SEN 3244

SOFTWARE ARCHITECTURE

M. Mangong Clement

❑ Chap1: Introduction to Software Architecture

Lesson Objectives

1. Understand the importance of software architecture.

❑ Chap1: Introduction to Software Architecture

- ✓ **Goal 1 : Satisfy customer's (end-users) needs**
- ✓ **Goal 2 : To continuously satisfy goal one even with new requirements and modification.**

- *To accomplish goal 2, you need to think about the initial decision taken before building the application ; the software Architecture*

❑ Chap1: Introduction to Software Architecture

Good judgment is usually the result of experience,

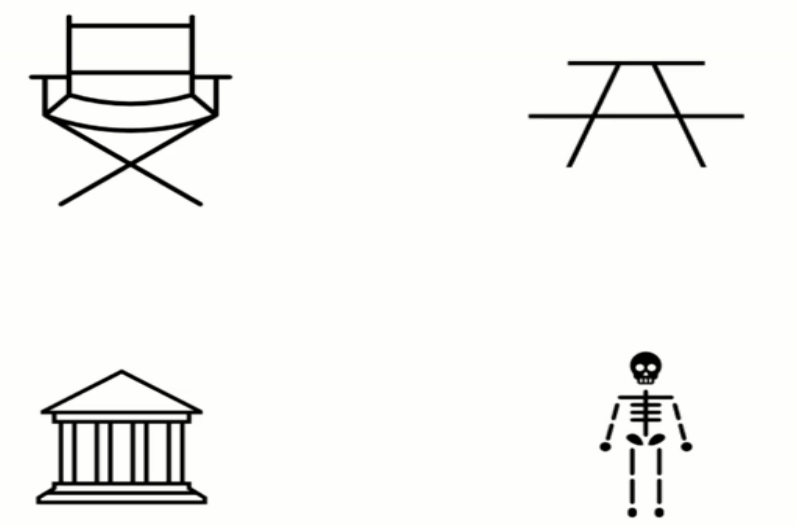
*And experience is frequently the results of bad judgment. But to learn from the experience of others requires those who have the experience to share the knowledge with those who follow. (- **Barry LePatner**)*

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is an architecture?

- Structural objects…limited by physics in the physical universe.



❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

Architecture vs software architecture

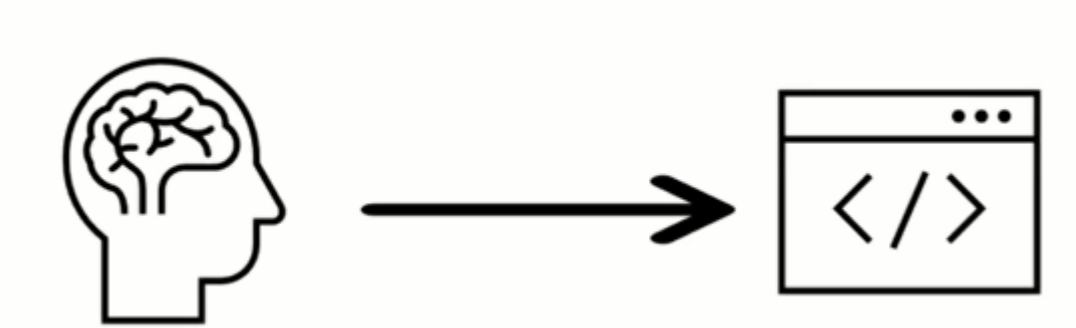
Architecture	Software Architecture
• Design a building that fits the constraints and required uses	• Design software that fits the constraints and required uses
• Once built, will remain fairly static over time	• Will be constantly changing throughout its lifetime

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (1)

- Mapping thoughts (from developers mind) to code in computer language.



What is the biggest obstacle?

