Semantic Data Web Technologies Lab

Administrative Budget Data Transformer (ABuDaT)

Mentor: Fathoni Musyaffa

Students:
Florian Weile, Tatiana Novikova, Aberham Gebreyohannes, Samuel Y. Ayele

[This document describes the technical system specification for implementing Administrative Budget Data Transformer]

Table of Contents

Generalities

1.1. Overview

Goals of the Lab:

- Implement transformation of government data published in various formats like .xml and .csv to into the openbudgets.eu RDF data model.
- Implement a user interface for non-technical users for the above task. Using the REST service of http://etl.linkedpipes.com/

1.2. Reference

The following are the reference document names:

- Understanding the datasets: https://github.com/openbudgets/datasets
- Linkedpipes documentation: http://etl.linkedpipes.com/documentation/
- Openbudgets.eu deliverables:
  - http://openbudgets.eu/assets/deliverables/D1.2.pdf
  - http://openbudgets.eu/assets/deliverables/D1.3.pdf
  - http://openbudgets.eu/assets/deliverables/D1.4.pdf

1.3. Definitions, Acronyms, Abbreviations

- ABuDaT: Administrative Budget Data Transformer.
- XML: a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable[1].
- CSV: a comma separated values file, which allows data to be saved in a table structured format.[2]
- RDF: Resource Description framework is a family of World Wide Web

[1] https://en.wikipedia.org/wiki/XML
[2] https://en.wikipedia.org/wiki/Comma-separated_values

Consortium (W3C) specifications used as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntax notations and data serialization formats.

- UML - Unified Modeling Language.
- OLAP - Online Analytical Processing.
- Data-Table: Any display of information in a tabular form, with rows and/or columns.
- Code-lists: a specific code that is reusable across the datasets, for example EUR code encodes Euro currency in the currency code list.
- ETL: refers to a process in database usage and especially in data warehousing that performs: Data extraction – extracts data from homogeneous or heterogeneous data sources.[3]
- Dataset: A named collection of data that contains individual data units organized (formatted) in a specific way.

3. Technical Software Requirement

3.1. Software Aspects

The following are the software requirements for ABuDaT:

- Ubuntu OS;
- Git;
- JDK (v8);
- Apache Tomcat;
- Nodejs;
- Nodejs-legacy;

[3] https://en.wikipedia.org/wiki/Extract,_transform,_load

- Npm;
- Maven;
- MySql server;
- Linkedpipes;
- Fuseki
- Selenium IDE.

4. Operational Specification

4.1. Installation

Clone the sources

```
$ git clone https://cowclaw@bitbucket.org/cowclaw/semweblab2016.git
```

Running ABuDaT standalone (Tomcat included)

```
$ cd semweblab2016/ABuDaT
$ ./gradlew bootRun
```

Abudat should be up and running on http://localhost:9000/

Configuration file:

semweblab2016/ABuDaT/src/main/resources/application.properties,

defaults are:

```
linkedpipes.etl.host=localhost
linkedpipes.etl.port=8080

#Path to the abudat output dir.
abudat.output-dir=/tmp/abudat

#The data endpoint of the fuseki installation.
fuseki.data.endpoint=http://localhost:3030/ds/data

server.port=${port:9000}
spring.thymeleaf.cache=false

spring.datasource.url=jdbc:mysql://localhost/abudatdata?autoReconnect=true&useSSL=false
spring.datasource.username=abudat
spring.datasource.password=abudat
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
```

Note: assumes running linkedpipes etl on localhost, see installation instructions below.

Note: Due to limitations and security concerns, linkedpipes etl needs to run on the same host as abudat. Otherwise the functionality for download of data and upload of data to a triple store will not work.

RUNNING IN A DEDICATED TOMCAT

It is possible to run ABuDaT in a dedicated tomcat server. Therefor simply assemble a war file:

```
$ cd semweblab2016/ABuDaT
$ ./gradlew war
```

You can find it the war here: semweblab2016/ABuDaT/build/libs/ABuDaT.war. It can then be copied to tomcats webapp folder. Note that in this case the server.port=${port:9000} setting will have no effect.

4.2. Database Setup

Install mysql-server

```
$ sudo apt-get install mysql-server
```

Create the database

```
$ mysql -u root -p
mysql> create database abudatdata default character set utf8 default collate utf8_bin;
mysql> create user 'abudat'@'localhost' identified by 'abudat';
mysql> grant all on abudatdata.* to 'abudat'@'localhost';
mysql> flush privileges;
mysql> quit;
```

TROUBLESHOOTING

If during container startup, you get exceptions regarding timezone check:

```
$ mysql_tzinfo_to_sql /usr/share/zoneinfo | mysql -u root mysql -p
```

Then add the default time zone to /etc/mysql/mysql.conf.d/mysqld.cnf, e.g.:

```
[mysqld]
...
default-time-zone='Europe/Berlin'
...
```

see: http://dev.mysql.com/doc/refman/5.7/en/time-zone-support.html, see: http://stackoverflow.com/a/32736024/4098376

Install fuseki as a triple store

see: https://jena.apache.org/documentation/fuseki2/#getting-started-with-fuseki

Download apache jena fuseki 2 from https://jena.apache.org/download/.

Unpack the dowloaded archive. On linux, cd into the unpacked directory.

Then run:

```
$ chmod +x fuseki-server
```

to make the server executable. To start the server run:

```
$ ./fuseki-server --update --mem /tmp/ds/
```

this create an non-persitent in-memory dataset for the server. To create a persistent file-based dataset run:

```
$ ./fuseki-server --update --loc=/tmp/ds /ds
```

The parameter given along with --loc is the path to the backing file.

4.3. Local LinkedPipes ETL installation

Setup process for Ubuntu Linux.

