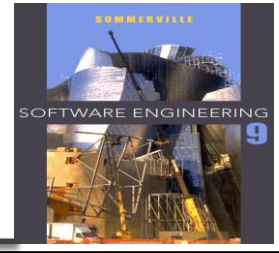


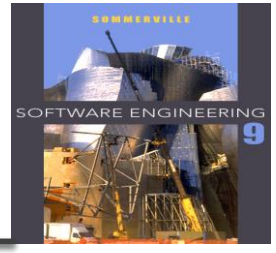
Figure 2.5: Graphical Illustration of the Agile Methods

1.9.6 Comparison of Different Software Process Models



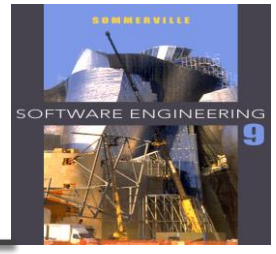
Process model	Approach	Advantages	Disadvantages	When to Apply
Waterfall	Linear sequential	Clear documentation, easy to manage	Inflexible, limited feedback, high risk	Simple, small projects with clear requirements
Incremental	Linear Sequential with Iterative	Easier to test and debug, more flexible, simple to manage risk.	Can be time-consuming, expensive, requires a high level of collaboration	Projects with changing requirements
Reuse-oriented	Reuse-based	Reduce total cost, low risk factor, save lots of time and effort.	Requires well-defined components, may not fit all projects	Projects that necessitate a high level of flexibility, scalability, maintainability
Spiral	Iterative, risk-driven	Highlights risk management, flexible, adaptive	Time-consuming, can be expensive	Large, complex, high-risk projects
Agile	Iterative	Flexible, adaptive, continuous feedback	Requires active user involvement, can be chaotic	Complex projects with fluctuating requirements

Key points



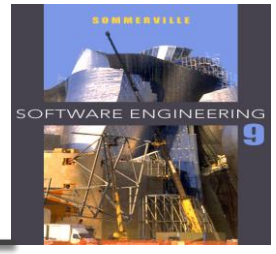
- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Essential software product attributes are maintainability, dependability and security, efficiency and acceptability.
- The high-level activities of specification, development, validation and evolution are part of all software processes.
- There are many different types of system and each requires appropriate software engineering tools and techniques for their development.
- Software processes are the activities involved in producing a software system. Software process models are abstract representations of these processes.

Key points



- General process models describe the organization of software processes. Examples of these general models include the ‘waterfall’ model, incremental development, and reuse-oriented development.
- The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model.
- Agile methods are incremental development methods that focus on rapid development, frequent releases of the software, reducing process overheads and producing high-quality code. They involve the customer directly in the development process.

Review Questions



1. Define: Software, Software Engineering, Software life cycle, Software process.
2. What are the 4P's of Software Engineering?
3. Differentiate between Generic product and Customized product.
4. What are the essential attributes of good software?
5. List down the challenges of software engineering practices, activities of Software process.
6. Explain in detail: Waterfall Model, Incremental development model, Reuse-oriented model, Boehm's spiral model, Agile method. What are the Advantages and Disadvantages?