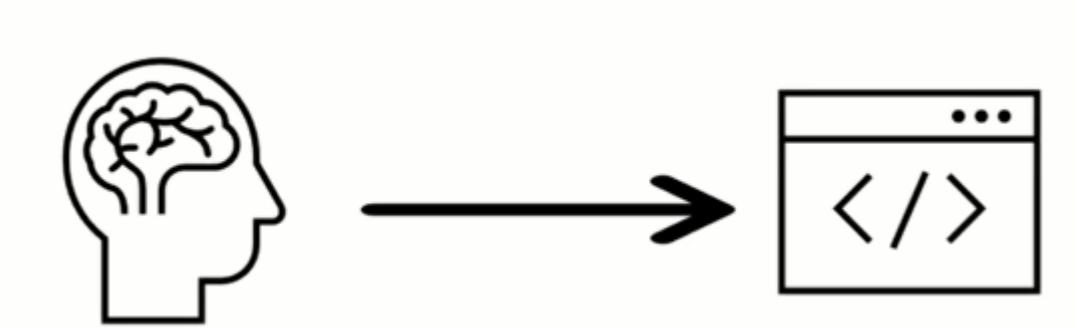
- Thoughts are never stable as our ideas comes and goes

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (1)

- Mapping thoughts (from developers mind) to code in computer language.



What is the biggest obstacle?

- Thoughts are never stable as our ideas comes and goes

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (2)

- Software architecture provides a limitation to the converting process of change of thoughts
  - so that it could follow certain **rules**
  - And so that the code can follow a certain ***pattern.***

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture?(3)

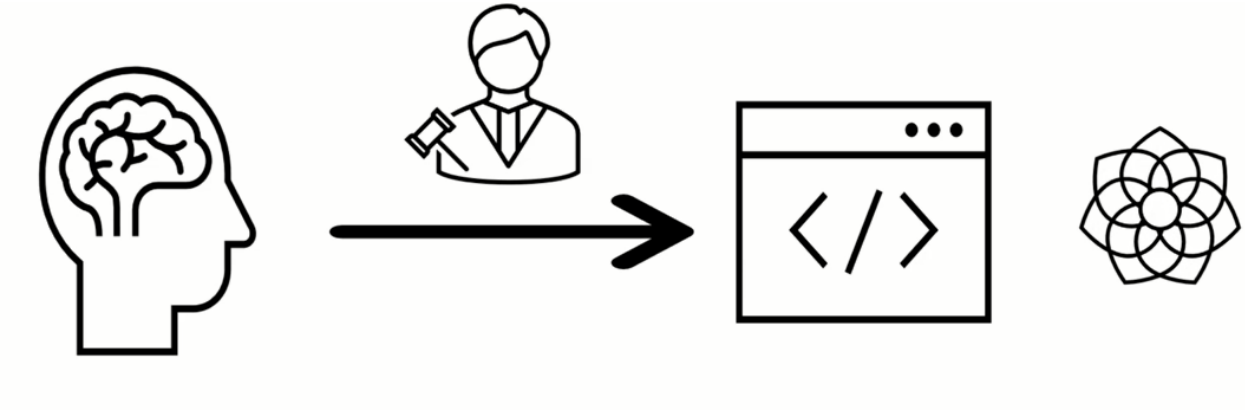
- Software is not limited my physic - as long as you can imagine something you can implement in code.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (4)

- It is the set of rules that provides a set of limitations to the software development process.



❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (5)

- It is a set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (6)

- ✓ Some partition systems into implementation units (modules), which are static
- ✓ Some are dynamic, focusing on the way the elements interact with each other at runtime to carry out the system's functions(component-and-connector)

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is software architecture? (7)

✓ Some describe the mapping from software structures to the system's organizational, developmental, installation, and execution environments (allocation).

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

- If we do not use a software architecture to create our software, the software will mostly likely to be complex especially the larger software or when the software get bigger and bigger.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

What is a good software architecture?

- ➢ Must be able to help us reduce the complexity and produce a simple software product.
- ✓ Complexity is the structure of a software system that makes it hard to understand and modify.
- ✓ We need the set of rules that will be use to make a software easy understand and to modify.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

