ACM Framework use case document

**UC #1 Framework Configuration**

|  |  |
| --- | --- |
| **Description:** Using the Configuration component, the user initiates the configuration of the framework. The outcome is a framework ready to pull artefact data from a repository of the user’s choice.  **Preconditions:** System requirements | |
| **User action** | **System response** |
| [1] User stores the original artefacts in a version control system. |  |
| [2] User specifies choice of database backend, repository local and remote paths, framework folder, and artefact types via calling appropriate methods. | Sets up database path, repository local and remote paths, framework path, and XSLT files used for transformation. |

**UC #2 Extract Artefacts**

|  |  |
| --- | --- |
| **Description:** The framework extracts XML-based representations of artefacts.  **Preconditions:** Original artefacts support an XML-based representation, which can be obtained either programmatically or by exporting from the tool. | |
| **User action** | **System response** |
| [1] User specifies the repository folder where artefact data resides. | Takes all the files in the specified folder and calls scripts to extract .xml files. The .xml files are placed in the framework subfolders. |
| Alternative flow [1.2] Artefact data is authored in a tool which needs to be used to export an xml-based representation and for which no automatic method exists. In this case the user manually supplies the .xml files. |  |

**UC #3 Transform Artefact Data to GraphML Representation**

|  |  |
| --- | --- |
| **Description:** The XML representations are transformed to a custom GraphML representation using XSLT transformations.  **Preconditions:** User supplies XSLT files that map a custom XML file to the specified GraphML representation. | |
| **User action** | **System response** |
| [1] | Transforms XML files to GraphML files, which are saved in the same framework folder. |

**UC #4 Trace Link Setup**

|  |  |
| --- | --- |
| **Description:** Using the Traceability component, traceability links are established.  **Preconditions:** Specific to implementation: currently, user-specified training data and test data | |
| **User action** | **System response** |
| [1] User sets up training data semi-automatically. | Generates data by accessing the GraphML store through the DAL. A CSV files is outputted to be used with machine learning algorithm. |
| [2] Runs ML algorithm and collects results |  |
| [3] Reviews results | Based on user review, produces a Relations.xml file |

**UC #5 Save Artefact and Trace Link Data to Graph Database**

|  |  |
| --- | --- |
| **Description:** Using DAL components, artefact and traceability data is saved to the graph database backend.  **Preconditions:** | |
| **User action** | **System response** |
| [1] Initiates data saving. | Saves specified GraphML and XML data to database. |

**UC #6 Perform Change Detection**

|  |  |
| --- | --- |
| **Description:** The user initiates the change detection process using the Interaction Manager component.  **Preconditions:** | |
| **User action** | **System response** |
| [1] Requests changes from repository. | Using the RepoAccessor component returns file level changes – added, edited, deleted files. Using the Change Detection component performs change identification and using the Transformation component XML and GraphML representations are produced. Change data is returned to the user. |

**UC #7 Perform Change Impact Analysis**

|  |  |
| --- | --- |
| **Description:** potentially impacted nodes and edges are identified in the graph database.  **Preconditions:** | |
| **User action** | **System response** |
| [1] User invokes impact analysis. | Based on the change data, it returns a set of potentially impacted nodes and edges using the Impact Analysis component. |

**UC #8 Perform Consistency Checking**

|  |  |
| --- | --- |
| **Description:** based on the impact set and pre-defined rules, consistency violations are identified.  **Preconditions:** | |
| **User action** | **System response** |
| [1] User invokes consistency checking. | Based on the change object (containing the change data) and impact set, it parses the rule base and returns consistency check results using the Consistency Check component. |

**UC #9 Propagate changes to data store and perform trace link maintenance**

|  |  |
| --- | --- |
| **Detailed description:** Following a change, the data store (graph database) is updated by using the DAL. | |
| **User action** | **System response** |
| [1] User invokes change propagation. | Based on the change object (containing the change data) it updates the database by editing and adding new nodes. Deleted entities are marked and subsequently deleted.  It also updates inter trace links based on pre-defined rules. |

**UC #10 Suggest Inconsistency Resolution**

|  |  |
| --- | --- |
| **Detailed description:** based the results of the consistency check, the user is presented with possible ways of resolving the inconsistencies by the Change Propagation component.  **Preconditions:** | |
| **User action** | **System response** |
| [1] User invokes change propagation. | Based on the consistency check results, it displays suggestions on inconsistency resolution to the user. |