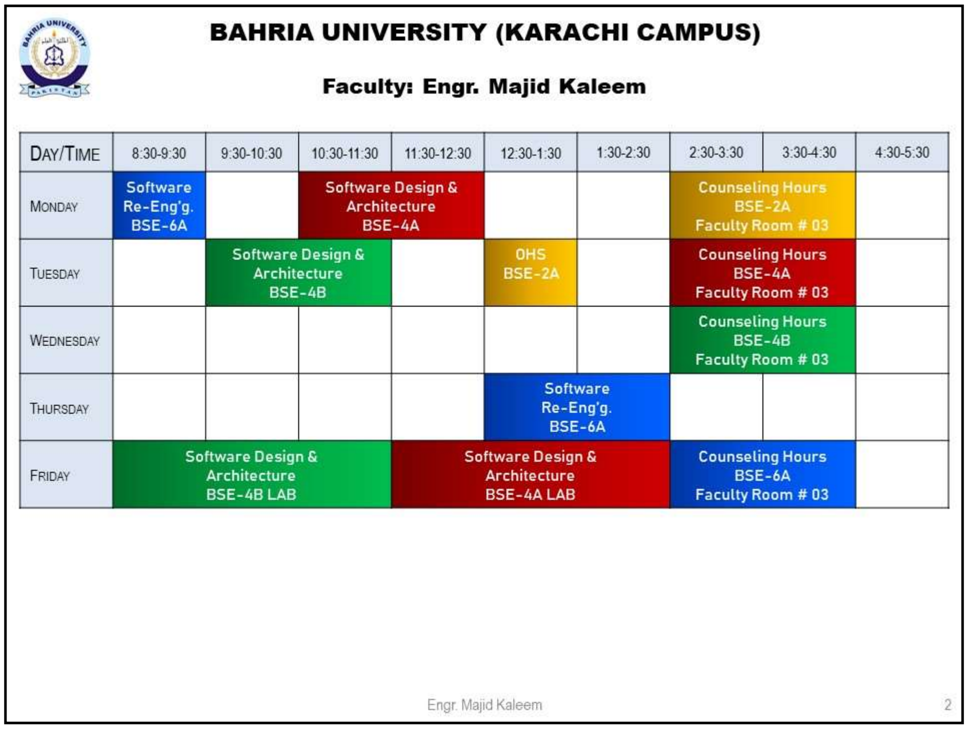
Bahria University-Karachi Campus

Software Design & Architecture

Lecture 1 of 16

Engr. Majid Kaleem

Engr. Majid Kaleem 1

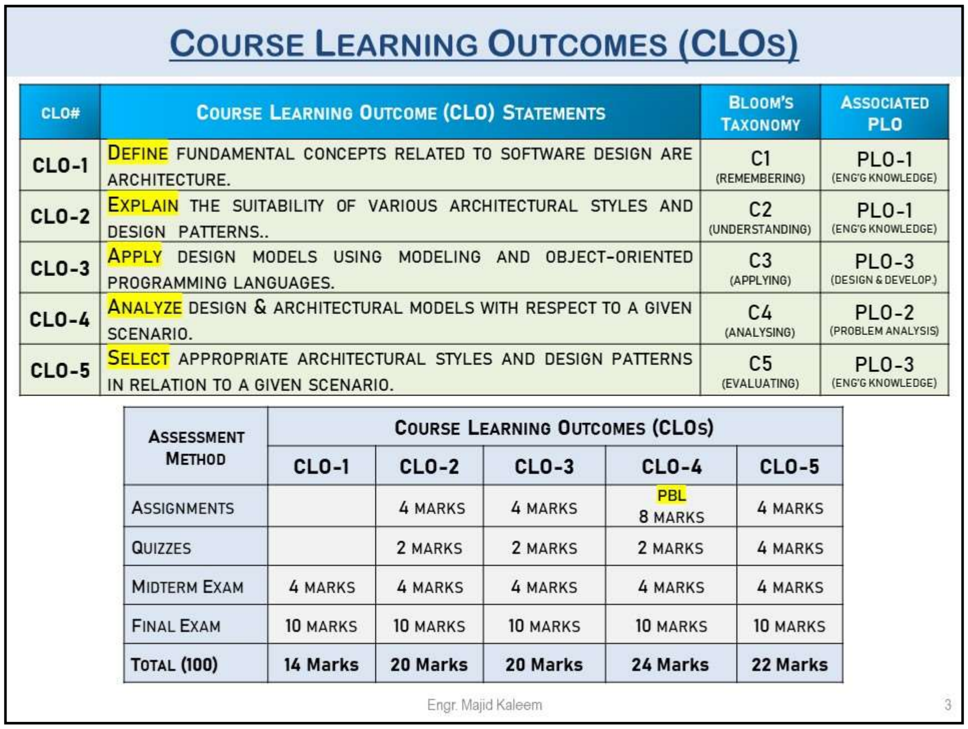


BAHRIA UNIVERSITY (KARACHI CAMPUS)

Faculty: Engr. Majid Kaleem

DAY/TIME	8:30-9:30	9:30-10:30	10:30-11:30	11:30-12:30	12:30-1:30	1:30-2:30	2:30-3:30	3:30-4:30	4:30-5:30
MONDAY	Software Re-Eng'g. BSE-6A		Software Design & Architecture BSE-4A				Counseling Hours BSE-2A Faculty Room # 03		
TUESDAY		Software Design & Architecture BSE-4B			OHS BSE-2A		Counseling Hours BSE-4A Faculty Room # 03		
WEDNESDAY							Counseling Hours BSE-4B Faculty Room # 03		
THURSDAY					Software Re-Eng'g. BSE-6A				
FRIDAY	Software Design & Architecture BSE-4B LAB			Software Design & Architecture BSE-4A LAB			Counseling Hours BSE-6A Faculty Room # 03		

