Architectural Thinking for Intelligent Systems

Assignment for Lecture A10

Team Members:
1. Ankit Agrawal, 2581532
2. Anika Fuchs, 2580781

1. Apply an architectural style to your system

Option A: SOA - Microservices

The architectural style that fits best to our system is ‘Software oriented Architecture’. The different modules of our architecture can then be easily implemented independently. The very loose coupling of the components makes the system appropriate for Microservices that can evolve independently of each other and share data. Together with a well structured architectural design it will be feasible to bring the components together and test the overall system. The complexity of connections, for example between Payment and User Context, is manageable.

2. Relevant forces and constraints, how does it support system qualities from A6

Constraints and Forces:

- Scalability
  Our architecture must be able to scale up and down based on the number of clients that are currently in the store. The architecture must be stable in order to be used reliably. The possibility of autoscaling in a containerized environment makes it easy to adjust performance and availability of the system.
- Completeness
  Each and every context must be complete. There is no space for errors in the single components, they do not have many points of contact and they do not share functionalities, hence it is intuitive to develop these components individually and make them fully functional and complete services. With that approach it is possible to extend the services and introduce new versions of the system.
- Fault tolerance
  A microservice architecture consists of many instances of a service and hence is able to quickly recover from errors with minimal impact on other services of the system.

- Individual testing of components
  By the ability of testing functionalities independently it becomes easier to fulfill all the distinct requirements of different stakeholders. It makes the development faster because one can test the implementation gradually without waiting for other components to be finished.
- Clear distinction between functionalities
  A clear distinction between functionalities eases the communication between developer and architect and desires can be explained in more detail. The implementation focuses on one function and the borders are clearly defined.
- Ability to develop new functionalities
  The microservices architecture is best when we want to implement new functionalities in the future. The expectations of customers are constantly growing and technologies evolve over the years. By using SOA we can adapt to these changes and several parts of the system can evolve over time, still being independent of each other.

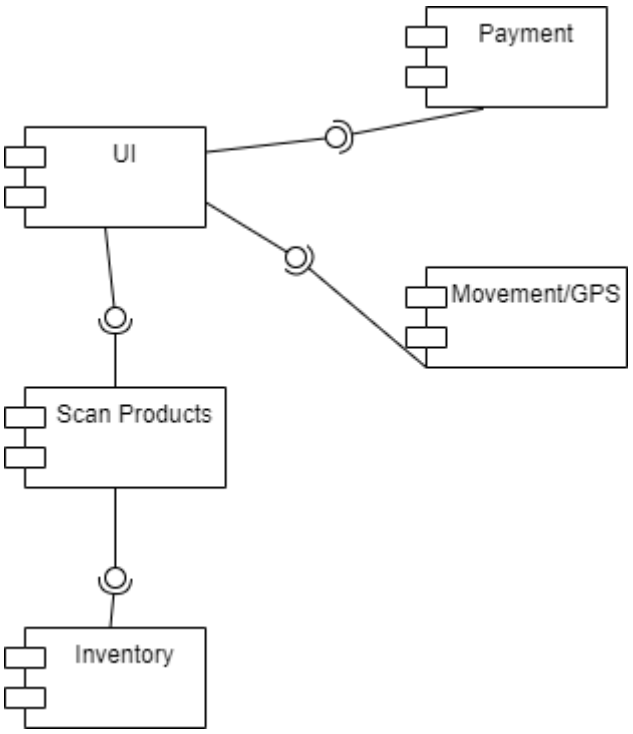
Selected system qualities from A6:

- Performance
- Reliability

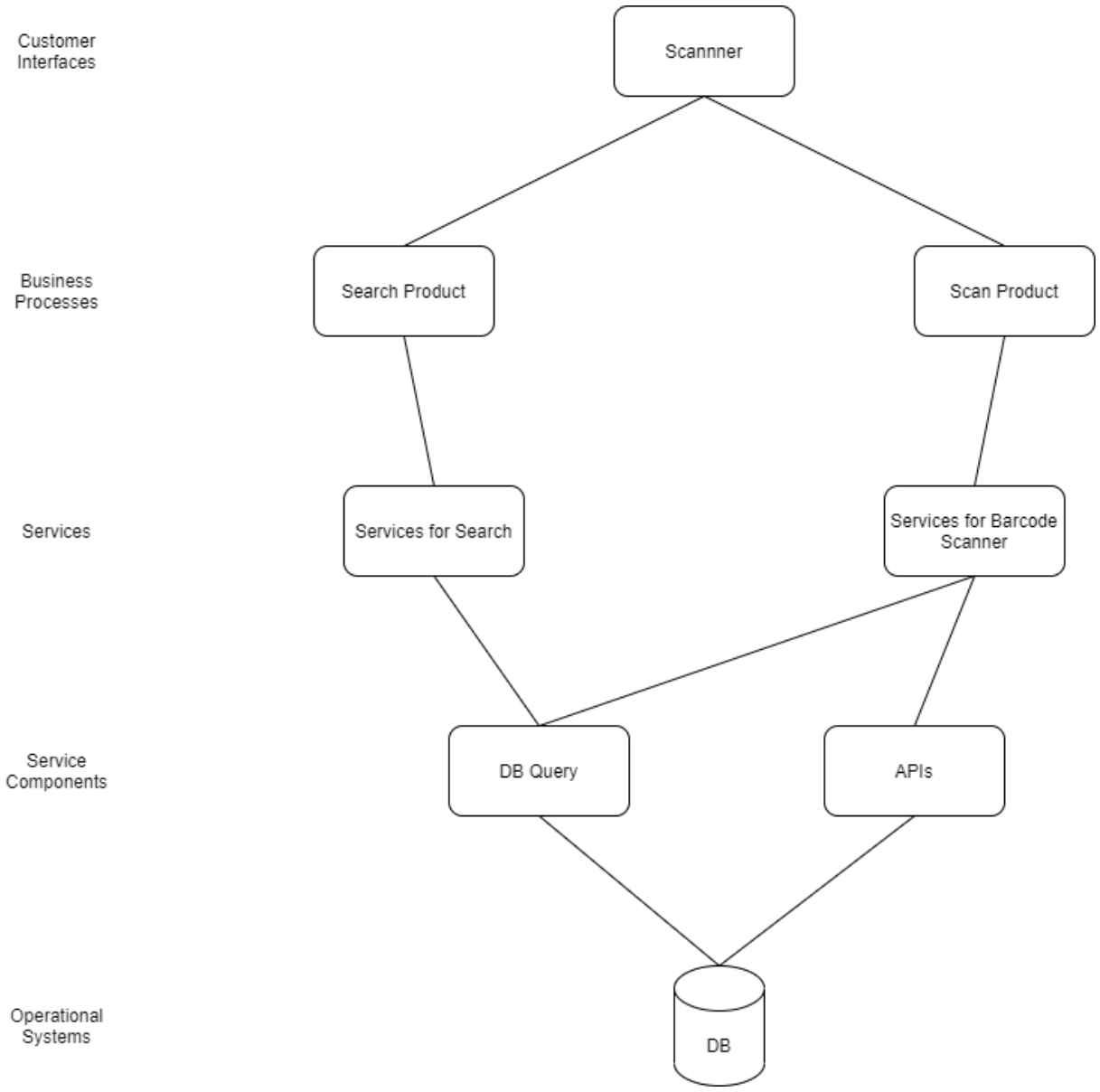
How does the selected architectural style support the system qualities:

By having a simple design and individually extendable components we can focus on efficient functionalities and test their performance directly to be able to improve it even further without waiting for other components to be implemented. This is especially important for the user side. Independently deployable microservices can easily recover from errors. By using a containerized technology services can run simultaneously and single containers can be restarted in case of errors. This makes the system reliable and prevents downtimes.

3. Package / component diagram for top-level architectural style



4. Refine one component: system structures and relationships (second level of detail) Which architectural style for this component? Same or different? Explain.



Component: Scan Products
Architecture Style: SOA Microservices

In the scan products component, there are 2 different components i.e. to scan the barcode and to search for the products and it connects with the business processes. The service components are further connected to Operational systems. Hence it is suitable for this component as we can connect all individuals' processes and components together. The very loose coupling of the components makes the system appropriate for Microservices that can evolve independently of each other and share data.

5. Viable alternative for architectural style

Option B: Client/Server; rich client

A viable alternative for software oriented architecture would be the ‘Client/Server Architecture’ combined with a rich client. The robot and its functionalities would be the client and it sends requests to the server, like location and product requests.

The C/S architecture makes it difficult to decide which functionalities should be implemented on the client side and which on the server side. The clear distinction between functionalities mostly important to the customer and functionalities mostly important to the store owner cannot be maintained as with Microservices.
Especially the centralization of product and inventory information on the server can be an advantage of C/S architecture. These informations constantly change/evolve and dependencies could be identified faster. Moreover the idea of robots as being the clients and a centralized server is quite intuitive to understand.