*R&D Project*

**ADEQUATe** - Analytics & Data Enrichment to improve the QUAliTy of Open Data

Project Number: 849982 Start Date of Project: 01/10/2015 Duration: 36 months

# Deliverable 1.1

# Catalogue of quality dimensions and concrete metrics to assess (meta-)data quality.

| | |
|---|---|
| Dissemination Level | Public |
| Due Date of Deliverable | March 2016 |
| Actual Submission Date | 31/03/2016 |
| Work Package | WP 1, Requirements & Specification |
| Task | T1.2 Requirements Specification for data quality improvement methods and technologies |
| Type | Report |
| Approval Status | |
| Version | 1.0 |
| Number of Pages | 24 |
| Filename | D1.1 Implementation Requirements.docx |

**Abstract:** This document provides a refined selection of quality dimensions and associated metrics based on Deliverable 1.3 and 1.2, as well as an overview of possible building blocks for the ADEQUATe architecture.



FFG

Project funded by FFG - IKT der Zukunft Programme

## History

| Version | Date | Reason | Revised by |
|---|---|---|---|
| 0.5 | 27/03/2016 | | JH, TL (DUK) |
| 1.0 | 31/03/2016 | Final Version | DUK, SWC |

## Author List


| Organisation | Name | Contact Information |
|---|---|---|
| JH (DUK) | Johann Höchtl | johann.hoechtl@donau-uni.ac.at |
| TL (DUK) | Thomas J. Lampoltshammer | Thomas.lampoltshammer@donau-uni.ac.at |
| SWC | Tomas Knap | t.knap@semantic-web.at |

# Executive Summary

This document summarizes the set of metrics and algorithms to be implemented on the ADEQUATe platform. It is the result of the State-of-the-Art elicitation in deliverable 1.3, the crosscutting with user requirements of D1.2, and the goals of the project according to the Description of Work (DoW).

# Table of Contents

# List of Figures



# List of Tables

# 1 Introduction

This document summarizes the set of metrics and algorithms to be implemented on the ADEQUATe platform. It is the result of the State-of-the-Art elicitation in deliverable 1.3, crosscutting it with user requirements collected in D1.2, and the goals of the project according to the Description of Work (DoW).

The metric requirements are:

- Objectivity
- Support for monitoring and integration
- Support for correctness and completeness
- Automatable or suitable to automatically improve data quality (DQ)
- Support for the linking context

In the second section of this document, we describe the architectural building blocks of the ADEQUATe portal. The selection of software components to be used will be guided by the requirement that the platform must be capable to assess the selected quality metrics and display them to the user, and should aim to satisfy the user requirements identified in D1.2 as best as possible.

# 2 Proposed Metrics

The following list contains the metrics to be implemented on the ADEQUATe portal. This list was assembled based on the assessment of available DQ metrics, delivered by D1.3 (State-of-the-Art Analysis of Open Data Quality Assessment Metrics and Algorithms), as well as the user surveys and focus groups results, delivered by D1.2 (Requirements Specification). According to the Description of Work, the project's focus is on monitoring, integration, correctness, and completeness. These goals serve as an additional filter for the DQ metrics.

