



**Software Product Lines**

**Assignment 3**

*Tasks 1-2 are designed as live exercises for the tutorial session – for this, we will split up in breakup groups. It's not required to submit anything for these tasks.*

*Tasks 3 is to be completed offline and to be submitted until the deadline.*

**Task 1: Revisions vs. Versions vs. Variants**

Explain the difference between the concepts *revision*, *version* and *variant* using an example. How could they be applied to the chat product line?

**Task 2: Differences between variability mechanisms:**

In the course of the lecture, you have now learned about the following variability mechanisms:

- Runtime variability with parameters (global variables or parameter passing)
- Design patterns
- Version control systems
- Build systems
- Preprocessors

Which implications do these mechanisms have to the following aspects? Which benefits and drawbacks can one expect when applying them to the chat product line?

- a) Performance at runtime
- b) Collaborative development in a large team of developers
- c) Buying functionality from a third-party vendor
- d) Fine-granular extensions

### Task 3: Preprocessor implementation

- a) Use a preprocessor (such as Munge or Antenna for Java) to make colors, authentication, both encryption algorithms, and logging optional. I recommend to use FeatureIDE for this task. The FeatureIDE tutorial at <https://github.com/FeatureIDE/FeatureIDE/wiki/Tutorial> includes a section on using Munge and Antenna.
- b) As an additional feature, implement an alternative user interface. If you already have a GUI, you can implement a command line interface, and vice versa (a read-only command line interface is OK, no input required). Ensure that the user can selected between two user interfaces during the configuration of your chat product line.
- c) Generate and manually test a few (at least three) variants.

Submit a zip archive with the source code of your implementation, and a test report PDF. The test report should include for each tested variant: a) the used feature selection, and b) a screenshot of the resulting system when executed.