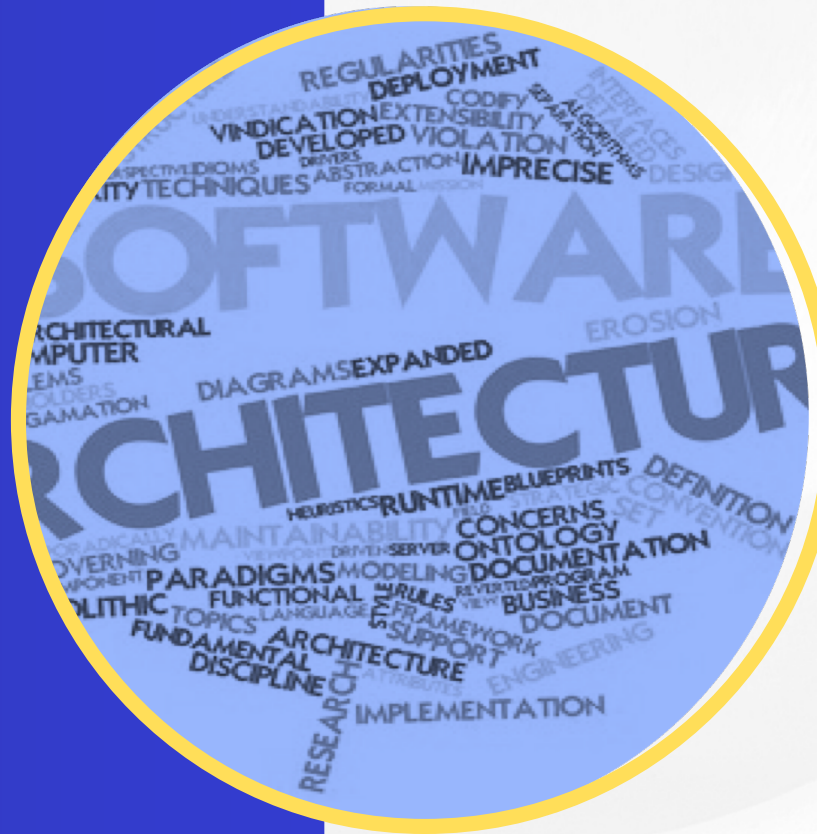
University of Tripoli – Faculty of Information Technology

Software Engineering Department

Software Design & Architecture

ITSE411



By Marwa Solla

Software architecture and design

for modern large-scale systems

Lecture 1 : Introduction

What We Learn In This Lecture

- Introduction of Software design
- Software Architecture Motivation
- Software Architecture Definition
- Software development Cycle

Introduction of Software Design

Objectives

- To understand the importance of design and architecture in the industrial process.
- To explain the mission of architecture and design activities among other SW process activities.
- To understand the difference between design and architecture.
- To understand the main roles of Architects.
- To understand the main roles of Designers.

Main Activities of many Industries

European Union (EU) Parliament defines industry as:

- *The automated production of material goods*.
- In order to understand if software (SW) development is an **industry, or not**, and how does it work, we need to understand the six main activities of any given industry. The main question is that is software Development is a real industry?

Main Activities of many Industries

Each, and every industry worldwide has **six main activities** that should be accomplished.

1. Planning.
2. Requirements Analysis.
3. Architecture and Design.
4. Build.
5. Test.
6. Deliver/Deploy.

Main Activities of many Industries

Example 1: Let us take the **planes production industry** as an example.

- If Boeing will produce a plane for Libyan Airlines, first of all, they have to understand how Libyan Airlines thinks in the potential plane. This needs to reach answers for lots of questions including:
  - What is the range of that plane?
  - How many seats should be in that plane?
  - Which engine Libyan needs the plane to come with?
  - What is the dedicated budget in Libyan Air for that plane?