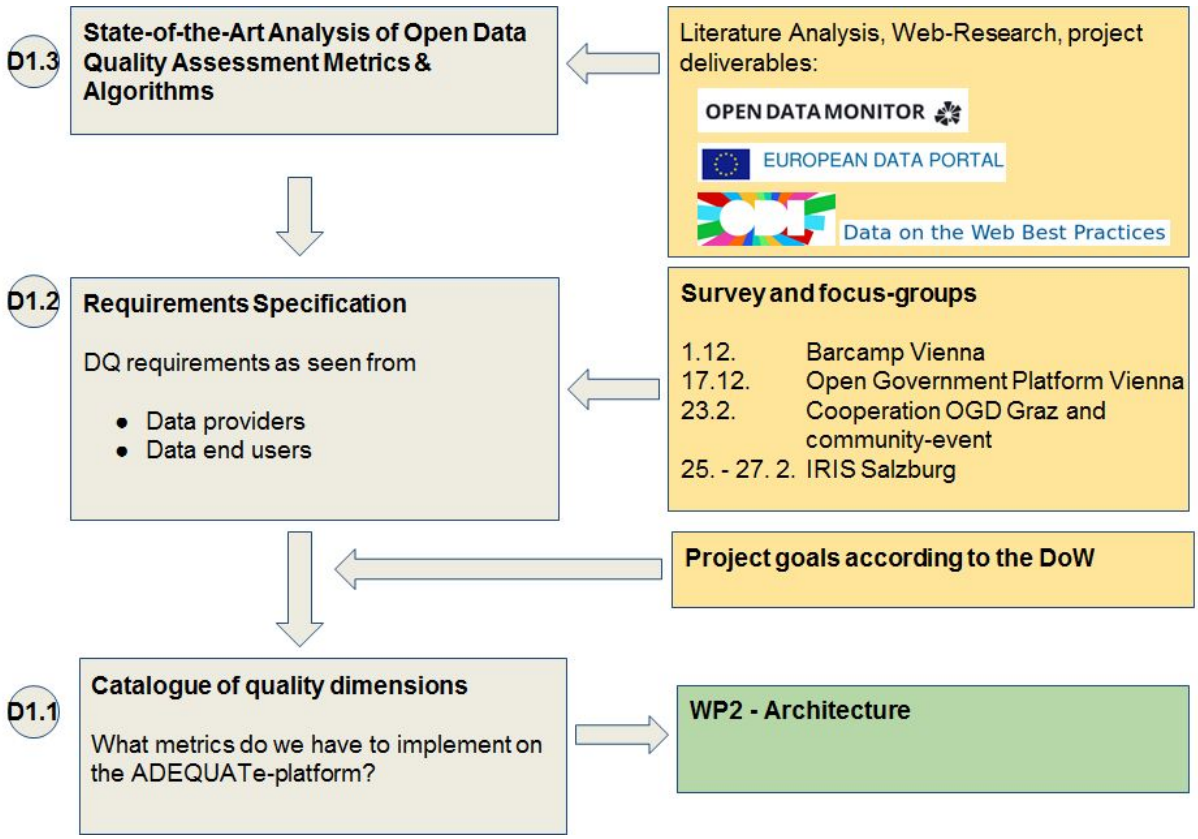


Figure 1: DQ metrics consolidation process

The following metrics therefore present the final results after considering the derived user requirements and project goal filters. This report structures their insights according to subjective (SC), objective (OC), and process criteria (PC) and highlights those which are suitable in a linking context (LC) and whose assessment can be automated (A).

**Table 1: Metrics to be implemented by the ADEQUATe-platform**

| Dimension | Metrics Definition | SC | OC | PC | LC | A | Reference to D1.3 | Reference to D1.2 |
|---|---|---|---|---|---|---|---|---|
| Documentation / Usage: The extent to which the data meaning is explained | Richness of Information expressed as the frequency-inverse document frequency of the description | | O | (P) | | A | 3.1.1 | Req. 10 |
| Clear Definition: How clear, readable, or understandable the data is. | Percentage of available associated meta data | | O | P | | A | 3.1.10, 3.1.4, 3.1.16 | Req. 10 |
| Feedback / Contactability: Possibility to give feedback on data; The extent to which the data publisher provides contact information | Contact details are provided; mail address exists | | O | P | | A | 3.1.10 | Req. 1 |
| Availability | Percentage of time an information source is “up” | | O | P | | A | 3.1.9 | |
| Versatility | Provision of the (meta) data in various languages: checking whether data is available in different languages | | O | P | | (A) | 3.1.22 | |
| Openness: The extent to which *licenses* conform to the open definition. | | | O | P | | A | 3.2.1 | Req. 14 |
| Completeness: Refers to the scope of the information in the data | Completeness = Number of not null values/total number of values | | O | | | A | 3.1.4 | Req. 15 |
| Accuracy: Data is correct (error-free) and reliable | Detection of outliers: by using distance-based, deviations-based and distribution- based method | | O | | | A | 3.1.3 | Req. 15 |
| (Structural) Consistency: Data continuously presented in the same format | Consistency = Number of consistent values/number of total values | | O | | | A | 3.1.5 | Req. 7 |
| (Structural) Consistency: Data continuously | Column format standardization | | O | | | A | 3.1.15 | Req. 7 |

