Mica Documentation

OBiBa

Jun 01, 2021

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. 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

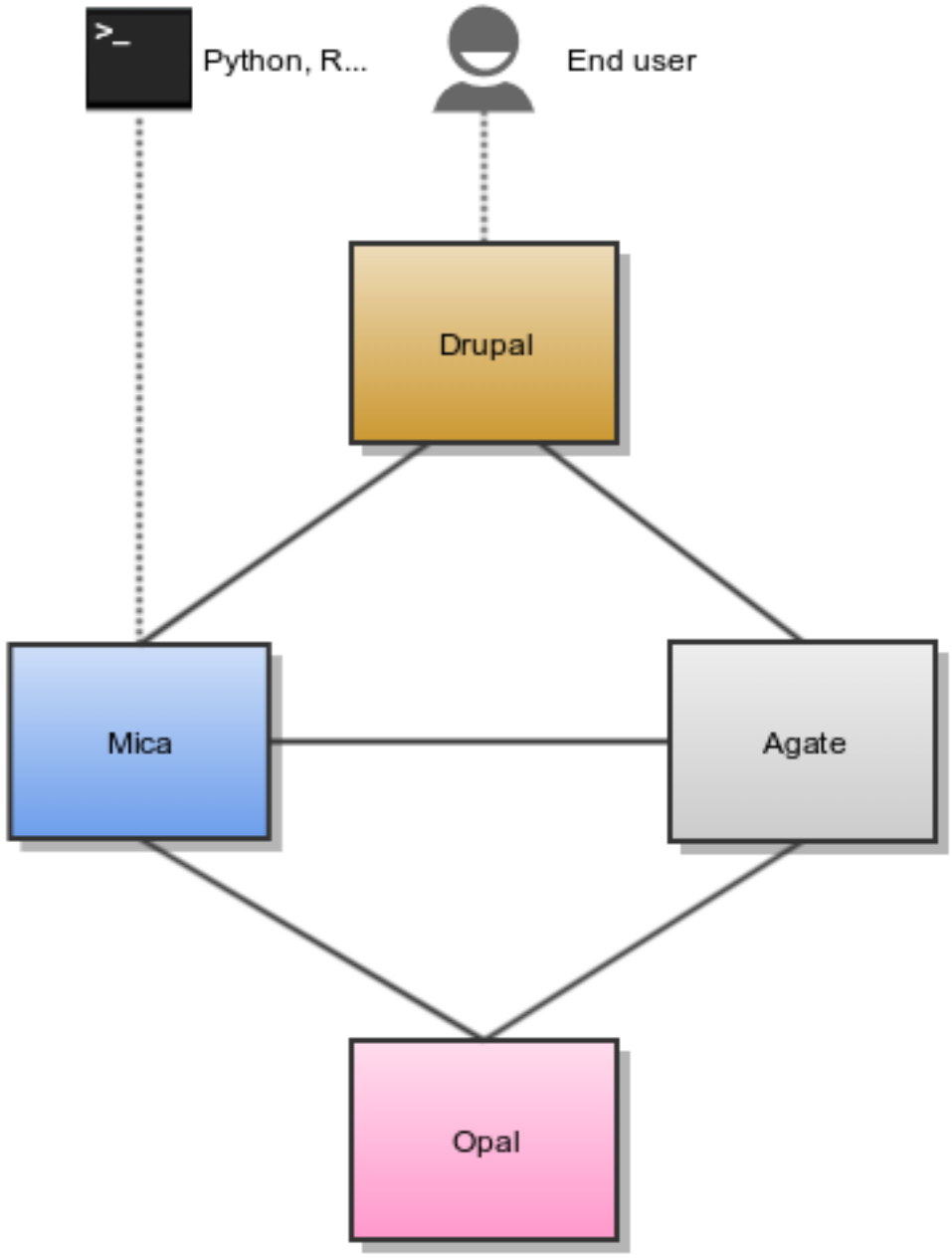
Introduction

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 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.

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.

