SEN 3244

SOFTWARE ARCHITECTURE

M. Mangong Clement

❑ Chap 8: Designing Software Architecture

Task - Class Work

"In groups of three students, download the ppt. notes, and answer the questions of each of the slides"

You shall present your answers towards the end of class.

You can use your tools in case to demonstrate your work

❑ Chap 8: Designing Software Architecture

"A designer knows he has achieved perfection not when there is nothing left to add, but when there is nothing left to take away."

- Antoine de Saint-Exupery

❑ Chap 8: Designing Software Architecture

❖ Architecture design overview

- It involves **making decisions**, and working with the **available materials** and **skills**, to achieve a **goals** and to **satisfy requirements and constraints.**
- **The output(final decisions) will reflect the goals, requirements and constraints.**

❑ Chap 8: Designing Software Architecture

❖ Architecture design overview

- We turn decisions about the architectural drivers / concerns into structures.
- The structures are then used to :-

✓ Guide the project (analysis and construction)

✓ Guide cost and schedule estimation, team formation , risk analysis and mitigation.

❑ Chap 8: Designing Software Architecture

❖ Architecture design overview

- It serves as the foundation for educating a new project member.
- It guides the cost and schedule estimations, team formation, risk analysis and mitigation , and implementation.

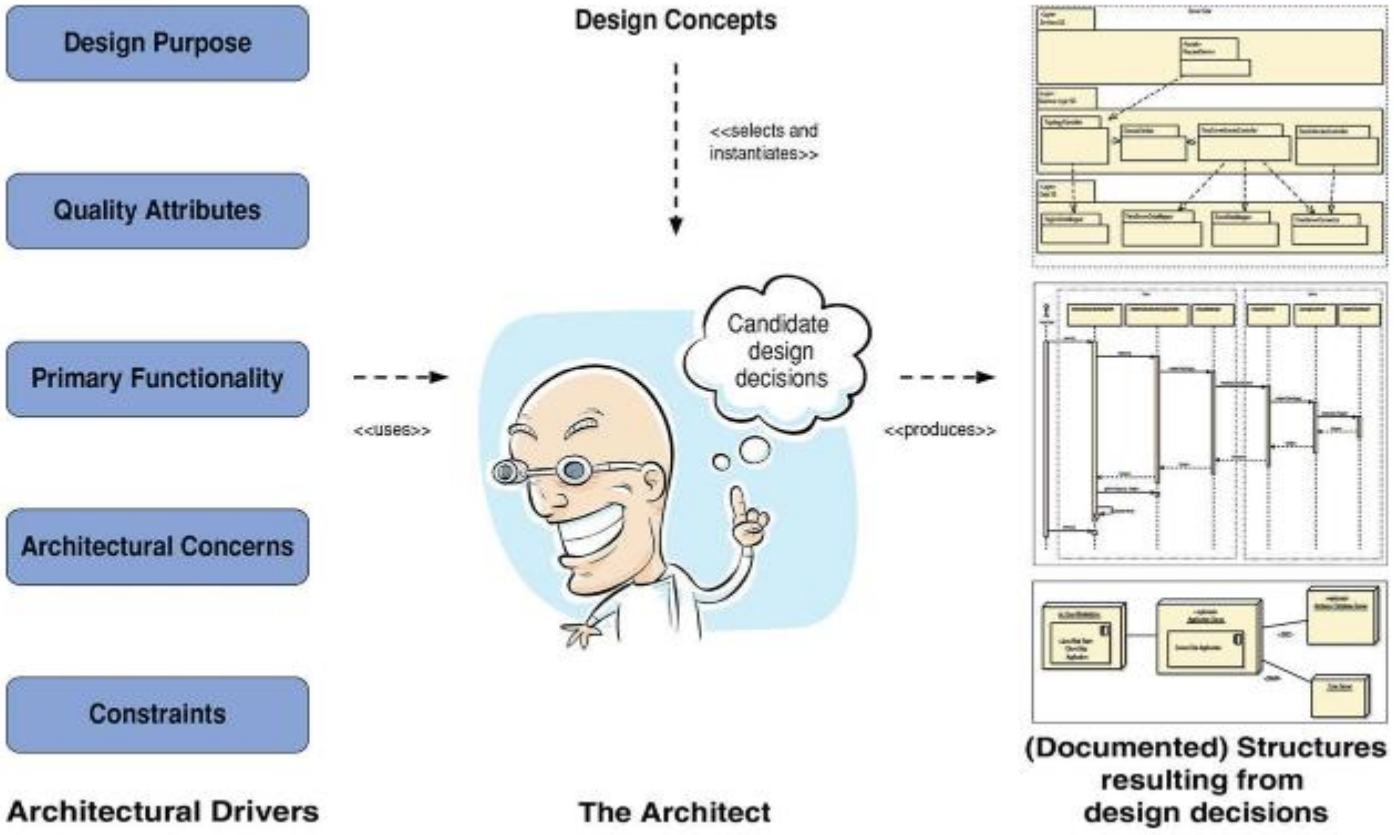
❑ Chap 8: Designing Software Architecture