| | | | | | | |
|---|---|---|---|---|---|---|
| presented in the same format | | | | | | |
| (Structural) Consistency: Data continuously presented in the same format | Number of tuples violating constraints, number of coding differences (Structural consistency) | | O | A | 3.1.5, 3.1.8 | Req. 7 |
| (Structural) Consistency: Data continuously presented in the same format | Encoding: detectability of encoding from the data set: true or false | | O | A | | Req. 7 |
| (Structural) Consistency: Data continuously presented in the same format | Encoding: availability of encoding information in the meta data: true or false | | O | A | | Req. 7 |
| (Structural) Consistency: Data continuously presented in the same format | Encoding: correctness of encoding information in the meta data: true when it coincides with the encoding automatically detected in the data set, otherwise false | | O | A | | Req. 7 |
| Accuracy: Data is correct (error-free) and reliable | Erroneous annotation, erroneous representation: 1 – (erroneous instances) / (total no. of instances) | | O | A | 3.1.18 | Req. 15 |
| Accuracy: Data is correct (error-free) and reliable | The extent to which certain meta data values accurately describe the resources. | (S) | O | A | 3.1.18 | Req. 6 |
| Latency, Throughput | Lag (e.g. ms) in terms of data streaming<br><br>Low latency: delay between submission of a request by the user and reception of the response from the system<br><br>High throughput: no. of answered HTTP-requests per second<br><br>Scalability of a data source: detection of whether the time to answer an amount of ten requests divided by ten | | | | 3.1.11 | |

| | | | | | |
|---|---|---|---|---|---|
| | is not longer than the time it takes to answer one request | | | | |
| Historical comparability | Existence of time stamps within the source data (e.g., dedicated column for time stamps in case of CSV files) | O | (A) | 3.1.7 | Req. 27 |
| Retrievability | Whether or not the description of a dataset and the content of a resource can be retrieved based on an HTTP GET operation. | O | A | 3.1.12 | |
| Understandability / Clarity / Clear Definition: How clear, readable, or understandable the data is. | Cognitive accessibility as the ease a user can comprehend what the resource is about after reading the metadata record | O | A | 3.1.13 | |
| Interpretability: The extent to which data is in appropriate languages, symbols, and units, and the definitions are clear | Use of self-descriptive formats: identifying objects and terms used to define these objects with globally unique identifiers | O | (A) | 3.1.14 | |
| Openness: The extent to which *file formats* conform to the open definition. | | O | A | 3.2.2 | |
| Timeliness / Currentness: The age of the data | Timeliness of the resource: a positive difference between current and expiry time of the resource implies data source to be out-dated.<br>Will be calculated according to *The Tau of Data* | O | A | 3.2.3 | Req. 9 |
| Accuracy | Checking syntax of the values using regular expression - e.g., postal codes, nuts codes, values within defined range, phone, email, dates. Note: This requires to identify the types of data for which it makes sense to define regular expressions | | | | Req. 15 |
| Linking | interlinking completeness : no. of types of resources (e.g., column “cities”) that | | | Linking Metrics B25 | Req. 24 |

| | | | |
|---|---|---|---|
| | are linked to external types (classified) / total no. of named entity columns in the dataset | | |
| | interlinking completeness: no. of resources in the dataset of certain type (e.g. Wien, Linz as instances of cities) that are interlinked (disambiguated) / total no. of resources in a dataset for the given type | Linking Metrics B25 | Req. 24 |
| | Interlinking completeness: average number of links to external datasets for interlinked resources (e.g. Linz, Wien) for interlinked types of resources (e.g. cities vs. administrative districts) | Linking Metrics B25 | Req. 24 |
| Accuracy - confirmation by external LD source | Checking accuracy of the data by looking into evidence for such data in external data sources (NOTE: requires conversion of at least some data to RDF data format - not just identifiers, but also relations) | Linking Metrics B47 | Req. 15 |

# 3 Platform User Requirements

The following list of platform user requirements has been derived from the focus groups and survey results which have been summarized in deliverable D1.2 and represents the subset of user wishes which are in line with the project's objectives. While the list of quality metrics of the previous section describe the quality dimensions the users would like to see on the platform, the following requirements describes a subset of the intended interaction with the ADEQUATe platform.

