Mica Documentation

OBiBa

Aug 16, 2019

Contents

Targeted at individual studies and study consortia, OBiBa software stack (Opal, Mica etc.) provides a software solution for epidemiological data management, analysis and publication. While Opal, the core data warehouse application, provides all the necessary tools to import, transform and describe data, Mica provides everything needed to build personalized web data portals and publish content of research activities of both studies and consortia. Based on the content defined in Mica, Drupal is the preferred platform to build your personalized web portal. Mica is to be used with Agate, the OBiBa's central authentication server which centralizes user related services such as profile management, and a notification system using emails.

CHAPTER 1

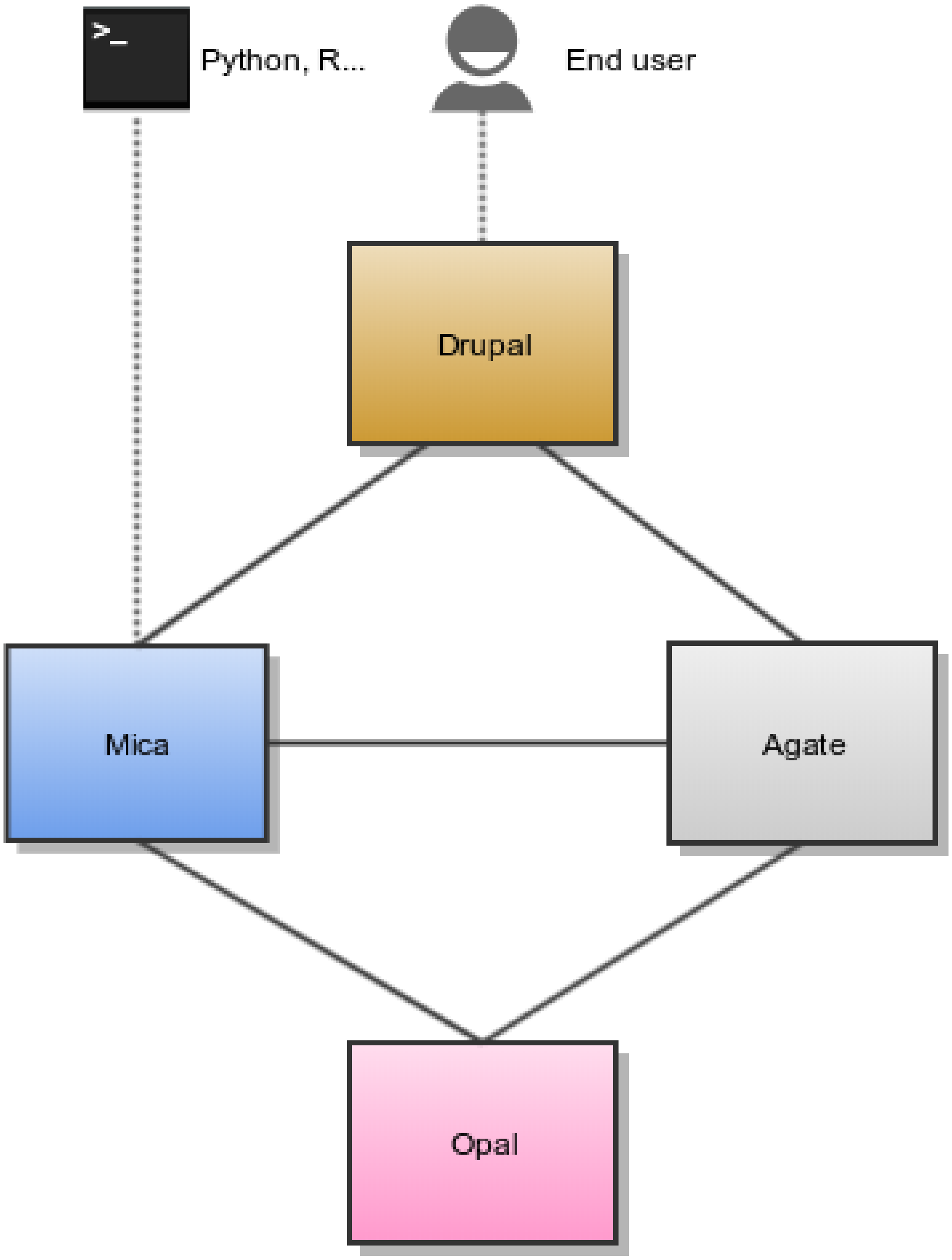
Introduction

Mica is an advanced web application designed to create data web portals for large-scale epidemiological studies or multiple-study consortia. It provides a structured description of consortia, studies, annotated and searchable data dictionaries, and data access request management.

Mica is built upon a multi-tier architecture consisting of several RESTful server and client applications. The table below list each application with a brief description:

Application	Description
Mica Server	Java server providing web services (REST) for managing, storing, searching Mica Domain content and communicating with other servers listed below.
Opal Server	Java server providing web services (REST) for importing, transforming and analyzing study variables.
Agate Server	Java server providing web services (REST) for user management and notifications.
Mica Web Application	Front-end to Mica Server providing client interface to manage Mica Domain content as well as to administrate and configure access permissions and secure connections.
Mica Drupal Client	Extension of the Drupal Content Management System (CMS) allowing to build a web data portal with Mica's published content.
Mica Python Client	Python front-end to Mica server providing services for administrative command-line and automation tasks.
Mica R Client	R front-end to Mica server providing services for Mica content analysis and reporting.

The diagram below illustrates the relationships between the Mica server and the other tiers:



1.1 Mica Server

Editors and reviewers of the Mica web portal content can access to the web interface of this server as described in the Mica Web Application User Guide. Data access request form can also be configured through this web interface.

Mica server is a client of Opal and Agate servers.

1.2 Opal Server

Opal application is used for:

- defining data dictionaries (variables),
- storing data,
- providing data summary statistics.

Opal offers well established security controls, allowing to NOT expose individual-level data. Note also that the Opal server is only accessed by the Mica server, reducing the risk of data compromisation from a malicious end user.

Installation and configuration guides can be found in the Opal documentation.

Mica expects at least one Opal server when some datasets are defined. Additional Opal servers can also be identified to access to distributed datasets.

1.3 Agate Server

Agate application is used for:

- having a user directory shared between OBiBa's applications,
- having centralized services such as profile management and email notifications.

Installation and configuration guides can be found in the Agate documentation.

1.4 Drupal Server

Drupal is a content management system, i.e. an application allowing to build fully customizable web portals. Drupal can be extended by modules and themes: Mica and Agate modules have been developed to access to the services of these servers. Drupal server is therefore a client of Mica and Agate servers.

Installation and configuration guides about Drupal as a Mica client can be found in the Mica Drupal Client User Guide documentation.

CHAPTER 2

Documents

Mica handles several type of documents, specific to the epidemiological studies domain: network, study, datasets etc. These document types have their own internal structure (to allow relationships between them and to ensure basic search), but can also be extended with custom fields. The default set of fields is the one promoted by Maelstrom Research. This default description model should fit with your needs in most of the cases.

All the documents follow the *Publication Flow* except the *Data Access Request* (which is a form privately exchanged between a researcher and the study/consortium).

2.1 Types

2.1.1 Network

A network is a group of epidemiological studies that has specific research interests. It is described using the following fields: name, aims, investigators, contact information and participating studies. It can also be related to other networks.

2.1.2 Individual Study

An individual study is defined as any epidemiological study (e.g. cohort, case control, cross sectional, etc.) conducted to better understand the distribution and determinants of health and disease. It is described using the following fields: name, objectives, investigators, contact information, design, data collection timeline, target number and characteristics of participants, and related scientific publications and documents. A study can include one or more populations described below.

Population

A population is a set of individuals sharing the same selection criteria for enrollment in a study. It is described using the following fields: name, sources of recruitment, participant characteristics, and number of participants. A population is linked to one or more data collection events according to the number of follow-ups.

Data Collection Event

A data collection event is a collection of information on one or more population(s) over a specific period of time (e.g. baseline, follow-up 1, follow-up 2). It is described using the following fields: name, start and end date, and data sources (e.g. questionnaires, physical measures, biosample measures, etc). A data collection event may be associated to one or more populations and it can include one or more datasets.

2.1.3 Harmonization Study

A harmonization study is defined as a research project harmonizing data across individual studies to answer specific reseach questions. It is described using the following fields: acronym, contact information, objectives, design and related documents. A harmonization study can include one population and one or more harmonized dataset (dataschema).