PREREQUISITES:

- have maven installed
- have nodejs-legacy, npm and nodejs installed

Create a Linkedpipes ETL data and working dir root and make the owner the current user:

```
$ sudo mkdir /usr/local/linkedpipes_etl
$ sudo mkdir /usr/local/linkedpipes_etl/working
$ sudo mkdir /usr/local/linkedpipes_etl/pipelines
$ sudo chown -R $USER.$USER /usr/local/linkedpipes_etl
```

Clone the sources:

```
$ cd /usr/local/linkedpipes_etl
$ git clone https://github.com/linkedpipes/etl.git
```

Deploy LinkedPipes

```
$ cd /usr/local/linkedpipes_etl/etl
$ mvn install
```

Create a configuration file:

```
$ cd deploy
$ touch configuration.properties
```

Create a configuration.properties under /usr/local/linkedpipes_etl/etl/deploy. It may look like this:

```
executor.webserver.port = 8085
executor.webserver.uri = http://localhost:8085

executor.execution.working_directory = /usr/local/linkedpipes_etl/working
executor.execution.uriPrefix = http://localhost:8080/resources/executions/
```

```
executor.log.directory = /var/log
executor.log.core.level = DEBUG

executor.osgi.lib.directory = /usr/local/linkedpipes_etl/etl/deploy/osgi
executor.osgi.working.directory = .felix/

executor-monitor.webserver.port = 8081
executor-monitor.webserver.uri = http://localhost:8081/api/v1/
executor-monitor.log.directory = /var/log
executor-monitor.log.core.level = DEBUG
executor-monitor.ftp.command_port = 2221
executor-monitor.ftp.data_ports_interval.start = 2222
executor-monitor.ftp.data_ports_interval.end = 2225
executor-monitor.ftp.uri = ftp://localhost:2221

frontend.webserver.port = 8080

storage.components.directory = /usr/local/linkedpipes_etl/etl/deploy/components
storage.components.path.prefix = file://
storage.pipelines.directory = /usr/local/linkedpipes_etl/pipelines

domain.uri = http://localhost:8080

external.fuseki.path =
external.working =
external.port.start = 3300
external.port.end = 3400
```

LINKEDPIPES ETL START / STOP SCRIPT

Save this script as /usr/local/bin/linkedpipes_etl.sh

```
#!/bin/bash

linkedpipes_etl_path="/usr/local/linkedpipes_etl/etl/deploy"

usage="Usage: linkedpipes_etl.sh [start|stop]"

if [ $# -eq 0 ]
then
        echo "No arguments given, "$usage
        exit
fi

if [ $1 == "start" ]
then
        cd $linkedpipes_etl_path

        echo Running executor
        ./executor.sh >> /tmp/lp_executor.log &

        echo Running executor-monitor
        ./executor-monitor.sh >> /tmp/lp_executor-monitor.log &

        echo Running frontend
        ./frontend.sh >> /tmp/lp_frontend.log &

elif [ $1 == "stop" ]
then
        echo Killing Executor
        kill `ps ax | grep /executor.jar | grep -v grep | awk '{print $1}'`

        echo Killing Executor-monitor
        kill `ps ax | grep /executor-monitor.jar | grep -v grep | awk '{print $1}'`
```

```
        echo Killing Executor-view
        kill `ps ax | grep node | grep -v grep | awk '{print $1}'`
else
        echo "Unknown argument "$1" "$usage
fi
```

Make it executable

```
$ sudo chmod +x /usr/local/bin/linkedpipes_etl.sh
```

Now you can start linkedpipes etl with:

```
$ linkedpipes_etl.sh start
```

To stop it run:

```
$ linkedpipes_etl.sh stop
```

5. About ABuDaT

ABuDaT is a software system that enables users to transform real world administrative budget data into the openbudgets.eu RDF data model. Transforming the raw data published by the authorities from various formats such as tabular (.csv) or hierarchical (.xml) and in various languages involves multiple steps which are described in this document.

The target format for ABuDaT is the OpenBudgets.eu RDF data model.

5.1. Navigation through ABuDaT User Interface

ABuDaT provides a web-based front end user interface that can be opened by navigating a browser to *http://localhost:9000.*

The front page contains 4 primary action buttons in the middle of the page (Image 1):

- "Transform Spending Data", allows users to transform spending data,
- "Transform Code Lists", allows users to transform code lists,
- "Information about ABuDaT", displays information about ABuDaT,
- "Information about Openbudgets.eu", displays information about Openbudgets.eu project.

The top navigation contains other buttons that become relevant once a transformation has been performed or saved:

- "Data Transformations", displays previously stored transformation of spending data,
- "Code Transformations", shows previously stored codelist transformations,
- "Home", helps users to navigate back to the homepage,
- "Openbudgets.eu", displays information about Openbudgets.eu project,
- "About ABuDaT", displays information about ABuDat.

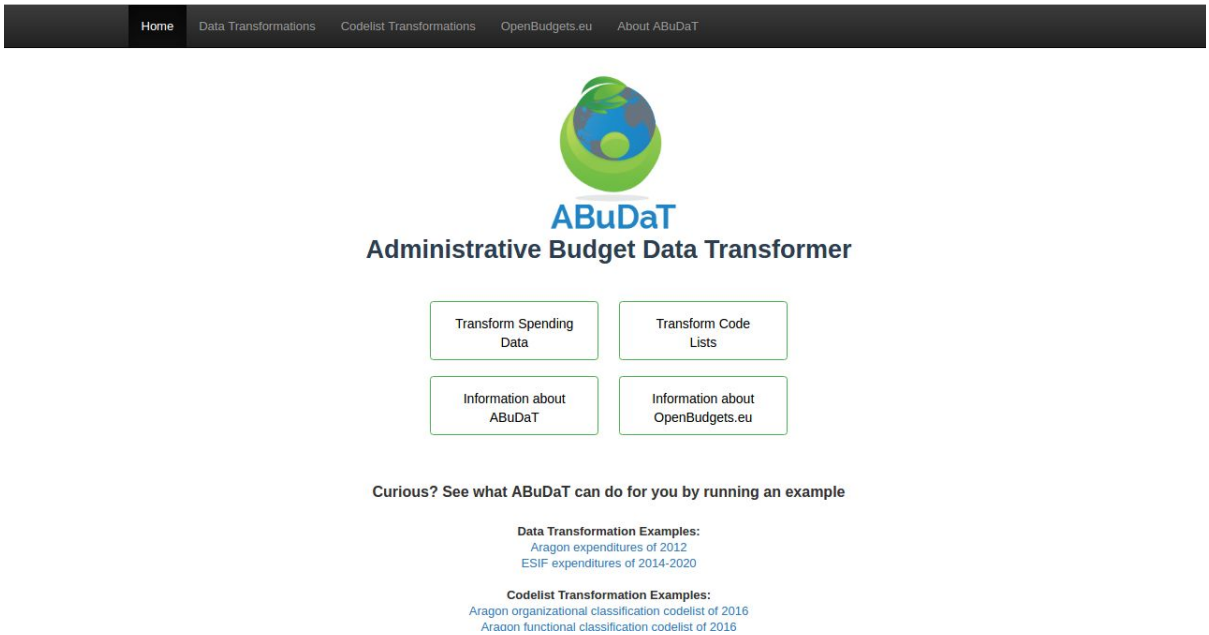

Image 1

5.2. Budget Data transformations

Step 1: Start by opening the "Transform Spending Data" page by clicking on the button (Image 2).

