# Sprint reflection 5

Context Project: Health Informatics
Group: HI4

| User Story | Task | Task Assigned To | Estimated Effort per Task | Actual effort per task | Done (yes/no) | Notes |
|---|---|---|---|---|---|---|
| The user wants to see the output of his program before it is saved and be able to edit it | - Create a screen that shows a preview of the output after analysis | Remi | 2 Hours | 2 Hours | Yes | This was shown at the demo. |
| The user wants to create relations between records based on a condition to link related data elements | - Extend language construct for connecting<br>- Implement a way to create relations<br>- Thinks of a way to output relations in an understandable way<br>- Store the relations in the data structure | Hans<br><br>Hans<br><br>Sven<br><br><br><br>Sven | 2 Hours<br><br>5 Hours<br><br>3 Hours<br><br><br><br>2 Hours | 2 Hours<br><br>5 Hours<br><br>3 Hours<br><br><br><br>3 Hours | Yes<br><br>Yes<br><br>Yes<br><br><br><br>Yes | |
| The user wants to compare time | - Extend language construct for comparing | Elvan | 2 Hours | 2 Hours | Yes | Comparison on numerical |

| | | | | | | |
|---|---|---|---|---|---|---|
| between events in the data for sequential analysis | - Implement (script) code for comparisons on the data<br><br>- Store the (time) differences in the data structure | Elvan<br><br><br>Elvan | 5 Hours<br><br><br>2 Hours | 5 Hours<br><br><br>2 Hours | Yes<br><br><br>Yes | values is also added. The parser automatically detects which comparison (time or value) to perform. |
| The user wants to add comments to (unexpected) results in the GUI as keywords or as memos of his own reflection | - Create a parser for the comment construct<br><br>- Add comment option in the GUI with every record that is shown | Hans<br><br><br>Matthijs | 3 Hours<br><br><br>2 Hours | 3 Hours<br><br><br>- | Yes<br><br><br>No | <br><br><br>GUI was updated too late to add this option. Moved to next sprint. |
| The user wants to select an XML file which fills in the file groups for him | - Add an option to load the XML file in the GUI<br>- Parse an XML file into groups<br>- Add the groups visually to the program | Remi<br><br>Matthijs<br><br>Remi | 1 Hour<br><br>2 Hours<br><br>3 Hours | 1 Hour<br><br>2 Hours<br><br>3 Hours | Yes<br><br>Yes<br><br>Yes | |
| The user wants to specify which columns need to be analyzed | - Add checkboxes to columns in the GUI to indicate if they need to be included for the analysis | Remi | 3 Hours | 3 Hours | Yes | |

| | | | | | |
|---|---|---|---|---|---|
| | - Modify the parser to not store the data of the excluded columns | Hans | 2 Hours | 2 Hours | Yes | |
| The user wants to save the created groups to an XML file | - Add an option to store the current configuration<br><br>- Write the current configuration to a file | Matthijs<br><br>Matthijs | 3 hours<br><br>3 hours | 3 Hours<br><br>3 Hours | Yes<br><br>Yes | |
| The user wants the code to be clean, documented and well tested so it can easily be extended for future use. | - Refactoring: Make sure your own code is fully tested (test coverage should be at least 75%)<br>- Refactoring: Make sure your own code is well documented<br>- Refactoring: Check your code with Cobertura, PMD, FindBugs and CheckStyle.<br>- Test the GUI on JavaFX Services with JUnit tests following the dependency injection pattern | Everyone<br><br><br>Everyone<br><br><br>Everyone<br><br><br><br>Sven | 2 Hours<br><br><br>1 Hour<br><br><br>1 Hour<br><br><br><br>5 Hours | 2 Hours<br><br><br>1 Hour<br><br><br>1 Hour<br><br><br><br>6 Hours | Yes<br><br><br>Yes<br><br><br>Yes<br><br><br><br>Yes | All Checkstyle issues were solved, JavaDoc was updated and we achieved a test coverage above 75%.<br><br><br><br>The coverage of the GUI was increased up to 21%. |
| The client wants an up to date emergent architecture document | - Update the architecture document | Elvan | 2 Hours | 2 Hours | Yes | A part on XML configuration and design patterns in the |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | SelectControll er was added. |
| The user wants the structure of the program to be well thought out so it can easily be extended for future use | - Create an UML or an other kind of diagram describing the structure of the program | Matthijs | 1 Hour | 1 hour | Yes | This diagram is added to the EAD. |

## Main Problems Encountered

**Problem 1**

**Description: Merge conflicts do cost a lot of time**

**Reaction:** Because of the midterm exams and resits this week, most of the programming work was done in the weekend and at the end of the sprint, leaving some branches open for a couple of days. This caused a lot of merge conflicts. Some conflicts are especially difficult to fix, because sometimes git thinks that the whole file has changed, even though the file has been adjusted partially. Often this is caused by differentiation of line-endings. Git is not able to recognize this. We have gotten a fix for this from our TA Tim, so this kind of conflicts can be prevented and fixed more quickly in future sprints.

# Adjustments for the next Sprint Plan

This week went really well. We were finally able to find out how to to test our GUI. Even though we were shorter on time because of the midterms, we worked hard to implement more C's. Test coverage has been improved greatly. And also all reasonable checkstyle issues have been fixed.
(Motivate any adjustments that will be made for the next Sprint Plan.)

- Frequent GUI testing
  Now we are finally able to test the GUI, we are planning to test the GUI more intensively and frequently from now on. This will improve our test coverage and the overall quality of the program.
- Reserve time to improve user friendliness
  Now that we have implemented a major part of the program's functionality, we want to reserve more time to process the feedback of Wenxin Wang and others to improve our program's usability. Up till now we have actually done this every week but from now on we want to spend more time on this. We also want to extend the error management of the program to improve the user guidance of the system.