❖ Architecture design overview

- The good news to software architect is that, there are proven designs and design fragments, or building blocks that we call design concepts , that can be reused and combined to reliably achieve these goals.

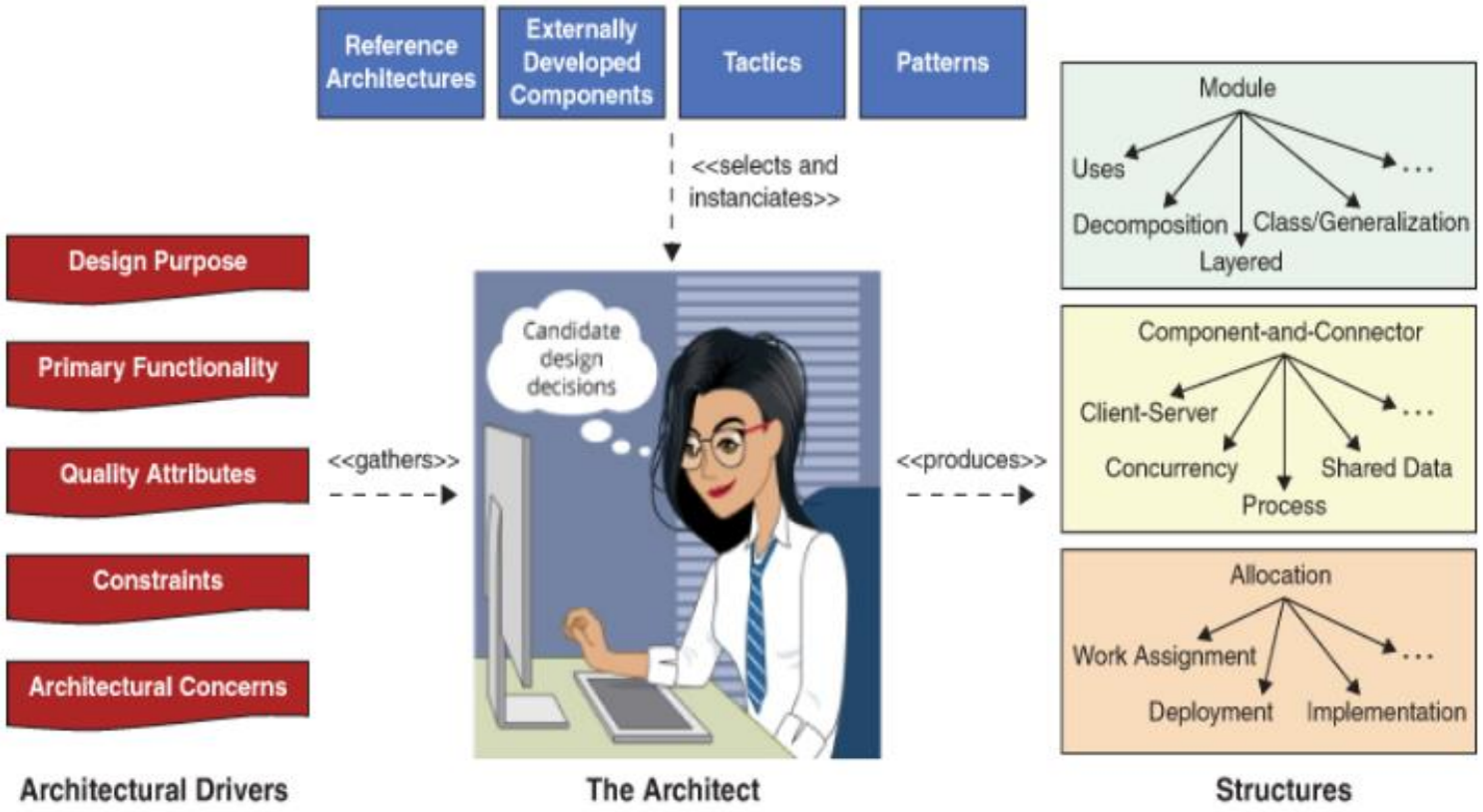
❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity(1)



❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity(2)



❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Architectural decisions do occur at three levels of design.**

1. **Architectural design – level one**
2. **Element interaction design – level two**
3. **Element internals design – level three**

However, level one can delve as deeply as elements internals design to achieve a particular driver.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural decisions do occur at three levels of design(1).
  - **One can not design an element's internals until the element themselves have been defined**
  - **One cannot reason about interaction until several elements and some patterns of interaction among them have been defined.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural decisions do occur at three levels of design(2).
- **Some will use the term - high level design and detailed design/ low level design.**
- **we prefer architectural design , element interaction design and element internals design**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural design is therefore a key step to achieving your product and project goals.
- The decision you take will have an implications for the achievement of these goals.
- The choice of a particular reference architecture may provide a good foundation for achieving your goal because of their familiarities.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **All architectures are design, but not all designs are architectures**
- **What makes a decision architectural?**

✓ If the decision taken has consequences and those consequences matter to the achievement of an architectural driver(*does it have consequence on performance, availability, modifiability*…).

- **No decision is inherently architectural or non-architectural.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ Element Interaction Design – 2nd level of design

- Architectural design identifies only a subset of elements that are part of the system's structure.
- Architect focus on the primary functionality of the system
- Primary use cases give the primary functionality
- The elements that support the non primary use case and their interfaces are part of the element interaction design

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Element Interaction Design – 2nd level of design(1)**
  - The location and relationships of these elements are constrained by the decisions that were made during architectural design
  - The elements can be unit of work called modules assigned to an individual or to a team.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ Element Internals Design – 3rd level of design

- Conducted as part of the element development activities
- The internals of the elements in the previous design level are established so as to satisfy the element's interface.
- Also called detailed design as the design is designing the internals of modules

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ Architectural Drivers

- We categorize “what” and “why” questions as architectural drivers.
- The Architectural drivers used

✓ Design purpose, quality attributes, primary functionality , architectural concerns, and constraints.

✓ The consideration drives and shapes architecture’s success.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ Architectural Drivers – Design Purpose
  - ▪ **You need a clear purpose of the design**
    - ✓ When and why are you doing this architecture design?
    - ✓ Which business goals is the organization most concerned about at this time?

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Architectural Drivers – Design Purpose(1)**

1. You may be doing architecture design as part of the project proposals.

- ✓ For pre-sales process in a consulting organization
- ✓ For internal project selection and prioritization in a company.
- Such will not be detailed it purpose is to understand and break down the architecture in sufficient details that the unites of work are understood and hence may be estimated.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Design Purpose(2)

2. Doing architecture design as part of the process of creating an exploratory prototype.

✓ Incase the purpose is not so much to create a releasable or reusable system, but rather to explore the domain, to explore new technology, to place something executable in front of a customer to elicit rapid feedback, or to explore a quality attribute.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Architectural Drivers – Design Purpose(3)**

3. You may be designing your architecture during development

- ✓ This could be for an entire new system, a portion of a new system, or existing system that is being refactored or replaced.
- ✓ In this case, the purpose is to do enough design work to satisfy requirements, guide system construction and work assignments, and prepare for an eventual release.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Design Purpose(4)

- **It will after the design process and the outputs of the design so the architect should be clear about these goals and should communicate them and establish a clear design purpose before beginning the design process.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – **Quality Attributes(1)**

- **These are measurable or testable properties of a system that are used to indicate how well the system satisfies the needs of its stakeholders.**
- **They are quality attributes that shape the architecture most significantly.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – **Quality Attributes(2)**
  - **You must worry about eliciting, specifying, prioritizing and validating the quality attributes.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(3)

- **Widely use technique that help here**

✓ **QAW (Quality Attribute Workshop)**

- **Facilitated brainstorming session involving a group of system stakeholders that covers the bulk of activities of eliciting, specifying, prioritizing and achieving consensus on quality attributes.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(3)

- **Widely use technique that help here(1)**

✓ **Mission Thread workshop**

- **Serve the same purpose as QAW , but for a system of systems**

✓ **The Utility Tree can be used by the architect to prioritize quality attribute requirement according to their technical difficulty and risk**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(3)

- Widely use technique that help here(2)

✓ **The use of scenario or set of scenarios is one of the best ways to discuss, document, and prioritize quality attribute requirements.**

✓ **A scenario describes the system's response to some stimulus.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Architectural Drivers – Quality Attributes(3)**

- **Quality attribute scenario**

✓ **Pairing of a stimulus with a response**

Example : suppose you are building a video game

"This game shall change view modes when the user presses the <C> button"