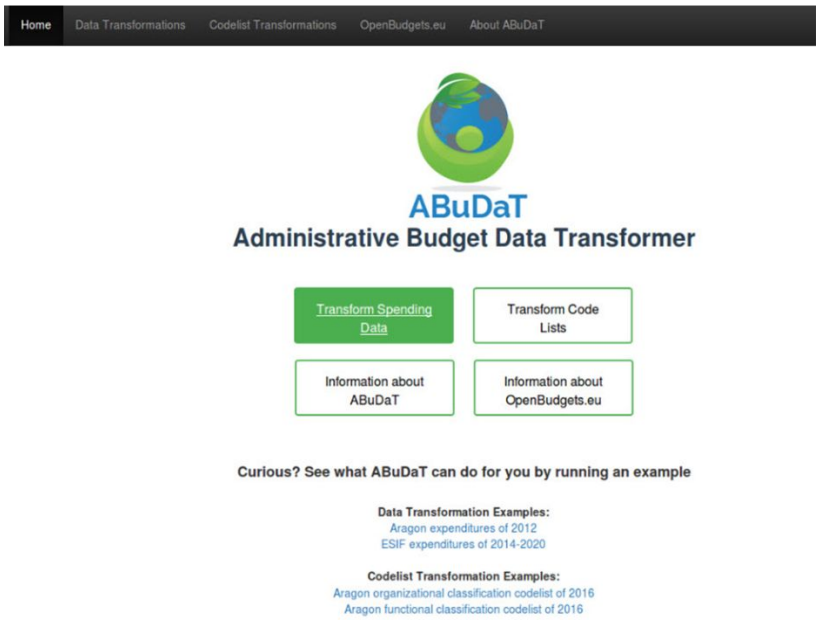

Image 2

Step 2: A new page opens for inserting the URL of the budget data link in the address bar (the data can be tabular (.csv) or hierarchical (.xml)), then click on submit button (Image 3).

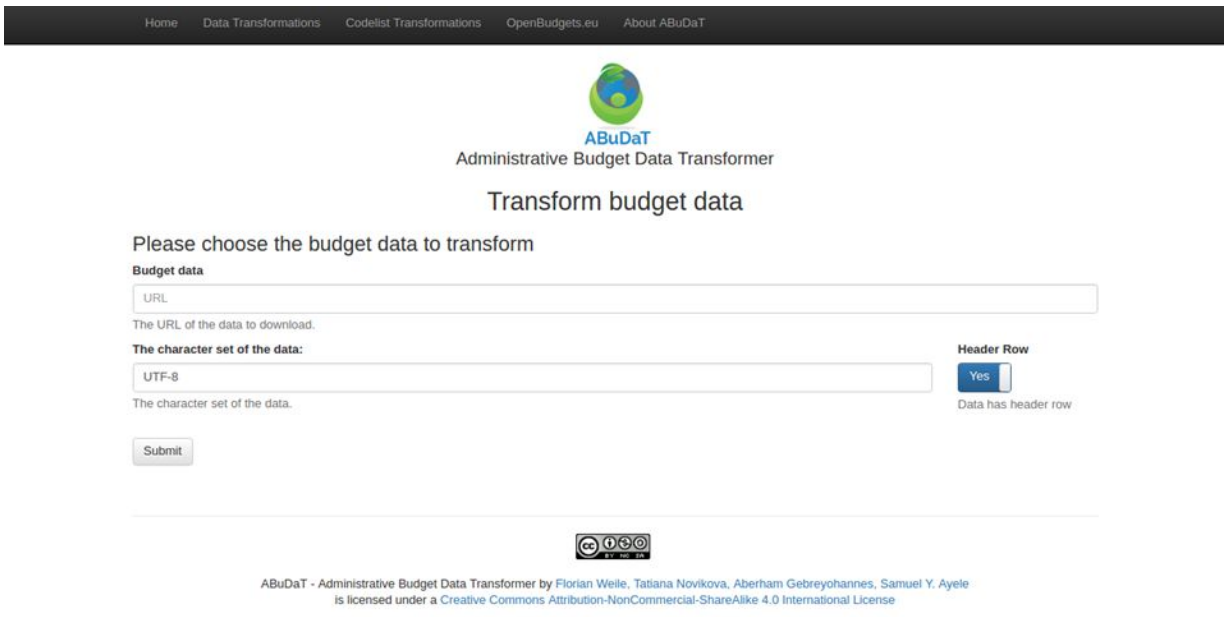

Image 3

Step3: After clicking the submit button the general transformation page opens as shown below on Image 4.

First: fill in general information:

- Enter a URL for budget data to transform;
- Provide the name for the data transformation;
- Enter the URL for the dataset;
- Add the description for the Transformation.

Second: fill in core dimension of the dataset:

- Enter budgetary unit of the budget;
- Mention fiscal period of the budget data;
- Operation character (optional);
- Budget phase (optional).

Third : core Attributes of the dataset:

- Currency of budget;
- Mention about tax.

Forth: customer dimensions:

- Insert IRI;
- Select Subproperty;
- Label;
- Codelist;
- Put a comment.

Image 4

After filling the form properly, at the end of the form 3 buttons are located:

- Execute in LinkedPipes ETL: execute the budget data in LinkedPipes;
- Save: save all the inputs that were filled in;
- Save a Copy: make a backup copy of the input (Image 5).