Main Activities of many Industries

- The previous questions helps in **defining the scope** of that project. Answers helps in having a clear consensus about if this production project could be conducted or not.
- If there some technical, legal, expertise– related, political, or other could arise, and couldn't be mitigated, then the production project is not feasible.
- If the dedicated budget was not reasonable at all then, Boeing will notify Libyan Airline that this plane couldn't be produced and delivered to Libya.
- In contrary, if the project shows feasibility then, the rest of information will help in defining the scope of the project.

Main Activities of many Industries

- Scope sets the boundaries of the project from every prospective that shouldn't be exceeded, including but, not limited to; timing, budgetary, resources.
- The scope in our context could be the type of the plane (e,g 777, 737,747,..etc). If they have chosen a 777 plane. The subtype of the plane must to be defined as well (777–200, 777–300, 777–300ER, 777–400 or 777– 400ER)?
- Without such kind of information, the scope couldn't be defined clearly. Moreover, it may lead to complete failure of the project due to failure in achieving all what has been agreed on during scope definition.
- These tasks are being accomplished as part of what is called **Planning Activity**.

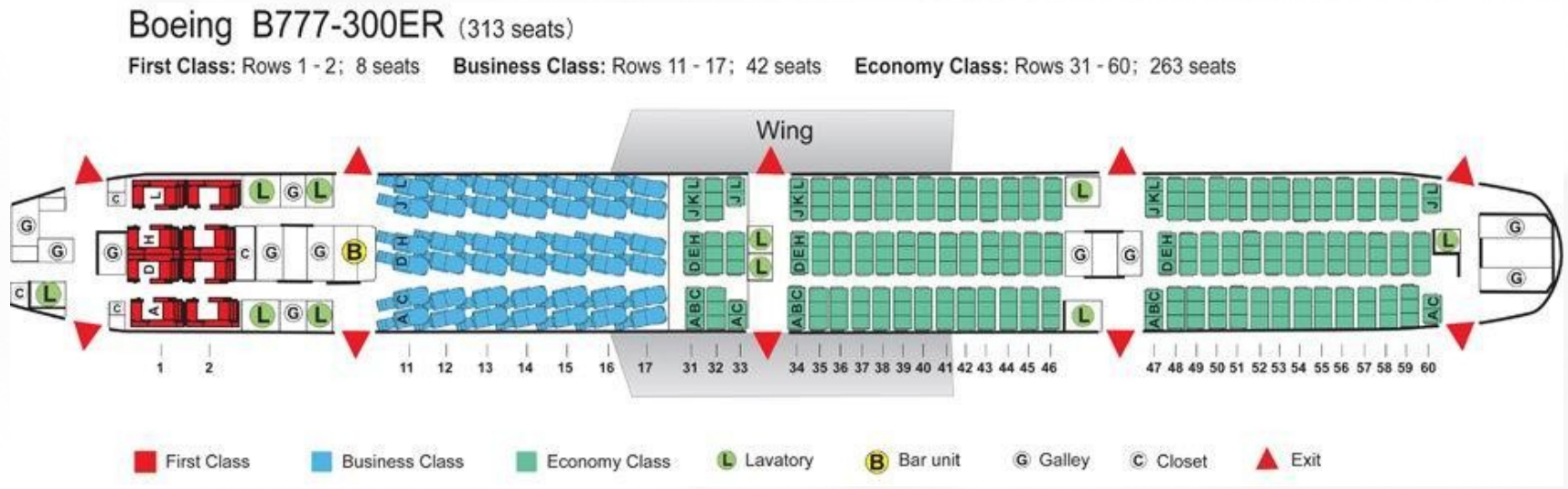
Main Activities of many Industries

- After contract agreement between Boeing and Libyan Airline, the second activity should begin. During this activity, Boeing should **gather the requirements, or wish list** of Libyan Airline in this plane.
- This wish list include answers for:
  - What will be the material of furnishing the seats (e.g. recycled leather, genuine leather, textiles, etc…)?
  - How many first–class, business–class, and economy class seats should be placed?
  - What will be spacing distance between each class's seats?
  - How will each seat be equipped (e.g. LCD screen, Sky Phone, First Class bed–like seat, etc…)?
- This activity is called **Requirements Analysis.**