- The functional requirement, if its important needs to be associated with quality attributes requirements

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(3)

- **Quality attribute scenario(1)**

Example : suppose you are building a video game

- ✓ **How fast should the function be?**
- ✓ **How secure should the function be?**
- ✓ **How modifiable should the function be?**

To address this problem, we use a scenario to describe a quality attribute requirement.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – **Quality Attributes(3)**
  - **Quality attribute scenario(2)**

✓ **It's a short description of how a system is required to respond to some stimulus.**

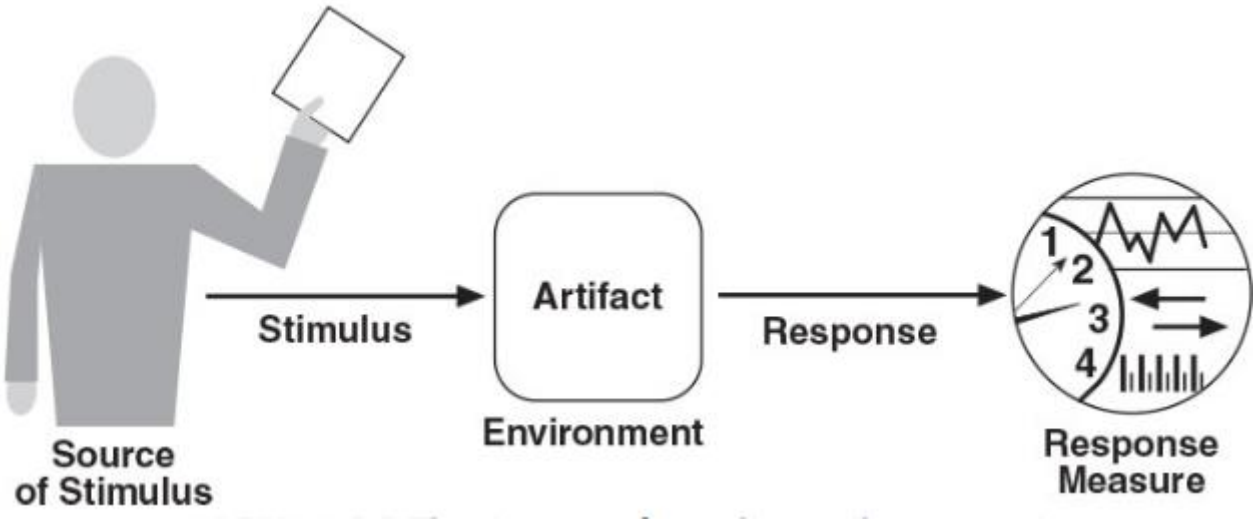
✓ **It adds the source of stimulus, the artifact affected is the entire system, and the environment.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ Architectural Drivers – Quality Attributes(3)
- ▪ Quality attribute scenario(3)

The six parts of a quality attribute scenario



❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Architectural Drivers – Quality Attributes(3)**
- ▪ **Quality attribute scenario(4) - Prioritization**

✓ *Consider two dimensions that are associated with each scenario and that assigned a rank of importance.*

1st Dimension : Corresponds to the importance of the scenario with respect to the success of the system. This is ranked by the customer.

2nd Dimension : Corresponds to the degree of technical risk associated with the scenario. This is ranked by the architect.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Architectural Drivers – Quality Attributes(3)**

- **Quality attribute scenario(4) – Prioritization(1)**

✓ *A low/medium/high (L/M/H) scale is used to rank both dimensions.*

✓ *Once it has been ranked, scenario are prioritized by selecting that have a consideration of (H,H), (H,M), or (M,H) rankings.*

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Architectural Drivers – Quality Attributes(4)**
- ▪ **You can always elicit at lease a range of possible responses**

✓ **Instead of saying the system should be "fast" ask the stakeholder**

❖ **if a 10-second response time is acceptable**

❖ **if that is unacceptable, ask if 5 seconds is ok, or 1 second and you will find out that the users knows more than they realize.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(5)

Assignment –

- ➢ **Research on Quality attribute workshop and the Utility Tree**
- ➢ **What are the steps in QAW?**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ Architectural Drivers – Quality Attributes(6)

➢ **Quality attribute workshop and the Utility Tree**

✓ The steps of the QAW

1. QAW Presentation and introductions
2. Business goals Presentation
3. Architectural plan presentation
4. Identification of drivers
5. Scenario Brainstorming
6. Scenario consolidation
7. Scenario prioritization
8. Scenario refinement

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Quality Attributes(7)

➢ **Class work assignment**

In groups of 3's members, Carry out a QAW and present the Utility tree of the Moodle system

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Primary Functionality

➢ Functionality is the ability of the system to do the work for which it was intended.

