DataShield/Opal
Notes on Deployment, Tools & Data

Dr. Ulrich Harttig
Team Leader of the Human Study Center (HSZ) Data Center

ENPADASI Workshop WP3
Bari, 08-09 June 2016
• Opal Deployment
  – Web access via proxy
  – Proxy setup example
• Data import
  – Data Dictionary
  – Data upload
• Opal Tools
  – CLI access
  – REST interface
• DataShield Proxy Demo setup
Opal Deployment - Web access via proxy

Access via proxy allows
- Deployment in standard environment, no opening of additional firewall holes
- Integration in existing infrastructure
- Flexible backend structure

setup @ DIfE

https://epi.dife.de/zopal/ (external access via proxy, IP unrestricted, OPAL accounts needed)
https://epi.dife.de/dstudio/ (Rstudio, external access via proxy, restricted: special account needed)
Opal Deployment - Proxy setup example

- using Apache 2.4 with modules: proxy proxy_connect proxy_fcgi proxy_html proxy_http
  proxy_wstunnel, rewrite, deflate
- part of an existing web server configuration
- for a setup of a complete Virtual Host dedicated to Opal see
  http://wiki.obiba.org/display/OPALDOC/Opal+Configuration+Guide

```
# disable forward proxy
ProxyRequests Off

<Proxy "*">
  SetOutputFilter INFLATE;proxy-html;DEFLATE
  Require all granted
</Proxy>

# for opal
# this works for jehanne.dyndns.org/opal/ and https://epi.dife.de/zopal/
RewriteRule "^/ws/(.*)$" "http://zimt:8880/ws/$1" [P]
<Location "/zopal/">
  LogLevel info rewrite:trace5 proxy:trace4
  ProxyPass "http://zimt:8880/"
  ProxyPassReverse "http://zimt:8880/"
  ProxyPreserveHost On
  RequestHeader set X-Forwarded-HTTPS ON
  
  ##
  
  <RequireAny>
  
  ##
  Require host dife.de
  
  ##
  
  </RequireAny>
  
  Require all granted
</Location>

# setup for the Rstudio/Server proxy
ProxyPassMatch ^/dstudio/p/([0-9]+)/websocket$ ws://localhost:8787/p/$1/$2/
<Location "/dstudio/">
  ProxyPass http://localhost:8787/
  ProxyPassReverse http://localhost:8787/

  AuthType Basic
  AuthName "DSstudio"
  AuthBasicProvider file
  AuthUserFile "/etc/apache2/security/ds.auth"
  
  <RequireAll>
  
  ##
  Require all granted
  Require user harttig
  
  ##
  Require ip 193.175.234.39 172.17.55.31
  
  </RequireAll>
</Location>
```

tested on 2 separate instances

URL of actual opal instance

IP/host based access restriction (recommended)
e.g. IP of the analysis machine(s)

OPTIONAL for debugging rewrite and proxy modules
Data Dictionary

- a structured documentation of detailed information for each variable in a study
- enhance comprehension and highlight heterogeneity across different studies.
- can be created before or after data import
- create and update Data Dictionary information directly in Opal or via ex/import of an Excel template

The template consists of 2 spreadsheets

- Variables Spreadsheet
  - is used to define variable attributes.
- Categories Spreadsheet
  - is used to define categories for the categorical variables

see also
http://wiki.obiba.org/display/OPALDOC/How+to+install+and+use+Opal+and+DataSHIELD+for+Data+Harmonization+and+Federated+Analysis
Variables Spreadsheet
The *name* column is mandatory.