Population

A population is a set of individuals sharing the same selection criteria for enrollment in the individual studies selected to create the harmonization study. It is described using the fields: name and description. A population is linked to one or more harmonized dataset.

2.1.4 Collected Dataset

A collected (study-specific) dataset holds metadata about the variables collected within a data collection event. The metadata is described using a standardized format of data dictionary which provides information on collected variables' definitions and characteristics (e.g. type, unit, categories, and area of information covered). It can be associated to a study by specifying a data collection event.

Collected Variable

A collected variable is a variable that was collected, measured, or constructed within a study protocol. It is described using the following fields: name, label, description, type, unit, categories, and area of information covered. If the collected dataset includes data, summary statistics of the collected variable can be published on the web portal (e.g. means, minimum, maximum, counts and percentages). Each collected variable is part of one and only one study-specific dataset.

2.1.5 Harmonized Dataset

A harmonized dataset holds metadata about core variables constructed from multiple collected datasets. The metadata is described using a standardized format of data dictionary which provides information on harmonized variables' definitions and characteristics (e.g. type, unit, categories, and area of information covered): this represent the data schema of the harmonized dataset. It can be optionally associated to the harmonized data.

Data Schema Variable

A data schema variable is the harmonized dataset reference variable. Each harmonized variable will *implement* a corresponding data schema variable.

Harmonized Variable

A harmonized variable is a core variable (common format) generated by multiple individual studies. It is described using the following fields: name, label, description, type, unit, categories, and area of information covered. If the harmonized dataset includes data, summary statistics of the harmonized variable (e.g. means, minimum, maximum, counts and percentages) can be published on the web portal. Each harmonized variable is part of one and only one harmonized dataset.

2.1.6 Research Project

A research project reports information about the work that was conducted thanks to the network/study data: research objectives and results, contact information, status timeline. It could be somehow related to a data access request but not necessarily.

2.1.7 Data Access Request

A data access request is different type of document (compared to the studies, datasets etc.):

- it is created by a final user (usually a researcher having an account on the data web portal),
- it has its own life cycle (submission, approval etc.),
- permissions (view and edition) are restricted to the researcher and the data access officer and depend on the state of the request.

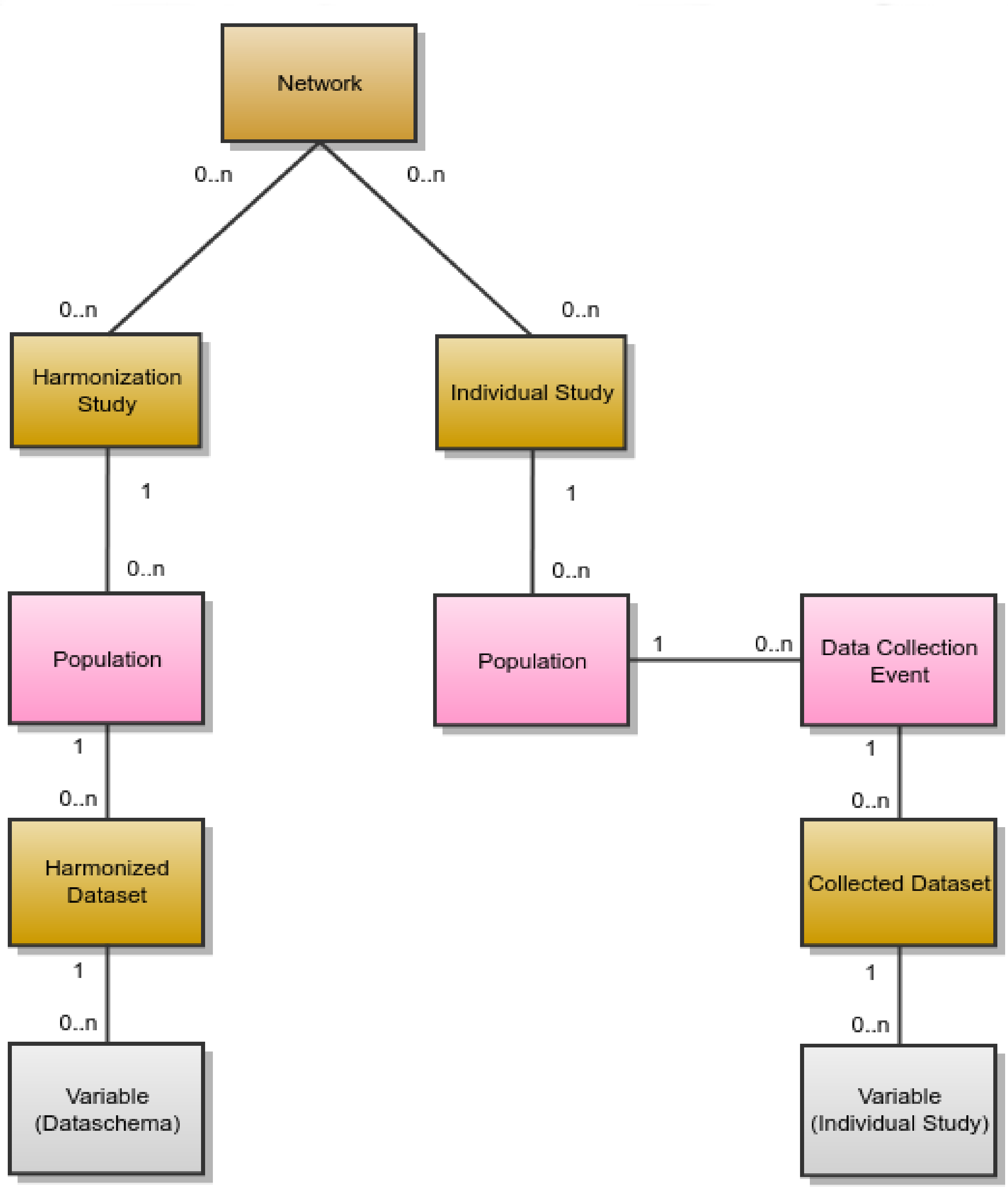
2.2 Search

Mica search engine allows to look into the domain while applying criteria on each type of document. The result of this combined query can be of any type. For example:

- search for variables about alcohol, associated to studies having collected biosamples, and being part of a network
- search all studies having collected biosamples and having variables about alcohol, and being part of a network
- ...

2.3 Associations

The following diagram describes the various documents that can be published in the Mica web portal. Each of them can be edited individually in the Mica Web Application administration interface (except variables, defined in the Opal servers).



2.4 Permissions

Three types of permissions can be granted to a user. Each permission is defined by a user role each of which applies different level of restrictions on a document. The table below lists each role and corresponding restrictions:

Role	Description
Reader	Read-only access to the document in draft mode with its revisions and its associated files.
Editor	Edit access to the document in draft mode with its revisions and its associated files. Publication or permanent deletion are not permitted.
Reviewer	Full access to the document, including its publication, permanent deletion and permissions.

2.5 Revision History

The revision history of a document is the succession of states after each edition (state refers to the content of the document, not its status). This history of changes allows to:

- view changes,
- reinstate a revision,
- identify which state is published.

2.6 Comments

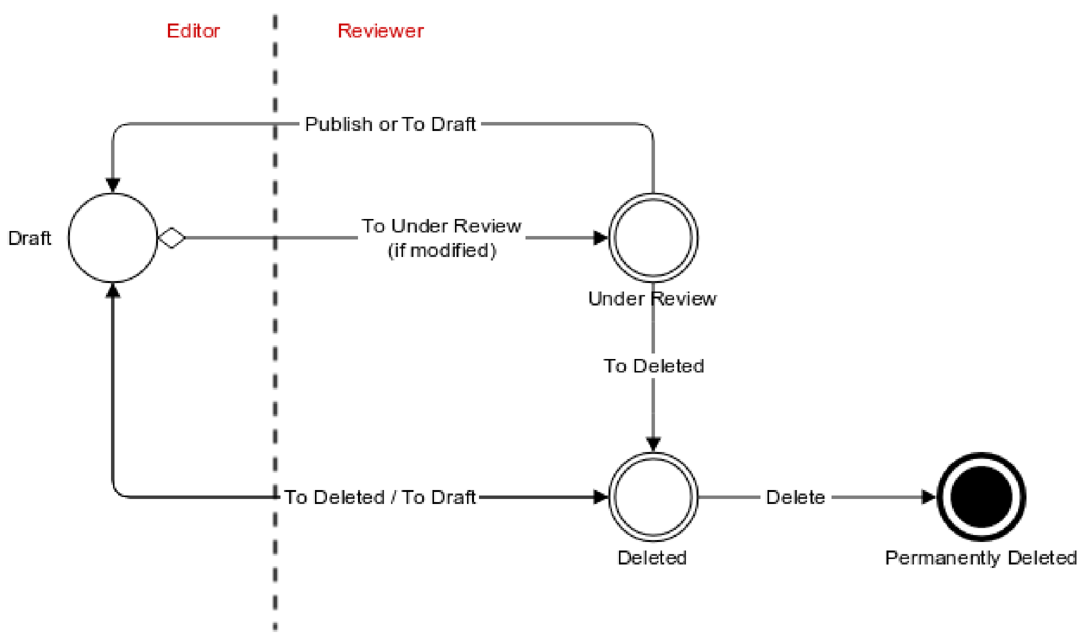
To enhance the collaboration between Mica users, each member can add a comment on any Mica domain document as well as data access requests documents. Mica can be configured to send email notifications when a comment is added or updated.

