Zoo

System Design Document

Authors:
Sebastian Strumbelj
Benjamin Moser
Simon Reufsteck

Contents

1 Changelog

Chance History			
Version	Author	Description of changes	Date
1.0	M,S,R	Inital release	1.1.2018
1.1	M	- Merged modules Order- and Animal-Reporting(*)	5.1.2018
1.2	S	- New module 'Controlling'(**)	6.1.2018

Legend:
S = Sebastian Strumbelj
M = Benjamin Moser
R = Simon Reufsteck

Merged modules *OrderReporting, AnimalReporting, etc. (*)* ...into a single Reporting module since reports almost always work on data from more than one of these domains. Splitting these would lead to many nearly identical submodules performing nearly identical tasks. This does not seem very maintainable.

New module *Controlling (**)* This functionality was missing in the original draft.

2 Introduction

2.1 Purpose

The system is supposed to become a central tool for administration of daily business and management tasks in the Zoo.

2.2 Design Goals

This document outlines the architecture of an implementation of the system.

The system will enable to

- Manage the ordering and storage of animal feed

- Manage invoices and budgets
- View reports on all gathered data

Furthermore, the system will

- Provide safe User Authentification
- Reasonable Access Restriction

The system will not contain

- Coordination of self-driving cars

2.3 Definitions, Acronyms, Abbreviations

SRS Software Requirements Document

2.4 References

- **Software Requirements Document** as per Assignment.
- **Statements from Zoo Employees** as per Assignment 3.
- **Letter from Susan Tapir** as per Assignment 2.

2.5 Overview

First, this document will investigate the current state of affairs , describing what systems are currently in use. Further, comparable systems will be listed and described shortly.

Then, a new software architecture adhering to the recently engineered requirements (cf. SRS) will be proposed. It is described in detail its structure (subsystem decomposition), hardware/software-mapping, management of persistent data, security concerns, control mechanisms (concurrency), as well as behaviour at startup/shutdown (boundary conditions).

3 Current Architecture

There is no comparable system currently in place at the site.

There is an existing hardware infrastructure that will be the basis for the newly proposed software system.

The new software system shows similarities to other business administration tools such as SAP.

4 Proposed Architecture

4.1 Overview

The system is composed of four major subsystems: The database, the business logic, the camera surveillance subsystem and the View subsystem.

Their interdependency and composition of sub-subsystems is illustrated in Figure 1. Note that among the *ViewingSubsys*, there are groups of modules. Each of these has an individual dependency as marked by *each* next to a line. How precisely these subsystem will interact is defined by the interfaces listed in Section 4.6.1.

4.2 Subsystem Decomposition

The sub-subsystems are as denoted in Figure 1. It follows a detailed description of the subsystems and their tasks.

Individual Databases, DBDriver will be part of a Database Management System such as MySQL.

Query Abstraction will abstract database queries from appllication logic requests.