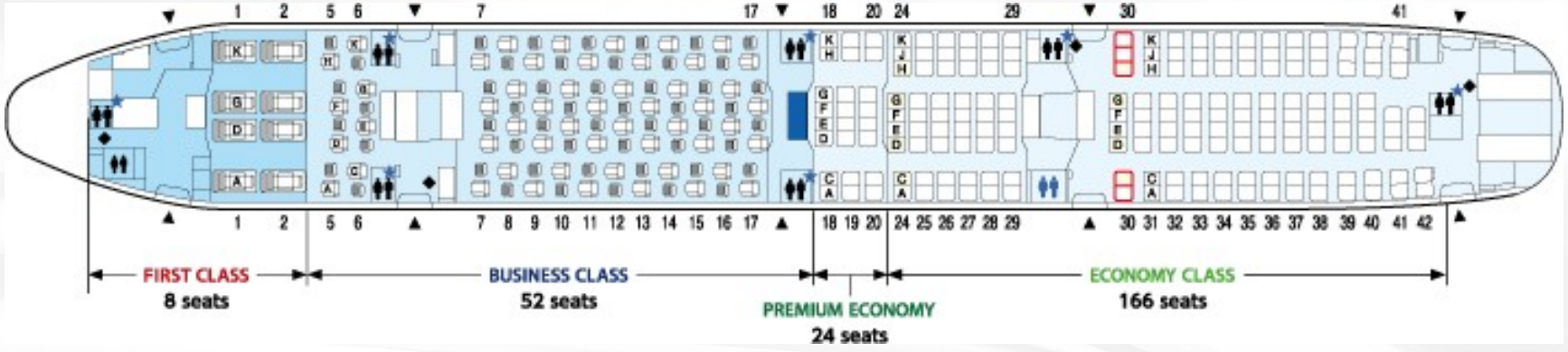
Main Activities of many Industries

- When both parties finally agree on all requirements, Boeing should pass the agreed specifications to the manufacturing team.
- This team is concerning with producing the product (the Plane in our example). In order to produce the plane, a blueprint should be there for the people who will build it.
  - The blueprint has lots of details that should be elaborated in order to let the producers able to build the plane.
- Usually, the blueprint includes two important parts; the product **Architecture & Design.**

Main Activities of many Industries



BOEING 777-300ER (Seats 250)



Main Activities of many Industries

- The same plane type that comes with same subtype can have multiple designs with similar predefined architecture.
- Previous Figures showed the architecture and internal design of the Boeing 777–300ER that are being built for the favor of three different Airline customers.
- The first design is for Air China, 313 seats, The second design is for ANA (a Japanese carrier), 250 seats.
- The blueprint includes most of the Architecture and Design specifications that are needed to enable plane developers to develop the plane.

Main Activities of many Industries

- The development, or build activity itself that ends with built plane is called **product Development** activity (the fourth main activity in the production process). This activity ends with producing the plane.

- This plane couldn't be delivered to the customer without being tested aggressively. Its engines should be tested under the expected load of range, and weight. This activity is called **Testing**.

Main Activities of many Industries

- At the end, the plane should be prepared to be deployed at the customer's predefined airport.
- To do this, the plane should be painted with the theme, and logo of the customer Airline.
- Finally, a crew flies by the plane to be deployed into the customer's airport. This activity is called **Product Deployment**.

Each SW development project includes those main six activities.

SW Development as an Industry

- According to International Organization for Standardization (**ISO**), software development is an industry.
- According to a study that has been conducted by Stanford University; SW is considered as an industry by US, and Japan.
- This mapping is important to grown the belief that architecture and design are core activities not in SW only, but also in any product development process.

Analogies from outside the software world

- Everything we build has a structure,
- The more we invest in building a product, the harder it becomes to change its structure.
- The thing is the structure of our system describes:
  - The intent of our product .
  - The qualities of the product.