CHAPTER 3

Publication Flow

Documents (and their associated files) are all publishable documents (except *Data Access Request*). Being a publishable document means that there can be different revisions of the document before being published.

The publication flow refers to the work flow from a draft document to its publication. The following diagram represent the life cycle of a document with its *Revision Status* and *Transitions*:



3.1 Revision Status

The publishable document goes through several states allowing to separate user privileges: some users will be responsible for the content edition only, while other users will be responsible for the reviewing and the publication of the document.

A draft document can be changed/edited as many times as necessary. When the edition work is done, the document is staged for being reviewed. The state of the document that is reviewed is the one that will be published. Once the review and the publication is done, the document is ready again for edition. When a document is to be removed, it is first marked as being deleted (without affecting the publication) before being permanently removed.

The revision status is an enumeration of named states:

Status	Description	Editable	Publishable	Deletable	From Status	To Status
`Draft`	The document is in the editable state. This state requires lesser privileges: the document cannot be published nor deleted, it can only be staged for these operations.	✓	✗	✗	• Under Review • Deleted	• Under Review • Deleted
`Under Review`	Staged for reviewing, allowing user with higher privileges to approve and perform publication. The document is not editable and it can be published. Once published it automatically goes back to the Draft status. If the changes are not approved, the status can be switch to Draft without affecting the publication.	✗	✓	✗	• Draft	• Draft • Deleted
`Deleted`		✗	✗	✓	• Draft • Under	• Draft

3.2 Transitions

The transitions between the different revision status are the following:

Transition	Description	Permission	From Status	To Status
To Under Review	Once changes have be saved, the document is ready to be reviewed.	• Edit • Review	• Draft	• Under Review
To Draft	If reviewed changes or the deletion are rejected, the document can return to the draft state for edition.	• Edit • Review	• Under Review • Deleted	• Draft
Publish	When changes have been reviewed and approved, the document can be published: the current state of the document is persisted in the publication repository.	• Review	• Under Review	• Draft
To Deleted	Approval for document deletion is requested.	• Edit • Review	• Draft	• Deleted
Delete	Deletion is approved and effective. If the document was published, it is removed from the publication repository.	• Review	• Deleted	

CHAPTER 4

Installation

Mica is a stand-alone Java server application that requires MongoDB as database engine.

4.1 Requirements

4.1.1 Server Hardware Requirements

Component	Requirement
CPU	Recent server-grade or high-end consumer-grade processor
Disk space	8GB or more.
Memory (RAM)	Minimum: 4GB, Recommended: >4GB

4.1.2 Server Software Requirements

Software	Suggested version	Download link	Usage
Java	>= 1.8.x	Java Oracle downloads	Java runtime environment
MongoDB	>= 2.4.x	MongoDB downloads	Database engine

While Java is required by Mica server application, MongoDB can be installed on another server.

4.2 Install

Mica is distributed as a Debian/RPM package and as a zip file. The resulting installation has default configuration that makes Mica ready to be used (as soon as a MongoDB server is available). Once installation is done, see *Configuration* instructions.

4.2.1 Debian Package Installation

Mica is available as a Debian package from OBiBa Debian repository. To proceed installation, do as follows:

- Install Debian package. Follow the instructions in the repository main page for installing Mica.
- Manage Mica Service: after package installation, Mica server is running: see how to manage the Service.

4.2.2 RPM Package Installation

Mica is available as a RPM package from OBiBa RPM repository. To proceed installation, do as follows:

- Install RPM package. Follow the instructions in the RPM repository main page for installing Mica.
- Manage Mica Service: after package installation, Mica is running: see how to manage the Service.

4.2.3 Zip Distribution Installation

Mica is also available as a Zip file. To install Mica zip distribution, proceed as follows:

- Download Mica distribution
- Unzip the Mica distribution. Note that the zip file contains a root directory named **mica-x.y.z-dist** (where x, y and z are the major, minor and micro releases, respectively). You can copy it wherever you want. You can also rename it.
- Create an `MICA_HOME` environment variable
- Separate Mica home from Mica distribution directories (recommended). This will facilitate subsequent upgrades.

Set-up example for Linux:

```
mkdir mica-home
cp -r mica-x-dist/conf mica-home
export MICA_HOME=`pwd`/mica-home
./mica-x-dist/bin/mica
```

Launch Mica. This step will create/update the database schema for Mica and will start Mica: see Regular Command.

For the administrator accounts, the credentials are “administrator” as username and “password” as password. See User Directories Configuration to change it.

4.3 Upgrade

The upgrade procedures are handled by the application itself.

4.3.1 Debian Package Upgrade

If you installed Mica via the Debian package, you may update it using the command:

```
apt-get install mica
```

4.3.2 RPM Package Upgrade

If you installed Mica via the RPM package, you may update it using the command:

```
yum install mica
```

4.3.3 Zip Distribution Upgrade

Follow the Installation of Mica Zip distribution above but make sure you don't overwrite your mica-home directory.

4.4 Execution

4.4.1 Server launch

Service

When Mica is installed through a Debian/RPM package, Mica server can be managed as a service.

Options for the Java Virtual Machine can be modified if Mica service needs more memory. To do this, modify the value of the environment variable `JAVA_ARGS` in the file **/etc/default/mica**.

Main actions on Mica service are: `start`, `stop`, `status`, `restart`. For more information about available actions on Mica service, type:

```
service mica help
```

The Mica service log files are located in **/var/log/mica** directory.

Manually

The Mica server can be launched from the command line. The environment variable `MICA_HOME` needs to be setup before launching Mica manually.

Environment variable	Required	Description
`MICA_HOME`	yes	Path to the Mica "home" directory.
`JAVA_OPTS`	no	Options for the Java Virtual Machine. For example: *-Xmx4096m -XX:MaxPermSize=256m*

To change the defaults update: `bin/mica` or `bin/mica.bat`

Make sure Command Environment is setup and execute the command line (bin directory is in your execution PATH)):

```
mica
```

Executing this command upgrades the Mica server and then launches it.

The Mica server log files are located in **MICA_HOME/logs** directory. If the logs directory does not exist, it will be created by Mica.

4.4.2 Usage

To access Mica with a web browser the following urls may be used (port numbers may be different depending on HTTP Server Configuration):

- http://localhost:8082 will provide a connection without encryption,
- https://localhost:8445 will provide a connection secured with ssl.

4.4.3 Troubleshooting

If you encounter an issue during the installation and you can't resolve it, please report it in our Mica Issue Tracker.

Mica logs can be found in **/var/log/mica**. If the installation fails, always refer to this log when reporting an error.

CHAPTER 5

Configuration

The file **MICA_HOME/conf/application.yml** is to be edited to match your server needs. This file is written in YAML format allowing to specify a hierarchy within the configuration keys. The YAML format uses indentations to express the different levels of this hierarchy. The file is already pre-filled with default values (to be modified to match your configuration), just be aware that you should not modify the indentations. In the following documentation, the configuration keys will be presented using the dot-notation (levels are separated by dots) for readability.

5.1 HTTP Server Configuration

Mica server is a web application and as such, you need to specify on which ports the web server should listen to incoming requests.

Property	Description
`server.port`	HTTP port number. Generally speaking this port should not be exposed to the web. Use the https port instead.
`server.host`	Web server host name.
`https.port`	HTTPS port number.

5.2 MongoDB Server Configuration

Mica server will store its data (system configuration, networks, studies, datasets, etc.) in a MongoDB database. You must specify how to connect to this database.