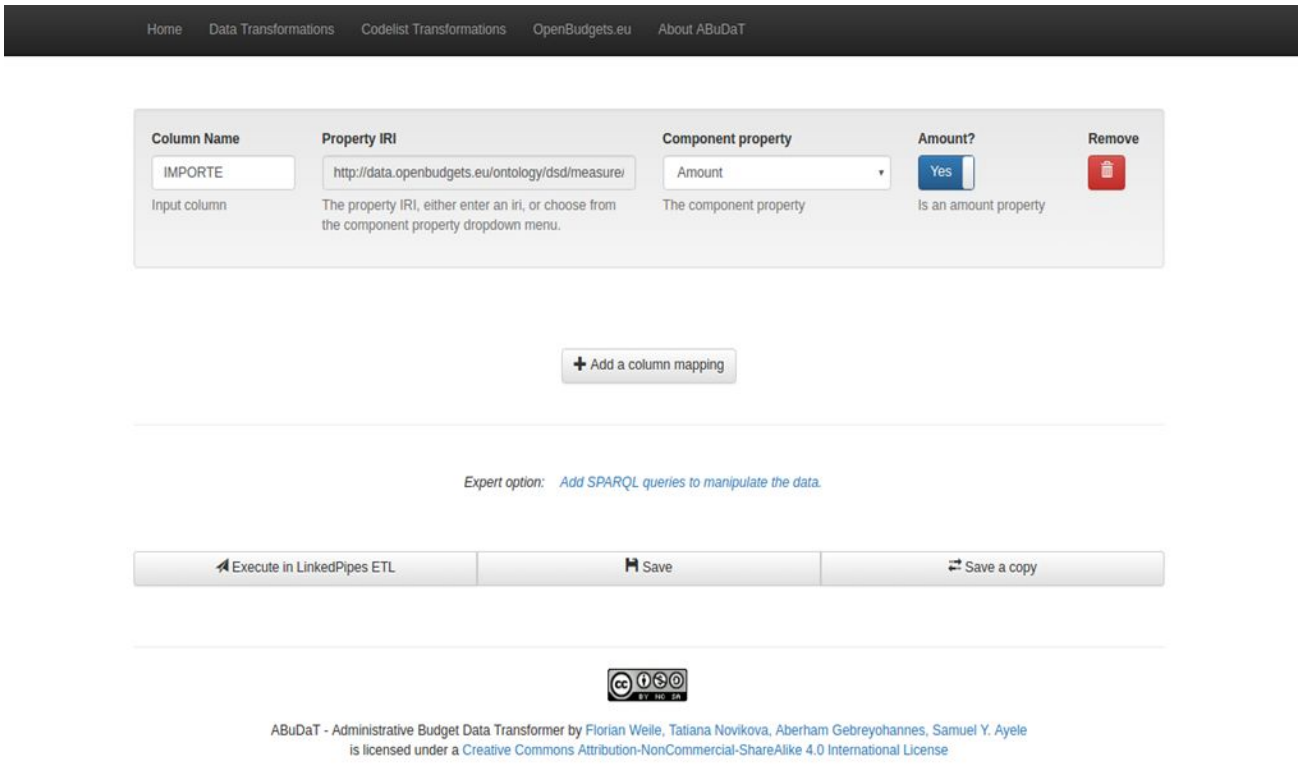

Image 5

Step5: after completing step 4:

- Click Executions button from the data transformations page (Image 6);
- Click Download for having RDF format of the budget data.