The meaning of column “priority” is defined by the MoSCoW-method[1].

| Requ# | Description | Motivation | Type | Priority |
|---|---|---|---|---|
| Requ. 2 | Users want to discuss and collaborate on particular data sets. | | Platform | MUST |
| Requ. 3 | Users want to be able to fork particular data sets in order to work independently on certain ideas/issues. | Users want to be able to merge other branches. | Platform | MUST |
| Requ. 5 | Users want to see changes in data sets to identify potential quality improvements. | User should also see quality score as was before and now | Platform | COULD |
| Requ. 12 | Users want to see the quality improvement over time of a particular data set. | | Platform | SHOULD |
| Requ. 13 | Users want to see the current status of reported issues regarding a particular data set. | | Platform | COULD |
| Requ. 14 | Users want clarity about legal aspects regarding available data sets. | So that users know under which conditions the dataset may be used -> provide short | Platform | SHOULD |

[1] https://en.wikipedia.org/wiki/MoSCoW_method

| | | | | |
|---|---|---|---|---|
| | | desc of the license consequences | | |
| Requ. 16 | Users want to have a ranking for data quality for each data set (summary score and for each dimension) | So that they can better decide about the quality of the dataset. No pre-defined overall ranking but user-definable set of metrics | Platform | SHOULD |
| Requ. 17 | Users want to be able to rank data sets based on certain criteria - dimensions of data quality | | Platform | SHOULD |
| Requ. 18 | Users want to see the overall reputation of data providers on the portal | | Platform | COULD |
| Requ. 21 | Provide powerful search functionality, including semantic information | ● concepts for columns in files<br>● create and modify filters eg., for single keywords, time, format<br>● well-indexed data sets to improve search results<br>● Include user provided concepts behind data sets<br>● search taking into account hierarchy of types of concepts | Platform | SHOULD |
| Requ. 25 | Users want transparent and clear processes regarding | So that consumers can examine provenance records of the | Platform | SHOULD |

| | | | | |
|---|---|---|---|---|
| | (pre-) processing when uploading data sets. | published data to understand how the data was adjusted during pre-processing phase | | |
| Requ. 27 | Users want to see the development (changes) over time of a particular data set | | Platform | COULD |
| Requ. 28 | Users want to see the dialog regarding issues of a particular data set | | Platform | COULD |
| Requ. 29 | Users want to be informed when changes to data sets happen, in which they are interested | | Platform | SHOULD |

## 3.1 DoW enriched platform requirements

The following list of enriched platform requirements have not been specifically mentioned by end users but are according to the project Description of Work and thus constitute a project requirement:

(Description of fields in Component / Comment-Column: LC=”Odalic” = TablemMiner++ + UI; PP=Poolparty; MD=Metadta for CSV Files)

| **Requ. #** | **Description** | **Motivation** | **Type** | **Prio** | **Component / Comment** |
|---|---|---|---|---|---|
| Requ. 30 | Users want to see overall reports on the OD portal quality | System monitors quality of Austrian data portals and creates reports | Platform | MUST | “PortalWatch” |
| Requ. 31 | Users want to classify/ | User uploads the file to be processed, | Platform | MUST | LC |

| | | | | | |
|---|---|---|---|---|---|
| | disambiguate input CSV files | optionally with metadata | | | |
| Requ. 32 | User want to discover relations in the input CSV files | | Platform | SHOULD | LC |
| Requ. 33 | Users want to provide feedback on the result of classification/ disambiguation | | Platform | MUST | LC, clarify “feedback” (approve, reject, suggest new), where/how to store user feedback |
| Requ. 34 | Users want to provide feedback to the result of relation discovery | | Platform | SHOULD | LC |
| Requ. 35 | Users want to export tabular data as Linked Data. Consider supporting specific format for statistical data | Manual effort is expected to tune the resulting format. | Platform | COULD | LC |
| Requ. 37 | Users want concepts (such as cities) to be available as Linked Data | LOD pilot will be used as inspiration | Platform | MUST | PP |
| Requ. 38 | Users want the converted Linked Data to be available via | At least class concepts | Platform | MUST | Virtuoso |

| | | | | | |
|---|---|---|---|---|---|
| | SPARQL endpoint | | | | |
| Requ. 39 | Users want to configure the knowledge bases (specify which KB to use) classification/ disambiguation/ relation discovery process is run against | | Platform | COULD | LC |
| Requ. 42 | Knowledge base maintainers want to create new concepts/ edit existing/ link concepts in the Adequate KB. | | Platform | MUST | PP. Defining relationships between datasets should be modeled on the platform level/UI, whereas PP will be used to model relationships between linked data/concepts |
| Requ. 43 | Users want to select concepts from Knowledge base and associate them with CSV columns | | Platform | MUST | PP (not per user profile), |
| Requ. 44 | Users want to propose new concepts to be | | Platform | COULD | PP |

| | | | | | |
|---|---|---|---|---|---|
| | associated with CSV columns | | | | |
| Requ. 45 | Knowledge base maintainers approve proposed concepts | | Platform | COULD | PP |
| Requ. 47 | Users want to create/ edit metadata records, where they can specify Knowledge Base concepts linked to certain columns | | Platform | SHOULD | PP/MD |