Property	Description
`spring.data.mongodb.uri`	MongoDB URI. Read Standard Connection String Format to learn more.

By default MongoDB does not require any user name, it is highly recommended to configure the database with a user. This can be done by enabling the Client Access Control procedure.

Follow these steps to enable the Client Access Control on your server:

- create a user with the proper roles on the target databases
- restart the MongoDB service with Client Access Control enabled

Note: Once the MongoDB service runs with Client Access Control enabled, all database connections require authentication.

MongoDB User Creation Example

The example below creates the *micaadmin* user for *mica* database:

```
use admin

db.createUser( {
  user: "micaadmin", pwd: "micaadmin",
  roles: [
       { "role" : "readWrite", "db" : "mica" },
       { "role" : "dbAdmin", "db" : "mica" },
       { "role" : "readAnyDatabase", "db": "admin" }
  ]
});
```

Here is the required configuration snippet in **/etc/mica/application.yml** for the above user:

```
spring:
  data:
    mongodb:
      uri: mongodb://micaadmin:micaadmin@localhost:27017/mica?authSource=admin
```

Note: Mica requires either **clusterMonitor** or **readAnyDatabase** role on the *admin* database for validation operations. The first role is useful for a cluster setup and the latter if your MongoDB is on a single server.

5.3 Opal Server Configuration

Mica server uses Opal to retrieve data dictionaries, data summaries and variable taxonomies. This server is sometimes referred as the Opal primary server (secondary servers can be defined at study level). If you want to publish datasets, the following Opal connection details needs to be configured.

Property	Description
`opal.url`	Opal server URL. It is highly recommended to use https protocol.
`opal.username`	User name for connection to Opal server.
`opal.password`	User password for connection to Opal server.

Mica server should connect to Opal and access to some selected tables only with the lowest level of permissions (View dictionary and summary, i.e. no access to individual data). Please refer to the Opal Table Documentation for more details about the permissions that can be applied on a table.

5.4 Mica Server Configuration

Mica server uses Mica as a user directory and as a notification emails service. From the Mica point of view, Mica is not a user: it is an application. Each time Mica needs a service from Mica, it will provide the information necessary to its identification. The application credentials registered in Mica are to be specified in this section. If you want to specify advanced permissions or allow users to submit data access requests, the following Mica connection details needs to be configured.

Property	Description
`agate.url`	Mica server URL. It is highly recommended to use https protocol.
`agate.application.name`	Application name for connection to Mica server.
`agate.application.key`	Application key for connection to Mica server.

5.5 Shiro Configuration

Shiro is the authentication and authorization framework used by Mica. There is a minimum advanced configuration that can be applied to specify how Shiro will hash the password. In practice this only applies to the users defined in the shiro.ini file. Default configuration is usually enough.

Property	Description
`shiro.password.nbHashIterations`	Number of re-hash operations.
`shiro.password.salt`	Salt to be applied to the hash.

5.6 Elasticsearch Configuration

Mica server embeds Elasticsearch as its search engine. Elasticsearch is a key functionality of Mica as the process of publication consist in indexing documents (networks, studies, variables etc.) in the search engine. Advanced queries can be applied on the published documents. Elasticsearch is embeded, i.e. it is not an external application. Mica's Elasticsearch can be part of a cluster of Elasticsearch cluster. The configuration of the Elasticsearch node and how it should connect to the other nodes of the cluster can be specified in this section. Default configuration is usually enough.

Property	Description
elasticsearch.dataNode	Boolean to specify if this node has data or if it is just a proxy to other nodes in a cluster.
elasticsearch.clusterName	Cluster identifier.
elasticsearch.shards	Number of shards.
elasticsearch.replicas	Number of replicas.
elasticsearch.settings	A string in JSON or YAML format to define other elasticsearch settings. See Elasticsearch Documentation for advanced settings.
elasticsearch.transportClient	Boolean to indicate to use the Transport Client instead of creating an elasticsearch Node.
elasticsearch.transportAddress	Elasticsearch service IP address and port when using the Transport Client, defaults to the localhost at port 9300.
elasticsearch.transportSniff	Boolean to indicate the Transport Client to collect IP addresses from nodes in an elasticsearch cluster.

Elasticsearch Cluster

Mica can be set to join or connect to an Elasticsearch cluster. You need to set *elasticsearch.clusterName* to the name of the cluster you want to join. There are different possible cluster topologies, each of which has different resource utilization profiles in terms or memory and CPU.

Note: To avoid API incompatibility issues, the recommended version of Elasticsearch server is 2.4.

An example of a configuration to join an elasticsearch cluster using a Client Node:

```
elasticsearch:
  clusterName: mycluster
  dataNode: false
  settings: '{"node.master": false, "node.local": false}'
```

An example of a configuration using the transport client:

```
elasticsearch:
  clusterName: mycluster
  transportClient: true
  transportAddress: "myhost:9300"
```

Elasticsearch Server Configuration

Mica uses the scripting capabilities of Elasticsearch. All the machines in the Elasticsearch cluster should have the scripting module enabled by setting the following values in the *elasticsearch.yml* configuration file (location of this file depends on how your elasticsearch service is installed):

```
script:
  inline: true
  indexed: true
```

5.7 User Directories

The security framework that is used by Mica for authentication, authorization etc. is Shiro. Configuring Shiro for Mica is done via the file **MICA_HOME/conf/shiro.ini**. See also Shiro ini file documentation.

Note: Default configuration is a static user 'administrator' with password 'password' (or the one provided while installing Mica Debian/RPM package).

By default Mica server has several built-in user directories (in the world of Shiro, a user directory is called a realm):

- a file-based user directory (**shiro.ini** file),
- the user directory provided by Agate.

Although it is possible to register some additional user directories, this practice is not recommended as Agate provides more than a service of authentication (user profile, notification emails etc.).

In the world of Shiro, a user directory is called a *realm*.

File Based User Directory

The file-based user directory configuration file **MICA_HOME/conf/shiro.ini**.

Note: It is not recommended to use this file-based user directory. It is mainly dedicated to define a default system super-user and a password for the anonymous user.

For a better security, user passwords are encrypted with a one way hash such as sha256.

The example shiro.ini file below demonstrates how encryption is configured.

```
# =======================
# Shiro INI configuration
# =======================

[main]
# Objects and their properties are defined here,
# Such as the securityManager, Realms and anything else needed to build the␣
↪SecurityManager


[users]
# The 'users' section is for simple deployments
# when you only need a small number of statically-defined set of User accounts.
#
# Password here must be encrypted!
# Use shiro-hasher tools to encrypt your passwords:
#   DEBIAN:
#     cd /usr/share/mica2/tools && ./shiro-hasher -p
#   UNIX:
#     cd <MICA_DIST_HOME>/tools && ./shiro-hasher -p
#   WINDOWS:
#     cd <MICA_DIST_HOME>/tools && shiro-hasher.bat -p
#
# Format is:
# username=password[,role]*
administrator = $shiro1$SHA-256$500000$dxucP0IgyO99rdL0Ltj1Qg==$qssS60kTC7TqE61/JFrX/
↪OEk0jsZbYXjiGhR7/t+XNY=,mica-administrator
```

(continues on next page)

(continued from previous page)

```
anonymous = $shiro1$SHA-256$500000$dxucP0IgyO99rdL0Ltj1Qg==$qssS60kTC7TqE61/JFrX/
↪OEk0jsZbYXjiGhR7/t+XNY=

[roles]
# The 'roles' section is for simple deployments
# when you only need a small number of statically-defined roles.
# Format is:
# role=permission[,permission]*
mica-administrator = *
```

Passwords must be encrypted using shiro-hasher tools (included in Mica tools directory):

```
cd /usr/share/mica2/tools
./shiro-hasher -p
```

5.8 Reverse Proxy Configuration

Mica server can be accessed through a reverse proxy server.