Other columns can be deleted; default values are used for certain columns (see below). Add any other columns to specify customized attributes for your variables. The built-in column names are reserved words and should not be used as customized attributes. Specify the attribute language by adding the language code to the end of the attribute name (e.g. label:en, label:fr)

<table>
<thead>
<tr>
<th>Default columns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table name the variable will be added to. Default value is Table.</td>
</tr>
<tr>
<td>name</td>
<td>The variable name (mandatory).</td>
</tr>
<tr>
<td>valueType</td>
<td>The value type of the variable. Default value is text. Opal can store data on different entities such as Participant, Instrument, Area, Drug, etc. Default value is Participant.</td>
</tr>
<tr>
<td>entityType</td>
<td>The unit in which values expressed (e.g. cm, kg ...).</td>
</tr>
<tr>
<td>unit</td>
<td>The mime type of the variable to help applications to display documents (e.g. image/jpeg, application/excel ...).</td>
</tr>
<tr>
<td>mimeType</td>
<td>1 if repeatable, 0 if not. (eg. Three measures of blood pressure). Default value is 0.</td>
</tr>
<tr>
<td>occurrenceGroup</td>
<td>Name of a repeatable variable group (e.g The group [measure value, measure date] is a group of variables that can be repeated)</td>
</tr>
<tr>
<td>referencedEntityType</td>
<td>If the variable values are entity identifiers, this is the type of the entities that are referenced</td>
</tr>
<tr>
<td>index</td>
<td>Position or weight of the variable in the list of variables of the table for ordering. Default value is 0.</td>
</tr>
<tr>
<td>label</td>
<td>Label of the variable. Can be localized (e.g. label:en, label:fr ...).</td>
</tr>
<tr>
<td>alias</td>
<td>Alternative name for the variable, usually used for defining a shorter name for the variable</td>
</tr>
</tbody>
</table>

Categories Spreadsheet
Columns *variable* and *name* are mandatory.

<table>
<thead>
<tr>
<th>Default columns</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>table</td>
<td>The table name the variable will be added to. Default value is Table.</td>
</tr>
<tr>
<td>variable</td>
<td>The variable name the category belongs to (mandatory).</td>
</tr>
<tr>
<td>name</td>
<td>The category name (mandatory).</td>
</tr>
<tr>
<td>missing</td>
<td>Some categories are interpreted as missing answers (e.g. ‘Don’t know’, ‘Prefer not to answer’). Use 1 for missing and 0 for not. missing (normal answer). Default value is 0.</td>
</tr>
<tr>
<td>label</td>
<td>Label of the category. Can be localized (e.g. label:en, label:fr ...).</td>
</tr>
</tbody>
</table>
Data Import

1. step: upload to OPAL file system - into personal (per user) or project file space

2. step: import from OPAL file space into Project/Table
   includes indexing = might take a while, progress of longer running tasks can be monitored via 'Tasks' tab

Data sources & Types

- **CSV Datasource**
  "delimiter separated values" format (default delimiter being comma). The first column will represent the entity identifiers and the subsequent column names will identify variables. Each row of the file (except the first row) are the values for one entity. The entity identifier must be unique: there cannot be two rows starting with the same identifier.

- **Opal Archive Datasource**
  This datasource comes as a .zip file containing a folder for each table having: the full data dictionary in a XML file, a XML data file per entity. This is the file format used when exporting data from Onyx.

- **SPSS Datasource**
  The SPSS source file must be a valid non-compressed binary file with a .sav extension. In Opal an SPSS file represents a table and its variables are used as the table's data dictionary.

- **Excel Datasource**
  Opal supports both Excel 97 and Excel 2007 formats. Only for data dictionary import, NOT data import

- **Opal Datasource**
  Opal datasource allows one Opal server to connect to a remote Opal server. This can be useful when syncing datasources in different Opal instances. (No working example yet!)

Examples see in CLI Section

see also http://wiki.obiba.org/display/OPALDOC/Datasource+Types
ulrich@zimt:~/projects/datashield/opal-datashield-vm$ opal -h
usage: opal [-h]
  {dict, data, entity, file, import-opal, import-csv, import-xml, import-spss, import-limesurvey, import-sql, import-ids, import-ids-map, export-
  xml, export-csv, export-sql, copy-table, delete-table, user, group, perm-project, perm-datasource, perm-table, perm-variable, perm-r, perm-datashield, perm-
  system, system, rest, encrypt, decrypt}
...
Opal command line.

