Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Lecture
Integration Architectures
Winter Term 2024 / 2025

Prof. Dr. Sascha Alda
Luca Ringhausen (B. Sc.)

Sankt Augustin, October 15, 2024

Exercise Sheet for Lecture Integration Architecture Winter Term 2024 / 2025

Exercise Sheet No. 1

(Due Date for Assignment No. 1-1: October 15th, 2024, 15:00 pm.
Due Date for Assignment No. 1-2: October 22th, 2024, 15:00 pm)

Assignment No. 1 (First Analysis of the Case Study; Due Date Oct 15th, 2024; Upload on LEA!):

On LEA, you find a collection of documents and artefacts that describe the *current* situation of the performance management process of the company SmartHoover Ltd (folder: “Case Study” → “Documents”). In this assignment, you should make comfortable with the case study, which we will be focusing on within the next months. Also, your semester project will be based on this case study. The case study will continuously be driven by the exercise sheets and by the discussions in the exercise courses.

a.)
Read and study the documentation carefully and answer the following questions:

1) What is the idea of the *current* performance management process *in general*?
2) Which actors are involved in the *current* process of the company SmartHoover?
3) What are the disadvantages of the *current* process?
4) How does the *current* software support looks like? Are there any drawbacks?

b.)
Model a UML-based Package diagram that represents a first high-level view (*context view*) of a software architecture of the *current* system. Please treat each component that you have identified as a black box. Also, include the major actors into your model and consider the most important associations the components. Do also consider usage dependencies between as well as basic data flow dependencies between the package. Further information on context views can be gathered from LEA (Chapter 1 / Material).

c.)
In the semester project, you have the task to implement a performance cockpit for improving the current performance process. Please give a first vision of this cockpit by phrasing 5-10 user stories. Where do you see the necessity for *integrating* software?

When presenting this assignment, the student(s) should also be able to explain and to demonstrate the current software support (OrangeHRM and OpenCRX) to the audience.

Assignment No. 2 (Software for managing Salesmen; Due Date Oct 22th, 2024; Upload on DIArchitect!):

As mentioned in one of the documents, the source code of the program for managing salesmen and their social performance records has been hidden by the admin. However, a portion of the source has been revealed. You can download the Java codes from LEA (folder: “Source Codes and Scripts”) or from the GitHub repository (link follows later!).

The code shows the interface of a control class, which can be used to manage information on salesmen and their social performance records (part B of a bonus computation sheet). The implementation of the interface will be used to access to a MongoDB database for storing and retrieving performance records of single employees. The internals of the MongoDB could not be identified, so you are asked to provide your own implementation.

Take the opportunity to introduce yourself to the database management system (DBMS) MongoDB. MongoDB is part of the MEAN-stack (the ‘M’ of MEAN), which has become a standard solution to implement full stack Web applications. We will also learn the remaining “letters” during the next weeks and months. However, feel free to learn at least a bit more on MEAN in terms of self-study. See the links below this assignment as a recommended pool of appropriate resources to these topics.

a.)
Please provide a Java program that uses the interface of a control class for managing salesmen and their performance records. In the backend, integrate a local MongoDB database that contains of two collections for storing general data for salesmen as well as their social performance records. Please make suitable assumptions on the schema of the two collections. Please do not use frameworks like Spring Boot, use the pure MongoDB driver.

b.)
Does the interface of the control class fulfill the CRUD pattern entirely? If not, please add the missing operations and refactor the given interface accordingly. No implementation of the “U” is required (exception: adding further social performance records to a salesman).

c.)
Test your application sufficiently by developing a Junit-Test Case. If possible, provide roundtrip test to ensure, that now data waste is given after the execution of a test.

Remark: for passing this assignment, you should at least provide a simple implementation covering the creation (C) or the reading (R) salesmen data from a MongoDB database.

For this assignment, you can upload the source codes as a ZIP-file. You can also provide a link to a public Git-Repository, so that I can clone your application. Please provide also short answers to the stated questions (e.g., in a text file).

Various Online resources for this and the next exercise sheets
(Last Access on all links: October 7th, 2024)

1. Resources for MongoDB:

On this GitHub repository, you find some coding examples for accessing MongoDB based on Java and Junit. Besides, you find various helpful links:
https://github.com/aldaGit/mongodbstart/tree/master

More useful links for tools, libraries etc. can be found in the readme.md of the GitHub repository.

2. The MEAN stack (not really relevant for this exercise sheet, just for a short introduction. You know: NO Angular in this exercise sheet!):

A first overview on the whole MEAN stack:
https://www.ibm.com/cloud/learn/mean-stack-explained