Aufgabe 7

Nathan Ritter
5566519
Lorenzo Tecchia
5581906

2023.06.04

Contents

Chapter 1

Task 1

- Architectural levels provide an abstraction method for Software Architectures, they are organised in an hierarchical order. Starting from the fundamental building block of the architecture the level of abstraction increases. The distinction from one level to another allows software architects to address different problems at different levels, and to avoid mixing these different aspects. Creating this structure of building blocks with abstraction facilitate the creation of a more uniform, consistent and therefore reliable system.

- The "change of level" is used in the context in which the system building block decomposition it's not a building block but a sub-system instead. So changing level means changing from the building block level to the system level.

- Macro-architecture covers the spectrum of the "large scale" (high level of abstraction)architecture, whereas the micro-architecture covers the spectrum of the building blocks so a lower level of abstraction.

- The transition between micro and macro architecture can be seen when recursively decomposing the building block of the system we encounter non-fundamental system-building blocks with a low level of abstraction.

- Architectural views are a mean that enable to focus on a specific problem at an appropriate time or to separate different aspects of the architecture of a system from one another. Architectural views differ from one another in a way that each of them depict a specific abstraction of the architecture of a system. The purpose of distinguishing them is that stakeholders that are interested on a specific view, don't have the need to understand all the parts of the architecture but they can laser focus their attention on that particular aspect.

- An architectural styles are a pattern of a structural organisation of a family of systems. They primarily reflect the fundamental structure of a software system and its properties. Architectural styles differentiate themselves from architectural patterns for the fact that they have different forms of description. [3]

Chapter 2

Task 2

(a) The architectural styles are the following:

- Client/Server architecture: Spotify(Any streaming software), it can be seen in the way that there is one copy of a song/album on a server that can be streamed to different clients via the app.
- Multi-layer architecture: ChatGPT, we can observe the layer architecture by the fact that a GUI appears to the user and the text inputed into the GUI prompt a calculation done and stored by a server.
- Event-based systems: Strava (Any sport tracking system), the system has to output different statistics to the user in correspondence of the event that happen (user does a km of running).
- Data flow networks: The Tor browser, uses the network routing to rout through an onion like layer of connections, the flowing of this connection is solely based on the input of the previous one thus ignoring the other connection traversed.
- Web architecture: Microsoft Azure(any web based cloud computing server) is clear how the cloud and therefore the web provides services for for computing pourposes.

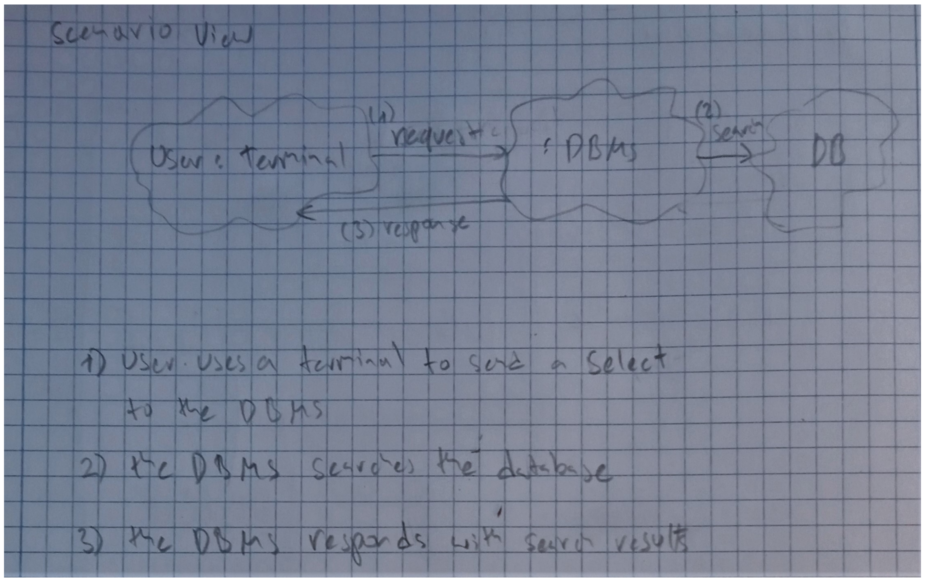
(b) 1. Real-time behaviour: Event-driven system style, or event-driven software architectures patterns, because they handle multiple events at ones, so users don't have to wait for one process to finish before starting the next. Thus the response time doesn't depend on whether a process has already been completed.[1]

