Reflection Report Group 12

Joachim R Baumann*, Phillip Nielsen*, Sebastian Revsbech Christensen*,
Simone Cosentino*, Oskar Præstholm*, Hans Askov*
University of Southern Denmark, SDU Software Engineering, Odense, Denmark
Email: {jobau19,phnie19,sechr17,sicos23,ospra20, haask19}
@student.sdu.dk

I. Contribution

Section	Main Contributor	Reviewed By
Abstract	Simone	Everyone
Introduction and Motivation	Sebastian, Phillip	Everyone
Problem, Research question, and approach	Oskar, Simone, Hans	Everyone
Literature Review	Sebastian, Joachim, Phillip	Everyone
Use case and QAs	Hans, Simone, Oskar	Everyone
Solution	Oskar, Simone, Hans, Sebastian, Joachim, Philip	Everyone
Empirical Evaluation	Joachim , Phillip	Everyone
Discussion/Future work	Joachim, Hans	Everyone
Conclusion	Simone	Everyone

TABLE I: Project contributions

In the stage for the use case and the design of the project, every decision was taken by the full group, with two meetings a week at the university. This means that the assumptions were created as a group, the use cases derived from these, the architecture of the systems and subsystems, and the QAs required for the system to solve the issues the ball pen production is facing were all done as a group.

Together we landed on the idea of creating an assembly line for pen production as it was a tangible and easy-to-understand process for us to visualize. So easy in fact, that we took apart a pen and labeled all of its parts, and made a small diagram of the process steps needed to put them together. This process included both sequential and parallel processes for assembling the pen. We used this concrete example which can be seen in figure 3 of the original report as the basis to build our architecture around which can be seen in the description of the use case and quality attributes. Also, potential challenges were identified like 24/7 availability and completely automated quality control. Based on this context and to align with the requirements we found, we decided that while building the architecture, we should have prioritized four main quality attributes: deployability, scalability, interoperability, and availability.

Trying to address the selected quality attributes, we tailored the general research questions our project was supposed to answer:

1) How can different architectures support the stated production system requirements?
2) Which architectural trade-offs must be taken due to the technology choices?

Transformed into our actual three research questions:

1) Which architectural patterns would best align with the system's quality attributes?
2) Which tech stack could contribute positively towards the required quality attributes?
3) Which framework should manage internal communication between subsystems?

We would meet after class together define the table of assumptions and reasoning behind these together. We worked on defining the assumptions together going back and forth between each other. When working together on the solution and our proposed and final architecture we landed on an Event-driven architecture that uses a message bus for communication between our microservices but also for all of our sensors to write to. We ended up choosing three systems: the scheduling, the log system, and the predictive maintenance system. These would all get data from the message bus and communicate it back to the other system allowing for many-to-many communications using one common interface. Sketches of both the production line and the system architecture were drawn to make sure that all the members of the group had the same idea of them.

II. Discussion

During our research project, our initial challenge came from our limited understanding of Industry 4.0 and the mechanics behind ball pen production. This prompted us to research both topics. However, obtaining information specifically about industrial ball pen production proved challenging. To overcome this challenge, we visualized the assumed machinery involved in ball pen manufacturing, defining the use of sensors, conveyor belts, and 'cells' responsible for assembling the pen components. Creating these visual representations helped in the team's understanding of the production process, giving a clearer understanding of the needs for the software architecture.

After our initial research, analysis, and visualizations of the production, the Quality attributes were defined as follows:

1) Interoperability
2) Availability
3) Deployability
4) Scalability

This led us to choose the Event-driven architecture, as it would fit well with the Quality Attributes based on [1]. However choosing the right message bus was not as easy, as there are many different message busses, which are good for different scenarios. After researching and discussing our findings. We ended up using two message busses, Kafka and MQTT due to having an event-driven architecture, the heart of the systems communication would go through a message bus, whereas Kafka is proficient in Enterprice systems but tends to use more resources. MQTT is proficient in smaller devices such as internet of things(IoT) and uses fewer resources to produce and subscribe to messages [2]. From here we started to design the test, we chose to test scalability in the form of the sensors, and the time for Kafka to obtain the messages. The flow of a message is from an MQTT publisher to the Kafka topic, which is then saved to Postgres. Implementing the test in code, proved to be a challenge, as we have not worked with MQTT before, and there was limited information about the setup of the MQTT-Kafka proxy. The problem was solved by looking at existing setups and how they were done we managed to get it to work.

