## Architecture

From a technical perspective, there are several choices to be made in terms of software language and libraries. The choices are dictated by the technical requirements of the application. In this section, all requirements are listed and motivate the architectural choices made.

### Requirements

Tables 1, 2, 3 and 4 show the data input, functional, non-functional and technical requirements, respectively.

Table 1: Data input requirements

|  |  |  |
| --- | --- | --- |
| Input ID | Description | Provided by |
| DI1 | Financial data as XBRL | Martijn Smet |
| DI2 | Financial data as Spreadsheet | Martijn Smet |

Table 2: Functional requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Depends on |
| FR1 | The application must transform the XBRL containing financial data to a spreadsheet. | DI1/DI2 |

Table 3: Non-functional requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Depends on |
| NFR1 | The application should transform the XBRL in an automatic way | FR1 |
| NFR2 | The application should be configurable in order to be flexible to changes | FR1 |
| NFR3 | As the transformation can require some time, the application should be able to display that the transformation process is advancing | FR1 |

Table 4: Technical requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Description | Depends on |
| TR1 | DI1 is an XML file within the XBRL namespace | DI1 |
| TR2 | The output format of FR1 should be an XLSX file | FR1 |
| TR3 | The application should use a JSON file as configuration file | NFR2 |
| TR4 | The application should have a command line interface | NFR1 |
| TR5 | The application should be multi-platform | FR1 |

### Programming language choice [TR6]

After looking at existing “XBRL to XLSX” applications (see spreadsheet in doc folder) it was possible to observe that:

* Most of the active libraries are done in Python followed by NodeJS and Java, which respect [TR5]
* The preliminary research shows that Python and Java have already standard libraries for XBRL parsing and XLSX serialization as follows:

|  |  |  |
| --- | --- | --- |
| Language | XBRL libraries | XLSX libraries |
| Python | Python-XBRL | LXML, Openpyxl |
| Java | Xerces | Apache POI |

Also NodeJS has some XBRL libraries but they are not maintained as well as the same for Java API for XBRL.

Among the 3, Python (version 3) has been chosen.

### XBRL parsers [TR1]

Once the programming language has been chosen, a detailed analysis of existing XBRL parsing libraries in Python has performed.

The research highlights that there are 2 existing libraries that could potentially be used:

|  |  |  |
| --- | --- | --- |
| Library | License | Notes |
| [Python-xbrl](https://github.com/greedo/python-xbrl) | Apache license | * The Python-XBRL library has a test folder in which the test case could be the starting point of the application. |
| [Arelle](https://github.com/Arelle/Arelle) | Apache license | * Arelle provide web services API to which Excel could connect and it has a plugin to save into Excel * So there could be 3 options: 1) connect Excel to the webservice using VBA 2) using a python script to connect to the webservice   3) With the plugin save into a Excel file and from there one can manipulate the excel file in python with the XSLX writers |

Arelle looks more complex, however allows different choices, if connecting to the web service and extracting the data from Python would not be difficult that would be the preferred choise.

Python-xbrl is simpler however is maintained by just one person which seems not have too much time to maintain it.

### XLSX writers [TR2]

Once the programming language was chosen, a detailed analysis of existing Excel writing libraries in Python was performed.

The research highlights that there are 4 existing libraries that could potentially be used:

|  |  |  |
| --- | --- | --- |
| Library | License | Notes |
| [PyExcelerate](https://github.com/kz26/PyExcelerate) | BSD | * It seems to be less maintained |
| [XLSXWriter](https://github.com/jmcnamara/XlsxWriter) | BSD | * It seems to be more performant in terms of time but consumes memory (<https://github.com/jmcnamara/XlsxWriter/blob/95334f999d3a5fb58d8da3197260e920be357638/dev/docs/source/working_with_memory.rst>) |
| [Openpyxl](https://bitbucket.org/openpyxl/openpyxl) | MIT/Expat | Compared with XSLXwriter has more contributions |
| [Pandas](https://github.com/pandas-dev/pandas) | BSD 3 | * Pandas can rely on XSLXWriter or Openpyxl to write XLSX files |

Among the 4 it seems that OpenPyxl could be a good candidate.

### Command line interface [TR5]

Python has a built-in command line interface however our research highlights that there are existing libraries that can facilitate the parsing of command line input:

|  |  |  |
| --- | --- | --- |
| Library | License | Notes |
| [Cement](https://github.com/datafolklabs/cement) | BSD 3 | * It is more a framework than a library, it supports:   + Config files (config,json,yaml)   + Log files   + output (json, yaml, genshi, handlerbars, jinjia2, mustache tabulate) |
| [Click](https://github.com/pallets/click) | BSD 3 | * It is a library with plugins supporting:   + Config files   + Log files   + documentation generation (Sphinx)   + completion (bash) * it integrates a progress bar functionality * The library has been developed by a developer who created other famous applications (Jinjia, Flask framework) |
| [Cliff](https://docs.openstack.org/cliff/ocata/install.html) | Apache license 2 | * It is more a framework with:   + documentation generation (Sphinx)   + completion (bash)   + interactive   + output (csv, table, value, yaml, json, html, custom) |

Among the 3, Click is chosen for the following reasons:

* It is a library which can be extended with plugins;
* It has a progress bar so that [NFR5] can be satisfied;
* It is used in [CookieCutter template for Data Science](http://drivendata.github.io/cookiecutter-data-science/) beneficial for applying a [Convention over Configuration](https://en.wikipedia.org/wiki/Convention_over_configuration) approach in the application.