Apache

Example of Apache directives that:

- redirects HTTP connection on port 80 to HTTPS connection on port 443,
- specifies acceptable protocols and cipher suites,
- refines organization's specific certificate and private key.

```
<VirtualHost *:80>
    ServerName mica.your-organization.org
    ProxyRequests Off
    ProxyPreserveHost On
    <Proxy *>
        Order deny,allow
        Allow from all
    </Proxy>
    RewriteEngine on
    ReWriteCond %{SERVER_PORT} !^443$
    RewriteRule ^/(.*) https://mica.your-organization.org:443/$1 [NC,R,L]
</VirtualHost>
<VirtualHost *:443>
    ServerName mica.your-organization.org
    SSLProxyEngine on
    SSLEngine on
    SSLProtocol All -SSLv2 -SSLv3
    SSLHonorCipherOrder on
    # Prefer PFS, allow TLS, avoid SSL, for IE8 on XP still allow 3DES
    SSLCipherSuite "EECDH+ECDSA+AESGCM EECDH+aRSA+AESGCM EECDH+ECDSA+SHA384␣
↪EECDH+ECDSA+SHA256 EECDH+aRSA+SHA384 EECDH+aRSA+SHA256 EECDH+AESG CM EECDH␣
↪EDH+AESGCM EDH+aRSA HIGH !MEDIUM !LOW !aNULL !eNULL !LOW !RC4 !MD5 !EXP !PSK !SRP !
↪DSS"
    # Prevent CRIME/BREACH compression attacks
    SSLCompression Off
    SSLCertificateFile /etc/apache2/ssl/cert/your-organization.org.crt
    SSLCertificateKeyFile /etc/apache2/ssl/private/your-organization.org.key
```

(continues on next page)

(continued from previous page)

```
    ProxyRequests Off
    ProxyPreserveHost On
    ProxyPass / https://localhost:8445/
    ProxyPassReverse / https://localhost:8445/
</VirtualHost>
```

For performance, you can also activate Apache's compression module (mod_deflate) with the following settings (note the json content type setting) in file */etc/apache2/mods-available/deflate.conf* :

```
<IfModule mod_deflate.c>
  <IfModule mod_filter.c>
      # these are known to be safe with MSIE 6
      AddOutputFilterByType DEFLATE text/html text/plain text/xml
      # everything else may cause problems with MSIE 6
      AddOutputFilterByType DEFLATE text/css
      AddOutputFilterByType DEFLATE application/x-javascript application/javascript␣
↪application/ecmascript
      AddOutputFilterByType DEFLATE application/rss+xml
      AddOutputFilterByType DEFLATE application/xml
      AddOutputFilterByType DEFLATE application/json
  </IfModule>
</IfModule>
```

CHAPTER 6

Plugins

6.1 Repository

Mica plugins available are:

Name	Type	Description	Depends	API
mica-search-es	mica-search	Mica search engine based on Elasticsearch 2.4. Can be used embedded in Mica (default) or configured to connect to an Elasticsearch cluster.	No dependencies	Search Plugin API

6.2 Installation

All plugins are to be deployed as a directory at the following location: **MICA_HOME/plugins**.

6.2.1 Automatic Installation

Because having a search engine is an absolute requirement, Mica server will check at startup that there is a plugin of type `mica-search` and if it's not the case, the latest version of the mica-search-es plugin (that applies to the current Mica server version) will be automatically downloaded and installed without needing a server restart. If for any reason this plugin cannot be automatically downloaded (network issue), the Mica start-up will fail and you will need to install the plugin manually.

6.2.2 Manual Installation

Available plugins can be downloaded from OBiBa Plugins Repository. The manual installation procedure should be performed as follow:

- Download the plugin of interest (zip file) from OBiBa Plugins Repository,

- Unzip plugin package in **MICA_HOME/plugins** folder. Note that the plugin folder name does not matter, Mica will discover the plugin through the plugin.properties file that is expected to be found in the plugin folder.
- Read the installation instructions (if any) of the plugin to identify the system dependencies or any other information,
- Restart Mica.

6.3 Configuration

The MICA_HOME/plugins folder contains all the Mica plugins that will be inspected at startup. A plugin is enabled if it has:

- A valid plugin.properties file,
- In case of several versions of the same plugin are installed, the latest one is selected.

The layout of the plugin folder is as follow:

```
MICA_HOME/
└── plugins
    └── <plugin-folder>
        ├── lib
        │   └── <plugin-lib>.jar
        ├── LICENSE.txt
        ├── README.md
        ├── plugin.properties
        └── site.properties
```

Inside the plugin's folder, a properties file, plugin.properties, has two sections:

- The required properties that describe the plugin (name, type, version etc.)
- Some default properties required at runtime (path to third-party executables for instance).

Still in the plugin's folder, a site-specific properties file, site.properties, is to be used for defining the local configuration of the plugin. Note that this file will be copied when upgrading the plugin.

6.4 Backups

Mica assigns a data folder location to the plugin: **MICA_HOME/data/<plugin-name>** where plugin-name is the name defined in the plugin.properties file. This folder is then the one to be backed-up.

CHAPTER 7

Web Introduction

The Mica Web Application is the administration web interface of the Mica server. It is NOT the end-user web portal and therefore firewall policies can (or should) be applied to restrict access to administrators or content editors.

See the *Documents* presentation page for a detailed description of the type of documents that can be edited through this web interface.

7.1 Requirements

This web interface is a javascript application requiring a modern web browser. There is no requirement regarding the operating system.

CHAPTER 8

Drupal Installation

Drupal is a Content Management System (CMS) allowing to build a web portal with a friendly administration interface and with extensible capabilities. What is referred to Mica Drupal Client in this documentation consists of a set of Drupal modules and theme. These modules/theme will get the published data from the Mica server (through its web services) and will deliver them as Drupal pages. Drupal supports user authentication which is itself extended to use Agate user directory. This way Drupal users can authenticate on Agate and get the Mica pages adapted to their permissions.

This guide describes how to set up a Drupal server with Mica client modules/theme configured. It is intended for the the system administrators.

8.1 Requirements

8.1.1 Server Hardware Requirements

Component	Requirement
CPU	Recent server-grade or high-end consumer-grade processor
Disk space	2GB or more.
Memory (RAM)	Minimum: 4GB, Recommended: >4GB

8.1.2 Server Software Requirements

Software	Suggested version	Usage
Drupal	7.x	Drupal application that will host Mica Client modules/theme.
Drupal requirements (PHP, database etc.)	PHP >=5.5	See Drupal Requirements

8.2 Dependencies

8.2.1 System dependencies

For Linux systems the following dependencies need to be installed:

- Debian

```
apt-get update
apt-get install mariadb-server php5.6 php5.6-mysql php5.6-curl php5.6-gd php5.6-cli␣
↪php5.6-xml
```

- CentOS

```
yum clean all
yum install mariadb-server php56w php56w-mysql php56w-gd php56w-cli php56w-xml
```

8.2.2 Drush and Composer

It is recommended to install Drush 7 (Drupal Shell) using Composer (Dependency Manager for PHP). See Drush install documentation.

Install Composer:

```
# Install Composer at system level (root access required)
curl -sS https://getcomposer.org/installer | sudo php -- --install-dir=/usr/local/bin␣
↪--filename=composer
```

Install Drush via Composer tool:

```
# Install Drush and add composer installation directory to your execution path
composer global require drush/drush:7.*
echo "export PATH=\$HOME/.composer/vendor/bin:\$PATH" | tee -a $HOME/.bashrc
# For CentOS 7 you have to use :
echo "export PATH=\$HOME/.config/composer/vendor/bin:\$PATH" | tee -a $HOME/.bashrc
source .bashrc

# Verify Drush install
drush status

# Install composer module for Drush (allows Drush to use Composer)
drush dl composer-8.x-1.x
```

8.2.3 Drupal Server

Now you can install Drupal 7. The installation with Drush is recommended. See Drupal Documentation for details (we recommend you the installation with drush).

Note: **CentOS** If you have problems about authorization (like httpd code 403 from apache), this error could be related to SELinux. You can disable SELinux (command : setenforce 0) to check if this resolves your problem (temporarily). See SELinux documentation for details.

8.3 Installation

The following modules and theme are required to have a fully functional Mica Drupal Client:

Name	Type	Drupal Link	Usage
obiba_mica	modules	https://www.drupal.org/project/obiba_mica	Uses Mica web services to render published content, data summaries and manage data access requests.
obiba_agate	module	https://www.drupal.org/project/obiba_agate	Uses Agate web services to authenticate Mica users.
obiba_bootstrap	theme	https://www.drupal.org/project/obiba_bootstrap	Bootstrap based Drupal theme with appropriate style sheets and page templates. Extension of bootstrap theme.

