# Requirement Specification

## Functional requirement

Table 1 Functional Requirement specification

|  |  |
| --- | --- |
| **RQ code** | **RQ description** |
|  | **Input Data** |
|  | File format “.csv” with the original data (see Appendix A) |
|  | File format “.csv” which contains one or several instance. Structure for both F1.1 (see Appendix A) |
|  | File format “.xml” which contains logical expression (see Appendix B) |
|  | Separator by “.csv” files (default separator is “;”. See NF5) |
|  | File F1.1 and F1.3 is a “.zip” archive |
|  | **Output data** |
|  | Task name |
|  | Agent state (returns at the request of the coordinator his state and if he is busy then the task name) |
|  | Result data calculation. Form a result calculation is a “.xml” file, who contains logical expression and values of D (F5.2.2), C (F5.2.3), A (F5.2.4). Structure in Appendix C |
|  | Values by each class |
|  | Values by each column |
|  | Values by each column in class |
|  | Result “.xml” file must be contain comparison result with original and object “.csv” files in bool value – true or false |
|  | For original file comparison result is false |
|  | **Communication with the coordinating agent** |
|  | All input data must be received from the coordinator's agent |
|  | Output data must be send to the coordinator's agent |
|  | All data calculation is stored only within the agent, and is not available to other agents |

Table 1(cont.) Requirement specification

|  |  |
| --- | --- |
|  | **Input processing** |
|  | Original “.csv” contains a set of characteristics (in the form of a columns). |
| **RQ code** | **RQ description** |
|  | Logical expressions is separated in many sub expressins. Separator is disjunctive operations. |
|  | – is a composition of unique value of characteristic, which are combined in the logical expression conjuncture |
|  | – is minus concrete value of sub logical expression |
|  | – is concrete value of sub logical expression |
|  | **Agent logic** |
|  | **Work with date storage** |
|  | Save original “.csv” data |
|  | Reset original data (if an instruction is received) |
|  | **Calculation** |
|  | Calculation number of the characteristics of an information expectation *()* (see F4.3) |
|  | Calculation (Degree of discrepancy). The formula is |
|  | Based on the formula, we form (the number of the not combined characteristics, see F4.4) |
|  | Calculation (Degree of combining). The formula is |
|  | Based on the formula, we form (the number of the combined characteristics, see F4.5) |
|  | Calculation (Adequacy). The formula is |
|  | Compare original and object calculation result |
|  | **Send result of comparison to coordinator agent** |
|  | Wait, if the coordinator agent is busy |

## Nonfunctional requirement

Table 2 Nonfunctional Requirement Specification

|  |  |
| --- | --- |
| **RQ code** | **RQ description** |
|  | Language is Java |
|  | Use JADE (Java Agent Development Framework) |
|  | “.xml” file will be parsed by JDOM Parser |
|  | “.csv” file will be transform to matrix. Use cycle, who parse row by method String[][] matrix =line.split(“;”) |
|  | Separator in “.csv” files by default is “;” |
|  | Structure of “.xml” file |
|  | Attribute “reverse” – if "true", then the negation operation is applied to the property. If "false" - a pure value is used. |
|  | “obj-in” – is not used |
|  | If the name of node names not contain “\_” the node is skipped during processing |
|  | Use “ByteZipArchiverJ8” to encode archive F1.5 |

## Test Data

### Explanation of input data

In this example, the data received from the “.csv” file. An example of a file is in appendix A. By default, characteristics are separated by the symbol ";".

In a table view, this file looks like this:

Table 3 File for Appendix A in a table view

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cb1 | Cb2 | Cb3 | Cb4 | Cb5 |
| 3 | 8 | 1 | 5 | 2 |
| 1 | 2 | 2 | 4 | 2 |
| 4 | 1 | 3 | 3 | 2 |
| 6 | 9 | 4 | 1 | 3 |
| 3 | 2 | 1 | 6 | 1 |

Next, we have A formula on “.xml file”. An example of a file is in appendix B.