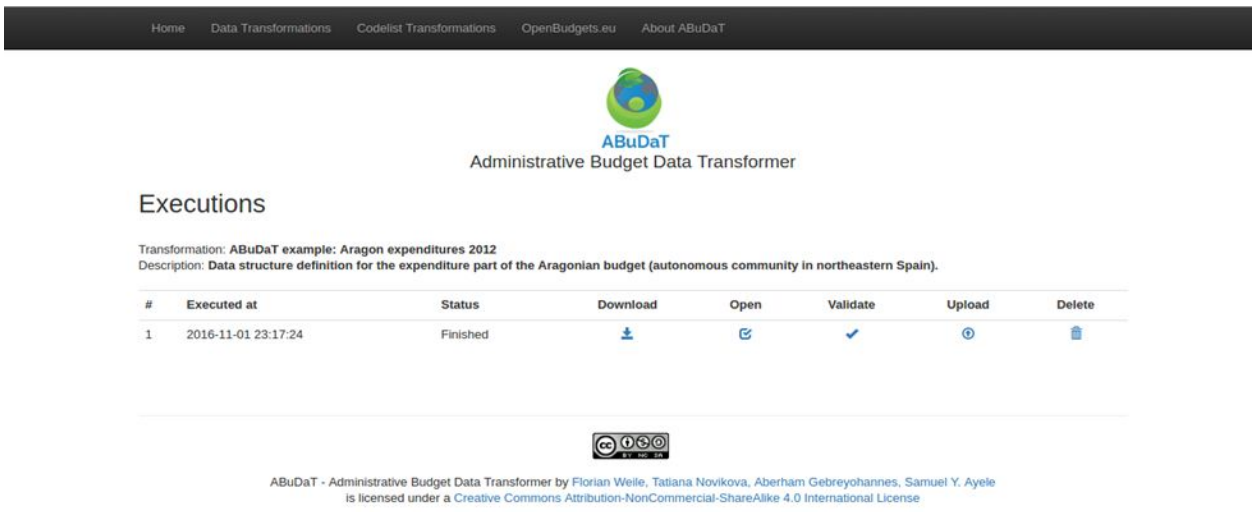

Image 6

Step 6: RDF format Result (Image 7):

```
@prefix obeu-measure: <http://data.openbudgets.eu/ontology/dsd/measure/> .
@prefix obeu-budgetphase: <http://data.openbudgets.eu/resource/codelist/budget-phase/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix fn: <http://www.w3.org/2005/xpath-functions#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix obeu-attribute: <http://data.openbudgets.eu/ontology/dsd/attribute/> .
@prefix obeu-dimension: <http://data.openbudgets.eu/ontology/dsd/dimension/> .
@prefix obeu-operation: <http://data.openbudgets.eu/resource/codelist/operation-character/> .
@prefix qb: <http://purl.org/linked-data/cube#> .
@prefix obeu-currency: <http://data.openbudgets.eu/resource/codelist/currency> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix sesame: <http://www.openrdf.org/schema/sesame#> .

<http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/observation/10> <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-
2012/dimension/administrativeClassification> <http://data.openbudgets.eu/resource/codelist/estructura_organica_aragon_2012/01010> ;
        <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/economicClassification>
<http://data.openbudgets.eu/resource/codelist/estructura_economica_aragon_2012/121001> ;
        <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/functionalClassification>
<http://data.openbudgets.eu/resource/codelist/estructura_funcional_aragon_2012/1111> ;
        <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/fundingClassification>
<http://data.openbudgets.eu/resource/codelist/estructura_financiacion_aragon_2012/91002> ;
        obeu-measure:amount "587.647,56"^^xsd:decimal ;
        a qb:Observation ;
        qb:dataSet <http://data.openbudgets.eu/resource/datasets/aragon-expenditure-2012> .

<http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/administrativeClassification> a qb:dimension , qb:CodedProperty , rdf:Property ;
        rdfs:label "Organization managing the planned budget" ;
        qb:codeList <http://data.openbudgets.eu/resource/codelist/estructura_organica_aragon_2012> ;
        rdfs:comment "The administrative classification is organized hierarchical in four levels." ;
        rdfs:isDefinedBy <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012> ;
        rdfs:range skos:Concept ;
        rdfs:subPropertyOf obeu-dimension:administrativeClassification .

<http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/economicClassification> a qb:dimension , qb:CodedProperty , rdf:Property ;
        rdfs:label "Economic Classification" ;
        qb:codeList <http://data.openbudgets.eu/resource/codelist/estructura_economica_aragon_2012> ;
        rdfs:comment "Identifies the type of expenditure incurred or source of revenues. The economic classification is organized hierarchical into chapters, articles, concepts,
and sub-concepts. This dimension is used for both, expenditure and revenue." ;
        rdfs:isDefinedBy <http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012> ;
        rdfs:range skos:Concept ;
        rdfs:subPropertyOf obeu-dimension:economicClassification .

<http://data.openbudgets.eu/ontology/dsd/aragon-budget-exp-2012/dimension/functionalClassification> a qb:dimension , qb:CodedProperty , rdf:Property ;
        rdfs:label "Functional Classification" ;
        qb:codeList <http://data.openbudgets.eu/resource/codelist/estructura_funcional_aragon_2012> ;
```

Image 7

5.3. Codelists' transformations

Start by clicking on Transform Code Lists from the index page (Image 8). This will take you to the "Transform Budget Data" page (Image 9).

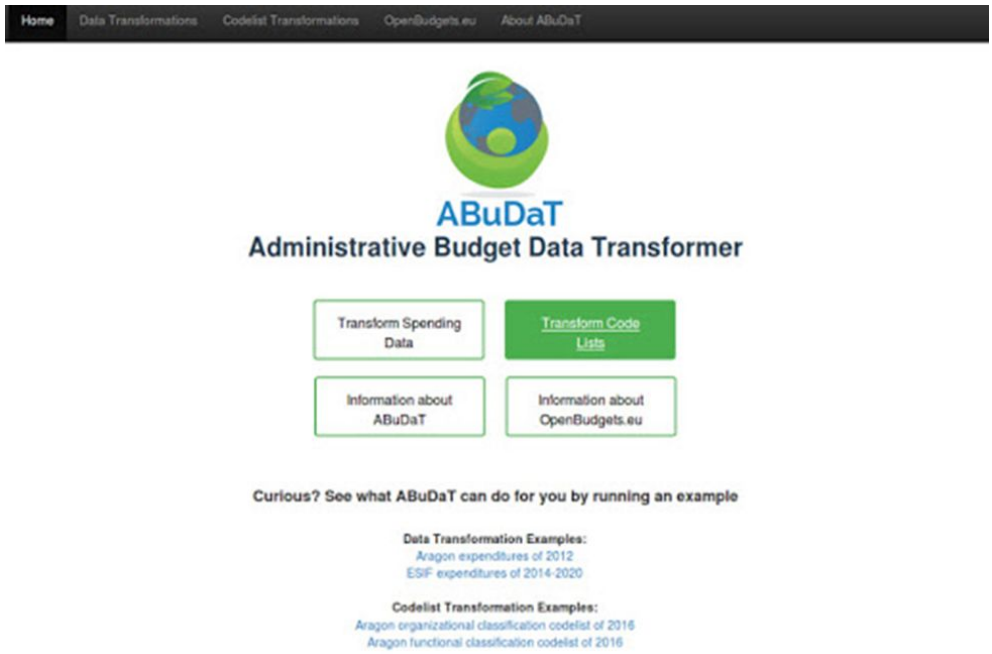

Image 8

Enter the URL of the spending data you would like to transform and click on submit button (Image 9).