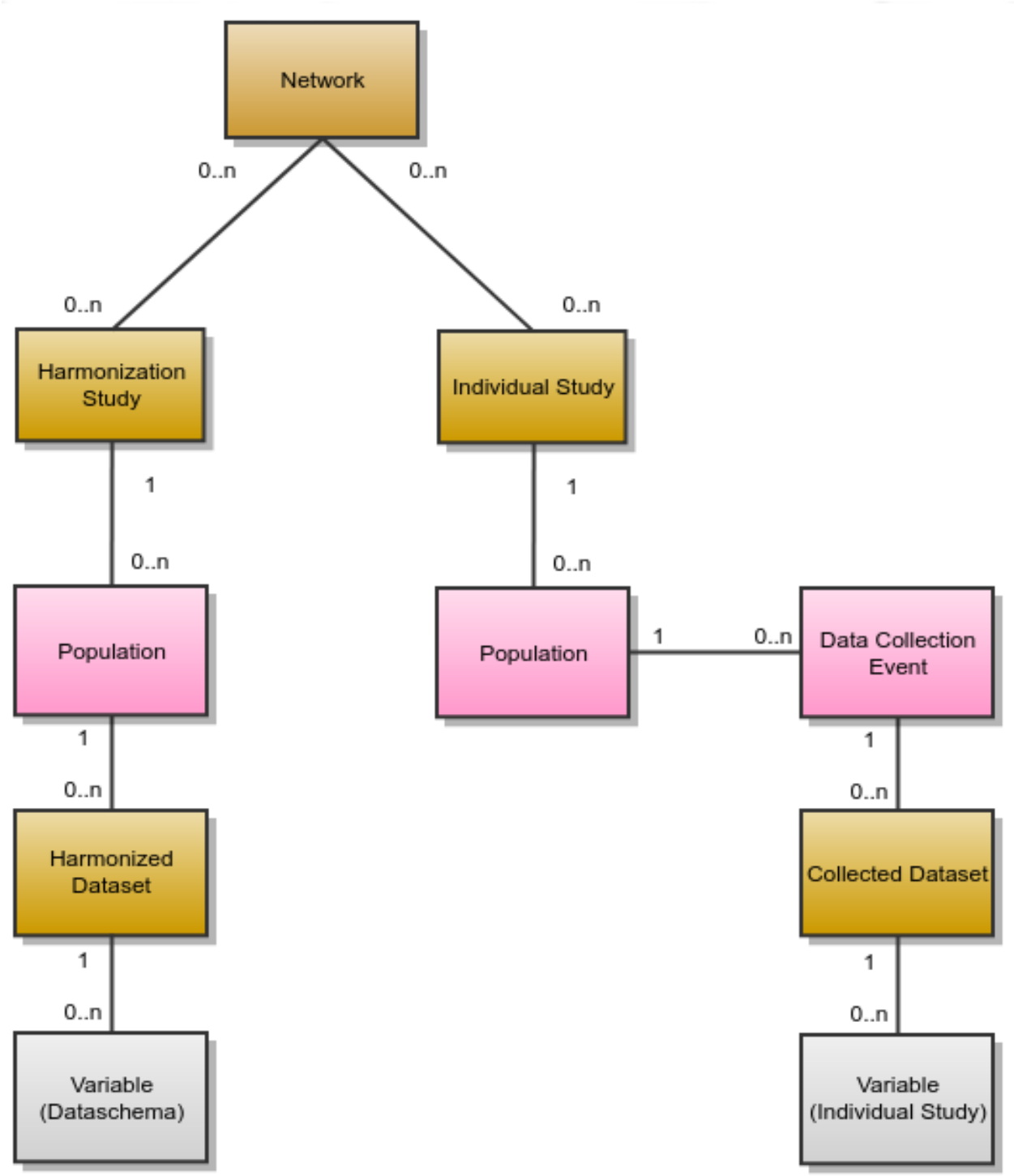
2.2 Search

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

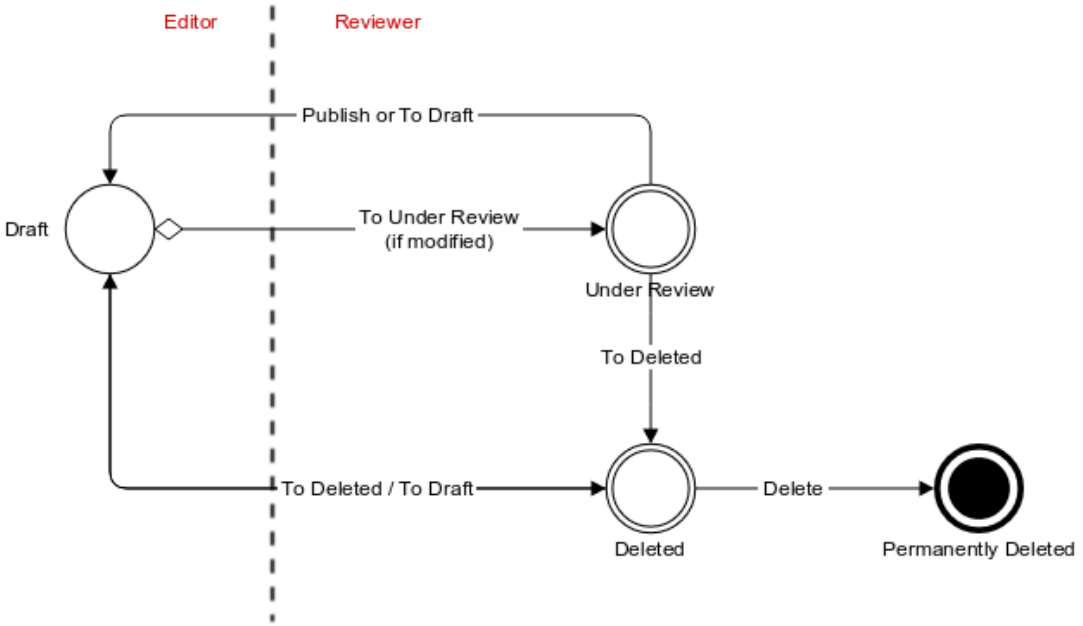
To enhance the collaboration between Mica users, each member can add a comment on any Mica domain document as well as data access requests documents. Mica can be configured to send email notifications when a comment is added or updated.

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	• 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	8	OpenJDK 8 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.

4.2.4 Docker Image Installation

OBiBa is an early adopter of the Docker technology, providing its own images from the Docker Hub repository.

A typical docker-compose file (including a MongoDB database) would be:

```
version: '3'
services:
      mica:
               image: obiba/mica
               ports:
                        - "8835:8445"
                        - "8872:8082"
               links:
```

(continues on next page)

(continued from previous page)

```
                - mongo
        environment:
                - JAVA_OPTS=-Xmx2G
                - MICA_ADMINISTRATOR_PASSWORD=password
                - MICA_ANONYMOUS_PASSWORD=password
                - MONGO_HOST=mongo
                - MONGO_PORT=27017
                - OPAL_HOST=opal
                - OPAL_PORT=8443
                - AGATE_HOST=agate
                - AGATE_PORT=8444
        volumes:
                - /opt/mica:/srv