The separation of concern principle

➢ *Consider you have several things; its complex to view*

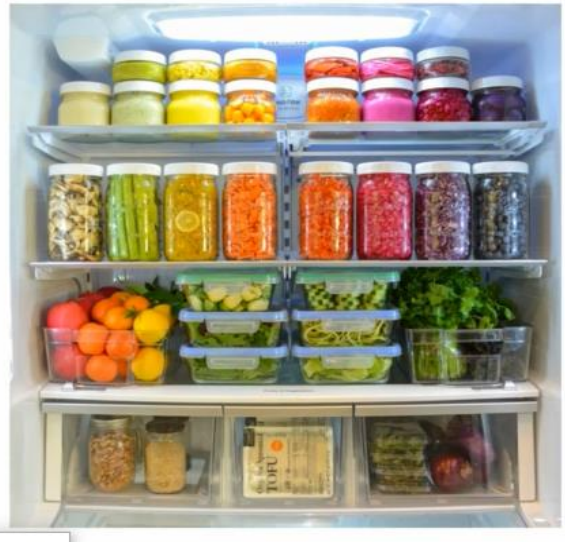
Here we need the arrangement design and patterns to arrange them

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ *Divide into different module and each module handle a set of concerns that are more or less related.*

Separation of Concerns

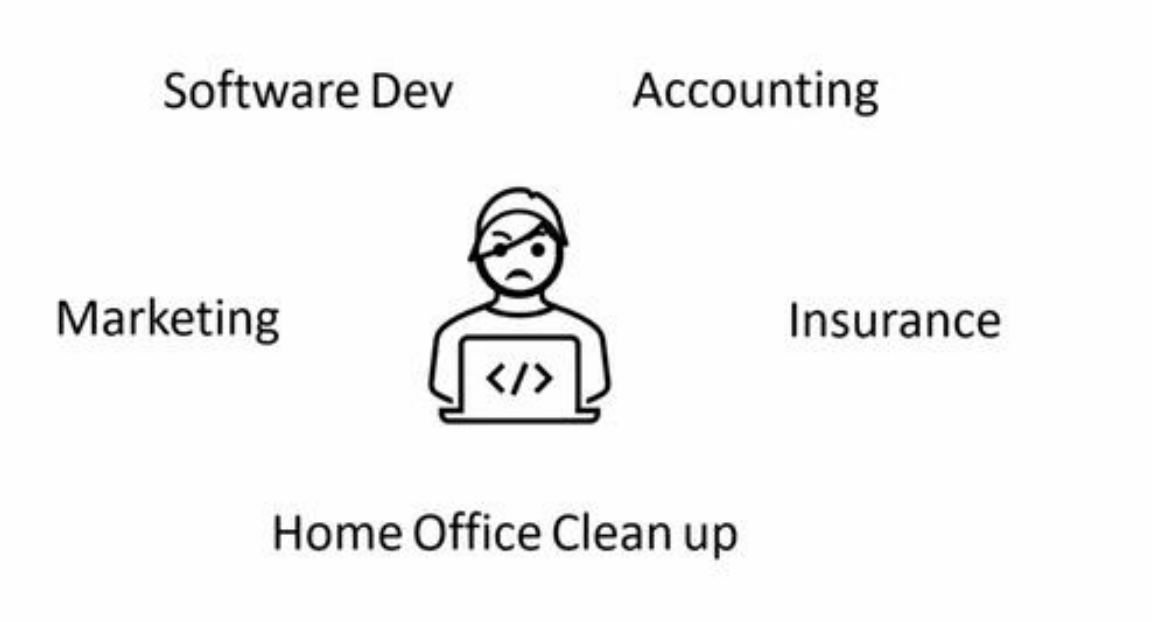


❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

The separation of concern principle

➢ *Take an example that you are a freelancer*

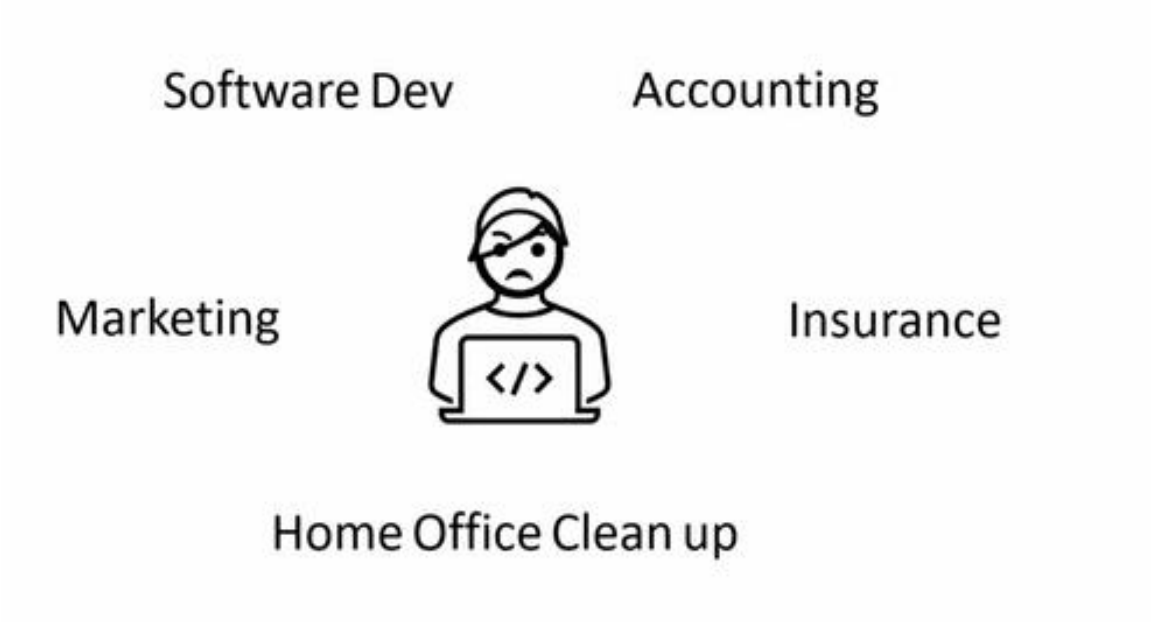


❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

The separation of concern principle

➢ *Take an example that you are a freelancer*

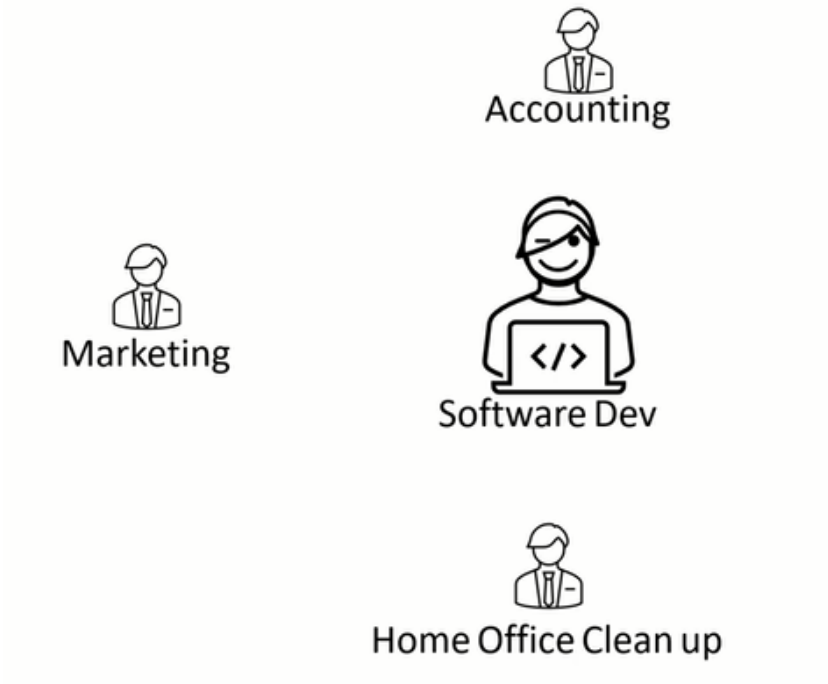


❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

The separation of concern principle