Once Drupal is installed on your system, run the following commands:

```
# Go to Drupal installation directory
cd DRUPAL_DIR

# Download and enable Obiba bootstrap theme
drush en -y bootstrap
drush en -y obiba_bootstrap

# Download and enable Obiba Mica module
drush en -y obiba_mica

# Download and enable Obiba Agate module
drush en -y obiba_agate

# Download and enable Obiba Mica Data Access module (optional)
drush en -y obiba_mica_data_access_request

# Download Obiba Javascript dependencies
drush download-mica-dependencies

# Generate the autoload composer dependencies
drush composer-json-rebuild
cd sites/default/files/composer/
composer update
composer dump-autoload -o
cd DRUPAL_DIR
# Choose option 9 (to clear registry cache)
drush cc registry

# Apply JQuery settings
drush vset -y --format=string jquery_update_jquery_version 1.10
drush vset -y --format=string jquery_update_jquery_admin_version 1.10

# Download and enable Autologout module (optional)
drush dl -y autologout
drush en -y autologout
drush vset -y autologout_redirect_url "<front>"
drush vset -y autologout_no_dialog TRUE
```

- Debian

```
# Apply some folder permissions
chown www-data:www-data ./sites/default/files/composer/
```

- CentOS

```
# Apply some folder permissions
chown apache\: ./sites/default/files/composer/
```

To enable the mode_rewrite on Debian:

```
sudo a2enmod rewrite
sudo service apache2 restart
```

On CentOS the rewrite_mode is enabled by default.

- Make sure that the apache config on Debian and CentOS allow overriding via .htaccess, to do so make sure the apache config file has the following directive:

```
<Directory "/var/www/html">
...
AllowOverride All
...
</Directory>
```

- Go to http://localhost/drupal/#overlay=admin/config/search/clean-urls
- Check "Enable clean URLs" and save.
- Due to an incompatibility with a nonvalid ssl certificate in CentOS, you need to set mica url and agate url without ssl. To do this :
  - Go to http://localhost/drupal/admin/config/obiba-agate/agate-settings
  - Replace Agate address with : http://localhost:8081
  - In Application Key, set : changeIt
  - Save
  - Go to http://localhost/drupal/admin/config/obiba-mica/obiba-mica-settings
  - Replace Mica address with : http://localhost:8082
  - Save

8.4 Upgrade

Before proceeding, make sure that the PHP version is 5.6 and Mica server version is >= 2.0.0.

The following instructions apply when upgrading from obiba_mica 7.x-1.3 or older.

```
# Go to Drupal installation directory
cd DRUPAL_DIR

# Upgrade Obiba modules
drush up obiba_mica
drush up obiba_bootstrap
drush up obiba_agate

# Install Obiba javascript dependencies
drush download-mica-dependencies
```

(continues on next page)

(continued from previous page)

```
# Replace the old search module with the new one
drush dis obiba_mica_search
drush en obiba_mica_repository

# Generate the autoload composer dependencies
drush composer-json-rebuild
cd sites/default/files/composer/
composer update
composer dump-autoload -o
cd DRUPAL_DIR
# Choose option 9 (to clear registry cache)
drush cc

# Install Obiba Agate module new dependency
drush en autologout

# Clear all caches
drush cc
```

If some templates have been overridden, please compare with the new original one.

If you have defined a sub-theme of obiba_bootstrap's theme, you might need to update your style sheet.

CHAPTER 9

Drupal Configuration

Drupal is turned into Mica Drupal Client via a set of Drupal modules that can be enabled/disabled in the Modules > OBiBa subsection of Drupal.

Note: If you decide to disable one of OBiBa Drupal module, make sure you know exactly what it does. As a general rule, all modules should be enabled in order to make Mica Client works. There are, however, two notable exceptions to this rule:

You may disable both the "Data Access Request" and the "OBiBa Auth" modules in the case you don't intend to use the Data Access Request feature provided by Mica. Not an OBiBa module per se, but one which Mica Client use extensively is the Google Chart module (in the Chart section). If you intend to use Highcharts in your portal, you may want to activate the module there and disable the Google Chart modules.

9.1 OBiBa Mica settings

Here, we will explain how to configure Mica's services. The sections enumerated here reflect the sections present in the section Configuration > OBiBa Mica settings of the administration panel.

9.1.1 OBiBa Study Server (MICA)

This subsection lists various fields that Mica Drupal Client uses to communicate with Mica Server. Here is a succinct description of each fields along with its name:

Field	Description
Mica address	The URL of Mica Server
Anonymous user name	The Anonymous user has read permission upon the content that has been published on Mica server. Here, you enter the name of the anonymous user as know by Mica Server.
Anonymous user password	Self-explanatory.
Copyright Notice Text	A copyright notice to be included if a user download a list of data.
Number of items per server response page	Determines the how many items that must be displayed in a server response page. For instance, this parameter affects the number of variables listed in a page.
Minimum number of items per server response page	Determines the minimum number of items that must to be displayed in a server response page. This parameter affects the number of studies, networks or datasets listed on a page.

9.1.2 Data Access Request

Either the name of a field is self-explanatory or the explanation located below that field is sufficient to understand what it is meant for except for the last item:

Access request commenting. If checked, data access request commenting is enabled. For a given Access Request form, there will be a comment tab aside the history tab. By checking on this option, the commenting area can be used for a discussion between the Data Access Officer (DAO) and the user who request access.

9.1.3 Statistics Settings

The explanation that lies below the checkboxes is self-explanatory.

9.1.4 Cache Image settings

The option for time image timeout is supposed to be clear. Now, you also have a button to clear the image cache. This might be useful as, for instance, logo of studies (or networks) don't tend to change much, so the image cache timeout tends to be long. If, however, you change an image, you can clear the cache right away.

9.1.5 Networks, Studies, Datasets and Variable Search

Depending on the purpose for which you intend to use Mica, you might want to deactivate the Networks (resp. Studies, Datasets or Variables) tab in the Search page. By deactivating the checkbox aside Show Networks (resp. Studies, Datasets or Variables) search, the Network (resp. Studies, Datasets or Variables) tab won't show up in the Search page.

Below each of these four configurations (Show Networks, Studies, Datasets or Variables search) are options to customize the result of a given search string entered in the left-hand-side column e.g., to show or not the studies in the results when one search for a network.

In **Datasets Search > Show dataset auto-complete search filter**. If selected, the auto-complete search filter will be displayed in the search page. By choosing "Checkbox" you have a checkbox selection. Finally, you can also dispable the display.

9.1.6 Study, Dataset and Variable Content

By clicking on a "specific" result on the Search page, that is, not a number of networks, variable, studies or dataset, you are brought on a page that describes that network, study, dataset or a variable. In the configuration panel, the options listed in the Study, Dataset or Variable Content boxes will set options concerning the display of information on a description page of that type.

9.1.7 Taxonomies

In this block, you may edit the appearance as well as the order of the taxonomies appearing in:

- **Figures**. This concerns the display (or its absence thereof) of all figures concerning Variable Classification e.g., the Area of information, the various constructs etc.
- **Search**. This concerns the display of the search panel (on the left) on the Search page under the Variable tab.

If the text area for **Taxonomy in Figure** is empty, it will display all taxonomies. This is the default state.

9.1.8 Translation

The last section is for translation of the web data portal created via Mica Drupal Client. The textarea concerns the pages that should not be translated. Suppose that your data web portal is translated in 2 languages (the primary language is English) and that a data access form for the data displayed therein is available only in English. Then, you can translate all the portal into the second language but not the pages related to data access. In order to do so, you need to enter the path of each of the page you don't want to be translated into the textarea separated by a coma and you're done: these pages will remain only in English.

9.2 Mica Drupal Client Templating

We will examine two distinct ways to do templating: with a sub-theme and with a custom module.

9.2.1 Dependencies

First of all, you need to get:

- Bootstrap theme
- OBiBa Bootstrap sub-theme

Further, see the Drupal Bootstrap Documentation.

9.2.2 Overriding templates via a new sub-theme

Overriding a template is useful if one wants to determine the way the information is displayed in a page and have a better control over the design. Thus, for every page to display in Mica Drupal Client, there is a file (or a set of) template file(s) located in the corresponding template repository of each OBiBa module.

It is not recommended to modify these files directly or the modifications will be overwritten the next time OBiBa Modules will be updated thus the idea of template overriding.

Note: The list of templates that we can override can be seen in the template.php file of obiba_bootstrap.

You may do template overriding as follow:

- First, create a sub-template as decribed in the documentations hyperlinked above
- Define obiba_bootstrap as the base theme in the .info file of that sub-theme.