Engr. Majid Kaleem 2

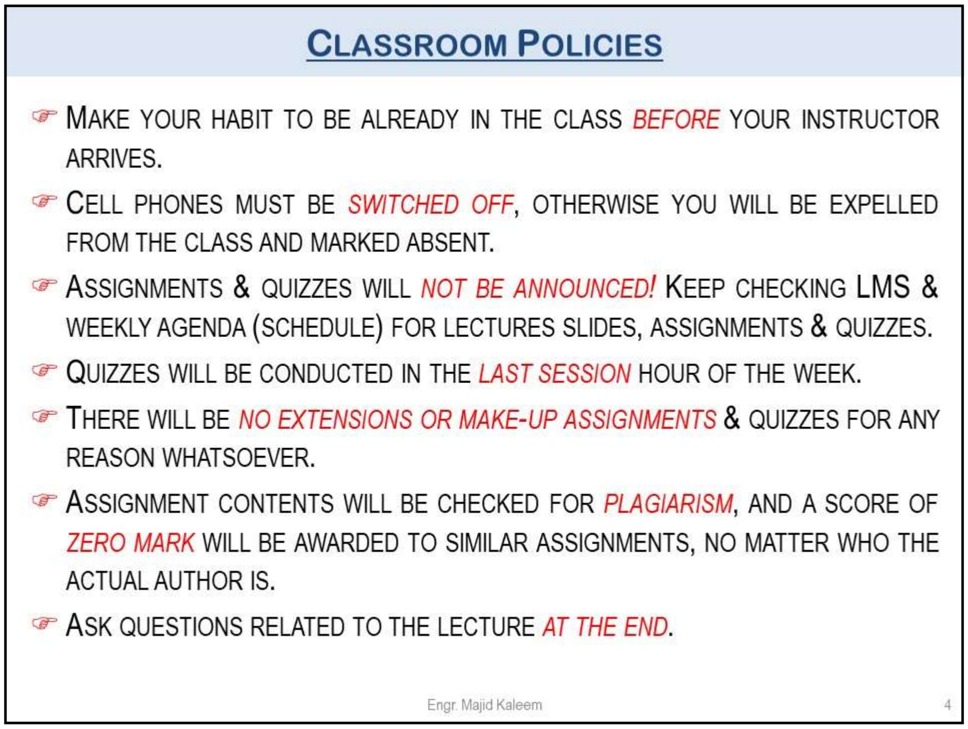


Course Learning Outcomes (CLOs)

CLO#	Course Learning Outcome (CLO) Statements	Bloom's Taxonomy	Associated PLO
CLO-1	Define fundamental concepts related to software design are architecture.	C1 (Remembering)	PLO-1 (Eng'g Knowledge)
CLO-2	Explain the suitability of various architectural styles and design patterns..	C2 (Understanding)	PLO-1 (Eng'g Knowledge)
CLO-3	Apply design models using modeling and object-oriented programming languages.	C3 (Applying)	PLO-3 (Design & Develop.)
CLO-4	Analyze design & architectural models with respect to a given scenario.	C4 (Analysing)	PLO-2 (Problem Analysis)
CLO-5	Select appropriate architectural styles and design patterns in relation to a given scenario.	C5 (Evaluating)	PLO-3 (Eng'g Knowledge)

Assessment Method	Course Learning Outcomes (CLOs)				
	CLO-1	CLO-2	CLO-3	CLO-4	CLO-5
Assignments		4 marks	4 marks	PBL 8 marks	4 marks
Quizzes		2 marks	2 marks	2 marks	4 marks
Midterm Exam	4 marks	4 marks	4 marks	4 marks	4 marks
Final Exam	10 marks	10 marks	10 marks	10 marks	10 marks
Total (100)	14 Marks	20 Marks	20 Marks	24 Marks	22 Marks

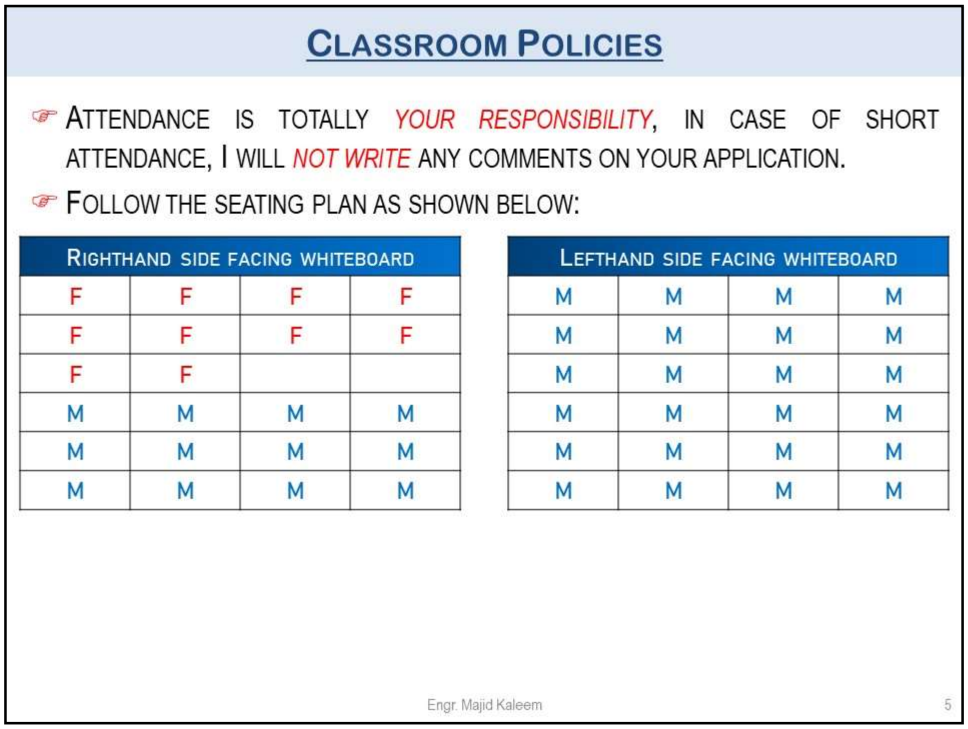
Engr. Majid Kaleem 3



Classroom Policies

- Make your habit to be already in the class *before* your instructor arrives.
- Cell phones must be *switched off*, otherwise you will be expelled from the class and marked absent.
- Assignments & quizzes will *not be announced!* Keep checking LMS & weekly agenda (schedule) for lectures slides, assignments & quizzes.
- Quizzes will be conducted in the *last session* hour of the week.
- There will be *no extensions or make-up assignments* & quizzes for any reason whatsoever.
- Assignment contents will be checked for *plagiarism*, and a score of *zero mark* will be awarded to similar assignments, no matter who the actual author is.
- Ask questions related to the lecture *at the end*.