2. High portability: Micro kernel styles, because in the style there is a main system that can run on the target operating system that contains everything the system needs to function and then it can run smaller plug in systems that are independent of each other. So once you have created one of these plugins you can use it directly on any system where the main system is already implemented.[2]

Chapter 3

Task 3

3.1 UseCase View



1) User uses a terminal to send a select to the DBMS

2) the DBMS searches the database

3) the DBMS responds with search results

Figure 3.1: UseCase View

3.2 Logical View

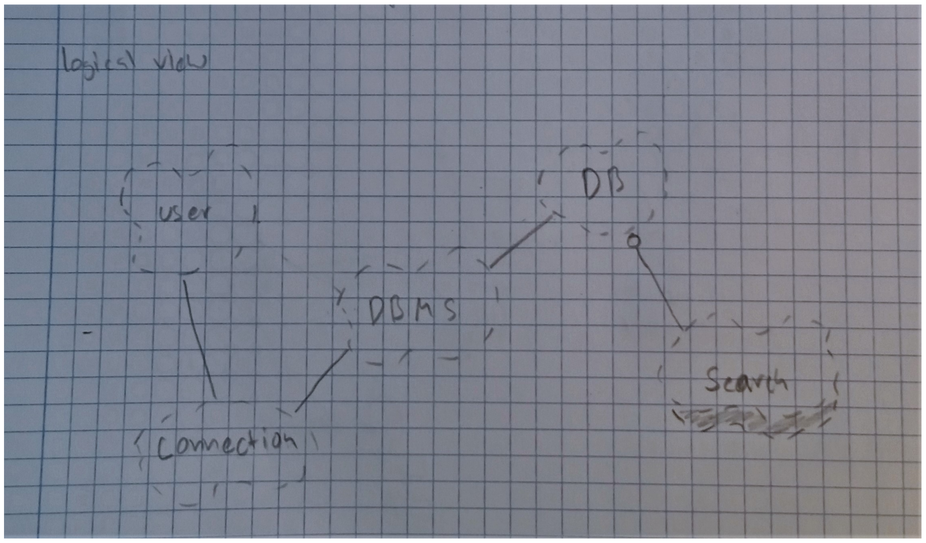


Figure 3.2: Logical View

3.3 Implementation View

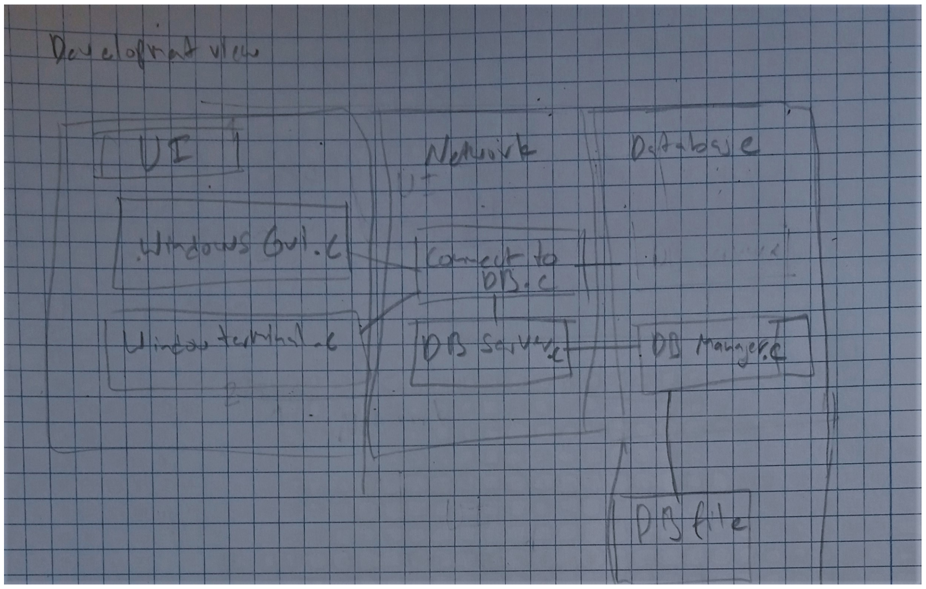


Figure 3.3: Implementation View

3.4 Process View

GUI event handler ensures that the user can press the buttons without having to wait for the connection to be completed. DB client establishes the connection, waits for the server. DB webserver connects and sends information to DB client and talks to B manager. DB manager processes data and searches data while server does other things.

