

Object Oriented Software Engineering

Dagmawi Lemma (PhD)

Assistant Professor, AAU

courses.dag@gmail.com

<https://sites.google.com/site/dagmawi2012/>

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

1

Software Architecture

Object Oriented Software
Engineering

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

150

Software Architecture

- What is architecture?
- Architectural Styles
 - Tiered vs layered architecture
 - Open vs Closed architecture
- Selecting the right architecture
- Further readings
 - Enterprise Architecture
 - SOA

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

151

What is Architecture?

- High level description of the software with respect showing
 - How the low-level design elements (modules/subsystems) are arranged
 - How the structured solution meets all of the technical and operational requirements, while optimizing common quality attributes such as performance, security, and manageability

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

152

What is Architecture?

- K. Philippe et. al, derived and refined a definition of architecture based on work in (Shaw and Garlan 1996). According to their definition is:
 - Software architecture is about describing how the software is organized
 - The description may include
 - the structural elements and their interfaces by which the system is composed;
 - behavior as specified in collaboration among those elements;
 - composition of these structural and behavioral elements into larger subsystems; and
 - an **architectural style** that guides this organization

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

153.

What is Architecture?

- Architecture focuses on how the major elements and components within an application are used by, or interact with, other major elements and components within the application
 - At low-level design, selection of data structures and algorithms or the implementation details of individual components are the concerns
 - However, some of the low-level design concern could be dictated by the high-level decision – the architecture
- Architecture and Design concerns very often overlap
 - In some cases, decisions are clearly more architectural in nature
 - In other cases, the decisions are more about design, and how they help you to realize that architecture

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

154.

Why Architecture?

- Architecture helps to provide solid foundation for the software
 - failing to consider key scenarios - *may affects funtionality,*
 - failing to design for common problems - *may affects end-user criteria,*
 - failing to appreciate the long term consequences of key decisions – *may affect dependability and maintainability*
- Such failures put your application at risk

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

155

Why Architecture?

- Software should be designed with consideration for the
 - user,
 - system (the IT infrastructure), and
 - business goals
- For each of these areas, one should outline key scenarios and
 - identify important quality attributes (e.g., reliability or scalability) and
 - key areas of satisfaction and dissatisfaction (e.g., usability)

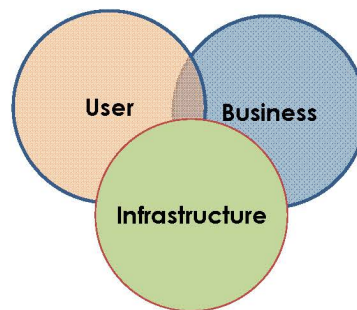
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

156

Why Architecture?

- It is needed to balance tradeoffs



Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

157

Why Architecture?

- following high level concerns are suggested (in MSDN) to be considered when thinking about software architecture:
 - How will the users be using the application?
 - How will the application be deployed into production and managed?
 - What are the quality attribute requirements for the application, such as security, performance, concurrency, internationalization, and configuration?
 - How can the application be designed to be flexible and maintainable over time?
 - What are the architectural trends that might impact your application now or after it has been deployed?

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

158

Architectural Styles

- An architectural style, also known as architectural pattern, is a set of principles – a coarse grained pattern that provides an abstract framework for a family of systems
- Architectural styles describe
 - deployment patterns,
 - some describe structure and design issues, and
 - others describe communication factors
- Include patterns such as
 - client/server,
 - layered architecture,
 - component-based architecture,
 - message bus architecture, and
 - service-oriented architecture (SOA)

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

159

Architectural Styles

- More specifically, an architectural style determines the vocabulary of components and connectors that can be used in instances of that style, together with a set of constraints on how they can be combined

David Garlan and Mary Shaw, January 1994, CMU-CS-94-166,
"An Introduction to Software Architecture"

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

160

Architectural Styles

- Architectural styles may vary by their key focus area
- Some of the major areas of focus and the respective architectural styles includes

Focus area	Architecture styles
Communication	Service-Oriented Architecture (SOA), Message Bus
Deployment	Client/Server, N-Tier, 3-Tier
Structure	Component-Based, Object-Oriented, Layered Architecture
Domain	Domain Driven Design

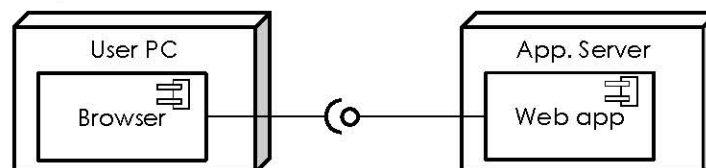
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

161

Tiered vs Layered

- Tier focus on the definition of the deployment style while layer refers to the structure of the software with respects to the function/service to be provided
- For example, in a web application, the web browser and the application server are deployed at different location – tiers; hence we have two tiers
 - In this example since we have a browser that request a service from the application server it can be referred as a client/server architecture – hence 2-tiered



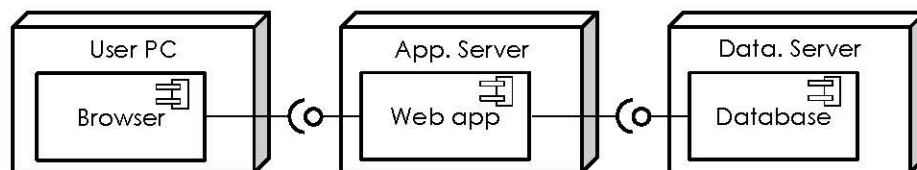
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

162

Tiered vs Layered

- It is also common the split servers based on their service (e.g. database, security, etc.); this could lead to another level of tier so we can have the third tier in such arrangement



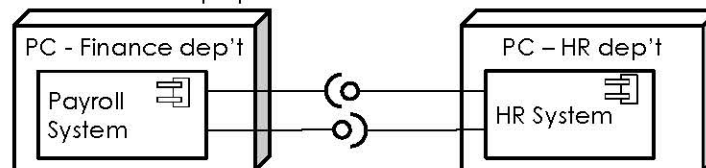
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

163.