Engr. Majid Kaleem 4

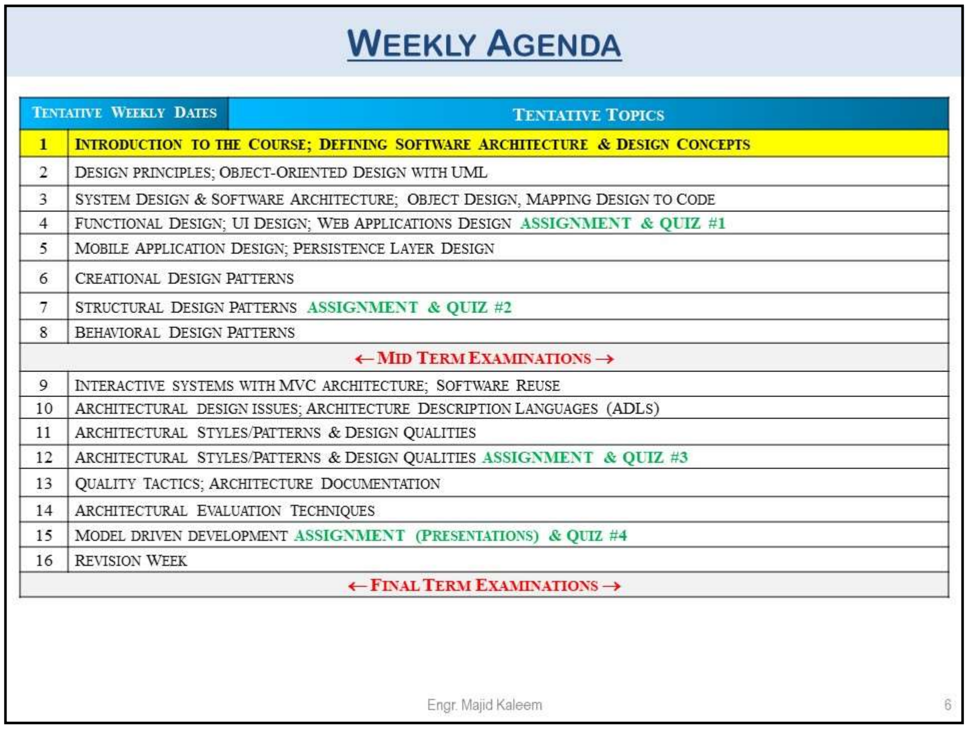


Classroom Policies

- Attendance is totally *your responsibility*, in case of short attendance, I will *not write* any comments on your application.
- Follow the seating plan as shown below:

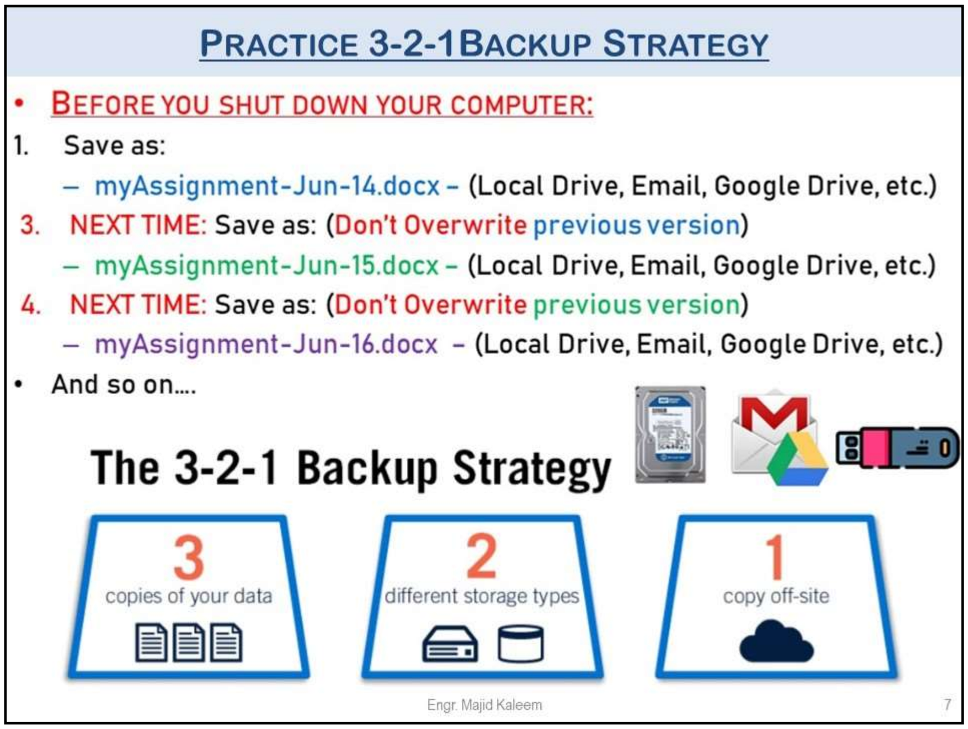
Righthand side facing whiteboard			
F	F	F	F
F	F	F	F
F	F		
M	M	M	M
M	M	M	M
M	M	M	M

Lefthand side facing whiteboard			
M	M	M	M
M	M	M	M
M	M	M	M
M	M	M	M
M	M	M	M
M	M	M	M



Weekly Agenda

Tentative Weekly Dates	Tentative Topics
1	**Introduction to the Course; Defining Software Architecture & Design Concepts**
2	Design principles; Object-Oriented Design with UML
3	System Design & Software Architecture; Object Design, Mapping Design to Code
4	Functional Design; UI Design; Web Applications Design **Assignment & Quiz #1**
5	Mobile Application Design; Persistence Layer Design
6	Creational Design Patterns
7	Structural Design Patterns **Assignment & Quiz #2**
8	Behavioral Design Patterns
	← Mid Term Examinations →
9	Interactive systems with MVC architecture; Software Reuse
10	Architectural design issues; Architecture Description Languages (ADLs)
11	Architectural Styles/Patterns & Design Qualities
12	Architectural Styles/Patterns & Design Qualities **Assignment & Quiz #3**
13	Quality Tactics; Architecture Documentation
14	Architectural Evaluation Techniques
15	Model driven development **Assignment (Presentations) & Quiz #4**
16	Revision Week
	← Final Term Examinations →



Practice 3-2-1Backup Strategy

- **Before you shut down your computer:**

1. Save as:
   - myAssignment-Jun-14.docx – (Local Drive, Email, Google Drive, etc.)
3. NEXT TIME: Save as: (Don't Overwrite previous version)
   - myAssignment-Jun-15.docx – (Local Drive, Email, Google Drive, etc.)
4. NEXT TIME: Save as: (Don't Overwrite previous version)
   - myAssignment-Jun-16.docx – (Local Drive, Email, Google Drive, etc.)