# 4 Architectural Building Blocks

The following list of tools, services and existing software components would be suitable to fulfil the described platform requirements. They have been assembled after brainstorming and serve as input to WP2:

## 4.1 Badge-Service

Badges are a means to quickly convey characteristic features. In software development for example, they signal build status, code coverage, or the availability of the author to answer questions. In the open data context they provide a quick way to signal data license openness, adherence to standard file formats, whether the data contains Unique Identifiers, etc. Their visual appearance is standardized[2] and their usage straightforward. They can be easily integrated into existing websites to display status information on the dataset. The following schematic overview describes the working:

[2] https://github.com/badges/shields

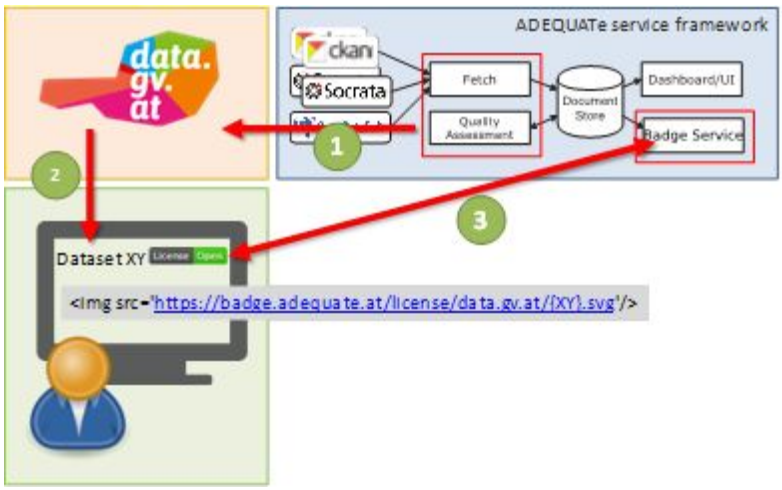


Figure 2: Anatomy of a badge service

In the first step ① the ADEQUATe quality monitoring framework harvests a data portal and performs various checks, and stores the results in its database (Document Store). A user fetches a particular data set ②, which contains an image link to the adequate badge service. The browser will try to fetch the image ③ and, depending on the request Url (here: /license/data.gv.at/XY.svg) the badge service accesses the quality assessment result database to return the appropriate badge for dataset XY of portal data.gv.at in the license domain.

**Relevance to the project:**

The application of Badges is possible in various data quality assessment domains, but they should be used sparingly and only in those cases where they can convey immediate value to an end user. User requirement 16 states: “Users want to have a ranking for data quality for each data set”, as well as requirement 17: “Users want to be able to rank data sets based on certain criteria”, underline user’s wish for clearly visible and ranked quality metrics. Based on the set of metrics identified above, possible application domains are:

Table 2: Examples of possible ADEQUATe-badges

| Application domain | Badge(s) |
|---|---|
| Openness of License | License ★<br>License ★★<br>License ★★★<br>License ★★★★<br>License ★★★★★ |
| Interpretability (availability of unique ID) | UID ok<br>UID fail |

| | |
|---|---|
| Openness of file Format | Format closed<br>Format problematic<br>Format open |
| Timeliness | Currency Up-to-date<br>Currency Outdated<br>(with possible steps in-between, according to Atz 2014, 261) |

## 4.2 Portalwatch

PortalWatch implements different modules to access and retrieve data. It can harvest data from CKAN, Socrata, and OpenDataSoft. Basic portal statistics become possible by accessing the portal's API to obtain meta data and by fetching the resources via http HEAD requests. A set of components calculates basic statistical information about the occurrence and distribution of various metrics, such as the number of datasets, resources, response code distribution, frequency count for licenses, formats, organisations, etc. A module gets previously computed data from the database to calculate metrics and for information retrieval to render the results as a web site.