➢ The way the system is structured does not affect the functionality.

➢ You can have all the functionality of a system coded in one module.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Primary Functionality(1)

➢ In terms of architectural design, allocation of functionality to elements is what matters.

➢ A good architecture is one in which the most common changes are localized in a single or a few elements, and hence easy to make.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Primary Functionality(2)

➢ **Consider primary functionality when designing an architecture.**

➢ **Primary functionality is functionality that is critical to achieve the business goals that motivate the development of the system.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Primary Functionality(3)

➢ You need to think how functionality will be allocated to elements (usually modules) to promote modifiability or reusability

➢ Some quality attribute scenarios are directly connected to the primary functionality.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Architectural Drivers – Primary Functionality(4)**

➢ **Refactoring**

✓ **If you refactor a software architecture , what you are doing is maintaining the same functionality but changing some quality attributes that you care about,**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Architectural concerns

➢ Additional aspects you need to consider as part of the architectural design.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Architectural concerns-1

➢ There are several concerns

✓ General concerns – broad issues with creating the architecture, such as establishing an overall system structure, the allocation of functionality to modules, the allocation of modules to teams, organization of the code base, startup and shutdown, and supporting delivery, deployment and updates.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – Architectural concerns-2

➢ There are several concerns

✓ Specific concerns – detailed system –internal issues such as exception management, dependency management , configuration, logging, authentication, authorization, cashing ,,,

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – **Architectural concerns-3**

➢ **There are several concerns**

✓ **Internal requirements – May address aspects that facilitate development, deployment, operation, or maintenance of the system.**

✓ **Issues – Result from analysis activities, such as design review**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – constraints
  - Write out the constraints ; it will be use as part of the architectural design process.
  - It may be a technical constraint or non technical
  - A decision over which you have tittle or no control as an architect.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – constraints(1)
  - It may be a form of mandated technologies, system you need to integrate, laws and standards that must be compiled with, the abilities and availability of your developers, deadlines that are non-negotiable, backward combability with order versions of system..

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural Drivers – constraints(2)
  - Example –

✓ Technical constraints is the use of open source technologies,

✓ Whereas a nontechnical constraint is that the system must be delivered by December 15.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts : - building blocks for creating structures
- Design is not random but planned, intentional , rational and directed.
- Different types of design concepts exist

✓ *Reference architecture, deployment patterns, architectural patterns , tactics and externally developed components.*

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Design Concepts : - Reference Architectures**
  - **Blueprints that provide an overall logical structure for particular types of applications.**
  - **It's a reference model mapped onto one or more architectural patterns.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Design Concepts : - Reference Architectures(1)**
  - ▪ **Example**
    - ✓ **Reference architecture for the development of web application is showed on the next slide.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Design Concepts : - Reference Architectures(2)**
- ▪ **Example**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Design Concepts : - Reference Architectures(3)**
  - **Reference architecture may be confused with architectural style.**
  - **Architectural style (such as pipe and filters) define types of components and connectors in a specific topology that are useful for structuring an application either logically or physically.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ Design Concepts : - Reference Architectures(4)

- Reference architecture provide a structure for application in a specific domains, and they may embody different styles.
- While architectural styles tend to be popular in academia, reference architectures seems to be preferred by practitioners.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ **Design Concepts :Architectural design Patterns**
- ▪ **Its solutions to recurring design problems that exist in a defined context.**
- ▪ **For example architectural pattern that is useful for structuring the system, the layer pattern.**
- ▪ **When you choose a pattern such as this one, you must decide how many layers you will need for your system.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Architectural design Patterns
  - Example:

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Design Concepts :Deployment Patterns**
  - **It provide models on how to physically structure the system to deploy it.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Tactics

- When architects use collections of fundamental design techniques to achieve a response for particular quality attributes, we call the architectural design primitive tactics.
- Techniques that architects have been using for years.
- Tactics is not invented but capture what architect have actually have done in practice.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Design Concepts :Externally Developed components**
- **Pattern and tactics are abstract in nature and when designing the architecture you need to make the concepts concrete and closer to actual implementation. We have two ways**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Externally Developed components(1)
  - **We have two ways (Buy or Build)**
    - ✓ Code the elements obtained from tactics and patterns.
    - ✓ Associate technologies with one or more of those elements in the architecture.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Externally Developed components(2)
  - **Not created as part of the development project**
    - ➢ Technology families ; RDBMS , ORM
    - ➢ Products; self-contained functional piece of software - Oracle or MS SQL Server
    - ➢ Application frameworks ;
    - ➢ Platforms ; Java, .Net or Google Cloud

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Externally Developed components(3)
  - **Criteria of selection**
    - ✓ Problem that it addresses : Is it something specific, such as a framework for OO,
    - ✓ Cost : what is the cost of the license and if its free, what is the cost of support and education.