- And so on….

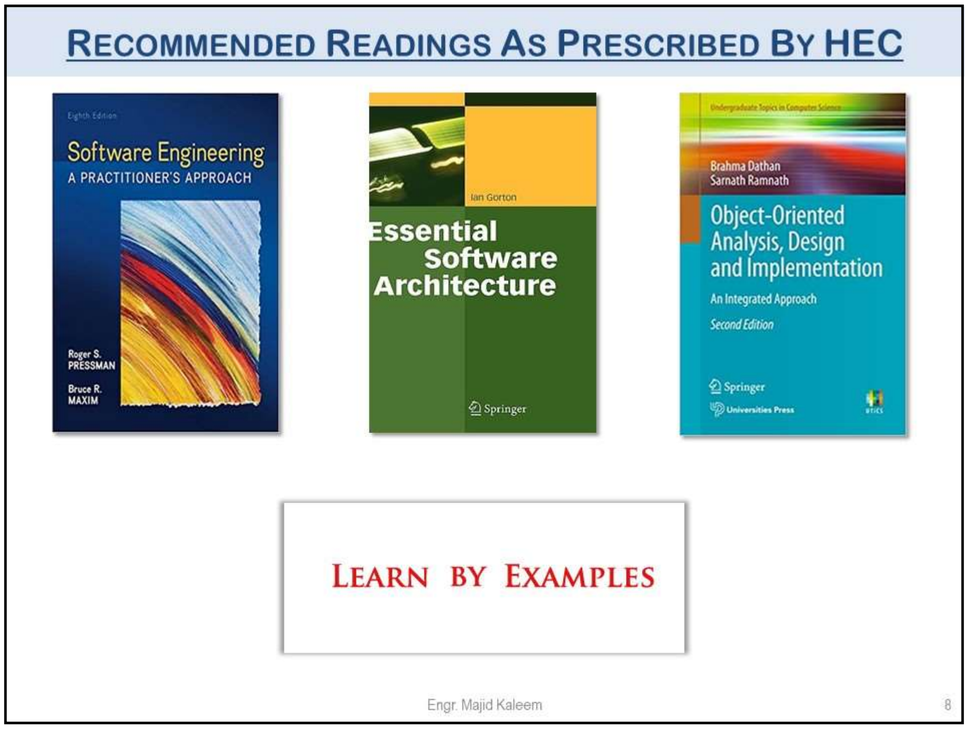
The 3-2-1 Backup Strategy

3 copies of your data

2 different storage types

1 copy off-site

Engr. Majid Kaleem 7



Recommended Readings As Prescribed By HEC

Eighth Edition
Software Engineering
A Practitioner's Approach
Roger S. Pressman
Bruce R. Maxim

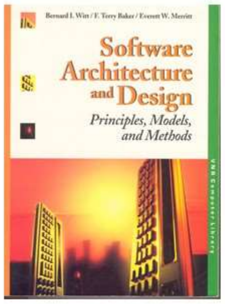
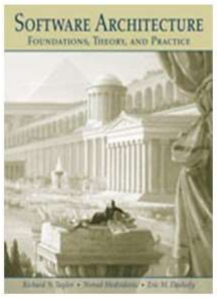
Ian Gorton
Essential Software Architecture
Springer

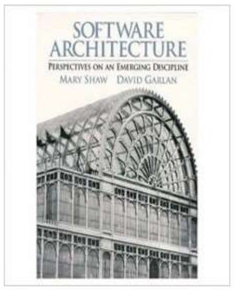
Undergraduate Topics in Computer Science
Brahma Dathan
Sarnath Ramnath
Object-Oriented Analysis, Design and Implementation
An Integrated Approach
Second Edition
Springer
Universities Press

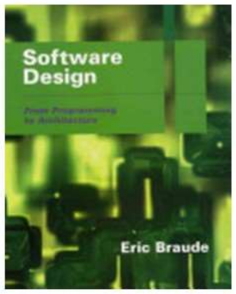
Learn by Examples

Engr. Majid Kaleem 8

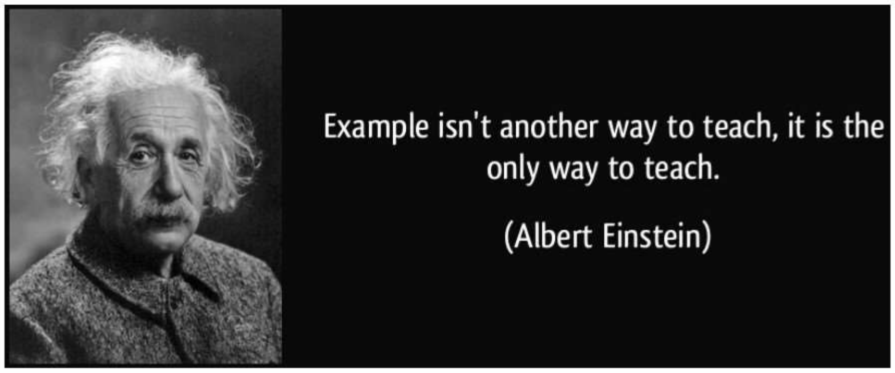
Reference Readings


Software Architecture and Design
Principles, Models, and Methods






That's My Style of Teaching!

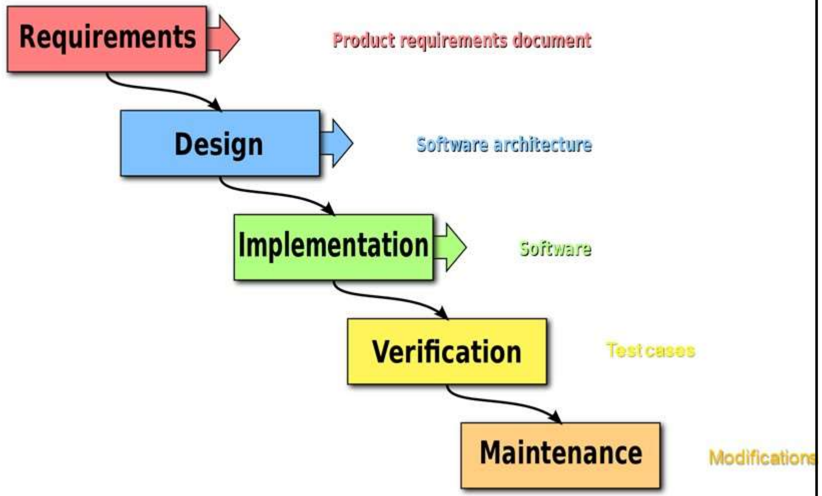

Example isn't another way to teach, it is the only way to teach.