optional arguments:
  -h, --help            show this help message and exit

sub-commands:
  {dict, data, entity, file, import-opal, import-csv, import-xml, import-spss, import-limesurvey, import-sql, import-ids, import-ids-map, export-
  xml, export-csv, export-sql, copy-table, delete-table, user, group, perm-project, perm-datasource, perm-table, perm-variable, perm-r, perm-datashield, perm-
  system, system, rest, encrypt, decrypt}

Available sub-commands. Use --help option on the sub-command for more details.
dict   Query for data dictionary.
data   Query for data.
dict   Query for entities (Participant, etc.).
file   Manage Opal file system.
import-opal   Import data from a remote Opal server.
import-csv   Import data from a CSV file.
import-xml   Import data from a ZIP file.
import-spss   Import data from a SPSS file.
import-limesurvey   Import data from a LimeSurvey database.
import-sql   Import data from a SQL database.
import-ids   Import system identifiers.
import-ids-map   Import identifiers mappings.
export-xml   Export data to a zip of Opal XML files.
export-csv   Export data to a folder of CSV files.
export-sql   Export data to a SQL database.
copy-table   Copy a table into another table.
delete-table   Delete some tables.
user   Manage users.
group   Manage groups.
perm-project   Apply permission on a project.
perm-datasource   Apply permission on a datasource.
perm-table   Apply permission on a set of tables.
perm-variable   Apply permission on a set of variables.
perm-r   Apply R permission.
perm-datashield   Apply DataSHIELD permission.
perm-system   Apply system permission.
system   Query for system status and configuration
rest   Request directly the Opal REST API, for advanced
       users.
encrypt   Encrypt string using Opal’s secret key.
decrypt   Decrypt string using Opal’s secret key.
# download folder as ZIP file

```bash
opal file -v -o http://zimt:8880 -u administrator -p xxxxx -dl /home/administrator/testdata/CNSIM > CNSIM.zip
```

# upload data file to project directory - NOT data import

```bash
```

# import data from project directory to table space - data import

```bash
```

```json
{
   "command": "import",
   "commandArgs": "import --destination CNSIM",
   "id": 1,
   "messages": [
     {
       "msg": "Job started.",
       "timestamp": 1465222955297
     },
     {
       "msg": "  Importing tables [CNSIM3] in CNSIM ...\\n",
       "timestamp": 1465222955298
     }
   ],
   "name": "import",
   "owner": "demo01",
   "project": "CNSIM",
   "startTime": "2016-06-06T16:22:35.297+0200",
   "status": "IN_PROGRESS"
}
```

Further progress can be checked via 'Task' tab of UI.
# transfer data from one opal instance to another

```
```


2016-06-06 23:46:24,971 [qtp2025390719-27481] ERROR org.obiba.opal.rest.client.magma.RestDatasource - Unexpected error while communicating with Opal server: Host name may not be null -> Cause: ???

see also

http://wiki.obiba.org/display/OPALDOC/Opal+Python+User+Guide
Opal REST Interface

Programmatic access to Opal - any web client - advanced users

ulrich@zimt:~/projects/datashield/opal-datashield-vm$ opal rest --help

usage: opal rest [-h] [--opal OPAL] [--user USER] [--password PASSWORD]
      [--ssl-cert SSL_CERT] [--ssl-key SSL_KEY] [--verbose]
      [--method METHOD] [--accept ACCEPT]
      [--content-type CONTENT_TYPE] [--json]
      ws

positional arguments:
  ws                    Web service path, for instance:
                        /datasource/[PROJECT]/table/[TABLE NAME]/variable/[VARIABLE]

optional arguments:
  -h, --help            show this help message and exit
  --opal OPAL, -o OPAL  Opal server base url
  --user USER, -u USER  User name
  --password PASSWORD, -p PASSWORD
                        User password
  --ssl-cert SSL_CERT, -sc SSL_CERT
                        Certificate (public key) file
  --ssl-key SSL_KEY, -sk SSL_KEY
                        Private key file
  --verbose, -v         Verbose output
  --method METHOD, -m METHOD
                        HTTP method (default is GET, others are POST, PUT,
                        DELETE, OPTIONS)
  --accept ACCEPT, -a ACCEPT
                        Accept header (default is application/json)
  --content-type CONTENT_TYPE, -ct CONTENT_TYPE
                        Content-Type header (default is application/json)
  --json, -j            Pretty JSON formatting of the response

