Magdeburg, Nov 10th 2017

Marcus Pinnecke, M.Sc.
Prof. Dr. Gunter Saake

Winter Term 2017/2018

Software Engineering

submission deadline: Nov. 24th 2017 11:59pm

Exercise sheet No. 3

After working on this sheet you will have learned the following

- ✓ one agile development method
- ✓ most popular software design pattern

REVISION 2: changes marked red

REVISION 1: changed marked gray

This is a *per-student exercise* sheet, i.e., you are not allowed to submit a group solution.

You must submit your solution in time via a pull request to the lectures exercise Git repository[1] (cf. sheet No. 1). For this, branch from the official branch `master`, and commit your solutions in a directory `sheet_03/<your last name>`. For your pull request, use the remote branch `submissions` as target for merging.

Prepare to present details of your solution during the tutorium.

This sheet consists of the following tasks:

Task 1	**Methods for Agile Software Development**	8 Point
Task 2	**Software Modeling with Design Patterns**	8 Point

You are now in the perfect shape for developing your project My Main Memory Database System (M3DB). Since you talked about a lot of your plans, several new teammates joined your project. Therefore, your development groups to a size that you have to think about applying a particular *agile software method*. Besides this, you make yourself familiar with solutions to ever occurring problems in designing applications and systems, *software design pattern*.

Good Luck!

[1] https://github.com/Arcade-Lecture/exercises.git

Task 1 Methods for Agile Software Development — 8 Point

During the last lecture, you come in touch with several agile software development methods:

- Extreme Programming
- Kanban
- Scrum
- Feature-Driven Development
- Lean Software Development
- Dynamic Systems Development Method
- Adaptive System Development

Pick one of those methods, and provide a textual solution formatted in Markdown in *<repo>*/sheet_03/*<your last name>*/task_1:

~~(1) Give a statement why you pick this particular method and why not another one!~~

(2) Give a definition (or at least an in-depth description) on what the characters of your particular method is! State the purpose and the principles you chosen development method!

(3) Provide an overview on the processes and their connections to each other! For instance, for *Scrum* you will describe the terms

a. *sprint*
b. *planning*
c. *meeting*
d. *product*
e. *feature*
f. *assignment to sprint*
g. *back log*
h. *daily scrum meeting*
i. *release*
j. *increment*
k. *sprint review*
l. *product backlog*
m. *prioritization*
n. *scrum master*
o. *product owner*
p. *team*

and give relationships between those terms in order to describe the *Scrum Process Framework* and *roles*. For instance, you will describe the relationship between the *scrum master* and the *product owner* w.r.t. responsibilities and deliveries.

(4) Provide a comparison with at least one other development method focusing on commonalities and differences!

(5) Discuss when it is more useful to choose your particular method and the other(s) that you add in (4).

Task 2 Software Modeling with Design Patterns — 8 Point

In the lecture you learned visualizing software design aspects using UML. This task is about the opposite direction. In software design, there occur problems over and over again – and over time, some design solutions were found that face these problems.

Popular patterns are listed below, categorized in their typical classes:

1. **Creational patterns**: abstract factory, builder, factory method, prototype, singleton
2. **Structural patterns**: adapter, bridge, composite, decorator, facade, flyweight, proxy
3. **Behavioral patterns**: chain of responsibility, command, interpreter, iterator, mediator, memento, observer, state, strategy, template method, visitor
4. **Concurrency patterns**: active object, balking, binding properties, double-checked locking, event-based asynchronous, guarded suspension, join, lock, monitor, proactor, reactor, read write lock, scheduler, thread pool, thread-local storage
5. **Architectural patterns**: front controller, interceptor, model–view–controller, action–domain–responder, entity–component–system, *n*-tier, specification, publish–subscribe, naked objects, service locator, active record, identity map, data access object, data transfer object, inversion of control, model 2
6. **Diverse**: blackboard, business delegate, composite entity, dependency injection, intercepting filter, lazy loading, mock object, null object, object pool, servant, twin, type tunnel, and method chaining

(1.) Chose one **creational** pattern, describe it and give an UML representation!
(2.) Chose one **structural** pattern, describe it and give an UML representation!
(3.) Chose one **behavioral** pattern, describe it and give an UML representation!
(4.) Chose one **concurrency** pattern, describe it and give an UML representation!
(5.) Chose one **diverse** pattern, describe it and give an UML representation!
(6.) Bonus task: Chose one **architectural** pattern, describe it and give an UML representation! (+ 1.6 points)

Tip: Feel free to make a choice by your own. We recommend the ones that are underlined.
Tip: For most patterns, an UML class diagram is the best choice for visualization.
Tip: Look for good books or web resources that provide you a tutorial on these patterns. You may start here: https://www.youtube.com/watch?v=vNHpsC5ng_E&list=PLF206E906175C7E07

For textual solution, use a Markdown-formatted file and store it in <*repo*>/sheet_03/<*your last name*>/task_2. Use the same directory for UML figures. Feel free to use a tool in order to create diagrams.