System design document for Coffee Break

Version: 0.1

Date: tisdag den 11 april 2017

Author: Felix

This version overrides all previous versions.

# 1 Introduction

General info. What is this? What does it describe?

## 1.1 Definitions, acronyms and abbreviation

Todo-list – A list of different tasks that the user wishes to get done.

Task – The simple item that the user can add to his/her Todo-list

List task – A more complex version of a task, which creates the Task with the specified name and also contains a list of different tasks that the user can use to specify a list of tasks in a single place. This way, the user receives another option when they wish to organize/structure their Todo-list.

Time Category – The category which tasks can be sorted into which involve a certain timeframe

Label Category – Custom categories that the user can create through either adding custom labels/tags onto their task during creation, or through setting up static categories which are always visible.

Test-driven development – Abbreviated as TDD. Before any new code is written for the application, a test for the specified component will be make using the different specifications for the component as guidelines. This will lessen the number of bugs in the end product. The procedure can be read in depth at <https://blog.jetbrains.com/idea/2016/12/live-webinar-the-three-laws-of-tdd/> .

MVC – “Model, View, Controller”, a design model used in the most applications/programs today. The Model is the database which handles all of the logic and calculations. The View is the what is actually shown to the user, and the Controller handles the interaction between the user and the Model.

Object Oriented implementation – A certain form of programming paradigm, where the coding is divided into different objects and classes. This is to break up the different tasks into smaller, more manageable parts and then tackling the problem by creating one “puzzle piece” at a time.

# 2 System architecture

The most overall, top level, description of the system. Which (how many) machines are involved? What software on each (which versions). Which responsibility for each software? Dependency analysis. If more machines: How do they communicate? Describe the high level overall flow of some use case. How to start/stop system.

An ​UML deployment diagram​, possibly drawings and other explanations. Possibly UML sequence diagrams​ for flow.​

(Persistence and Access control further down)

Any general principles in application? Flow, creations, ...

# 3. Subsystem decomposition

For each identified software above (that we have implemented), describe it ...

## 3.1 “...First software to describe” ...

Recap: What is this doing (more detailed)

Divide it into top level subsystems. An ​UML package diagram for the top level. Describe responsibilities for each package (subsystem). Describe interface. Describe the flow of some use case inside this software. Try to identify abstraction layers. Dependency analysis Concurrency issues.

If a standalone application

* Here you describe how MVC is implemented
* Here you describe your design model (which should be in one package and build on the domain model)
* A class diagram for the design model.

else

* MVC and domain model described at System Architecture

### Diagrams

* Dependencies (​STAN​ or similar) - UML sequence diagrams for flow.

### Quality

* List of tests (or description where to find the test)
* Quality tool reports, like ​PMD​ (known issues listed here)

NOTE: Each Java, XML, etc. file should have a header comment: Author, responsibility, used by.., uses ...

## 3.2 “...next software to describe” ...

As above….

# 4. Persistent data management

How does the application store data (handle resources, icons, images, audio, …). When? How? URLs, pathe’s, … data formats… naming..

# 5. Access control and security

Different roles using the application (admin, user, …)? How is this handled?

# 6. References

The three laws of Test Driven Development - <https://blog.jetbrains.com/idea/2016/12/live-webinar-the-three-laws-of-tdd/>