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **Design Concepts :Externally Developed components(4)**
  - **Criteria of selection**
    - ✓ **Type of license : Does it have a license compatible with the project**
    - ✓ **Support**
    - ✓ **Learning curve**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Design Concepts :Externally Developed components(5)
  - **Criteria of selection**
    - ✓ Popularity
    - ✓ Compatibility and ease of integration
    - ✓ Support for critical quality attributes
    - ✓ Size

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- Architectural design decision
  - Design is a process of making decisions
  - The act of act of making decisions is a process, not a moment of time

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- ○ The Architecture Design Process
- ➢ The need for a principle method
- ✓ The question is , how do you actual perform design?
- ✓ Performing design to ensure that the drivers are satisfied requires a principled method

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **"First things first"**

➢ **Determine the scope of the system**

✓ **– What is inside and what is outside of the system you are creating,**

✓ **- And which external entities the system will interact with.**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

- **"First things first"**

➢ **Determine the scope of the system (context)**

✓ **– What is inside and what is outside of the system you are creating,**

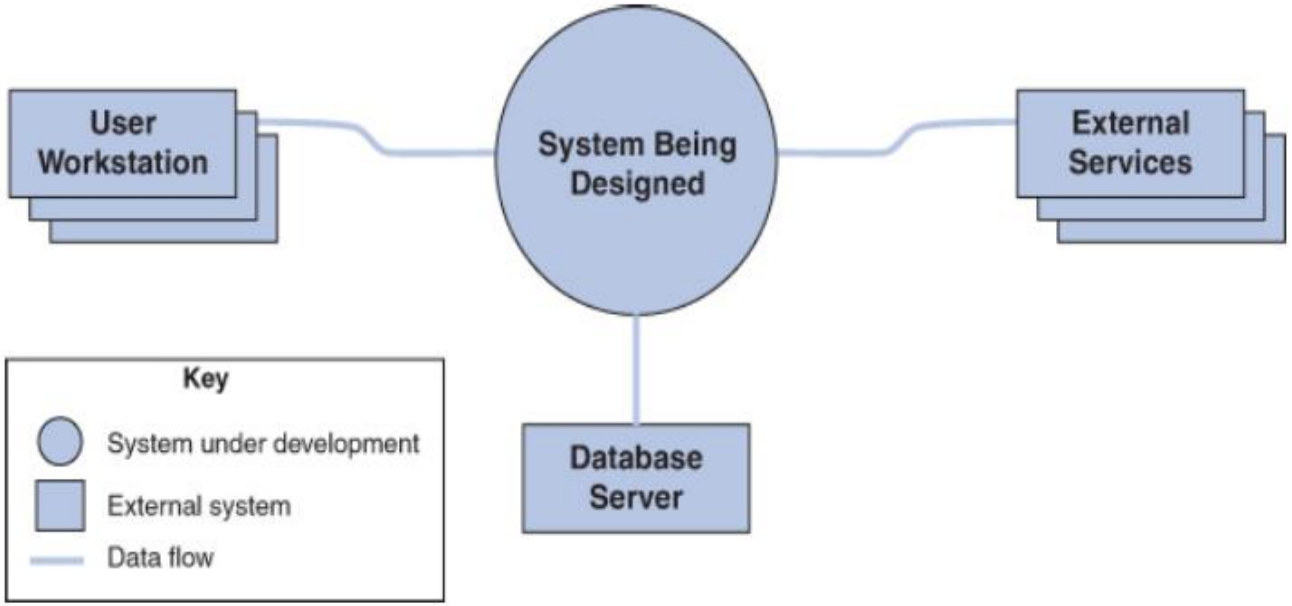
✓ **- And which external entities the system will interact with**

❑ Chap 8: Designing Software Architecture

❖ Overview of the architecture design activity

○ "First things first"(1)

➢ The context can be represented by a context diagram: Example of a system context diagram



❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

- ADD allows an architecture to be designed in a systematic , repeatable, and cost-effective way.
- With ADD, architecture design is performed in rounds, each of which may consist of series of design iterations.