Reporting will create aggregated reports based on raw data from multiple sources

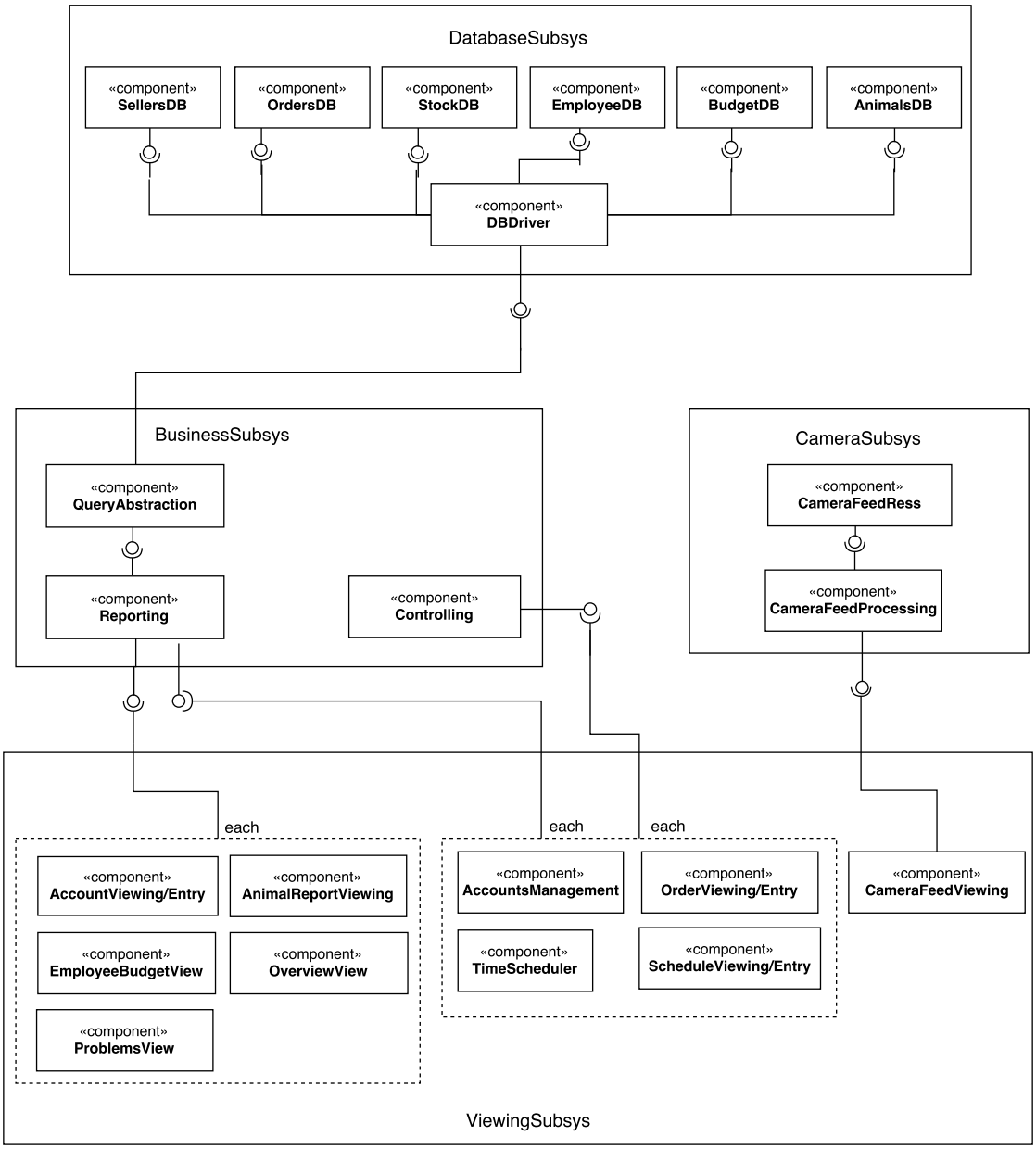


Figure 1: Subsystem Component Diagram

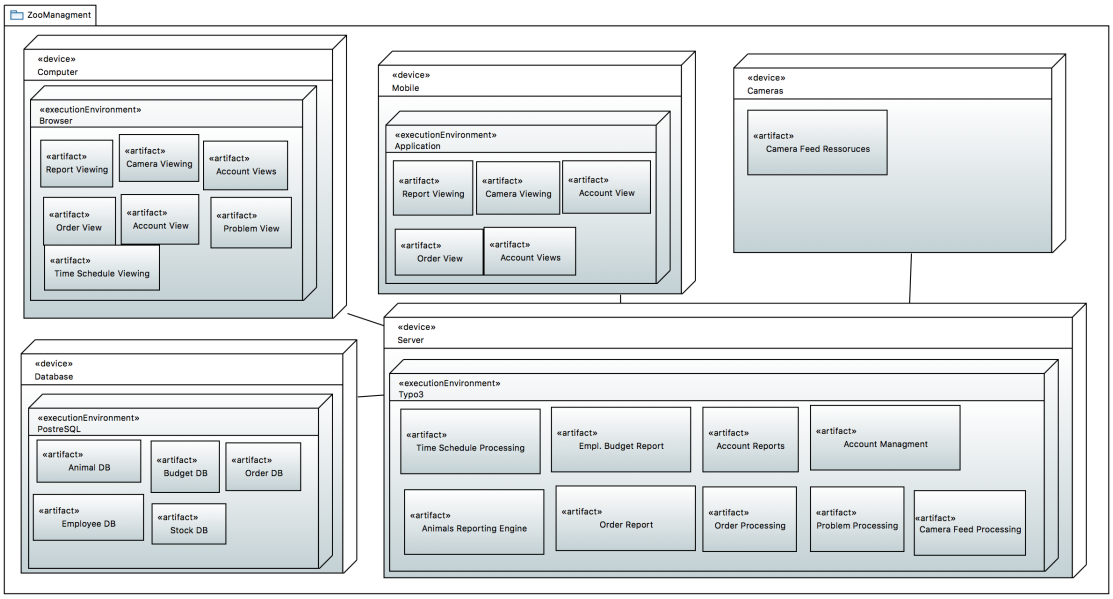


Figure 2: Deployment Diagram

Controlling is responsible for handling user input and computing user-initated tasks, e.g. scheduling a slot for an employee's vacation. (This corresponds to the MVC controller.)

CameraFeedRess makes available the raw footage provided by the surveillance cameras.

CameraFeedProcessing fetches video data from *CameraFeedRess* and applies conversion/compression methods to make it viewable by a client.

ViewingSubsys Components in this subsystem will provide data views customised to their domain. Components in the second group will additionally provide a User Interface to enter/modify data.

4.3 Hardware-Software-Mapping

The deployment is illustrated in Figure 2.

We have five big components for the zoo managment system. The devices of the user are computers and mobile phones. While the computer is using the

browser for viewing the data, the mobile phone is using an app. These programs fetch their data from a server, where all the processing is located. The server is able to access data from a database. The only purpose of the cameras is to record and send the data to the server.

TODO: Describe each area (Server, Database, Mobile) in Detail (e.g. in Browser, what kind of DBMS, in App...)