(Albert Einstein)

https://cms.bahria.edu.pk/

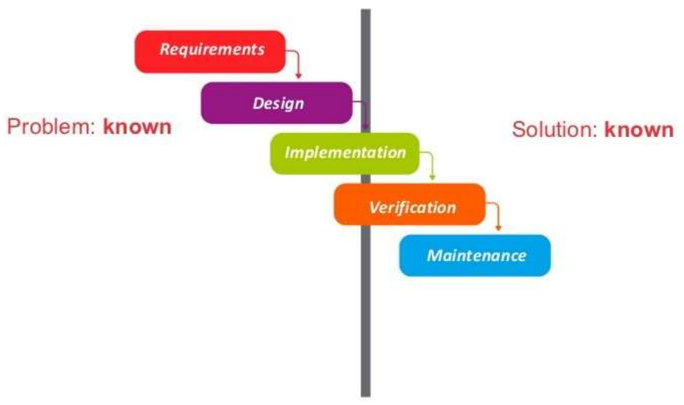
MAIN PHASES OF SOFTWARE PROCESS

Remember this?



Main Phases of Software Process

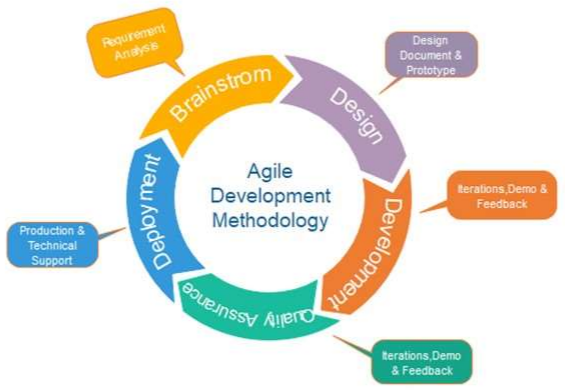
Remember this?



Engr. Majid Kaleem 13

Main Phases of Software Process

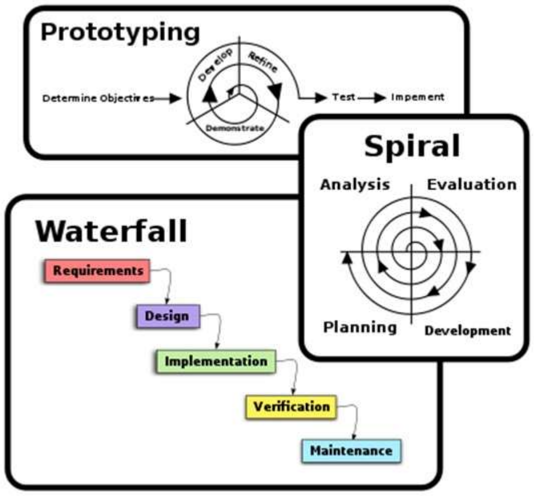
Remember this?


Agile Development Methodology
Brainstrom
Design
Development
Quality Assurance
Deployment
Requirement Analysis
Design Document & Prototype
Iterations,Demo & Feedback
Iterations,Demo & Feedback
Production & Technical Support

Engr. Majid Kaleem 14

Main Phases of Software Process

Remember this?

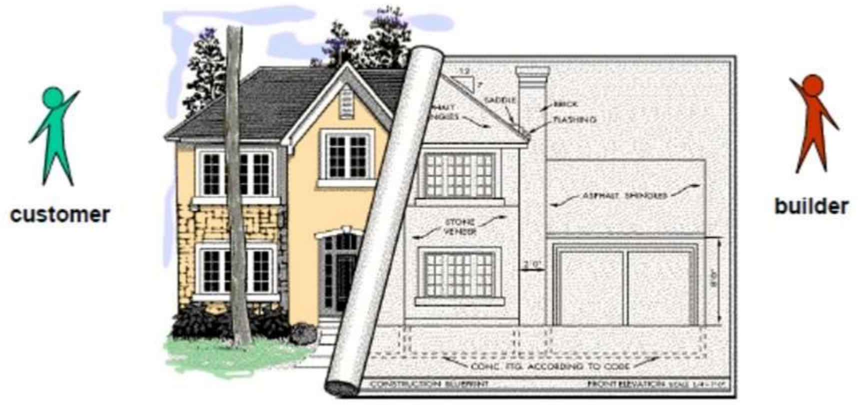


Main Phases of Software Process

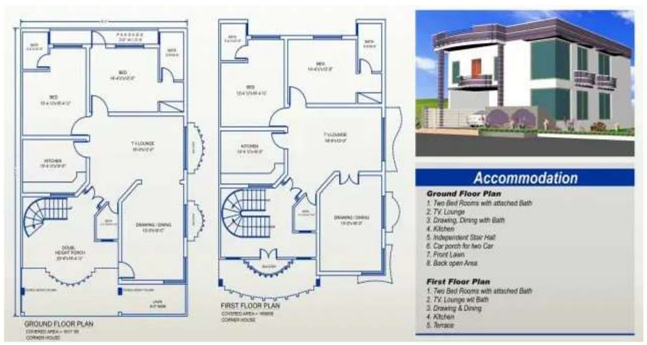
- Software Process: a procedure followed by the development team to produce an application.

1. Requirements Analysis (answers WHAT?)
   - Specifying what the application must do
2. **Design (answers HOW?)**
   - Specifying what the parts will be, and how they will fit together
3. Implementation (A.K.A. CODING)
   - Writing the code
4. Testing (type of VERIFICATION)
   - Executing the application with test data for input
5. Maintenance (REPAIR or ENHANCEMENT)
   - Repairing defects and adding capability

DILEMMA!