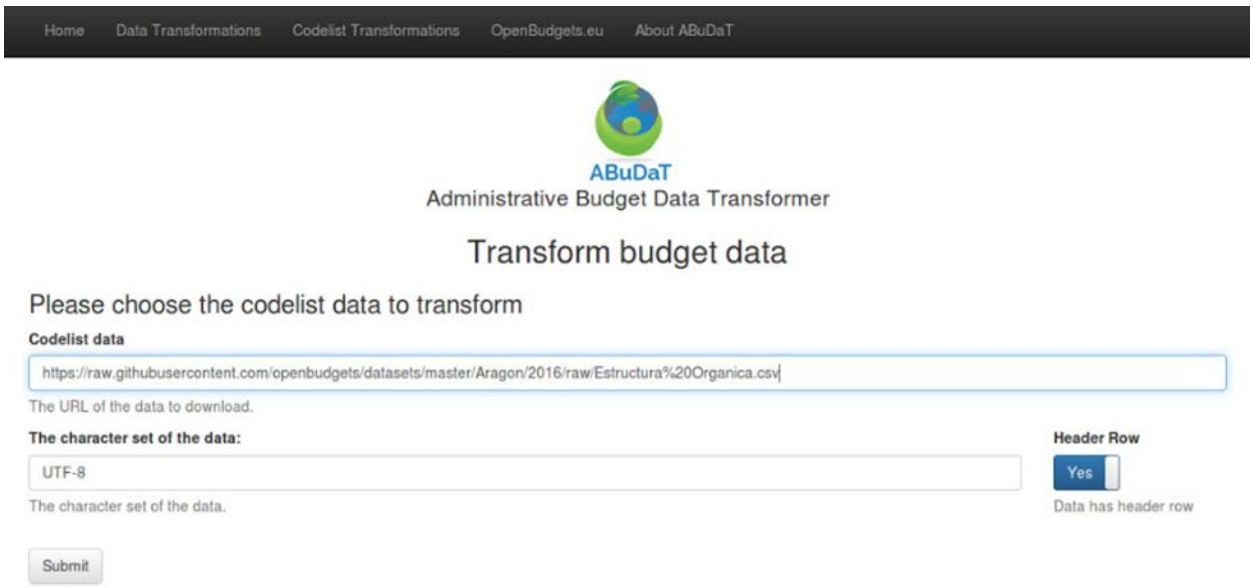

Image 9

This will open up the “General Information” (Image 10) page where you can:

- Give a name for the transformation;
- Add a description;

- And enter URL for the code list.

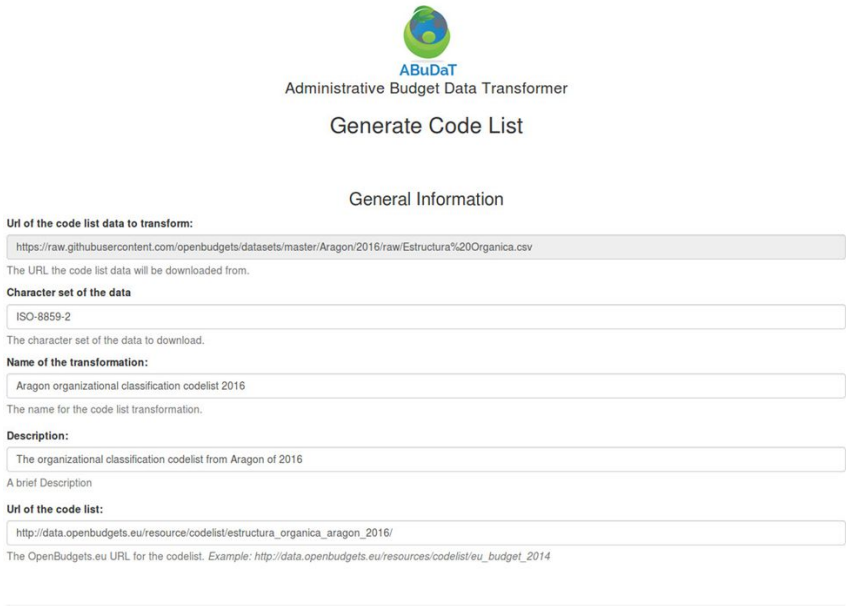

Image 10

Furthermore this page allows you to perform the following (Image 11)

- Choose key column;
- Choose Label column;
- Set start and end row;
- Decide whether Slicing is enabled or not.

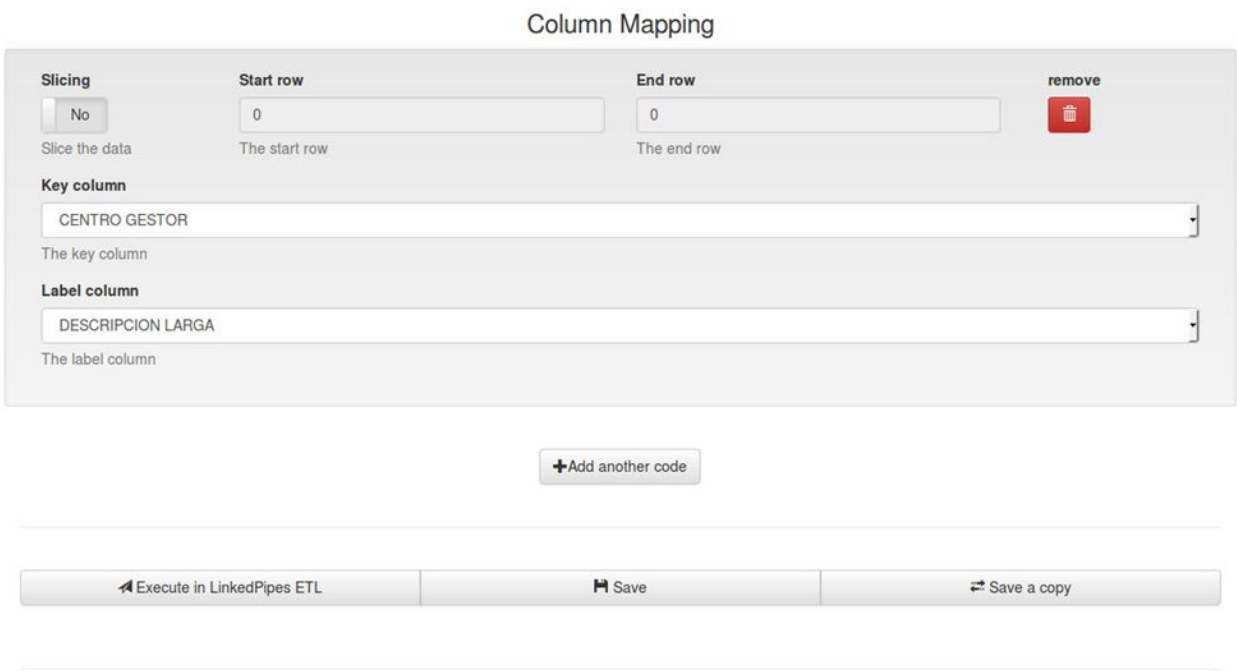

Image 11

You can add another code by clicking on the “Add another code” button or remove them by clicking on the “Remove” button next to the end row input field.

When you are ready to execute, simply click on “Execute in LinkedPipes ETL” at the bottom of the page, this will display an alert message depending on the outcome of the result. A Successful transfer to LinkedPipes or Save displays a Success message (Image 12) while an error causes ABuDaT to display an Error message (Image 13).

Image 12

Image 13

Once your codelist transformation has been successfully sent to LinkedPipes, you can open another tab in your browser and type in *https://localhost:8080*. You will see your newly added execution for codelists transformation in your executions list (Image 14).