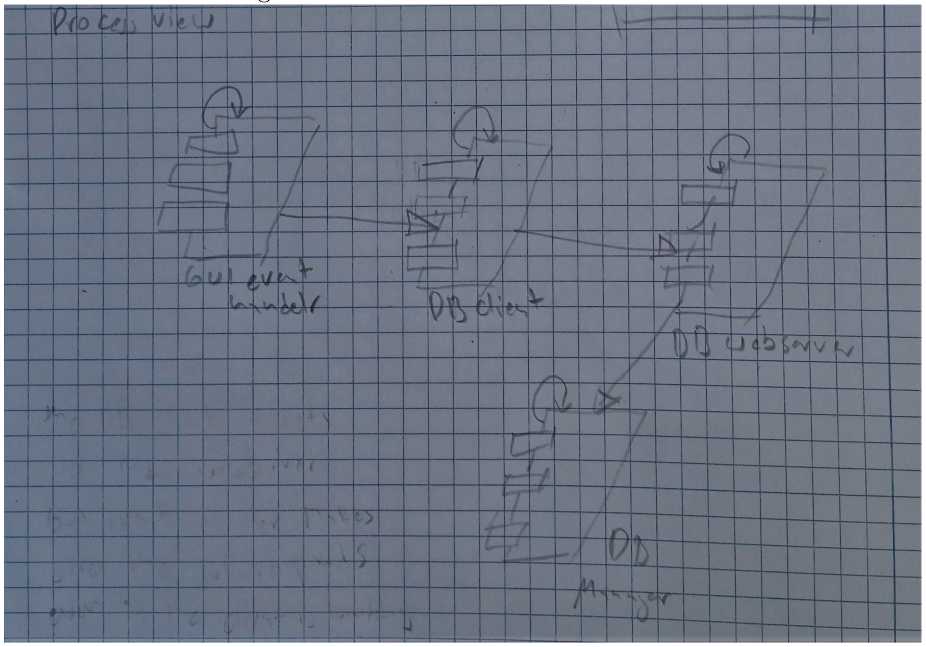


Figure 3.4: Process View

3.5 Distribution View

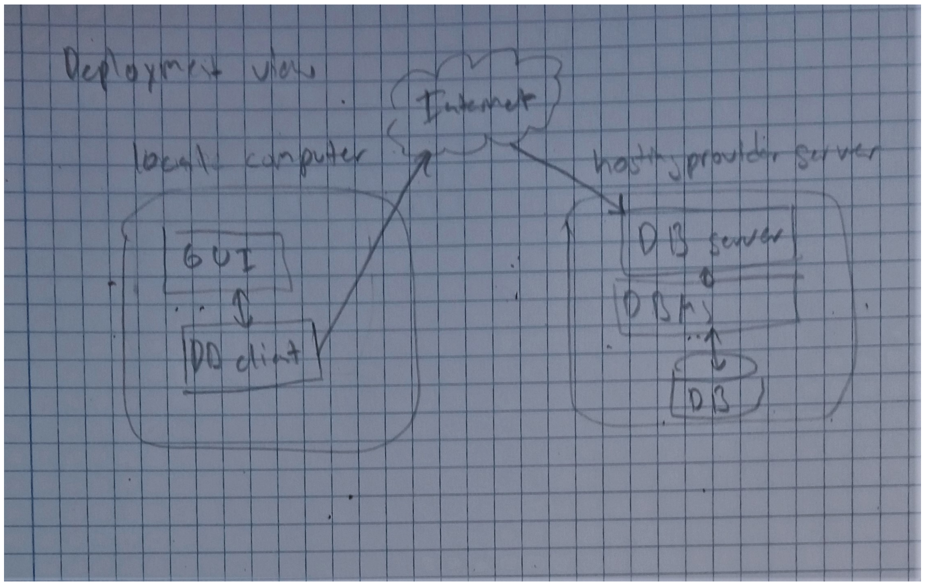


Figure 3.5: Distribution View

Bibliography

Aley, James. *The benefits and challenges of event-driven architecture — infoworld.com.* `https://www.infoworld.com/article/3669414/the-benefits-and-challenges-of-event-driven-architecture.html`. [Accessed 04-Jun-2023].

Software Architecture Patterns; Spirit Of Soft LLC — spiritofsoft.com. `https://www.spiritofsoft.com/software-architecture-patterns/`. [Accessed 04-Jun-2023].

Vogel, Oliver. *Software architecture: a comprehensive framework and guide for practitioners.* ISBN: 9783642197352.