➢ *You wish someone to assist you in the other things.*



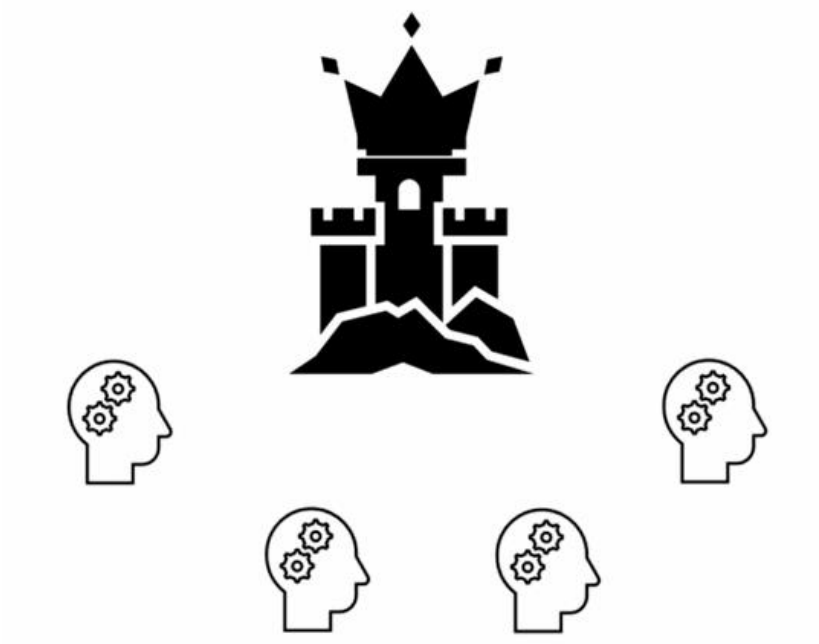
➢ ***It brings more efficiency and a bigger chance of success is high***

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

The separation of concern principle

- *You have to separate in different module.*

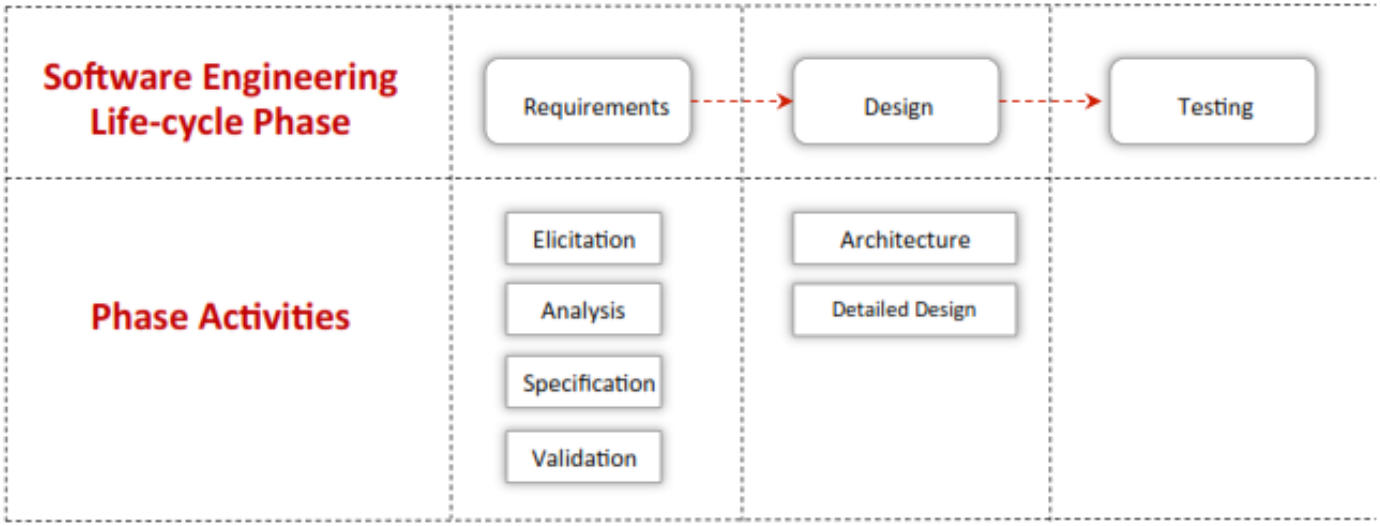


- *Pattern will tell us how to separate into the various modules.*
- *To separate related things into different modules.*