mongo:
        image: mongo
opal:
        image: obiba/opal
        ports:
                - "8833:8443"
                - "8870:8080"
        links:
                - mongo
        environment:
                - OPAL_ADMINISTRATOR_PASSWORD=password
                - MONGO_HOST=mongo
                - MONGO_PORT=27017
                - AGATE_HOST=agate
                - AGATE_PORT=8444
        volumes:
                - /opt/opal:/srv
agate:
        image: obiba/agate
        ports:
                - "8834:8444"
                - "8871:8081"
        links:
                - mongo
        environment:
                - AGATE_ADMINISTRATOR_PASSWORD=password
                - MONGO_HOST=mongo
                - MONGO_PORT=27017
                - RECAPTCHA_SITE_KEY=6Lfo7gYTAAAAAOyl8_MHuH-AVBzRDtpIuJrjL3Pb
                - RECAPTCHA_SECRET_KEY=6Lfo7gYTAAAAADym-vSDvPBeBCXaxIprA0QXLk_b
        volumes:
                - /opt/agate:/srv
```

Then environment variables that are exposed by this image are:

Environment Variable	Description
JAVA_OPTS	
MICA_ADMINISTRATOR_PASSWORD	Agate administrator password, required and set at first start.
MICA_ANONYMOUS_PASSWORD	Agate anonymous password, required and set at first start.
MONGO_HOST	MongoDB server host.
MONGO_PORT	MongoDB server port, default is 27017.
OPAL_HOST	Opal server host (optional).
OPAL_PORT	Opal server port, default is 8443.
AGATE_HOST	Agate server host.
AGATE_PORT	Agate server port, default is 8444.