Tiered vs Layered

- Also, if we have applications deployed at different peers where each has their complete implementation but might support one another we can have peer-to-peer (p2p) architecture, so for n peers existing in the system we can have the p2p can form n-tier architecture



NB: being p2p does not justify n-tier, even in client/server configuration we can have n-tier as long as we have more than one server (deployment location)

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

164.

Tiered vs Layered

- Layer on the other hand focus on the structural configuration of the software, such as functionalities, regardless of the deployment location
 - Functionality within each layer is related by a common role or responsibility
- For example, continuing the web application example, perhaps the designer can decide every data request should be validated before the access to the data is permitted

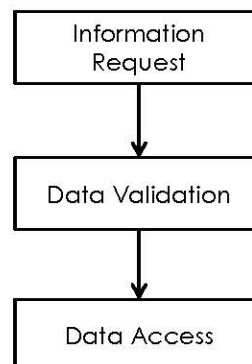
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

165

Tiered vs Layered

- In this arrangement, let us assume that after the data request is made the validation has to be done by the 2nd layer before it returns the requested data
- Where should it be validated would be the focus of tier (client – java script, server – server script (JSP, ASP)); on the other hand the functionality, when to validate and to whom to return the requested information is structural defined functionality, hence layer perspective



Dagmawi L.
(courses.dag@gmail.com)

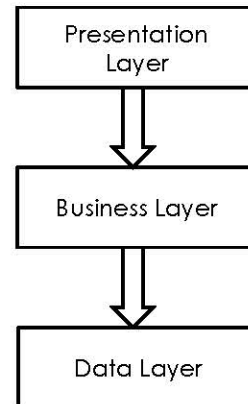
Object Oriented Software
Engineering

166

Tiered vs Layered

- The layers of an application may reside on the same physical computer (the same tier) or may be distributed over separate computers (*n*-tier)
- The components in each layer communicate with components in other layers through well-defined interfaces
- For example, a typical Web application design consists of
 - a presentation layer (functionality related to the UI),
 - a business layer (business rules processing),
 - and a data layer (functionality related to data access, often almost entirely implemented using high-level data access frameworks)

- Example

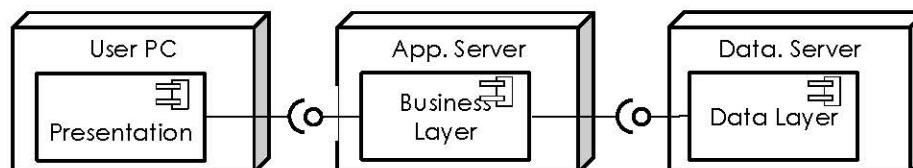
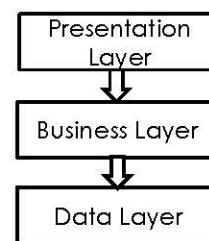


Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

167

Tiered vs Layered



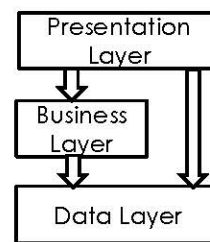
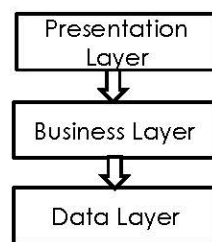
Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

168

Open vs Closed

- Closed Layered Architecture
 - components in one layer can interact only with components in the same layer or with components from the layer directly below it
- Open Layered Architecture
 - allows components in a layer to interact with components in the same layer or with components in any lower layer



Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

169

Selecting the Right Architecture

- As discussed in Chapter 1, software design requires technical skill; this includes knowledge on the existing types of architectural styles so that one can compare and contrast them prior to selection
- Selection of the architecture is also guided by the design goal
 - For example, the overall user experience of the solution is very often a function of the business and the IT infrastructure, and changes in one or the other can significantly affect the resulting user experience. Similarly, changes in the user experience requirements can have significant impact on the business and IT infrastructure requirements. Performance might be a major user and business goal, but the system administrator may not be able to invest in the hardware required to meet that goal 100 percent of the time. A balance point might be to meet the goal only 80 percent of the time

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

170

Client/Server

- A 2-tier architecture where one tier (client) request service and the other tier (server) provides the requested service
- Other variation of client/server may include
 - P2P architecture
 - Repository architecture
 - Application server
- The main benefits of the client/server architectural style are:
 - Higher security
 - Centralized data access
 - Ease of maintenance

Dagmawi L.
(courses.dag@gmail.com)

Object Oriented Software
Engineering

171