CAS 703 Term Project Winter 2023

Your term project (which will be done in pairs, exceptionally a group of three) involves building a domain-specific language (DSL) and editor using the tools and techniques you learned in the course. The deliverables for the project include a short report (due **12 April)** and a zipfile (emailed to the instructor, also due **12 April**) containing your implementation.

The project objectives and deliverables are as follows.

1. Come up with a concept for a DSL that you would like to build. It can be from any domain that interests you. Focus on identifying a task that you would like to automate and draw inspiration for a language from that. For example, perhaps you'd like to automate the layout of objects in a drawing tool, or to generate guest lists for a wedding, or check constraints on a feature model. Get inspiration from the literature on DSLs, MDE and programming languages and come up with a short (maximum one paragraph) description of the DSL: what features might it possess, and what task will be automated? Send this to the instructor, who will provide feedback. **Suggestion:** think about *mandatory requirements* and *nice-to-have requirements.* **Deadline: 22 February by email.**

2. Use Emfatic/Ecore to define a metamodel for the domain-specific language described in Part 1. Include a class diagram of the metamodel in your report, discuss the metamodel, state any assumptions you have made, and explain any alternative design decisions that you have considered and discounted [Maximum 2 pages in your report.] **Deadline: 12 April**

3. Use any of the technologies introduced during the course to define a concrete syntax and implement a supporting editor for your language. Provide a screenshot of your editor, discuss and justify your syntax design and editor implementation decisions, and reflect on the strengths and weaknesses of the selected concrete syntax compared to alternatives. [Maximum 2 pages in your report] **Deadline: 12 April**

4. Use the Epsilon Validation Language to implement any validation constraints which cannot be expressed in the metamodel itself. Briefly explain the rationale and implementation of each constraint. Integrate the implemented constraints with the editor discussed in (3) to provide in-editor errors/warnings and briefly discuss how you achieved this integration. [Maximum 2 pages in your report] **Deadline: 12 April**

5. Implement a model management operation on your language, e.g., a model transformation, a model-to-text transformation, an analysis. Describe your model management operation, including any challenges associated with its implementation, and any design choices you had to make. [Maximum 2 pages in your report.] **Deadline: 12 April**