

Software Engineering

Prepared by Dr. Yasser Abdelhamid

IT 382
Software Engineering

Lecture 3



LECTURE OUTLINE

- ◉ **Software process models**
- ◉ **Waterfall model**
- ◉ **Incremental model**
- ◉ **Reuse model**

THE SOFTWARE PROCESS

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

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.

SOFTWARE PROCESS DESCRIPTIONS

- 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 MODEL

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

PLAN-DRIVEN PROCESSES

- **Plan-driven** processes are processes where all of the process activities are **planned** in advance.
- Progress is measured **against this plan**.

AGILE PROCESSES

- In agile processes, planning is **incremental**.
- It is easier to change the process to reflect changing customer requirements.

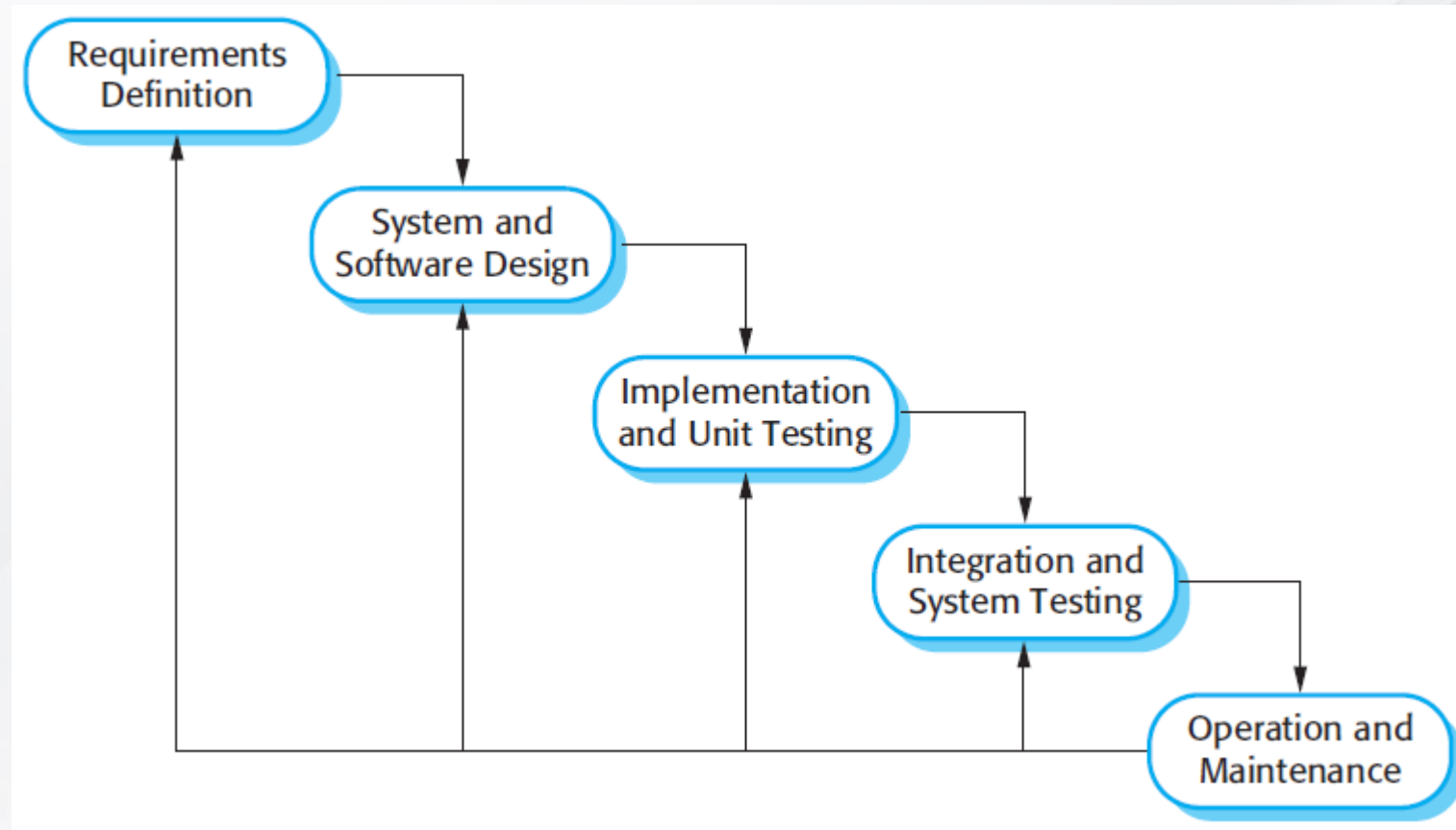
REAL LIFE PRACTICE

- 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

- **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.
- **Reuse-oriented software engineering**
 - The system is **assembled** from existing 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

DRAWBACKS OF WATERFALL MODEL

- The main drawback of the waterfall model is **the difficulty of accommodating change** after the process is underway.
- In principle, a **phase has to be complete before moving onto the next phase.**