Analogies from outside the software world

Examples

Application to Software

- different organizations will give us different properties.
- The software architecture impacts:
  - performance and scale of the product.
  - easy to add new features.
  - response to failures or security attacks.
- The cost of a redesign will be significant, both in terms of time and money.

Software Architecture & Design

- Software Architecture Definition
- Software design Definition.

Definition of software architecture

many ways define software architecture

The definition we're going to use is as follows:

> ***“the software architecture of a system is a high-level description of the system structure, it's different components, and how those components communicate with each other to fulfill the system's requirements and constraints.”***

Unpacking the Definition

"the software architecture of a system is a high-level description of the system structure, it's different components, and how those components communicate with each other to fulfill the system's requirements and constraints."

- An abstraction that shows us the important components while hiding the implementation details.

Unpacking the Definition

"the software architecture of a system is a high-level description of the system structure, it's different components, and how those components communicate with each other to fulfill the system's requirements and constraints."

- The components defined by their behavior and APIs.
- The components may themselves be complex systems with their own software architectural diagrams.

Unpacking the Definition

"the software architecture of a system is a high-level description of the system structure, it's different components, and how those components communicate with each other to fulfill the system's requirements and constraints."

- The components are coming together to do what the system must do, which is basically our requirements.
- the system does not do what it shouldn't do, which is described in the system constraints.

Levels of Abstraction

software architecture can have many different levels of abstraction:

- Classes,
- modules, packages, or libraries.
- Services (processes or groups of processes)

Advantages

This more distributed, multi–service approach allows us to architect systems that can:

- handle large amounts of requests.
- process, and store very large amounts of data.
- serve many users every day.

Large Scale Systems –Examples

Examples of such systems include:

- Ride–sharing.
- video–on–demand.
- Social media.
- Online video games.
- Investing services
- Banks.

Design vs. Architecture

- Most of Technical people mix between architecture and design.

❑ Software Architecture

- Architecture **(IEEE definition):** is the fundamental organization of a software system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution.

- Software architecture: is the high–level structure or blueprint of a software system. It defines how different components of a system interact with each other, the overall organization of the system,

Design vs. Architecture

- **Software Design :**

- Software design is a lower–level activity compared to architecture, concerned with the detailed specifications of the components that make up a system. It focuses on how the system will be implemented

- software design is a process of building a program while satisfying a program's functional requirements and not violating it's non functional constraints.

- Detail design is the process of dealing with individual component. Particularly, with respect to their data structure and their algorithms.

Design vs. Architecture

- The design of any given SW encompasses two kinds of technical decisions; components **design decisions** and **architectural decisions**. Both of those decisions are key pillar in finishing the SW final blueprint/design.

Design vs. Architecture

- **Architecture decisions** involve choices that impact the entire system, such as the selection of architectural patterns, the distribution of responsibilities, and the identification of key architectural styles.
- Architecture Decisions are mostly related to fulfilling Non–Functional Requirement. This includes but not limited to; *scalability*, *reliability*, *availability*, *maintainability*.
- **Design decisions:** are more granular and focus on the internal structure of components, algorithms, data structures, and interactions between classes or modules.
- Design Decisions are mostly concerned with fulfilling the Functional requirements. This includes but not limited to; identifying the applications domain object model, business objects, and data entities that will be physically implemented through code writing.

Design vs. Architecture

Table 1.1

Architecture	Design
Performed by architect.	Performed by designer.
high–level structures and the overall organization of the system. It focuses on major components or modules, their relationships, and how they interact at a high level.	delves into the details of individual components or modules. . making specific decisions about the internal structure of each component.
Architecture provides a blueprint for the entire system. It is more about the "big picture" and abstract concepts.	This deals with fine–grained details, addressing specific algorithms, data structures, and implementation details
Concerned mainly with application quality attributes, and nonfunctional requirements.	Concerned mainly with fulfilling customer's functional requirements.

NOTE: The architect unusually has more experience than designer.