Once the sub-theme is set, you can override the different vues generated by a module by copying the template file for that module in the template folder of that sub-theme, that is:

```
cp /site/all/modules/obiba_mica/<module to overide>/templates/<template to overide>
↪<drupal>/sites/all/themes/<Sub_theme_bootstrap>/templates
```

9.2.3 Overriding templates via custom module

If you want to use default template obiba_bootstrap, which entails making smaller edits to the design, you may override the templates in a custom module that you can install in your instance of Mica Drupal Client:

- Copy the template that you want to override in the folder "Template" of the custom module,
- Use the hook_theme() function to override the templates.

For instance, you can use the following in a .module file:

```
/*
* hook_theme()
*/
function MYMODULE_theme($existing, $type, $theme, $path){
 $theme = array();
      $theme['obiba_mica_dataset-detail'] = array(
        'template' => 'obiba_mica_dataset-detail',
        'path' => drupal_get_path('module', 'MYMODULE') . '/templates',
       );
      return $theme;
}
```

CHAPTER 10

Python Introduction

Mica Python client, a command line scripting tool written in Python, enables automation of tasks in a Mica server.

10.1 Requirements

Python 2.x must be installed on the system. See more about Python.

10.2 Installation

You can install Mica Python Client via the following two methods:

- use the Debian/RPM package manager
- use a Python package

10.2.1 Debian Package Installation

Follow the OBiBa Debian Repository instructions and run:

```
sudo apt-get install mica-python-client
```

10.2.2 RPM Package Installation

Follow the OBiBa RPM Repository instructions and run:

```
sudo yum install mica-python-client
```

10.2.3 Python Package Installation

This type of package is cross-platform (Linux, Windows, Mac).

Install on Linux or Mac

1. Download the most recent version
2. Decompress the file and enter the installation folder:

```
tar xvzf mica-python-client-X.XX.tar.gz
cd mica-python-client-X.XX
```

3. Install the package:

```
sudo python setup.py install --record installed_files.lst
```

Note: The *–record* will generate a list of installed files on your system. Since there is no uninstaller, you can use this file to remove the Mica Python Client package. You can do this by executing the following command: `sudo cat installed_files.lst | xargs rm -rf`

Install on Windows

- Using Cygwin

You can install Cygwin, making sure that CURL, Python, gcc are included and follow these steps inside a Cygwin BASH window:

```
cd /usr/lib
cp libcurl.dll.a libcurl.a
cd <your-desired-dir>
curl -C - -O http://download.obiba.org/mica/stable/mica-python-client-X.XX.tar.gz
tar xzvf mica-python-client-X.XX.tar.gz
cd mica-python-client-X.XX
python setup.py install --record installed_files.lst
```

- Using plain Windows tools

This Windows installation is the most complicated one but does not required any third party tools. You are required to do a few manual installations before the package is fully usable. The following steps were tested on a Windows 7.

1. You must have Python installed on your Windows system. Run this installer in case you don't have one.
2. Download the Google protobuf binary and make sure that its containing folder is in your path.
3. Download the Google protobuf source package containing the setup.py file and follow these steps:

```
unzip protobuf-2.5.0.zip
cd protobuf-2.5.0/python
python setup.py install
```

4. Go to the Python Libs site and download the file pycurl-7.19.0.win-amd64-py2.7.exe
5. Run the installer and follow the instructions until the package is installed
6. Download the most recent version and follow these steps:

```
unzip http://download.obiba.org/mica/stable/mica-python-client-X.XX.zip
cd mica-python-client-X.XX
python setup.py bdist_wininst
cd dist
```

7. Execute the generated installer and follow the instructions (mica-python-client-X.XX.win-amd64.exe)

10.3 Usage

To get the options of the command line:

```
mica --help
```

This command will display which sub-commands are available. Further, given a subcommand obtained from command above, its help message can be displayed via:

```
mica <subcommand> --help
```

This command will display available subcommands.

CHAPTER 11

Authorization Commands

Document authorization (on draft and published versions) management.

11.1 Document Access

This command is used to manage the access to a document. This access affects the **published** version and also applies to all associated files in their published version (unless the access to the files is explicitly excluded).

```
mica access-<DOCUMENT> ID <CREDENTIALS> [OPTIONS] [EXTRAS]
```

11.1.1 Arguments

Argument	Description
`DOCUMENT`	Mica document: `network`, `individual-study`, `harmonization-study`, `collected-dataset`, `harmonized-dataset` (see *Documents*)
`ID`	Identifier of the document

11.1.2 Options

Option	Description
`--add, -a`	Add an access
`--delete, -d`	Delete an access
`--no-file, -nf`	Do not grant access to associated files
`--subject, -s`	Subject name to which the access will be granted
`--type TYPE, -ty TYPE`	Subject type: `user` or `group`

11.1.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

11.1.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

11.1.5 Example

Network

Add access for the user demouser on the network demo:

```
mica access-network --mica http://mica-demo.obiba.org --user administrator --password password --type USER --subject demouser --add demo
```

Remove the above permission:

```
mica access-network --mica http://mica-demo.obiba.org --user administrator --password password --type USER --subject demouser --delete demo
```

Individual Study

Add access for the user demouser on the individual study demo:

```
mica access-individual-study --mica http://mica-demo.obiba.org --user administrator --password password --type USER --subject demouser --add demo
```

Remove the above permission:

```
mica access-individual-study --mica http://mica-demo.obiba.org --user administrator --password password --type USER --subject demouser --delete demo
```

11.2 File Access

This command is used to manage the access to a file in the Mica file system. This access affects the **published** version.

```
mica access-file PATH <CREDENTIALS> [OPTIONS] [EXTRAS]
```

11.2.1 Arguments

Argument	Description
`PATH`	Path to the file in the Mica file system

11.2.2 Options

Option	Description
`--add, -a`	Add an access
`--delete, -d`	Delete an access
`--subject, -s`	Subject name to which the access will be granted
`--type TYPE, -ty TYPE`	Subject type: `user` or `group`

11.2.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

11.2.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

11.2.5 Example

Add access for user demouser on demo individual-study files:

```
mica access-file /individual-study/demo --mica http://mica-demo.obiba.org --user␣
↪administrator --password password --type USER --subject demouser --add
```

Remove the above access:

```
mica access-file /individual-study/demo --mica http://mica-demo.obiba.org --user␣
↪administrator --password password --type USER --subject demouser --delete
```

11.3 Document Permission

This command is used to manage the permissions of a document. These permissions affects the **draft** version and apply to all associated files in their draft version.

```
mica perm-<DOCUMENT> ID <CREDENTIALS> [OPTIONS] [EXTRAS]
```

11.3.1 Arguments

Argument	Description
`DOCUMENT`	Mica document: `network`, `individual-study`, `harmonization-study`, `collected-dataset`, `harmonized-dataset` (see *Documents*)
`ID`	Identifier of the document

11.3.2 Options

Option	Description
`--add, -a`	Add a permission
`--delete, -d`	Delete a permission
`--permission, -pe`	Permission to apply: `reader`, `editor` or `reviewer`
`--subject, -s`	Subject name to which the access will be granted
`--type TYPE, -ty TYPE`	Subject type: `user` or `group`

11.3.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

11.3.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

11.3.5 Example

Network

Add reader permission for the user demouser on the network demo:

```
mica perm-network --mica http://mica-demo.obiba.org --user administrator --password␣
↪password --type USER --subject demouser --add --permission reader demo
```

Remove the above permission:

```
mica perm-network --mica http://mica-demo.obiba.org --user administrator --password␣
↪password --type USER --subject demouser --delete demo
```

Individual Study

Add reader permission for the user demouser on the individual study demo:

```
mica perm-individual-study --mica http://mica-demo.obiba.org --user administrator --
↪password password --type USER --subject demouser --add --permission reader demo
```

Remove the above permission:

```
mica perm-individual-study --mica http://mica-demo.obiba.org --user administrator --
↪password password --type USER --subject demouser --delete demo
```

CHAPTER 12

Document Commands

Document management, upload, download, import, publication, search etc.

12.1 Update Collected Dataset

This command is for updating and/or publishing an existing Collected Dataset. The goal is to automate the linkage between a table in Opal with a collected dataset in Mica.

```
mica update-collected-dataset ID <CREDENTIALS> [OPTIONS] [EXTRA]
```

12.1.1 Arguments

Argument	Description
`ID`	The collected dataset identifier

12.1.2 Options

Option	Description
`--study STUDY, -std STUDY`	The associated study.
`--population POP, -pop POP`	The population of the associated study.
`--dce DCE, -dce DCE`	The data collection event in the population of the associated study.
`--project PROJECT, -prj PROJECT`	The associated Opal project.
`--table TABLE, -tbl TABLE`	The table in the associated Opal project.
`--publish, -pu`	Publish the collected dataset.
`--unpublish, -un`	Unpublish the collected dataset.