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

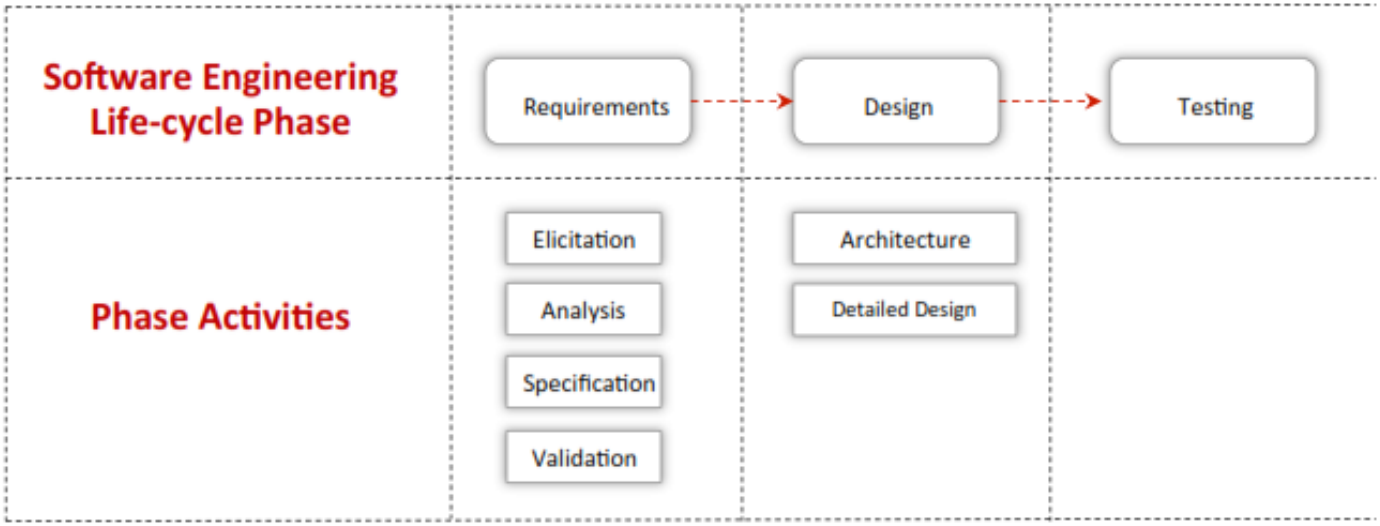
➢ Where is software architecture located in the software development life cycle?



❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Where is software architecture located in the software development life cycle?



❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Is software architecture different from “design”?

✓ All software architecture are designs but not all software designs are architecture.
✓ Architectural decisions span the entire system or major subsystems.
✓ Architectural decisions are driven by the systems’ goals, which must be articulated.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Functionality

- Functionality does not determine architecture

✓ Functionality is assigned to specific elements of the system(responsibility drive design)

✓ If functionality were the only thing that mattered, you wouldn't have to divide the system into pieces at all; a single monolithic app with no internal structure would do just fine.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Functionality

- The architect's interest in functionality is in how it interacts with and constrains other qualities.
- The functional requirements are not (usually) architecturally significant… leaving the non-functional requirements – quality attributes.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Quality Attribute