In a formula view, this files looks like this:

# Class Diagram

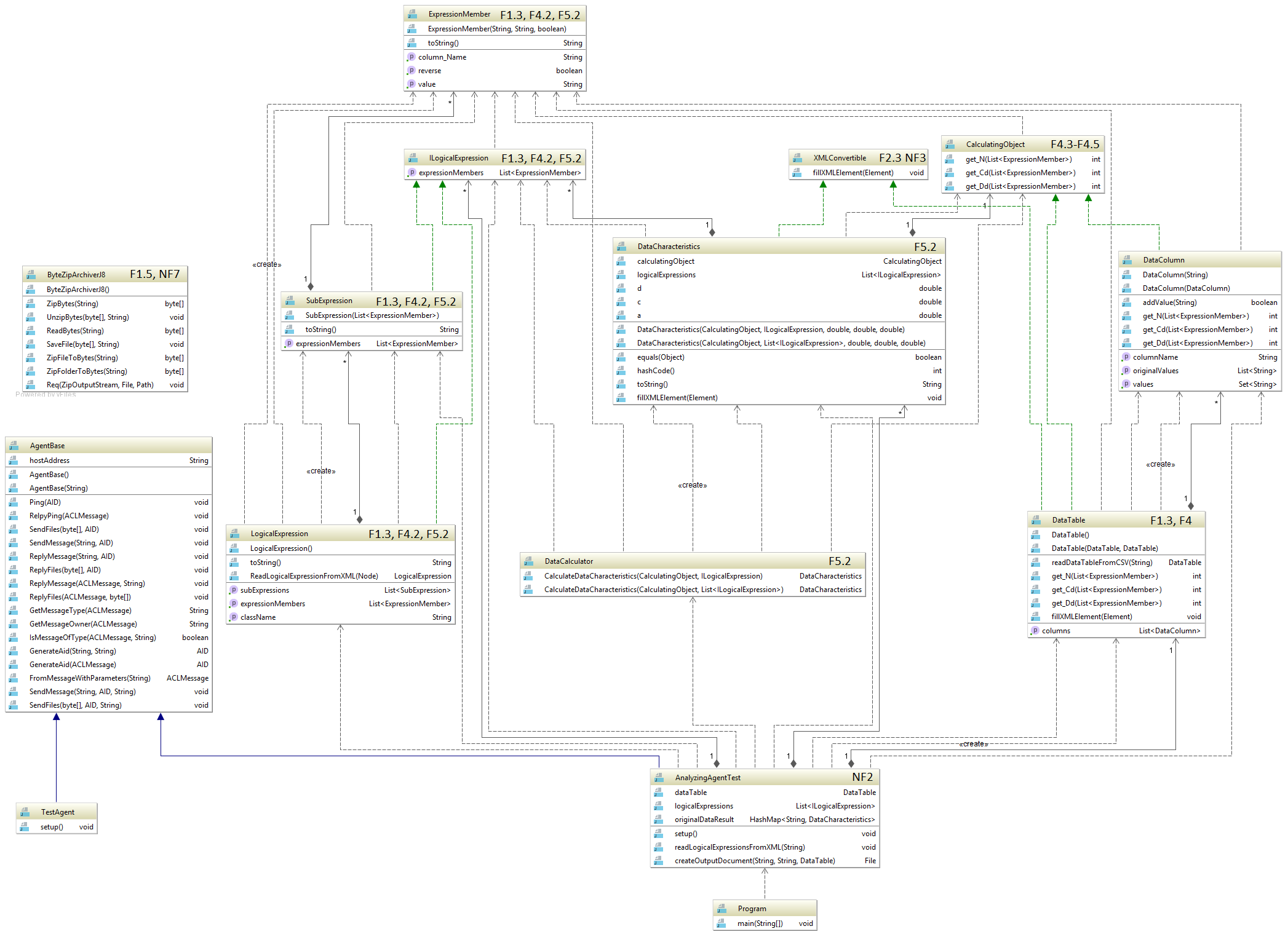


Figure 1 Class diagram

**Рисунок 2 – Діаграма діяльності для визначення поставленого завдання**

# Used programming tools

## language