12.1.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

12.1.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

12.1.5 Example

Link a collected dataset in local Mica to a table in Opal.

```
mica update-collected-dataset -u administrator -p password --project CLS --table
↪Wave1 cls-wave1
```

Associate a collected dataset to a study data collection event in Mica.

```
mica update-collected-dataset -u administrator -p password --study cls --population 1
↪--dce 1 cls-wave1
```

Publish a collected dataset.

```
mica update-collected-dataset -u administrator -p password --publish cls-wave1
```

12.2 Update Collected Datasets

This command is for updating and/or publishing a list Collected Datasets which are ID is filtered by a regular expression. The goal is to automate the linkage between a table in Opal with a collected dataset in Mica.

```
mica update-collected-datasets ID <CREDENTIALS> [OPTIONS] [EXTRA]
```

12.2.1 Arguments

Argument	Description
`ID`	A regular expression to filter the collected dataset identifiers.

12.2.2 Options

Option	Description
`--project PROJECT, -prj PROJECT`	The associated Opal project.
`--dry DRY, -d DRY`	Dry run of the command to list the collected datasets matching the regular expression.
`--publish, -pu`	Publish the collected datasets.
`--unpublish, -un`	Unpublish the collected datasets.

12.2.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

12.2.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

12.2.5 Example

Link the collected datasets which ID starts with 'cls-wave' in local Mica to a project in Opal and publish them.

```
mica update-collected-datasets -u administrator -p password --project CLS --publish '^
↪cls-wave'
```

12.3 File Management

This command is for advanced users wanting to directly access to the File System API of Mica server.

```
mica file PATH <CREDENTIALS> [OPTIONS] [EXTRA]
```

12.3.1 Arguments

Argument	Description
`PATH`	Path of file or folder in the file system, for instance: /study/foo

12.3.2 Options

Option	Description
`--download, -dl`	Download file.
`--upload UPLOAD, -up UPLOAD`	Upload a local file to a folder in Mica file system, requires the folder to be in DRAFT state. If the destination folder does not exist it will be created.
`--create CREATE, -c CREATE`	Create a folder at a specific location, requires the file to be in DRAFT state.
`--copy COPY, -cp COPY`	Copy a file to the specified destination folder.
`--move MOVE, -mv MOVE`	Move a file to the specified destination folder, requires the file to be in DRAFT state.
`--delete, -d`	Delete a file on Mica file system, requires the file to be in DELETED state.
`--name NAME, -n NAME`	Rename a file, requires the file to be in DRAFT state.
`--status STATUS, -st STATUS`	Change file status.
`--publish, -pu`	Publish a file, requires the file to be in UNDER_REVIEW state.
`--unpublish, -un`	Unpublish a file.

12.3.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

12.3.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

12.3.5 Example

Get the JSON representation of file /study/foo/bar.pdf

```
mica file /study/foo/bar.pdf -mk https://mica-demo.obiba.org -u administrator -p␣
↪password -j
```

Download file /study/foo/bar.pdf

```
mica file /study/foo/bar.pdf -mk https://mica-demo.obiba.org -u administrator -p␣
↪password --download > bar.pdf
```

Upload a file to /study/foo

```
mica file /study/foo -mk https://mica-demo.obiba.org -u administrator -p password --
↪upload ~/bar.pdf
```

Change status and publish file /study/foo/bar.pdf

```
mica file /study/foo/bar.pdf -mk https://mica-demo.obiba.org -u administrator -p␣
↪password --status UNDER_REVIEW
mica file /study/foo/bar.pdf -mk https://mica-demo.obiba.org -u administrator -p␣
↪password --publish
```

12.4 Search

This command allows to extract published information from the search API of Mica server. The output is in CSV format.

```
mica search <CREDENTIALS> [OPTIONS] [EXTRA]
```

12.4.1 Options

Option	Description
`--target TARGET, -t TARGET`	The type of document to be listed: variable, dataset, study, population, dce (data collection event) or network.
`--query QUERY, -q QUERY`	The search query, in RQL (Resource Query Language), that can be copied from the search page. If not specified, no filter is applied.
`--start START, -s START`	Start search at document position (default is 0).
`--limit LIMIT, -lm LIMIT`	Max number of documents to be listed (default is 100).
`--locale LOCALE, -lc LOCALE`	The language of the labels (default is 'en').
`--out OUT, -o OUT`	Output file path. If not specified, result is printed on the console.

12.4.2 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

12.4.3 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

12.4.4 Example

Get 1000 published variables.

```
mica search -mk https://mica-demo.obiba.org -u anonymous -p password --target␣
↪variable --limit 1000
```

Get 1000 (max) published variables about Alcohol from cohort studies:

```
mica search -mk https://mica-demo.obiba.org -u anonymous -p password --target␣
↪variable --limit 1000 --query 'variable(in(Mlstr_area.Lifestyle_behaviours,
↪(Alcohol))),study(in(Mica_study.methods-design,cohort_study))'
```

Get the cohort studies having collected data about Alcohol:

```
mica search -mk https://mica-demo.obiba.org -u anonymous -p password --target study --
↪query 'variable(in(Mlstr_area.Lifestyle_behaviours,(Alcohol))),study(in(Mica_study.
↪methods-design,cohort_study))'
```

12.5 Import Zip

This command allows to import a zip-archived file produced by Mica. The result of the import will be the creation or the update of the packaged documents and their attachments.

```
mica import-zip  <CREDENTIALS> [EXTRA] PATH
```

A very useful usage of this command is when a series of associated documents should be imported together. For instance, this command permits to import an individual-study, its network and all its associated collected-datasets. Here is how the documents should be organized into sub-folders and archived such that the import command recognizes it as a valid input:

```
- study
  - individual-study-name
    - network-something.json
    - collected-dataset1.json
    - collected-dataset2.json
    - collected-dataset3.json
    - individual-study-name.json
    - attachments
      - attachment-id1
      - attachment-id2
```

Note: attachment-id is the ID used in the document attachments list in the JSON file, this should not be the filename.

Warning: Use this command with special care to prevent overriding existing documents and breaking associations.

12.5.1 Arguments

Argument	Description
`PATH`	Path to the zip file or directory that contains zip files to be imported.

12.5.2 Options

Option	Description
`--add, -a`	Add an access
`--delete, -d`	Delete an access
`--no-file, -nf`	Do not grant access to associated files
`--subject, -s`	Subject name to which the access will be granted
`--type TYPE, -ty TYPE`	Subject type: `user` or `group`

12.5.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

12.5.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

12.5.5 Example

Import the file import.zip in Mica server running on localhost with user administrator.

```
mica import-zip -mk https://localhost:8445 -u administrator -p password  /path/to/the/
↪file/import.zip
```

Import all the zip files located in a directory with user editor.

```
mica import-zip -mk https://localhost:8445 -u editor -p password  /path/to/the/zips/
↪directory
```

CHAPTER 13

Other Commands

Other commands for advanced users.

13.1 Web Services

This command is for advanced users wanting to directly access to the REST API of Mica server.

```
mica rest ws <CREDENTIALS> [OPTIONS] [EXTRA]
```

13.1.1 Arguments

Argument	Description
`ws`	Web service path, for instance: /user/xxx

13.1.2 Options

Option	Description
`--method METHOD, -m METHOD`	HTTP method: GET (default), 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.

13.1.3 Credentials

Authentication is done by username/password credentials.

Option	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

13.1.4 Extras

Option	Description
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose output

13.1.5 Example

Get all the published studies visible to an anonymous user.

```
mica rest /studies -m GET -mk https://mica-demo.obiba.org -u anonymous -p password -a application/json -j
```

Add a new individual study document:

```
mica rest /draft/individual-studies -m POST -u administrator -p password -mk https://mica-demo.obiba.org -a application/json < patate-study.json
```

Search all files of the draft version of a network:

```
mica rest /draft/files-search/network/some-network -m GET -mk https://mica-demo.obiba.org -u administrator -p password -a application/json -j
```

CHAPTER 14

Partners and Funders

The development of this application was made possible thanks to the support of our partners and funders:

mælstrom	Epigeny	canarie	EUCAN connect

CHAPTER 15

Support

Please visit OBiBa support page.