4.4 Persistent Data Management

In our Subsystem Component Diagram there are six different databases. A database for the orders, the stock, the employees, the sellers, the budget and the animals, specified as follows:

- *OrdersDB:* This DB/Schema(*) includes a table 'orders' where the employee who is ordering, the seller, the procuct id, the price, the date and a unique id for the order is stored. The information about the employee is linked with the EmplDB via an EmplID.

- *StockDB:* This DB/Schema includes a table 'stock', where the product/feed, the number of remaining items and a unique id is stored. It can be joined with the Orders and Animals.

- *EmplDB/SellersDB:* This DB/Schema contains a table 'Employees', where their personal information, like unique id, adress, birthday, salary, etc., is stored. Also it contains a table 'Sellers', where all the sellers are stored with informations like id, company, location, etc. Both tables are specific tables for another table 'Persons', where only ID and name is stored. It can be joined with the Orders.

- *BudgetDB:* This DB/Schema is all about the finance. It contains a table 'Budget' where all the transactions are stored. Every transaction has it's own id, from where money is transferred, the destination, the amount of money and the date. It can be linked with the Stock and Orders.

- *AnimalsDB:* This DB/Schema contains all the animals. They have their own id, a name, the type of animal, the feed, the sex, etc. It can be joined with the Stock.

(*) We don't have a DBMS for every instance/table. Instead we have one big DBMS, with all the schemas and tables in it.

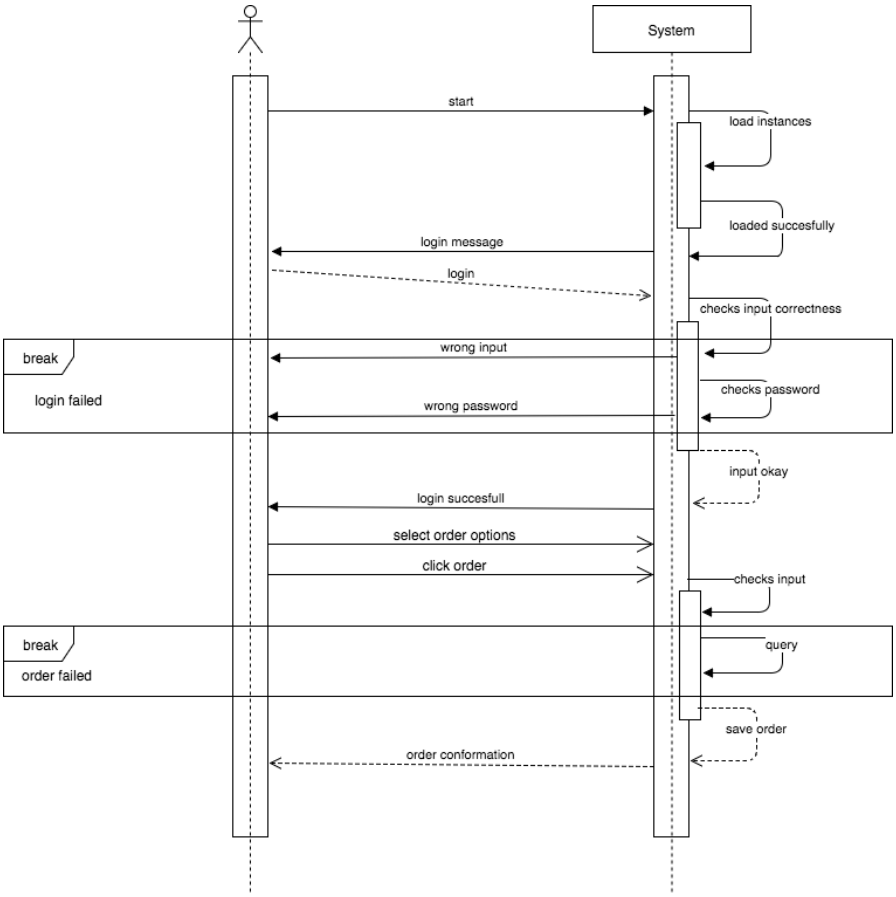
4.5 Access Control and Security

All persons have to login. Only employess, secretaries and the zoo keeper can access the system with their own password. Ordering feed is started by the zoo keeper, then he has to enter the invoice and this has to be stored. During the order process the manager can optionally choose feed dealers or the zoo keeper or the secretary can delete the order. The zoo manager is also able to increase the budget. Additional views for the budget are an extension and can only be viewed by the manager. Anyone else can't access the system.

4.6 Global Software Control

4.6.1 Interfaces

4.7 Boundary Conditions



The sequence diagram shows the sequences for the login and ordering process. The system first initializes and load its instances. Then it's possible for employees to login. The system first checks for correct input (e.g. correct username, no numbers, etc.), then checks the password. If one of both fails, the system has to restart the login process. If both are correct, the employee can order now. Again, the system checks, if the input parameters for the order are correct. Then it runs a query to check the stock and if the feed is available. If it's not, the query fails and the employee has to repeat the order process. If the order was successful, the system saves it and sends a confirmation.

5 Appendices

5.1 Changed Requirements

List changed requirements in old and new form.