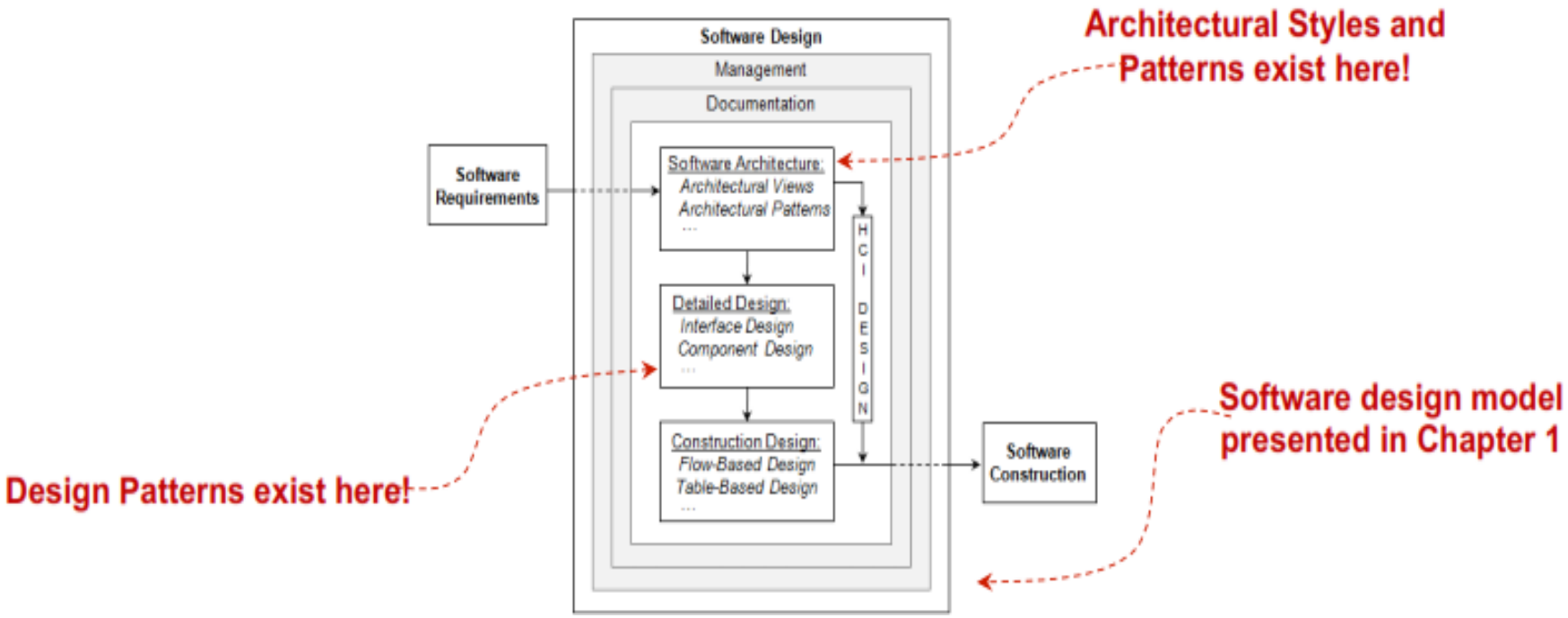
- A measurable or testable property of a system that is used to indicate how well the system satisfies the needs of its stakeholders.
- Some common quality attributes

Avaliability, interoperability, modifiability, performance, security, testability, usability.

❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Software Design stack



❑ Chap1: Introduction to Software Architecture

❖ Overview of Software Architecture

➢ Choice of applying architectural patterns

It depends on

✓ The type of system

✓ The requirements

✓ The desired quality attributes

It helps guide the choice of selecting one particular pattern over another.

- More than one pattern can be use too.

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

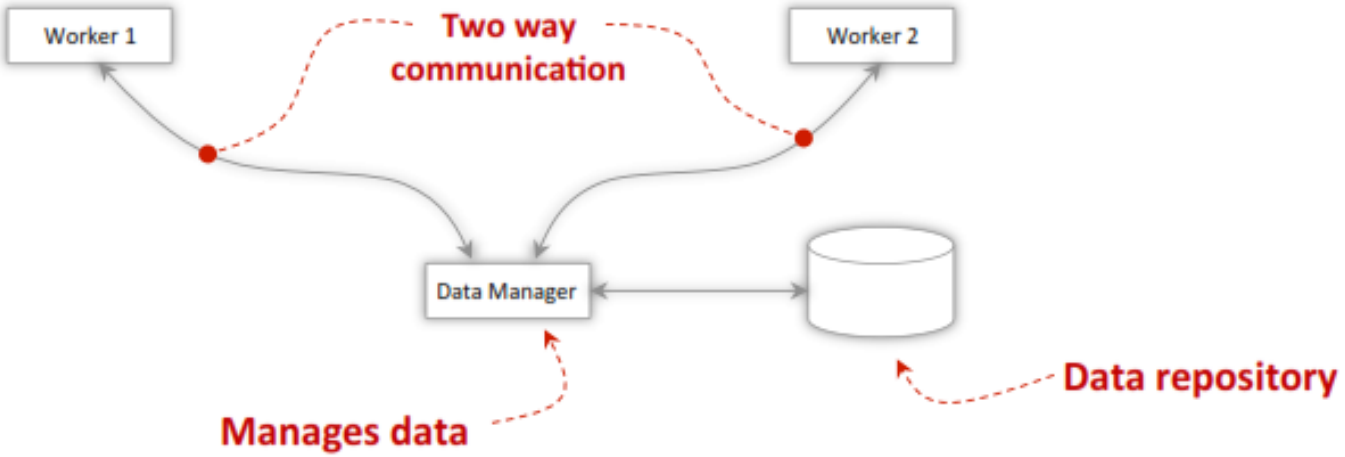
1. **Data- centered :**

- *Systems that serve as a centralized repository for data, while allowing clients to access and perform work on the data.*
- *Example: Blackboard Pattern*