see also http://wiki.obiba.org/display/OPALDOC/REST+API+Command
Opal REST Interface - query information

```bash
opal rest -v -o https://epi.dife.de/zopal/ -u demo01 -p xxxxxx -m GET -j /datasource/CNSIM/table/CNSI
```

```json
{
  "datasourceName": "CNSIM",
  "entityType": "Participant",
  "link": "/datasource/CNSIM/table/CNSIM",
  "name": "CNSIM",
  "timestamps": {
    "created": "2016-01-31T15:30:06.000+0100",
    "lastUpdate": "2016-01-31T15:33:28.000+0100"
  }
}
```

```bash
opal rest -v -o https://epi.dife.de/zopal/ -u demo01 -p xxxxxx -m GET -j \\n/datasource/mytpr/table/eveAge/variable/PROBAND_ALTER
```

```json
{
  "attributes": [
    {
      "name": "script",
      "value": "if($(`PROBAND_ALTER`).any(-99).not().value()){$`PROBAND_ALTER`.value();\n }\nelse {\n ;\n}"
    },
    {
      "name": "derivedFrom",
      "namespace": "opal",
      "value": "/datasource/mytpr/table/BMBFeve/variable/PROBAND_ALTER"
    }
  ],
  "categories": [
    {
      "isMissing": true,
      "name": "miss"
    }
  ],
  "entityType": "Participant",
  "index": 0,
  "isRepeatable": false,
  "mimeType": "",
  "name": "PROBAND_ALTER",
  "parentLink": {
    "link": "/datasource/mytpr/table/eveAge",
    "rel": "eveAge"
  },
  "referencedEntityType": "",
  "unit": "",
  "valueType": "integer"
}
```
These are calls from the installation scripts preparing **databases** and default **projects** from pre-defined *.json files

```bash
opal rest -o http://localhost:8080 \
  -u administrator -p $OPAL_PWD \
  -m POST /system/databases \
  --content-type "application/json" < "${DATABASES_DIR}/${i}.json"
```

```bash
opal rest \
  -o https://localhost:8883 \
  -u administrator \
  -p $OPAL_PWD \n  -m POST /projects \
  --content-type "application/json" < CNSIM.json
```

```
{
  "name": "CNSIM",
  "title": "CNSIM",
  "description": "Simulated data",
  "database": "sqldb"
}
```

**see also**

[https://wikis.bris.ac.uk/display/DSDEV/Importing+data+into+Opal+with+the+API](https://wikis.bris.ac.uk/display/DSDEV/Importing+data+into+Opal+with+the+API)
#load opal/datashield libraries
library(opal)
library(dsBaseClient)
library(dsStatsClient)
library(dsGraphicsClient)
library(dsModellingClient)

# login details
server <- c("zimtproxy")
# access via proxy to be accessible from Rstudio Server or other client
url <- c("https://epi.dife.de/zopal/")
user <- c("demo01")
password <- c("aileeGhe")
table <- c("CNSIM.CNSIM")
logindata <- data.frame(server, url, user, password, table)

# Create an 'opals' object by passing the 'logindata' data frame to the  datashield.login function
opals <- datashield.login(logins=logindata, assign = TRUE)

ds.dim(x='D')  # dimension of data

ds.quantileMean(x='D$LAB_HDL')  # Quantiles of the data

ds.table1D(x='D$GENDER')

ds.histogram(x='D$LAB_HDL')

ds.heatmapPlot(x='D$LAB_TSC', y='D$LAB_HDL')

Errors:

> ds.histogram(x='D$LAB_HDL', type='split')
Fehler: Command 'rangeDS(D$LAB_HDL)' failed on 'zimtproxy': Error while evaluating 'dsGraphics::rangeDS( D$LAB_HDL )'

-> check the R SERVER instance ([OPAL-URL]/ui/index.html#!admin/!datashield) if ds packages have been loaded -> here , the package dsGraphics is missing