Software Product Line Engineering

Summer 2019, Assignment 4

Bauhaus-Universität Weimar

Prof. Dr.-Ing. Norbert Siegmund
Stefan Mühlbauer, M. Sc.

Assignment issued:	Friday, 24 May, 2019
Submission due:	Thursday, 6 June, 2019, 23:59 CEST
Presentantion:	Friday, 7 June, 2019

Task 1: Frameworks and Components (20 marks)

(a) In your own words, briefly define the following terms (3 marks)

(i) *software framework*

(ii) *software component*

(iii) *service-oriented service architecture (SOA)*

(b) What are the characteristics of a *white-box* and a *black-box framework* (2 marks)? Under what circumstances is the use of either option preferable, when developing a SPL? (2 marks)

(c) Certain design pattern are associated with the use of a white-box and black-box frameworks. For both framework types, describe, which design patterns should be used to implement variability (2 marks) and why (2 marks).

(d) How can components and services be used to develop a SPL? (1 marks)

(e) What are the pros and cons in doing so? What are strengths and weaknesses of using components and services that purpose, compared to other implementation techniques? (4 marks)

(f) Briefly discuss the use of component-/SOA-based and framework-based approaches with respect to the following aspects (4 marks):

(i) Is this practical for for proactive, reactive, and extractive SPL development?

(ii) Is this practical for embedded systems (i.e., systems with strict limitations on resource utilization)?

(iii) Are special skills required for developers (i.e., effort to learn additional techniques etc.)?

(iv) Does this scale for projects with many features and/or files?

Task 2: Cross-Cutting Concerns / Preplanning Problem (DE only, 7 marks)

(a) In your own words, define the terms *cross-cutting concern* and *tyranny of the dominant decomposition*. Provide a brief example for each one, respectively. (4 marks)

(b) In your own words, what is the *preplanning problem* (1 mark)?

(c) How does the preplanning problem relate to class inheritance? (2 marks)

Task 3: Framework Implementation (16 marks)

Re-implement the chat application from the previous assignments as a ***black-box framework*** (10 marks). You can and are advised to reuse your own implementation and feature model. As with the last practical task, the following features need to be implemented: Color, Authentication, Two Encryption modes, Logging, CLI/GUI. In addition, a ***new optional feature Spam Filter*** (4 marks) needs to be implemented, which suppresses all messages that contain words from a blacklist of words on server-side.
For the blackbox-framework, all selectable features should correspond to an individual plugin. Again, every possible configuration should compile and run. For submission, provide a variant with the maximum number of optional plugins enabled.

Hint: Create an interface that provides all necessary extension points such that different spam filter plugins can be used. You can specify a variant (spam filter implementation) using the main method; you do not have to create a generic plugin loader. During the implementation, keep an eye on the following questions: How much effort is spent creating a plugin? How frequently do you have to revise the framework or interfaces, because your initial draft does not fit?

Submit your own answers (PDF) and implementation (Eclipse project) as an archive with name and matriculation number until 6 June, 23:59 CEST to `stefan.muehlbauer@uni-weimar.de`.