Design vs. Architecture

Architecture	Design
Some example decisions: –Platform, operating system to be used. –Programming language to be used. –User experience technology (e.g. HTML5, CCS3, Silverlight, AJAX, etc…) –Component messaging (e.g. SOAP).	Some example decisions: –Entity Relationship Diagram design. –Class diagram design (in case of object Oriented programming) –Classes data and behavior.
Two main types of architect roles - Application architect. - Infrastructure architect.	Three main types of designer roles –User Interface (UI) designer. –Application component designer. –Data designer.

Why is architecture important?

Architecture is important because the architecture of a system has a fundamental influence on these non–functional properties.

A centralized security architecture

In the *Star Wars* prequel *Rogue One* (https://en.wikipedia.org/wiki/Rogue_One), the evil Empire has stored the plans for all of their equipment in a single, highly secure, well-guarded, remote location. This is called a centralized security architecture. It is based on the principle that if you maintain all of your information in one place, then you can apply lots of resources to protect that information and ensure that intruders can't get it.

Unfortunately (for the Empire), the rebels managed to breach their security. They stole the plans for the Death Star, an event that underpins the whole *Star Wars* saga. In trying to stop them, the Empire destroyed their entire archive of system documentation with who knows what resultant costs. Had the Empire chosen a distributed security architecture, with different parts of the Death Star plans stored in different locations, then stealing the plans would have been more difficult. The rebels would have had to breach security in all locations to steal the complete Death Star blueprints.

Importance of design & architecture

Benefits :

If we get the architecture right we can:

- Go from a small startup to a multi–billion dollar company.
- Making a positive impact on millions of people.

Risks :

- Not doing a good job at the design phase can:
  - Waste months of engineering time.
  - Building a system that doesn't meet our requirements.
- Restructuring a system that was not architected correctly is very hard and expensive.
- the stakes here are high.

Architectural design decisions

- Is there a generic application architecture that can act as a template for the system that is being designed?
- How will the system be distributed?
- What architectural patterns or styles might be used?
- What control strategy will be used to the operation of the components in the system?
- What architectural organization is best for delivering the non–functional requirements of the system?
- How will the architectural design be evaluated?
- How should the architecture of the system be documented?

Design vs. Architecture

Types of Architect Roles:

- **Application architect.**
- **Infrastructure/technical architect.**

Design vs. Architecture

- **Application architect.**
  - he is the one who is concerned with defining the way that will be used for coupling, and cohering the application's components, and layers.
  - Application architect has to take major decisions that affect the rest of the development team, and the SW production cycles.

Design vs. Architecture

- **Application architect.**

  Some decisions that are taken by AA, aligned with the gathered requirements and agreed scope and affect the development team:

  - OS, DBMS, UI types, Component Messaging,…
  - Platform to be utilized (on premise, cloud, hybrid),
  - The type of client devices (desktop PCs, tablets, mobile, etc…)

Design vs. Architecture

- **Application architect.**

  One of the most important decisions is the Application Architectural Style or Architectural Pattern.

Application architect styles

Architecture Style	Description
Client/ Server	Divide the application into two parts, where the client sends requests to the server. Usually, the server is a database with application logic represented as stored procedures.
Component–Based Architecture	Decomposes application design into reusable functional or logical components that expose well–defined communication interfaces.
Layered Architecture	Segregates the concerns of the application into stacked sets (layers).

Application architect styles

Architecture Style	Description
Message Bus	In this style, the application can receive and send messages using one or more communication channels autonomously, so that applications can interact without the need to know specific details about each other.
N-Tiers/3Tiers	Divide functionality into loosely couple segments in a way that is close to the layered style, however in this style, each segment is being a tier that is usually deployed on a physically separate machine.
Model View Controller (MVC)	MVC separates an application into three interconnected components, each with its own distinct role and responsibility

Design vs. Architecture

- **Infrastructure/ technical architect.**
  - Infrastructure architect is the one who defines how these components and layers should be physically deployed on the hardware, and network infrastructure.
  - Infrastructure architect decisions may include; defining the security zones, defining the kinds of server clusters that will be used.
  - Some Quality Attributes are used.

The End