Linnæus University

2DV103 – Software Engineering Project

In this project you should work *in groups (3 members)*.

If you have questions, post them on the forum. It is ok to help each other in public forums. For any question about the Project use **"SEP Forum"**.

Please note that all tasks should be documented, that is, a model without text that explains it has little or no value. Use tools to prepare documentation about modeling.

Time log

Track the time you spend on this assignment. Create a table where you log time spent on every activity for each task. Include the log in your submission.

Grading

Your submission will receive a grade from A to F where F is Failed. You are allowed to improve your work after the initial submission before the deadline. The grade is final, i.e., you will not get an opportunity to correct/improve after grading.

Your answers should be your own! You are not allowed to copy code, models, or texts (books articles, blogs, wikis) in your answers! Each submission will pass through a plagiarism/clone detection system before correction. If plagiarism is detected, the assignment is failed and a formal investigation will be initiated.

Submissions that arrive after the due date will be downgraded by 25 %-units.

Evaluation Process

1. Submit a zipfile, containing the report and the source code, to MyMoodle by **13/05/2018**
2. Have a project demo with examiners on **w20** (Room 1173A/Skype)

Mauro Caporuscio mauro.caporuscio@lnu.se

Department of Computer Science Lnu.se

Re-engineering Legacy Software

The goal of this Project is to create a system to manage the front-desk activities of the “Linnaeus Hotel”, which manages 2 different buildings, 1 within the Växjö Campus and 1 within the Kalmar Campus.

You have been contracted to re-engineer the existing legacy software since your customers believe a modern and more efficient system will save money and help them to serve guests better. The system will be used to enter reservations as well as to check guests in and out of both hotel’s building.

Each building contains rooms in which guests can stay. Some hotel rooms adjoin others; that is, there are internal doors between them. Each hotel room is assigned a quality level (e.g. a larger room or a room with a view would be better than a smaller room without a view). Each room also has a certain number and type of beds, a room number, and a smoking/non-smoking status. Each quality level has a maximum daily rate, although the rate that a guest pays may be less.

When a hotel guest wishes to make a reservation, the hotel clerk asks him or her which nights he or she wants to stay and the type of room he or she wants. The system must verify if rooms are available in both buildings on those nights before allowing a reservation to be made. If rooms in a specific campus (e.g., Växjö) are not available, the guest should be offered to stay in a room within the other campus (e.g., Kalmar), according to current availability. To avoid the potential guest getting anxious while waiting for the response about room availability, the verification procedure must take less than 2 seconds to complete.

The hotel needs to record basic information about each guest, such as his or her name, address, telephone number, credit card, passport number, etc. A reservation can be canceled at any time but some fees (a percentage of the room price) may be charged if the cancelation is done too late.

When a guest checks in, a room is allocated to him or her until he or she checks out. When the customer requests a specific room, this can be allocated in advance at the discretion of the manager. The system must keep track of the guest’s account and print his or her bill. To avoid guests standing in a line while waiting for the checkout, the whole procedure must take in average less than 60 seconds to complete.

Task 1 – Requirements Specification Document

Starting from (1) the above description, (2) the 3 Requirement Specification documents submitted as Assignment 1, and (3) the analysis of the legacy system (see attached zip), develop a new and refined Requirements Specification document for the system-to-be, including both functional and quality requirements (informal descriptions and models).

Task 2 – Re-engineering Legacy Software

Re-engineer the Legacy Software to implement the set of functional and non-functional requirements specified in Task 1. In order to develop a high-quality software, apply principles and practices from the following areas:

- Quality Management
- Performance engineering
- Architecting and Designing
- Refactoring and Re-architecting or Rewriting

Provide a **Design Document** reporting the following info:

1. **Purpose**:
    - What system or part of the system this design document describes
    - Make reference to the requirements that are being implemented by this design (traceability)
2. **General priorities**:
    - Describe the priorities used to guide the design process
3. **Outline of the design**:
    - Give a high-level description of how you will re-engineer the application (Re-factoring/Re-architecting or Re-writing) and motivate your choice.
    - Give a high-level description of the design that allows the reader to quickly get a general feeling for it
4. **Major design issues**:
    - Discuss the important design issues that had to be resolved
    - Give the possible alternatives that were considered, the final decision and the rationale for the decision
5. **Design details**[1]:
    - Software Architecture
        - Reference Architectural Pattern (if any)
        - The set of components
            - For each component: discuss both requires and provides interface
        - The UML Component Diagram of the architecture
    - Component Implementation
        - For each component identified in Software Architecture
            - Design Principles addressed (if any)
            - The UML Class Diagram

Task 3 – Application Demo

1) Prepare a 15-mins presentation where you summarize your report
2) Prepare a 5-min live demo showing how your application works

[1] **NB**: Motivate all Design decisions