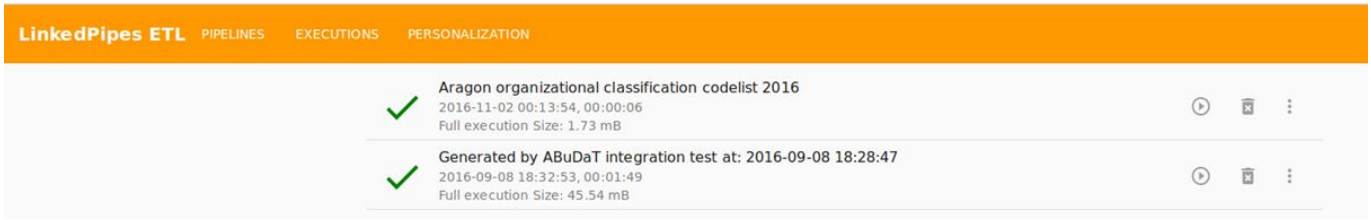

Image 14

5.4. How to View Expenditure Data Transformations

Clicking on the "Data Transformations" button from the top navigation opens up the Data Transformations page which will display "No transformations yet" message if no transformations have been saved yet and gives an option to start creating a new transformation or to try one of the examples (Image 15).

Image 15

Once you have saved a data transformations, you are able to view them here (Image 16).

Image 16

There are four options for saved Data Transformations.

- 'Edit' - opens up the transformation and allows you to edit;
- 'Execute' - sends the transformation to LinkedPipes ETL;
- 'Executions' - displays the Executions page (Image 17) for a particular data transformation which gives information on last execution date and time, its status, and provides links for.
    - Downloading the result of execution as a "turtle" file format;
    - Opening the execution in LinkedPipes ETL (using a new browser tab);
    - Validating - using the Data Cube Validation;
    - Uploading - to a fuseki triple store;
    - Deleting - deletes the execution from the list.
- 'Delete' - Removes the Expenditure Data Transformation entry from the Data Transformations page (Image 18).

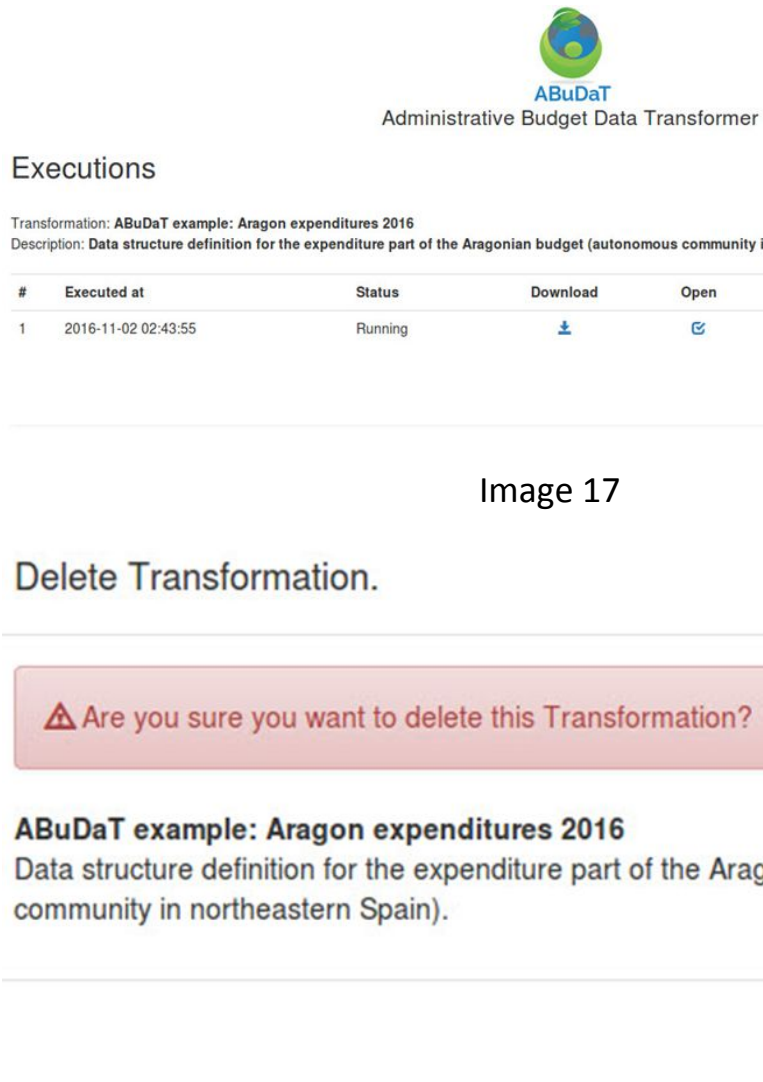

Image 17

Image 18

5.5. How to View Codelist Transformations?

Clicking on the "Codelist Transformations" button from the top navigation opens up the Codelist Transformations page which would display a "No transformations yet," message if no transformations have been saved yet and gives an option to start creating a new transformation or to try one of the examples (Image 19).

Once you have saved a data transformations, you are able to view them here (Image 20).