**Relevance to the project:**

Portalwatch is a data quality monitoring tool that assesses a number of the data quality metrics outlined in this document and therefore relevant to the ADEQUATe project, at the moment focussing on the assessment of process / portal metrics. In the course of this project, Portalwatch will be developed further to support the assessment of quality metrics that go beyond meta data assessment and fetch the data itself, in order to be able to assess quality metrics within the data.

## 4.3 UnifiedViews

The framework UnifiedViews[3] originates from the LOD 2 FP7[4] project back in 2012. It is a linked data management suite, which covers all relevant steps regarding ETL (extract-transform-load) processes. These processes can be split into several smaller steps, represented by DPUs (Data Processing Units). Figure 3 shows the DPU workbench within Unified Views.

[3] http://www.unifiedviews.eu
[4] http://lod2.eu/Welcome.html

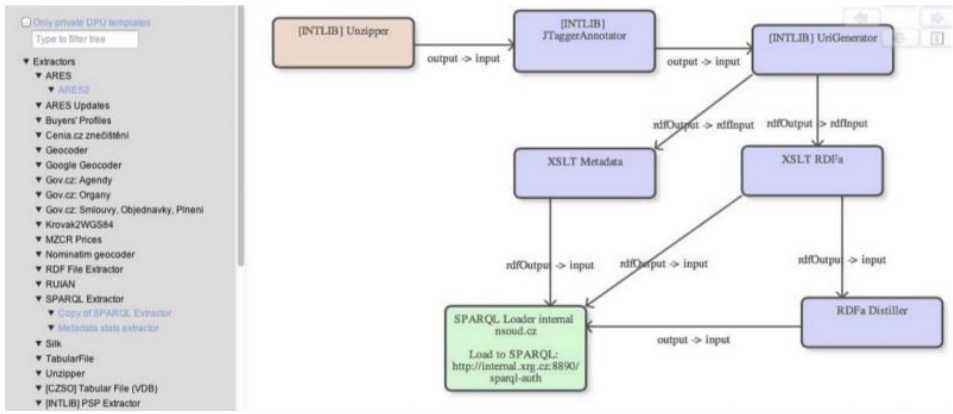


**Figure 3: Unified Views DPU Workbench**

These DPUs are then connected to model the intended data flow between the DPUs and within the ETL processes. Each individual DPU contains programmable modules that handle a specific task such as connecting to and loading data from a database, transforming data from one format into another format, or cleaning processes for “polluted” or erroneous data. The suite also offers templates for developing custom DPUs.

**Relevance to the project**:

The ADEQUATe project can benefit from the application of UnifiedViews in several ways. As the suite is based on the programming language Java, integration into existing environments is convenient as integration via RESTful interfaces is compatible with a plethora of other existing technologies. Furthermore, the before-mentioned plugins enable the project team to develop their own required tools regarding data cleaning, transformations or data set fixing. In addition, DPUs offer the possibility to implement the selected open data metrics, and therefore automate their calculations. By combining several DPUs, powerful ADEQUATe data correction and enrichment processing chains can be realized.

## 4.4 TableMiner+

In the recent days, governmental organizations publish their data as open data (most typically as tabular data - CSV files). To fully exploit the potential of such data, the publication process should be improved, so that data are published as Linked Open Data. By converting open data to Linked Data, we increase usefulness of the data by providing global identifiers for things and we enrich the data with links to external sources.

Leveraging tabular open data (further in the text we consider them to be CSV files) to Linked Data involves the steps as follows: 1) classify CSV columns based on their content and context against existing knowledge bases and assign globally unique HTTP URL identifiers for the columns according to Linked Data principles, 2) assign globally unique HTTP URL identifiers to the particular cell values according to Linked Data principles, 3) discover relations between columns based on the evidence for the relations in the existing knowledge bases, and 4) convert CSV data to RDF data, properly using data types, language tags, well-known Linked Data vocabularies, etc.

TableMiner+[5] is an algorithm for such (semi)automatic leveraging of tabular data to Linked Data, which realizes the steps 1) - 4) described above.