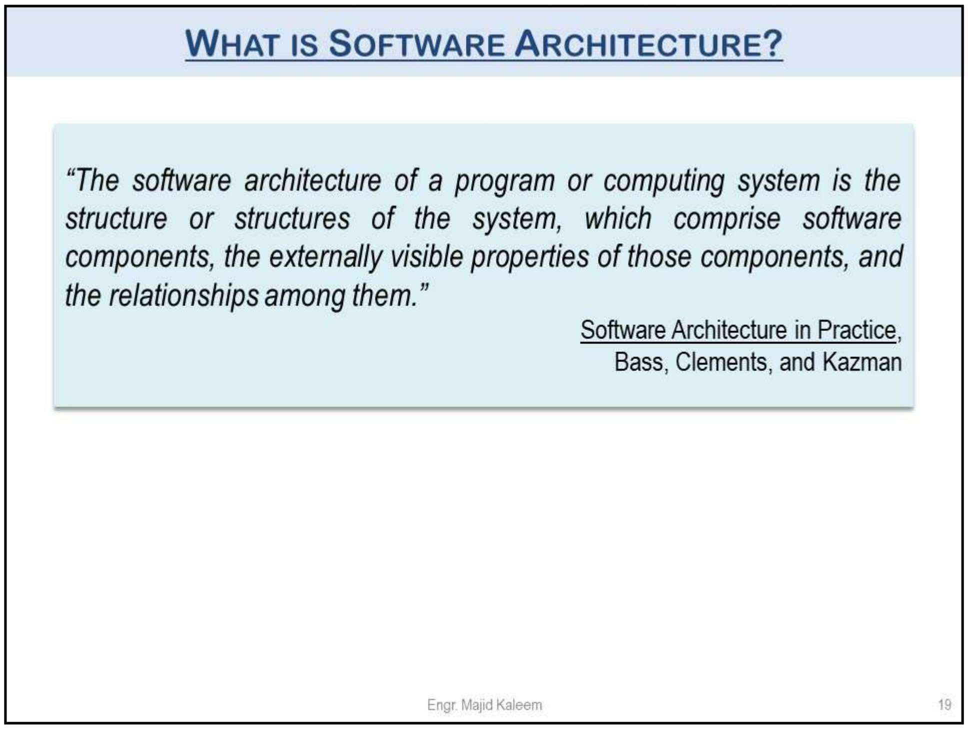
DILEMMA!



Accommodation

Ground Floor Plan
1. Two Bed Rooms with attached Bath
2. TV. Lounge
3. Drawing, Dining with Bath
4. Kitchen
5. Independent Stair Hall
6. Car porch for two Car
7. Front Lawn
8. Back open Area

First Floor Plan
1. Two Bed Rooms with attached Bath
2. TV. Lounge wit Bath
3. Drawing & Dining
4. Kitchen
5. Terrace

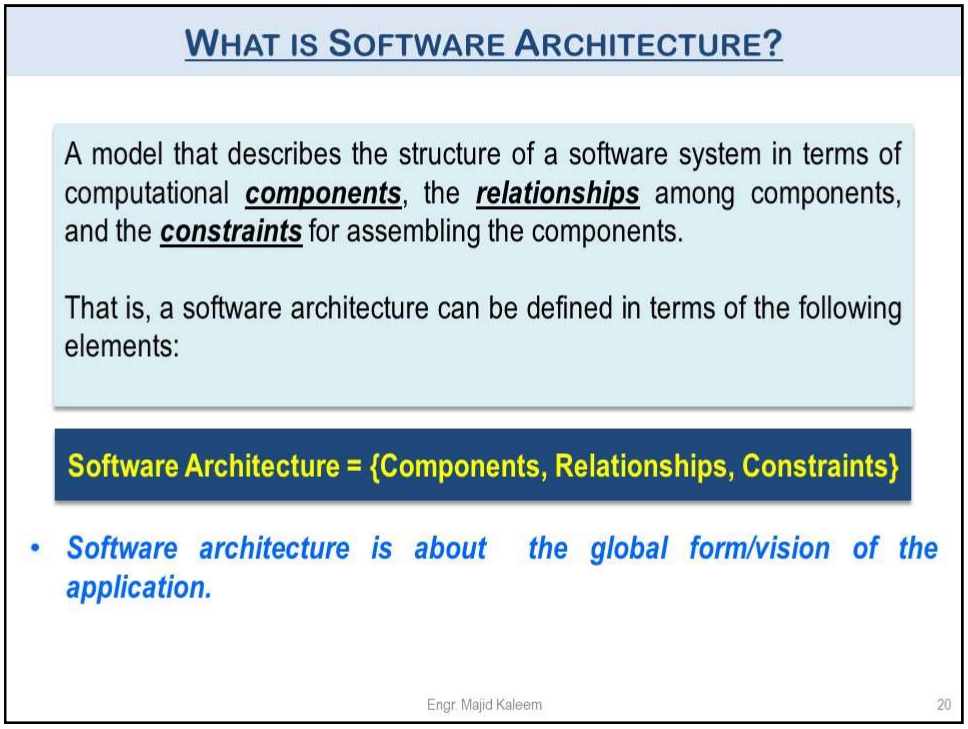


What is Software Architecture?

"The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them."

Software Architecture in Practice,
Bass, Clements, and Kazman

Engr. Majid Kaleem 19



What is Software Architecture?

A model that describes the structure of a software system in terms of computational ***components***, the ***relationships*** among components, and the ***constraints*** for assembling the components.

That is, a software architecture can be defined in terms of the following elements:

Software Architecture = {Components, Relationships, Constraints}

- ***Software architecture is about the global form/vision of the application.***

Engr. Majid Kaleem 20

What is Software Architecture?

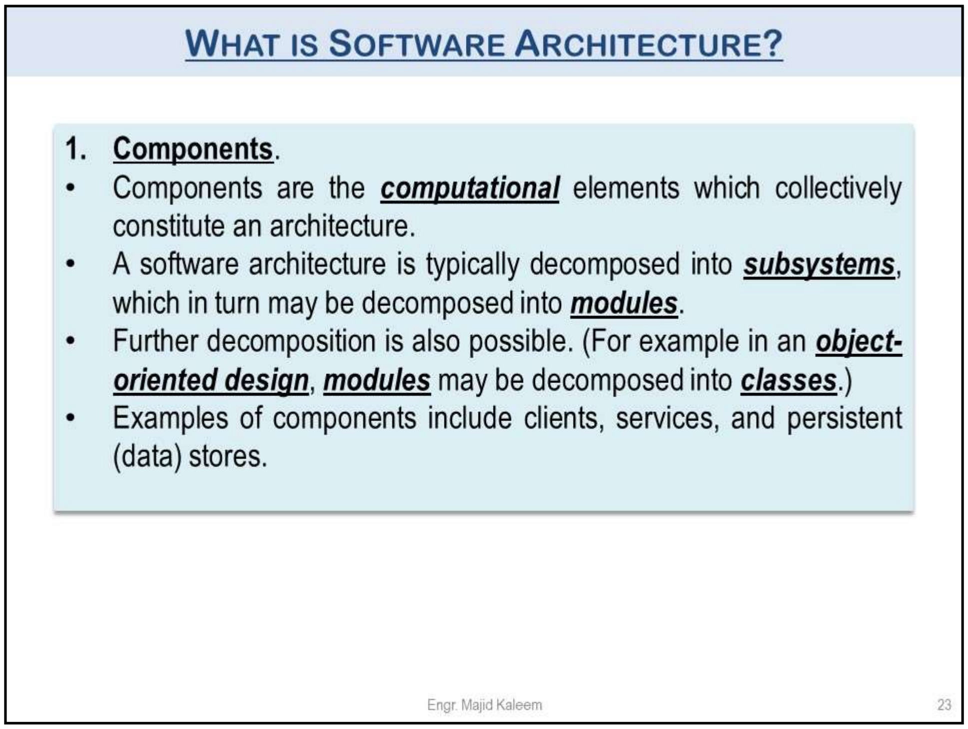
Perry and Wolf define software architecture using the following formula:

Software Architecture = {Elements, Patterns, Motivations}

- Within the context of object-oriented approach:
  - *The elements are the objects and classes,*
  - *Patterns are grouping of objects and classes,*
  - *Motivations explain why a particular grouping is better adapted than another in a given context.*

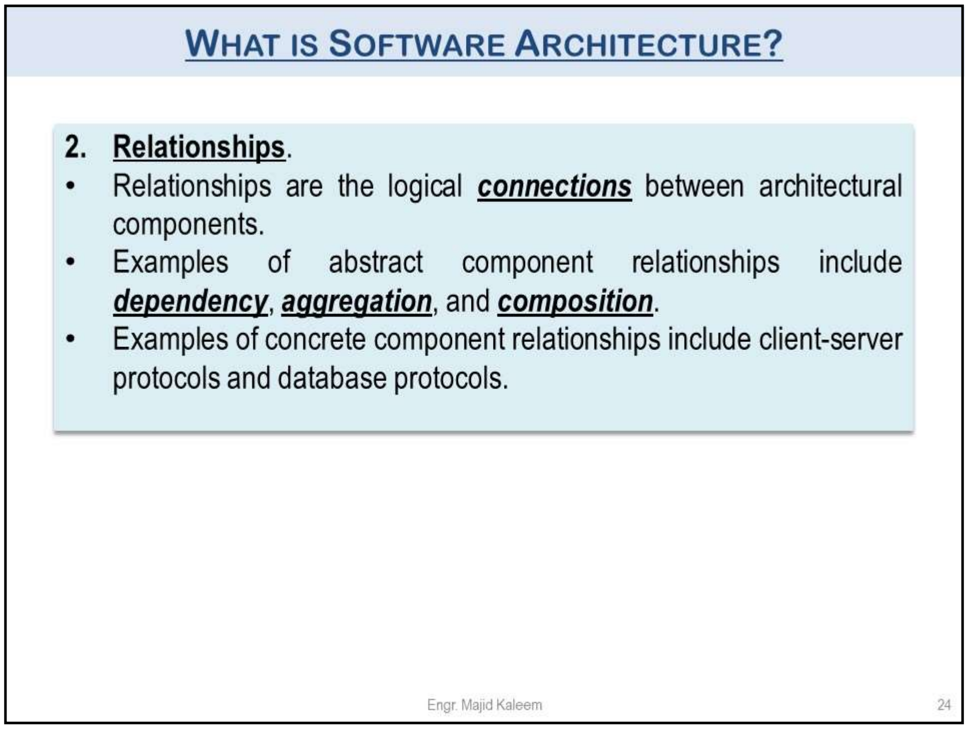
What is Software Architecture?

Software Architecture = {components, relationships, constraints}



WHAT IS SOFTWARE ARCHITECTURE?

1. **Components**.
- Components are the ***computational*** elements which collectively constitute an architecture.
- A software architecture is typically decomposed into ***subsystems***, which in turn may be decomposed into ***modules***.
- Further decomposition is also possible. (For example in an ***object-oriented design***, ***modules*** may be decomposed into ***classes***.)
- Examples of components include clients, services, and persistent (data) stores.



WHAT IS SOFTWARE ARCHITECTURE?

2. **Relationships**.
- Relationships are the logical ***connections*** between architectural components.
- Examples of abstract component relationships include ***dependency***, ***aggregation***, and ***composition***.
- Examples of concrete component relationships include client-server protocols and database protocols.

WHAT IS SOFTWARE ARCHITECTURE?

3. **Constraints**.
- Constraints provide ***conditions*** and ***restrictions*** for ***component relationships***.
- They connect the architecture to system requirements.
- Examples of constraints include restrictions on parameters types for communication protocols and high availability requirements for fault tolerance.

- https://www.ibm.com/developerworks/rational/library/feb06/eeles/index.html
- http://www.iso-architecture.org/ieee-1471/defining-architecture.html

WHAT IS SOFTWARE ARCHITECTURE?

- A software architecture is typically a set of design decisions to address various ***non-functional*** requirements and attributes of a software system/application.

- It primarily focuses on aspects such as performance, reliability, scalability, testability, maintainability and various other attributes, which can be key both structurally and behaviorally of a software system.

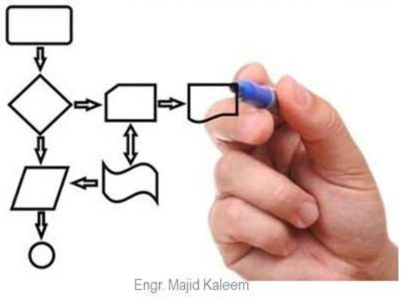
- The architecture of a software system defines the system in terms of ***computational components*** and ***interactions*** among those components." [Shaw and Garlan]

- Software architecture is a description of the subsystems and components of a software system and the relationships between them.

What is Software Design?

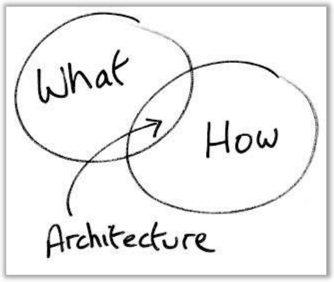
- The IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990) defines software design as ***"the process of defining the architecture, components, interfaces, and other characteristics of a system or component"*** and ***"the result of [that] process"***.

- Software design is the process of defining software ***methods***, ***functions***, ***objects***, and the overall ***structure*** and ***interaction*** of your ***code*** so that the resulting ***functionality*** will satisfy your users requirements.