Image 19

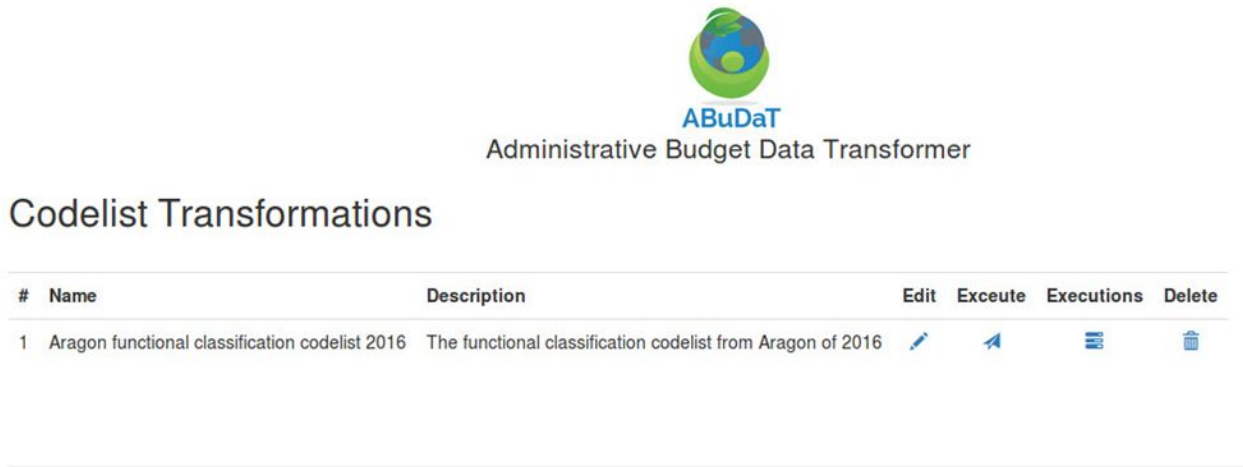

Image 20

There are four options for saved Data Transformations.

- 'Edit' - opens up the transformation and allows you to edit;
- 'Execute' - sends the transformation to LinkedPipes ETL;
- 'Executions' - displays the Executions page (Image 21) for a particular data transformation which gives information on last execution date and time, its status, and provides links for.
    - Downloading the result of execution as a "turtle" file format;
    - Opening the execution in LinkedPipes ETL (using a new browser tab);

- Validating - using the Data Cube Validation;
- Uploading - to a fuseki triple store;
- Deleting - deletes the execution from the list.

'Delete' - Removes the Expenditure Data Transformation entry from the Data Transformations page (Image 22).

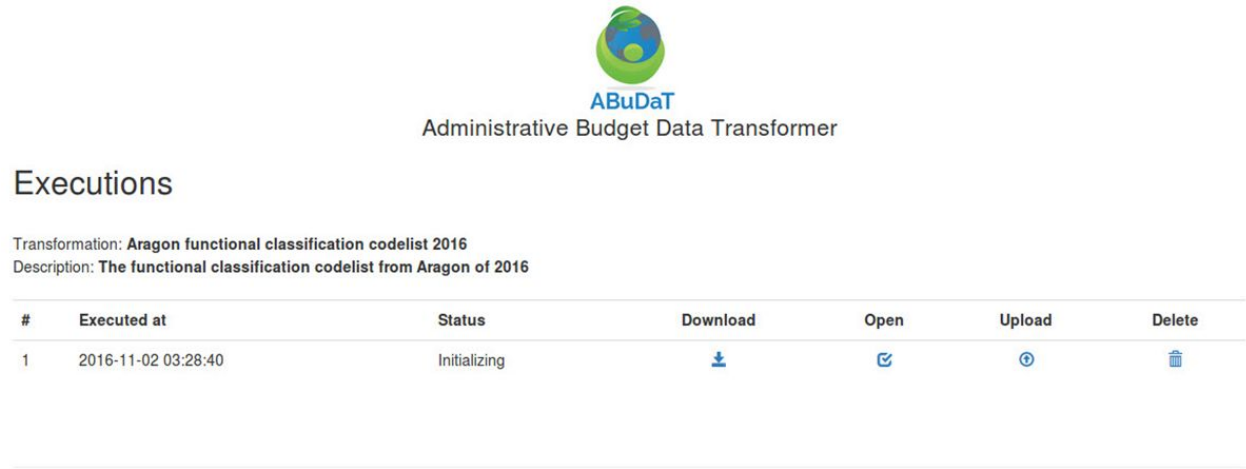

Image 21

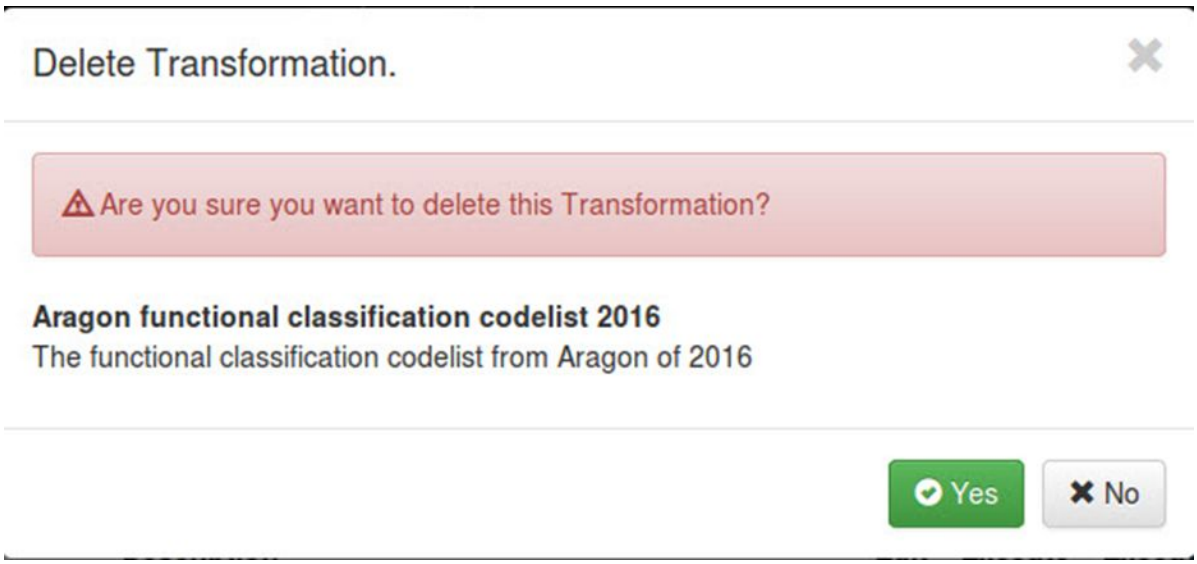

Image 22

References

Authors of this document are members of “Group E”, part of the students who participated in a lab module “MA-INF 3232/4313 EIS/Semantic Web Technologies” given by the Enterprise Information Systems working group at the University of Bonn, Germany during the Summer Semester of 2016.

Members of “Group E”:

- Florian Weile (L)
- Tatiana Novikova
- Aberham Gebreyohannes
- Samuel Y. Ayele

“Group E” was supervised by Fathoni Musyaffa.