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

1. Data- centered : *Blackboard Architectural Pattern*



Two major components needed around the main central repository of data

- Data management component
- Worker components

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

2. Data flow:

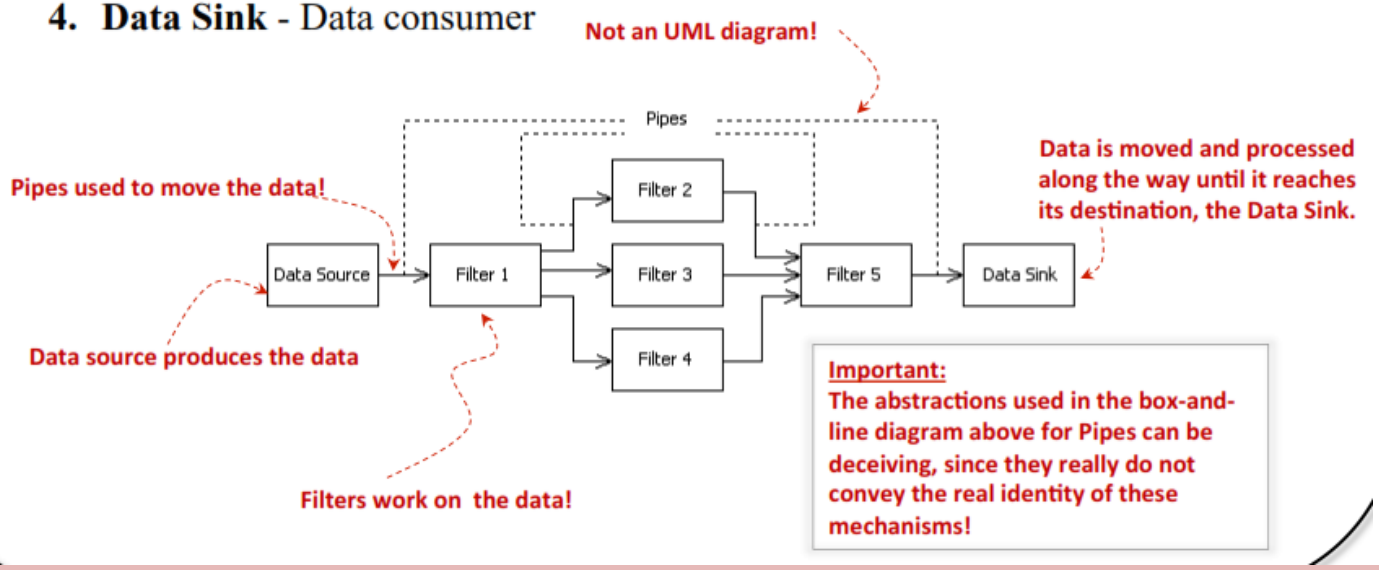
- *Systems oriented around the transport and transformation of a stream of data.*
- *Example: Pipes-and-Filters*

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

2. Data flow: Pipes-and-filters

1. **Data source -** Produces the data
2. **Filter** - Processes, enhances, modifies, etc. the data
3. **Pipes** - Provide connections between data source and filter, filter to filter, and filter to data sink.
4. **Data Sink** - Data consumer



SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

3. Distributed (for Distributed systems)

- *Systems primarily involve interactions between several independent processing units connected via a network.*
- *Example: Client-Server, Broker*

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

4. Interactive

- *Systems that serves users or user-centric systems.*
- *Example: Model-View-Controller*

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

5. Hierarchical

- *Systems where components can be structured as hierarchy (vertically and horizontally) to reflect different levels of abstraction and responsibility.*
- *Example: Main program and subroutine, Layered*

SOFTWARE Architecture Patterns & Styles

- The various classes of architectural patterns and styles

2. Data flow

1. Distributed

2. Interactive

3. Hierarchical

II. SOFTWARE Architecture Patterns

- They are several Software stakeholders
- Each stakeholder views the software architecture following their point of interest.
- The 4+1 view model shows the various stakeholder architectural perspectives.

Welcome!

This course is design for you to understand the ways software architectures are represented, both in UML and other visual tools.

QUESTIONS