Software Architecture vs. Software Design

- ***Architecture***: is more about the design of the ***entire*** system.
- ***Design***: emphasizes on ***module/component/class*** level aspects.
- ***Architecture***: focuses on "***what***" are we building.
- ***Design***: describes "***how***" we are building.



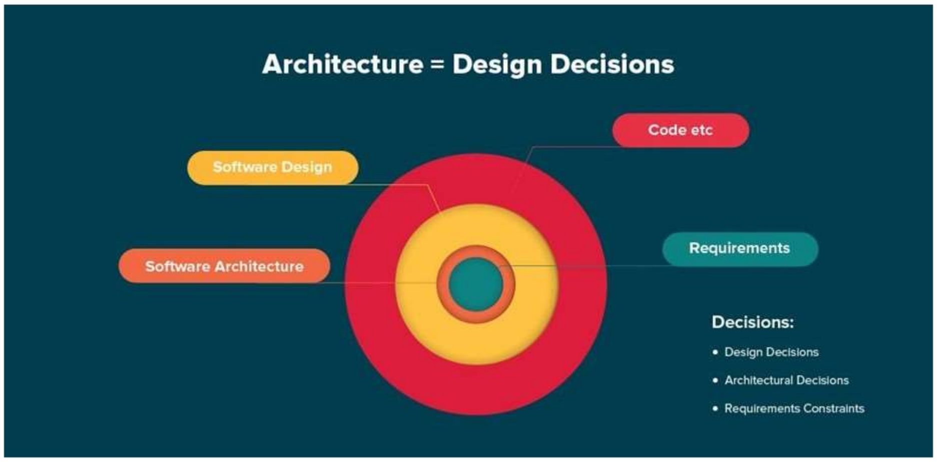
Architecture is mainly a design, while not all designs are architecture.

Functional vs. Non-functional Requirements

- Functional requirements describe ***what the system should do*** and non-functional requirements place ***constraints*** on how these functional requirements are implemented.
- Functional requirements describe ***what*** behaviors it does and non-functional ***how*** it does them.
- ***Example:***
  - *A functional requirements might state that a system must provide some facility for authenticating the identity of a system user; a non-functional requirement might state that the authentication process should be completed in four seconds or less.*

Engr. Majid Kaleem 29

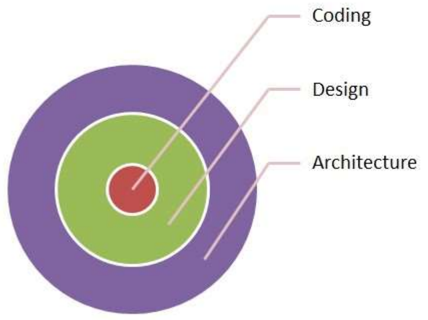
Software Architecture vs. Software Design

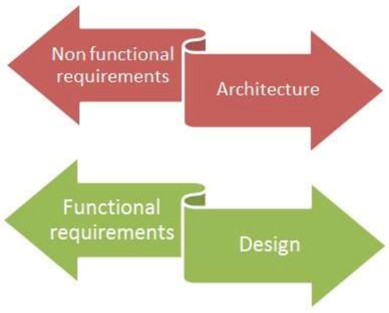


Engr. Majid Kaleem 30

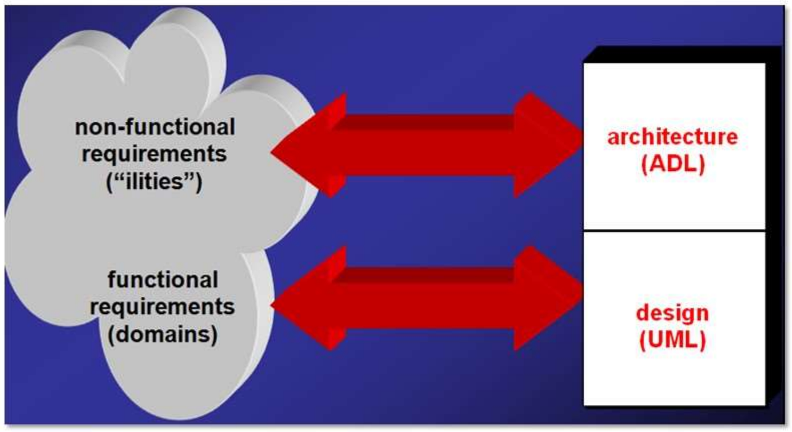
SOFTWARE ARCHITECTURE VS. SOFTWARE DESIGN

- ***Architecture:*** where non-functional decisions are cast, and functional requirements are partitioned
- ***Design:*** where functional requirements are accomplished.





SOFTWARE ARCHITECTURE VS. SOFTWARE DESIGN


non-functional requirements ("ilities")
functional requirements (domains)
architecture (ADL)
design (UML)

Software Architecture vs. Software Design

- Architecture and design is quite similar to the federal and provincial government.

Software Architecture vs. Software Design

- ***Federal Government (Software Architecture)***
- *"Federal government legislate matters common to more than one province"*

Federal government	Architecture
Inter-Provincial highways	Inter-Module dependencies
Postal service	Interfaces
Military	Behavior
...	…

- Architecture documents matters common to more than one module/component

Legislating is fancy word for documenting

Software Architecture vs. Software Design

- ***Provincial Government (Software Design)***
- *"Provinces may legislate on matters of a merely local or private nature"*

Provincial government	Software design
Education	Classes and objects
Provincial officers	Software design patterns
Municipal government	Dependencies
...	...

- Software design may document on matters of a merely local or private nature

"may" because you don't need to document everything

Engr. Majid Kaleem 35

Software Architecture vs. Software Design

EXAMPLE

Engr. Majid Kaleem 36

Software Architecture vs. Software Design

- The architecture of a system is its '***skeleton***'. It's the highest level of abstraction of a system.
- What kind of data storage is present, how do modules interact with each other, what recovery systems are in place.
- Just like design patterns, there are architectural patterns: ***MVC***, ***3-tier layered*** design, etc.
- Software design is about designing the ***individual modules/ components***.
- What are the responsibilities, functions, of ***module X***? of ***class Y***? What can it do, and what not? What design patterns can be used?

So in short, Software architecture is more about the design of the entire system, while software design emphasizes on module / component / class level.

```
If(anyQuestions)
{
 askNow();
}
else
{
 thankYou();
 submitAttendance();
 endClass();
}
```