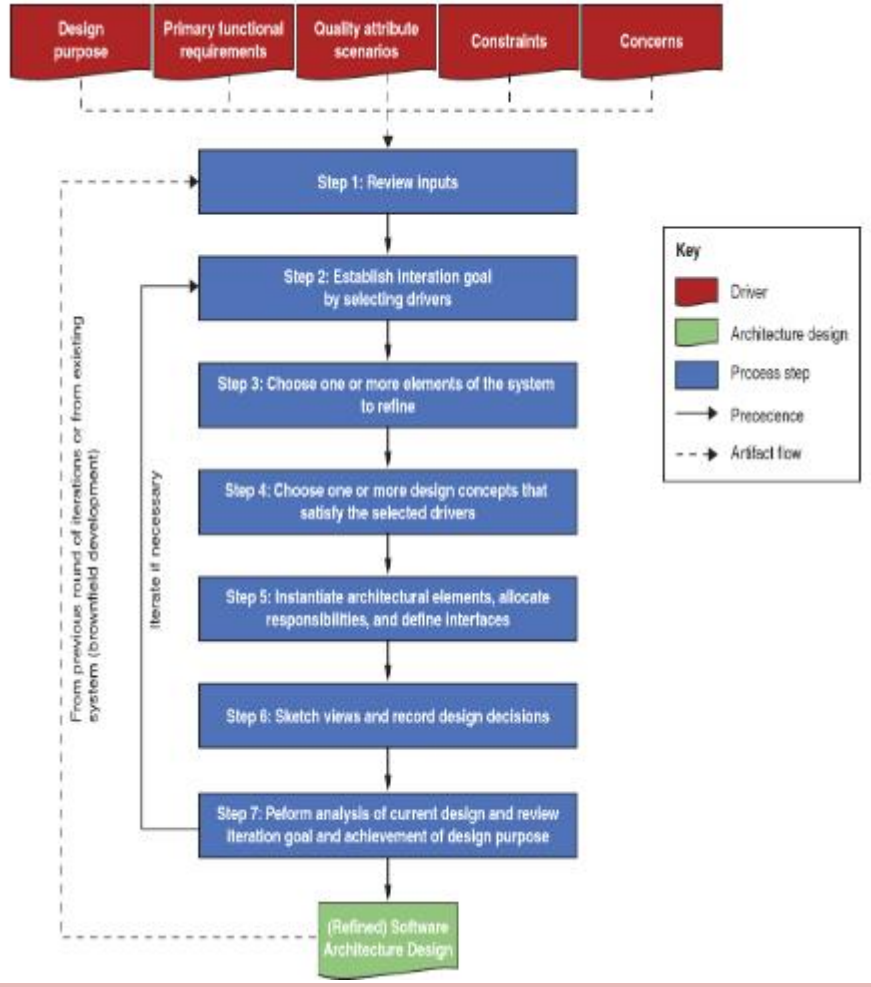
❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

- A round comprises the architecture design activities performed within a development cycle.
- Through one or more iteration, you produce an architecture that suits the established design purpose for this round.
- Within each iteration, a set of design steps is performed.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)



❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

1. Review inputs
2. Establish interaction goal by selecting drivers
3. Choose one or more elements of the system to refine
4. Choose one or more design concepts that satisfy the selected drivers
5. Instantiate architectural elements, allocate responsibilities, and define interfaces
6. Sketch views and record design discipline
7. Perform analysis of current design and review iteration goals and achievement of design purpose.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs

- Before starting a design round,

✓ You need to ensure that the architectural drivers (the inputs to the design process) are available and correct.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs

- Requirement drivers such as,

✓ The purpose of the design round

✓ The primary functional requirement

✓ The primary quality attribute (QA) scenario

✓ Any constraints

✓ Any concerns

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs(*why?*)

- Why capture the design purpose?
- Why capture primary functional requirements?
- Why primary quality attribute (QA) scenario?
- Any constraints
- Any concerns

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs(*1*)

- **Why capture the design purpose?**

✓ **To make sure the goal of the iteration is clear**

The purpose may (Examples)

1. **To produce a design for early estimation**
2. **To refine an existing design to build a new increment**
3. **To design and generate a prototype to mitigate certain technical risk.**

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs(*why?*)

- **Why capture primary functional requirements?**

✓ **Typically captured as a set of use cases or user stories.**

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs(*why?*)

- **Why primary quality attribute (QA) scenario?**

✓ **Captured QA should be prioritized, ideally by the most important project stakeholder**

✓ **You need to employ several techniques to elicit and prioritize them.**

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 1: Review inputs

- The drivers becomes part of an architectural design backlog that you should use to perform the different design iterations.

Example:

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 2: Establish iteration goal by selecting drivers.