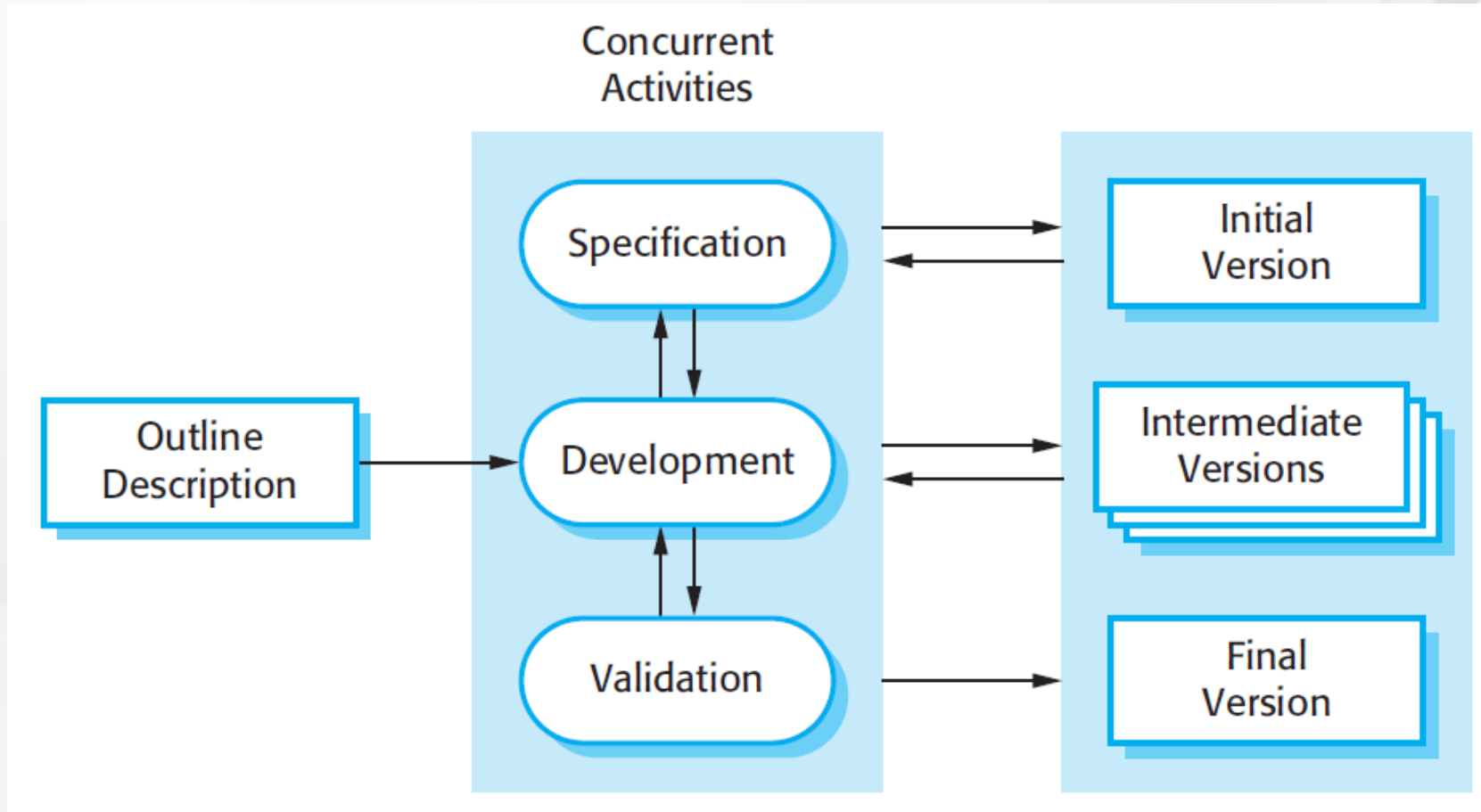
WATERFALL MODEL PROBLEMS

- **Difficult to respond to changing customer requirements.**
 - Therefore, this model is only appropriate when:
 - The requirements **are well-understood**
 - **Changes will be fairly limited during the design process. .**

WATERFALL MODEL PROBLEMS

- 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 DEVELOPMENT



INCREMENTAL DEVELOPMENT BENEFITS

- **The cost of accommodating changing customer requirements is reduced.**
 - The amount of analysis and documentation that has to be **redone is much less** than is required with the waterfall model.

INCREMENTAL DEVELOPMENT BENEFITS

- **It is easier to get customer feedback** on the development work that has been done.
 - Customers can **comment on demonstrations of the software** and see how much has been implemented.

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.

INCREMENTAL DEVELOPMENT PROBLEMS

- **The process is not visible.**
 - 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.