For most of the project, up until the evaluation part, we felt that we were able to make informed decisions based on the literature presented in the lectures, which worked well as a guide for the different trade-offs we made during the initial creation of the systems and subsystems and later the design. The first wrong assumption about the system was having everything connect to the Kafka message bus. We went back and changed the design to having the PLCs connect to MQTT instead based on the new knowledge we gained each week. We felt that we were able to draw on some of the different experiences people had previously, such as a group member from the robot technology bachelor's degree who had some experience working with PLCs and production setups. The first major challenge we encountered was in the creation of the first QA test. The initial test was supposed to test the deployability of the system. As such we compared two resource orchestrators, namely Kubernetes and Docker-Compose. The test calculated the time it takes for the orchestrators to deploy a new service, perform automatic restarts during crashes, and update a previous deployment to a new version. Furthermore, the orchestrators were limited to the same resources available. The test was completed and data was collected in each of the predefined criteria of success. However, we felt that this test was not good for a few reasons. Firstly, many of the best features of Kubernetes were not accounted for such as more sophisticated error handling, rollback options, and version handling. Secondly, the test was conducted on another system before the mocked version of our current system was created. Therefore, we decided to create a QA test more relevant to our specific system, testing another quality attribute that our system should also have.

III. Reflection

Creating and designing a system from close to no knowledge about industry 4.0, started out being difficult, but as our understanding of the subject became clearer, it became easier to work with. There was a clear jump in the understanding after the initial drawings were made, so everyone could visualize the components present for the production. During the research, and related works, we gained insights into how other people design architecture and how useful related works can be for research. We believe we have learned from the experience of designing an Architecture and testing the Quality Attributes. However, the architectural design has not been tested, and therefore it is not possible to conclude if we have succeeded in creating an architecture that lives up to all the requirements. While the experiment took a lot of time to conduct, the knowledge and experience we gained from going through the whole process seems irreplaceable. Thoroughly researching existing solutions in scientific papers Even though finding the scientific papers proved difficult, once we found a few, the amount of information was great. By using a scientific paper instead of researching solutions on our own, we gained a deeper understanding of not only the technologies but also the problem domain, understanding that it's not just about lines of code; it's about making decisions that impact the real-world performance and scalability of a system. Going through the whole process of researching the subject in existing articles, while modifying it towards our system's specific quality attributes, to testing and analyzing the data felt like the right way to learn to design a system. Looking back at earlier semester projects, the process we used seems very chaotic in comparison, as we would usually simply guess about the system required to fulfill our requirements, instead of doing thorough research about previous implementation and state-of-the-art like we did for this project. For the data analysis of our tests, we believe a more statistical analysis approach would improve our evaluation. We concluded that the response time was less than 1s, but another approach could be to calculate the 99 percentile response rate or 99.9 percentile as well as define a hypothesis for the test which we then could disregard based on the test result.

IV. Conclusion

It has been an educational experience to do research on Industry 4.0 with little to no knowledge on the subject. Both in the personal, and educational view. Having to create a software architecture has been the most important takeaway from this experience, as making our own architecture gained a good insight into the realm of software architecture, both in form of defining the quality attributes to conducting tests on these. Doing research on related works, gives insights into how other engineers design systems was helpful, and a learning full experience. Working in a six-man group was a learning experience, as coordinating this amount of people requires a structured approach, which we were able to do with multiple weekly design meetings. It also requires diplomacy and adapting to different people's backgrounds and ways of working. In the current version of the system, the data is mocked data from the sensors. To further assess the quality attribute of scalability, more parts of the system would be mocked, to see if Kafka could still maintain a low latency. Making the data more similar to the real data of a sensor is also an improvement. The next step would be to start creating tests for the other quality attributes that the system requires, for interoperability, availability, and deployability. Finally, once the requirements of the systems have been confirmed working, it would be time to implement the full system. Nevertheless, we now have a clear conception of what I4.0 is and how challenging it might be to design and implement advanced software architectures capable of controlling complex production systems.

References

[1] Z. Liu, P. Sampaio, G. Pishchulov, N. Mehandjiev, S. Cisneros-Cabrera, A. Schirrmann, F. Jiru, and N. Bnouhanna, "The architectural design and implementation of a digital platform for industry 4.0 sme collaboration," *Computers in Industry*, vol. 138, 6 2022.

[2] P. Sommer, F. Schellroth, M. Fischer, and J. Schlechtendahl, "Message-oriented middleware for industrial production systems," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 1217–1223.