- Each design iteration focuses on achieving a particular goal.
- Goal involves designing to satisfy a subset of the drivers.
- You need to establish a goal before you start a particular design iteration.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 3: Choose one or more elements of the system to refine

- Satisfying drivers requires you to make architectural design decisions which manifest themselves in one or more architectural structures.
- These structures are composed of interrelated elements-modules and/or components .

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 3: Choose one or more elements of the system to refine(1)

- The elements are generally obtained by refining other elements that you previously identified in an earlier iteration.
- Refining may mean decomposing them (top-down approach) or combination of elements (button-up approach)

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 3: Choose one or more elements of the system to refine(2)

- The elements you will select has ones involved in the satisfaction of specific drivers.
- When the design address an existing system, you need to have a good understanding of the elements that are part of the as-built architecture of the system.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 4: Choose one or more design concepts that satisfy the selected drivers

- Choose the design concept(s) of the type of design concepts available – for example , tactics, patterns , reference architectures, and externally developed components.
- Choose from a number of alternatives that need to be analysed before making the choice.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces.

- You have to make another type of design decision : how to instantiate elements out of the design concepts that you just selected.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces(1).

- How do you instantiate elements out of the elements out of the design concepts selected?

✓ For example : If you selected the layers pattern as a design concept, you must decide how many layers will be used, and their allowed relationships, since the pattern itself does not precise these.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces(2).

- After instantiating the elements, you then need to allocate responsibilities to each of them
- Example : in an app, at least three layers are usually present: presentation, business and data layers.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces(3).

- A least three layers are usually presentation , business and data layers.
- The responsibilities of these layers differ

✓ Presentation manages all user interactions

✓ Business layer manages application logic

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces(4).

- The elements that are instantiated need to be connected for their communication
- This requires the existence of relationships between the elements and the exchange of information through some kind of interface.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 5: instantiate architectural elements, allocate responsibilities, and define interfaces(5).

- The interface is then specified indicating how information should flow between the elements.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 6: Sketch views and record design decision

- At this stage you have finish the design activities of the iteration but you have to present the view representing the structures you have created
- The views made in the iteration may be revisited in the next iteration.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose.

- You must have created a partial design that addresses the goal established for the iteration.
- You can perform the analysis by reviewing the sketches of the views and design decisions that you captured.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

Iterate if Necessary

- You should perform additional iterations and repeat step 2-7 for every driver that was considered.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- They are dozen of design patterns and externally developed components that you could use to address any particular issue.
- You can and often will use and combine different types of design concepts

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- Example : To build a security driver, you might employ a security pattern, a security tactic, a security framework, or some combination of these.
- What are the techniques you can use

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- Technique to use(1)

✓ **Make use of existing best practices** –

✓ **Make use of your own knowledge and experiences** – if the system you are developing is similar to other systems you have designed in the past, you will probably want to begin with some of the design concepts that you have used before.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- Technique to use(2)

✓ **Make use of the knowledge and experience of others**

✓ You have a background and knowledge that you have gained through the years

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- Selection of Design Concepts

✓ When you have identify several alternatives design concepts, you need to select which one of the alternatives.

✓ Create a table that list the pros and the cons associated with each alternative and selecting one of the alternatives based on those criteria, such as cost, and SWOT(strength, weakness, opportunities, threats)

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 4: Choose Design concepts

- Selection of Design Concepts

✓ Also consider the constraints when taking such decisions.

✓ You can create prototypes

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- The structures could be grouped into three major categories

1. **Module structures** – which are composed of elements that exist at development time, such as files , modules and classes.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- The structures could be grouped into three major categories(1)

2. Component and connector(C&C) structure – which are composed of elements that exist at runtime, such as processes and threads.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- The structures could be grouped into three major categories(1)

3. **Allocation structure–** which are composed of both software elements (from a module or C&C structure) and non-software elements that may exist both at development and at runtime, such as file systems, hardware and development teams.

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- Instantiating Elements

✓ Reference architecture

✓ Pattern

✓ Tactics

✓ Externally developed components

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- **Associating responsibilities and identifying properties**

✓ **The important aspect that you need to consider when instantiating designing concepts is the properties of the elements.**

❑ Chap 8: Designing Software Architecture

❖ Attribute-Driven Design(ADD)

More on ADD step 5: Producing Structures

- **Establishing relationship between the elements**

✓ **External interfaces are interfaces between the elements that result from the instantiation of design concept.**

✓ **To identify the relationships and the interface details, you need to understand how the elements interact with each other to support use cases or QA scenarios.**

Welcome!

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

QUESTIONS