INCREMENTAL DEVELOPMENT PROBLEMS

- **System structure tends to degrade** as new increments are added.
 - Unless time and money is spent on **refactoring** to improve the software, regular change tends to corrupt its structure.
 -

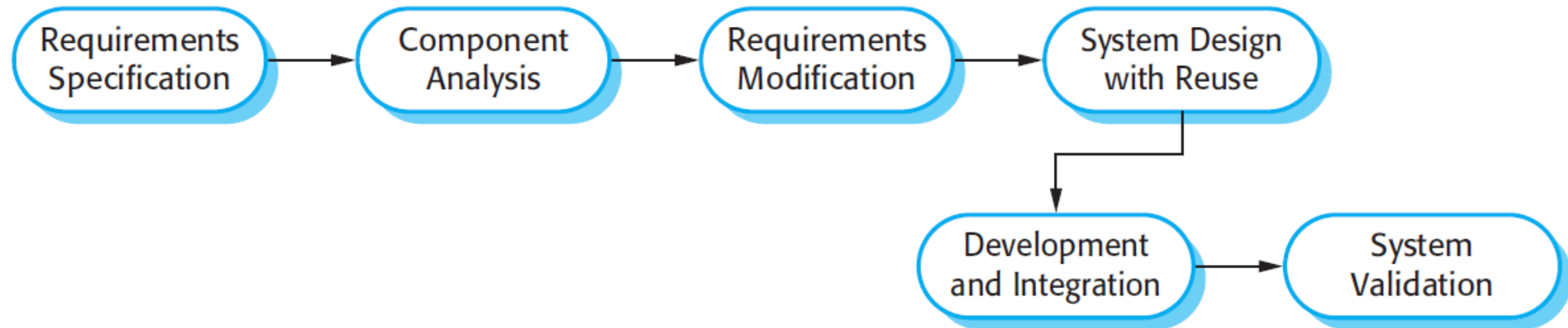
REUSE-ORIENTED SOFTWARE ENGINEERING

- Reuse-oriented software engineering is based on systematic reuse where **systems are integrated from existing components** or COTS (Commercial-off-the-shelf) systems.

REUSE-ORIENTED SOFTWARE ENGINEERING

- Process stages
 - Requirements specification
 - Component analysis;
 - Requirements modification;
 - System design with reuse;
 - Development and integration.
- Reuse is now the standard approach for building many types of business systems.

REUSE-ORIENTED SOFTWARE ENGINEERING



TYPES OF SOFTWARE COMPONENT

- **Web services** that are developed according to service standards and which are available for remote invocation.

WEB SERVICES

- Social Media Platforms:
 - Social media platforms like Facebook, Twitter, and Instagram provide web services that allow users to create profiles, post content, interact with others, and access various features through web APIs.

WEB SERVICES

- E-commerce Websites:
 - E-commerce platforms like Amazon and eBay offer web services for product listings, shopping carts, payment processing, and order management. These services enable businesses to sell products online.

WEB SERVICES

- Weather Services:
 - Websites and applications that provide weather information, such as AccuWeather or OpenWeatherMap, offer web services that allow developers to access real-time weather data, forecasts, and weather-related functionalities.

WEB SERVICES

- Mapping and Geolocation Services:
 - Services like Google Maps and MapQuest provide web APIs that allow developers to integrate mapping and geolocation features into their applications, such as displaying maps, finding routes, and geocoding addresses.

WEB SERVICES

- **Payment Gateways:**
 - Payment gateways like PayPal and Stripe offer web services for securely processing online payments. These services enable businesses to accept credit card payments, handle transactions, and manage billing processes.

WEB SERVICES

- Messaging and Communication Services:
 - Applications like WhatsApp, Slack, and Twilio provide web services for sending and receiving messages, making voice or video calls, and integrating communication functionalities into other applications.

WEB SERVICES

- Travel and Booking Services:
 - Websites and applications like Expedia and Airbnb offer web services for searching and booking flights, hotels, and other travel-related services. These services allow developers to access travel information and facilitate bookings.

WEB SERVICES

- Financial Services:
 - Financial institutions and fintech companies provide web services for banking operations, account management, transaction processing, and accessing financial data. Examples include banking APIs, stock market data APIs, and payment processing APIs.

WEB SERVICES

- News and Content Aggregators:
 - Services like Feedly and Flipboard offer web APIs for aggregating and delivering news articles, blog posts, and other content from various sources. Developers can integrate these services to provide personalized content experiences.

WEB SERVICES

- Cloud Storage and File Sharing:
 - Cloud storage providers like Dropbox and Google Drive offer web services that allow users to store, share, and access files and documents securely over the internet. These services provide APIs for integrating cloud storage functionalities into applications.

TYPES OF SOFTWARE COMPONENT

- **Collections of objects** that are developed as a package to be integrated with a component framework such as .NET or J2EE.

TYPES OF SOFTWARE COMPONENT

- **Stand-alone software systems (COTS)** that are configured for use in a particular environment.

SUMMARY

- 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.

SUMMARY(CONT.)

- Types of components that can be used in the reuse process model.
 - **Web services**
 - **Reusable Objects**
 - **Stand-alone software components**



Thank You

