Dr. Volker Riediger
M.Sc. Mahmood Al-Doori
Dipl.-Inf. Julian Flake
M.Sc. Veronika Vasileva

— Engineering Web and Data-intensive Systems - Winter Term 2024/25 —

Assignment 1.

From: Nov 18th, 2024 To: Nov 25th, 2024

Notes on the sumbission

Please store your solution in the OLAT group folder. We have already created a `solutions` folder for you.

Create a sub-folder for each assignment according to the scheme `solutions/assignmentNN`.

Put all solution files into that subfolder. We prefer PDF documents. You shall also add other files contributing to the solution, for example Astah models, database scripts (in later assignments).

Serious Advice

If you repeat the EWADIS course, we *strongly recommend* to redo the assignments from scratch. It is not very useful to copy solutions from the last semester, even if you think you have a perfect solution or a reference solution that we provided.

The intent of the assignments is to actively practice, not to present a perfect solution!

1 Architecture - Logical View - Domain Model

Part of the **logical view** of an architecture description is the so-called *domain model*.

Domain models deal with important concepts of an application, i.e., business objects and their relations. In a commercial project, the domain model represents the view of the customer on a system's data. It is the base for more detailed and more technical (implementation) data models.

Domain models are of Entity-Relationship (ER) type. There exist many ER languages, more or less expressive. A suitable diagram language for such models is a UML class diagram.

In principle, you could also use a sketching tool like Visio or draw.io to create the class diagram. Unlike in a simple drawing, in a UML model the various diagrams are *linked* to each other. Using a UML modeling tool enables consistent reuse of the model elements. Model elements can be used and viewed in more than one diagram. For example, if you rename a class, this change is automatically propagated to all diagrams that show the class.

Task:
TSS (Time Sheet System) is a system that is developed to support documenting employment data such as working hours, employment status and vacation time periods. This system is intended for employees whose income is less than 850 Euros. In the context of a university, this system is mainly used by student assistants to report their precise working times on a weekly basis. The working times are documented on time sheets and these time sheets are signed by both the student assistant and the supervisor or secretary representing the university (employer).

Personal data of all the involved aforementioned stakeholders are stored in the system including first name, last name, data of birth and email address. Prior to using the system, users have to give consent to store their data and confirm that they understand the regulations and the protection measures that protect their data. Each person may or may not have a concrete role in the system.

A person with a role has a contract. The system stores contract information such as status of the contract, name, contract starting date, contract ending date, time sheets documentation frequency, termination date (stored as year-month-day), actual working hours in a week, vacation hours, due hours that are yet to be occupied by the employee, working days per week, and vacation days per year.

Contract status is stored in the system as prepared, started, terminated, or archived. Time sheet frequency can either be weekly or monthly. Each role in the system has exactly one contract and each contract has exactly one supervisor, secretary and employee and may or may not have an assistant. Each contract has many time sheets, and each time sheet should have exactly one contract.

Time sheets should have a status, starting date, ending date, due hours that are yet to be occupied by the employee, employee signature and employer signature (each of which should also have a (year-month-day) date. Time sheets status needs to be documented as either a time sheet in progress, signed by employee, signed by supervisor, or archived. Time sheets are dependent on contacts. On the other hand, each time sheet should have many time sheet entries, and each entry should be tied to one time sheet at a time. Time sheet entries are dependent on time sheets. Each entry should have a report type, description, hours, starting time, ending time, and entry date. Starting and ending time should be save as (hour-minute-second), and entry date as (year-month-day). A report type can be either work, vacation or sick leave.

Based on the description above, please create a **UML class diagram** as a domain model for the TSS system.

Remarks:

We expect a solution with around 10-15 classes with some attributes and their relations.

Please submit your solution as PDF file in your group folder under `solutions/assignment01`.

Additionally, please add the *source file* for your model.

We *recommend* to use the Astah modeling tool (see forum posts for additional information).

If you use Astah, please submit the model file `*.asta` in addition to the PDF.

If you are not familiar with UML class diagrams, we recommend the text book *UML @ Classroom* and the online resources related to that book (see link list in OLAT).

Please prepare to discuss your solution in one of the next tutorials!

The goals of this task are:

- Practice to use the Astah tool (we will use it throughout the whole course)
- Recap your UML knowledge
- Review modeling conventions (see the UML reference card)
- Prepare the running example of this lecture
- We get an impression of your modeling skills

Non-Goals:

- You don't need to provide a *comprehensive* model
- You don't need to represent *all* features on detailed level
- You don't need to add *operations* to the classes, it shall only contain data items, their properties with appropriate types, and their relations