**Relevance to the project**:

TableMiner+ may be used for (semi)automatic leveraging of tabular data to Linked Data as described in WP4. So the basic workflow of publishing tabular data as Linked Data is as follows:

1. User runs the TableMiner+ algorithm on top of input tabular data
    a. As a result, user is presented with the results of the algorithm, suggesting how the columns of the input tabular data should be classified, the cell values disambiguated, and also shows relations between columns if any were discovered
2. User may provide feedback to the results of the algorithm, e.g., use may re-classify certain column, or mark certain disambiguation as being wrong
3. User may re-run the algorithm, which takes into account the feedback provided by the user (WP5)
4. User may iterate steps 2-3 until she is satisfied with the results of the algorithm
5. User may configure how the data is being published to RDF data format
    a. User may adjust the template describing how the data is being produced
    b. User may get preview of RDF data being produced
6. User may publish the data as RDF data
7. User may save the configuration of the transformation, so that it may be applied later/to similar input data

## 4.5 Poolparty

PoolParty[6] is a tool for organizing knowledge. It provides several components, such as the PoolParty Thesaurus manager for management of master data, the PoolParty extractor for annotating unstructured data and extracting terms and concepts and the PoolParty search for searching among annotated documents. PoolParty also provides a Linked Data frontend to serve master data as Linked Data

**Relevance to the project**:

PoolParty may be used as a tool for management of master data - concepts (tabular data columns, such as cities, districts etc.) and entities (such as particular city, district) - being discovered among the resources processed by Adequate platform. PoolParty may cooperate with TableMiner+ in a way that results in classification, entity disambiguation, and relation discovery may be stored in PoolParty Thesaurus Manager. PoolParty may be also used as Linked Data frontend to serve master data as Linked Data.

[5] https://drive.google.com/drive/u/1/folders/0By9GI12mR4fJZUE1OVByN2ZlYVE?ths=true

[6]

In this way, PoolParty would support the fulfilment of user requirements that target interlinked concepts and data sets and improved search (requirements 21, 32-47).

## 4.6 OpenRefine

OpenRefine[7] (formerly known as Google Refine) aims at providing a complete suite for data cleaning (see Figure 4). It is a standalone tool that focuses mainly on spreadsheet-based input. It works on a row/column-based point of view, represented via a single big table. All operations that are executed inside of OpenRefine are stored in a database, which enables not only an overview of performed actions but also a repetition of an entire action if required. The overall set of changes can be applied as „package“ on any other dataset similarly structured. Instead of formulas, transformation rules are used. These rules can be formulated in the General Refine Expression Language (GREL)[8], Jython[9], and Clojure[10].

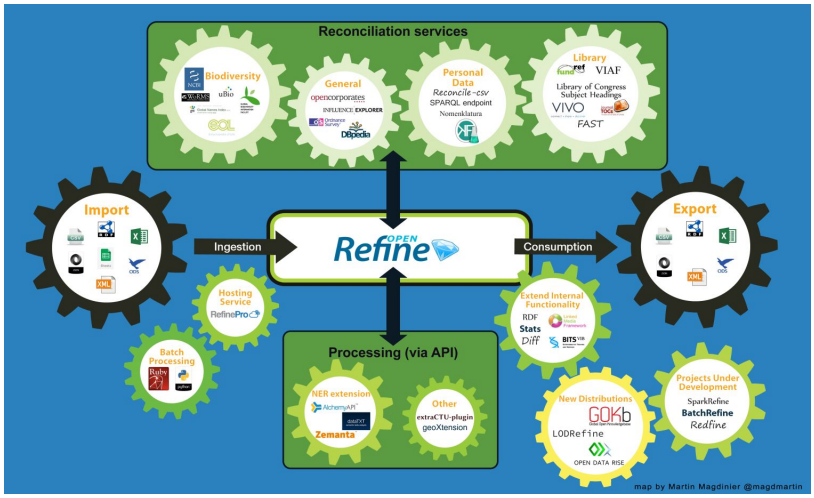


Figure 4: OpenRefine Ecosystem[11]

**Relevance to the project**:

The ADEQUATe platform can potentially profit from OpenRefine similar to the capabilities as provided by the UnifiedViews suite. Users can work on a certain problem in a particular data set. After resolving the given issue, the applied set of changes can be provided for other users that have data sets impacted by the same problem. This solution makes it only necessary to store the change sets as these can be applied to the original data set at any time, also stacked on top of each other. This reduces the required storage space and data traffic significantly, while on the other hand, increases the requirements in terms of computational resources.

