Course Unit Software Language Engineering

Masters in Informatics and Computing Engineering (M.EIC)

Department of Informatics Engineering, University of Porto/FEUP

1º Semester - 2024/2025

João Bispo

A DSL for Data Extraction from Arbitrary Sources

Objectives: To learn and to apply the knowledge acquired in the Software Language Engineering course unit by developing a DSL for extracting tabular data from arbitrary sources. The DSL should be able to define adapters that extract tables from structured sources (e.g. JSON, XML, YAML), and provide table manipulation capabilities.

Introduction and Assessment

In this course you will develop a semester-long project that will allow you to apply the knowledge acquired within the Software Language Engineering course unit.

The objective of the project is to develop a Domain-Specific Language that allows the extraction, manipulation and writing of tabular data from arbitrary sources.

Figure 1 shows the high-level flow of the project. The DSL will be able to specify configurations for input files that indicate how to extract tables from them, specify actions to allow basic processing of the extracted tables, and a way of outputting the tables in a given format.

You will receive three assignments, where each assignment will take several weeks to solve, and will build upon the previous assignment. Each assignment has a corresponding checkpoint during the semester, with the following dates and weights for the final project grade:

- 1st checkpoint: 10%, 2024/10/11
- 2nd checkpoint: 30%, 2024/11/15
- Final presentation: 60%, 2024/12/20

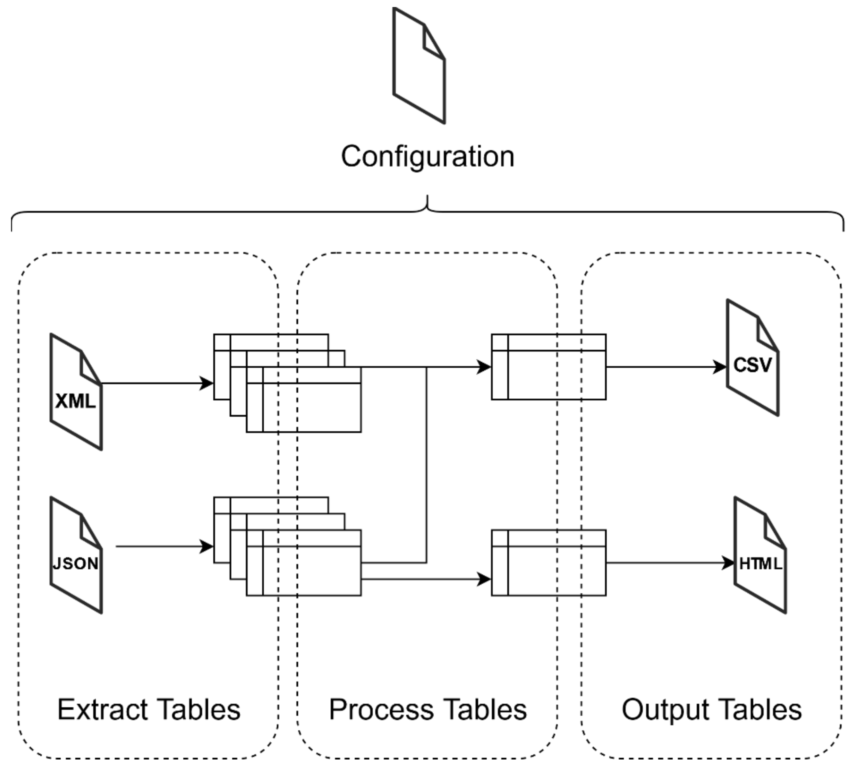

Figure 1. High-level flow of the project

The project will be developed in groups of 3 people, using a provided base Gradle project and a Git repository, with a report at the end of each checkpoint in the form of a "readme" file.

It is expected that the group manages and balances the dedication of each member to the project. The work distribution must be reported to the professors and documented in the final report.

Project Overview

The complete project has several milestones, which will be gradually achieved until the final delivery:

- Creation of a semantic model to represent the data tables and their manipulation.
- Creation of configurable table readers, to obtain tables from different kinds of sources.
- Creation of an internal DSL that allows working with the readers and the semantic model.
- Creation of an external DSL that will use the internal DSL and the semantic model.
- Expansion of the supported functionality of the semantic model and DSL.

The benefit of a DSL usually comes from being extremely focused on a problem. Consequently, good DSLs should have limited functionality, and do not try to do "too much". In the context of this project, the objective of this DSL is to extract and aggregate data that will be processed by other tools that are more suited to that job (e.g., Excel, Matlab, R), or visualize the data (e.g., outputting the tables as

HTML). However, it might be convenient to have certain features for data processing embedded in the DSL (e.g., calculation of averages, sums).

Keep this in mind when working on your project, one of the goals of this course is for you to develop your skills in designing a language (or API) and extract requirements based on the needs of the users. The assignments will be based on use cases, which include example files that your project must be able to process.

Assignment #1

Alice has several YAML report files, in a single folder, from which she wants to extract data related to some parameters of a decision tree classifier. She was able to locate the data in the YAML files as being under the item 'params', and would like to automatically extract some values to a CSV file, using slightly different names for the columns, and a specific order: criterion (becomes "Criterion"), splitter (becomes "Splitter"), ccp_alpha (becomes "CPP Alpha"), and min_samples_split (becomes "Min Samples Split"). Each YAML file contributes to a line in the final table, which would have an additional column File, as the first column, and with the name of the file.

By the end of this assignment, it is expected that you have:

- An initial version of the semantic model which will include the tables, as well as mechanisms for extracting tables from a set of sources, basic interaction between tables, and writing the tables to a file.
- As an initial interface, for now consider that your application accepts as input a configuration file (e.g. JSON, XML, YAML) which will define how to extract the data from which sources, and the necessary actions to generate the final table.

Tips:

- Think of the configuration file as a first version of your DSL, which might need to declare and configure certain elements, and define actions over those elements.
- For now, you can consider that your application only supports one extraction configuration and one source. In further assignments it is expected that the application supports multiple configurations and sources.
- We recommend separation of interfaces (e.g., *interface* or *abstract class* in Java) from implementations (e.g., *class* in Java)
- IMPORTANT: do not write a YAML parser! There are already many parsers, use a library. However, you can (and should) write your own functions/API over existing libraries, focused on your specific need. Remember that designing an API *is* designing a language.