[7] http://openrefine.org
[8] https://github.com/OpenRefine/OpenRefine/wiki/General-Refine-Expression-Language
[9] http://www.jython.org
[10] https://clojure.org
[11] http://openrefine.org/2015/01/26/Mapping-OpenRefine-ecosystem.html

Giving users of the ADEQUATe platform the opportunity to use OpenRefine to work on the data sets would fulfil user requirements 2 (platform for communication and collaboration on data sets), 5 (change tracking), 12 (tracking quality improvement over time) and 13 (visibility of status of current issues).

## 4.7 Jupyter

The Jupyter Notebook[12] is a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, machine learning and much more. Jupyter is open source, multi-user capable and integrates well with existing infrastructure, for example, Jupyter notebooks as the file containing data definitions are called, can be rendered on Github[13]. Besides that, Jupyter notebooks are (programming) language agnostic and language kernels support R, Javascript, Python, among many others[14].

**Relevance to the project**:

The ADEQUATe platform could embed the Jupyter notebook as a means to perform data manipulations, corrections, enhancements and visualisations. The notebooks can be stored alongside the data as a means to describe data transformations much in the same spirit as OpenRefine transformations can be considered as a step-model to transform data from one representation to another. An even more in-depth integration would become possible by integrating JupyterHub[15] into the ADEQUATe UI and by being able to access data stored on the platform.

Similar in use to OpenRefine, Jupyter could also contribute to fulfilling user requirements 2 (platform for communication and collaboration on data sets), 5 (change tracking), 12 (tracking quality improvement over time) and 13 (visibility of status of current issues).

## 4.8 Git+GUI (Gogs)

One important building block in a sustainable community platform is the possibility to cooperatively work on data sets. To achieve this, simultaneous actions on one and the same data set are required. In order to be able to manage all originating versions and to solve potential conflicts, a version control system is necessary. Git[16] represents the major platform for version control in the open source community. It is free, lightweight, fast, and offers all necessary functionalities that commercial tools provide. One of the most important features of Git is its easy branching system (see Figure 5). This system enables users to simultaneously work on different versions of a data set, e.g., to try new formats, add new data, or fix existing issues. When they have completed their work, users can merge their new versions, if desired, into the original dataset, to create an updated version. Possible conflicts regarding modifications in the data set are identified and, if possible, automatically resolved as well. Yet the core functionalities of Git are console-based, which is not suitable

[12] http://jupyter.org/
[13] https://github.com/blog/1995-github-jupyter-notebooks-3
[14] https://github.com/ipython/ipython/wiki/IPython-kernels-for-other-languages
[15] https://github.com/jupyter/jupyterhub
[16] https://git-scm.com

for embedment into an online platform. Therefore, a server variant with an adaptable GUI is necessary. Several free and commercial Git environments exist. The solution suggested for the ADEQUATe project is Gogs[17] (Go Git Service). This Git server environment is based on the fast and lightweight programming language Go[18]. It only requires a minimum of resources in comparison with other solutions such as GitLab[19], while at the same time providing a fully featured Web-based front-end that could be customized and embedded on the ADEQUATe community platform.

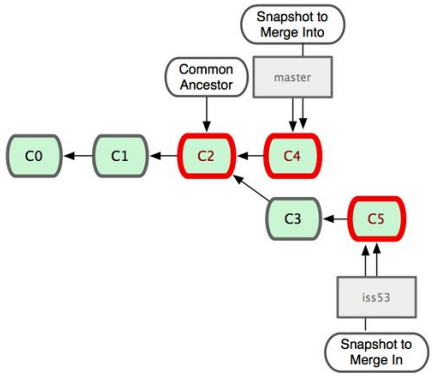


Figure 5: Branching and merging process in Git

**Relevance to the project**:

The ADEQUATe project plans to improve the quality of data sets available on open data portals, and re-publish improved versions of these data sets to make them available to the community. One of the approaches to improving open data quality the ADEQUATe project plans to employ are crowd sourcing approaches. In order to enable the community to work on data sets and publish improved / changed versions, a version management tool such as Git becomes indispensable.

In addition, D1.2 clearly identified that users wish to be able to fork data sets, create their own versions, track changes and versions of data sets made available by other users, and be able to trace the changes done to data sets by other users and/or the ADEQUATe algorithms (user requirements 3, 5, 12, 13, 27 and 28).

[17] https://gogs.io
[18] https://golang.org
[19] https://about.gitlab.com