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.
`server.context-path`	The URL's context path, starting with a /. For instance when setting */repo*, the base URL will be *https://example.org/repo*.
`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.
`opal.token`	Opal personal access token for connection to Opal server. If defined it has the priority over the username/password credentials.

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 Agate Server Configuration

Mica server uses Agate as a user directory and as a notification emails service. From the Agate point of view, Mica is not a user: it is an application. Each time Mica needs a service from Agate, 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`	Agate server URL. It is highly recommended to use https protocol.
`agate.application.name`	Mica application ID for connection to Agate server.
`agate.application.key`	Mica application key for connection to Agate 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.maxConcurrentJoinQueries`	Maximum count of ES queries that can be executed concurrently. Default value is `4`.
`elasticsearch.concurrentJoinQueriesWaitTimeout`	Wait timeout when executing concurrent ES queries in millis. Default value is `30000` milliseconds.

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 Miscelaneous Configuration

Advanced settings.

Property	Description
`dar.reminder.cron`	Schedule (cron syntax) of the email reminder for Data Access report. Default value is `0 0 0 * * ?` (every day at midnight).
`sets.cleanup.cron`	Schedule (cron syntax) of the expired cart/sets cleanup. Default value is `0 0 * * * ?` (every hour).

5.8 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
```

(continues on next page)

(continued from previous page)

```
# 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
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
```

5.9 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>
```

(continues on next page)

(continued from previous page)

```
<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
    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

Public Pages Configuration

Starting from Mica 4.0, the administration user interface is distinct from the public pages, i.e. pages that are to be accessed by regular users. These pages are based on templates that can be customized, extended or overridden. The template engine that is used is FreeMarker which has a clean and powerful syntax.

6.1 Page Templates

6.1.1 Configuring Pages

The main public pages are:

Page	Description
index	The home page
profile	The user profile page for updating personal information and password
signin	The login page
signup	The user registration page
signup-with	The user registration page, with form pre-filled with personal information extracted from a OpenID Connect server
forgot-password	The page to ask for password reset
just-registered	The welcome page after a user has registered
networks	The list of networks
network	The network page
studies	The list of studies
study	The study page (can be individual or harmonization)
datasets	The list of datasets
dataset	The dataset page (can be collection or harmonization)
variable	The variable page (can be collected, data schema or harmonized)
search	The catalog search page
projects	The list of approved projects
project	The approved project page
data-access-process	The data access process presentation page
data-accesses	The list of data access requests (restricted access)
data-access	The data access request main page (there are other pages for each of the data access request forms and features)
contact	The "Contact Us" form to send a contact request to the administrators or data access officers
cart	The variables cart page

The templates structure is organized in a way that it should not be necessary to override these main pages definitions. Instead of that, it is recommended to change/extend the theme/style as described in this guide.

The process of configuring these pages is the following (by order of priority):

1. Alter default settings

Some template variables (date formats, branding, favicon etc.) are defined in libs/settings.ftl and can be altered in the file **models/settings.ftl** that would be added in your configuration folder as follows:

```
MICA_HOME
└── conf
    └── templates
        └── models
            └── settings.ftl
```

If enough, this is the less intrusive approach. Note that you do not need to redefine all the settings, just reassign the ones of interest.

General settings

Variable	Description
defaultLang	The default language to be used when extraction labels from documents. If no text version is found for the page's language, this default language's version will be looked up.
datetimeFormat	The format in which the date-time values should be rendered.
date	The format in which the date values should be rendered.
faviconPath	The location of the favicon, to be modified to match your own.
brandImagePath	The location of your organization's logo.
brandImageCss	CSS classes to apply to the logo.
brandTextEnabled	Logical to show/hide a text aside of the logo.
brandTextCss	CSS classes to apply to the text aside of the logo.
networkIconCss	CSS classes to apply to the <i> element to render the icon that represents a network.
studyIconCss	CSS classes to apply to the <i> element to render the icon that represents a study.
datasetIconCss	CSS classes to apply to the <i> element to render the icon that represents a dataset.
harmoDatasetIconCss	CSS classes to apply to the <i> element to render the icon that represents a harmonization dataset.
variableIconCss	CSS classes to apply to the <i> element to render the icon that represents a variable.
projectIconCss	CSS classes to apply to the <i> element to render the icon that represents a project.
taxonomyIconCss	CSS classes to apply to the <i> element to render the icon that represents a taxonomy.
adminLTEPath	The location of the AdminLTE theme if this one has been modified (see the **Theme** section in this documentation).

Home page settings

Variable	Description
networksLink	The link to list the networks. Default is the **Networks** menu.
studiesLink	The link to list the studies. Default is the **Studies** menu.
datasetsLink	The link to list the datasets. Default is the **Datasets** menu.
portalLink	The link applied to the logo. Default is the data portal itself (same as **Home** menu), but it could also be the organization's main portal.

Cart page settings

Variable	Description
cartEnabled	Logical to show/hide the cart links (**Cart** menu, addition/removal to/from cart buttons). Default is consistent with the application's general configuration, but can be fine-tuned to make the cart visible to users within roles or groups.
listsEnabled	Logical to show/hide the lists links (**Lists** menu, addition to list buttons). Default is consistent with the application's general configuration, but can be fine-tuned to make the lists visible to users within roles or groups.
showCartDownload	Logical to allow downloading the content of the cart. Default is restricted to users with administration-related role.
showCartOpalDownload	Logical to allow downloading the content of the cart in the format of Opal views (for creating views in Opal from a variable selection). Default is restricted to users with administration-related role.

Contact Us page settings

Variable	Description
`contactEnabled`	Logical to show/hide the **Contact** menu. Default is **true**, but can be restricted to users within roles or groups.

User Profile page settings

Variable	Description
`showProfileRole`	Logical to show/hide the role to which the user belongs.
`showProfileGroups`	Logical to show/hide the groups to which the user belongs.

Repository list pages settings

Variable	Description
`listDisplays`	Enumerate the different ways of rendering the lists of documents (networks, studies or datasets). Possible values are **lines**, **table** and **cards**. Some can be omitted (at least one is required) and the order matters.
`listDefaultDisplay`	Default display of a list of documents (networks, studies or datasets). Default is **lines**.
`networkListDisplays`	Specific enumeration of the different ways of rendering the lists of networks. Default is the same as specified by `listDisplay`.
`networkListDefaultDisplay`	Default display of a list of the networks. Default is the same as specified by `listDefaultDisplay`.
`studyListDisplays`	Specific enumeration of the different ways of rendering the lists of networks. Default is the same as specified by `listDisplay`.
`studyListDefaultDisplay`	Default display of a list of the studies. Default is the same as specified by `listDefaultDisplay`.
`datasetListDisplays`	Specific enumeration of the different ways of rendering the lists of networks. Default is the same as specified by `listDisplay`.
`datasetListDefaultDisplay`	Default display of a list of the studies. Default is **cards**.

Search page settings

Variable	Description
defaultSearchStatus	The start state of the Search interface when entering the page. Default is showing the list of studies or the list of variables when there is only one study.
downloadQueryEnabled	Logical to show/hide the button for downloading the results of the query. Default is **true**, but can be restricted to users within roles or groups.
showCopyQuery	Logical to show/hide the button for copying the query string, that can be used in the R or Python API. Default is restricted to users with administration-related role.
mapName	Map name to be used in the graphic **geographical-distribution-chart**. Default is **world**, possible values are **world**, **europe**, **north-america**, **south-america**, **asia**, **africa** or **oceania**.
searchCharts	Show/hide and order the graphics by specifying their name. Possible values are **geographical-distribution-chart**, **study-design-chart**, **number-participants-chart**, **bio-samples-chart** or **study-start-year-chart**.
searchVariablesEnabled	Logical to show/hide the list of variables resulting from the search. Default is consistent with the application's general configuration.
searchDatasetsEnabled	Logical to show/hide the list of datasets resulting from the search. Default is consistent with the application's general configuration.
searchStudiesEnabled	Logical to show/hide the list of studies resulting from the search. Default is consistent with the application's general configuration.
searchNetworksEnabled	Logical to show/hide the list of networks resulting from the search. Default is consistent with the application's general configuration.
searchVariableColumns	Show/hide and order the column names for the list of variables. Possible values are **label**, **label+description** (variable label with a tooltip that shows the description), **valueType**, **annotations**, **type**, **study**, **population**, **data-collection-event/dce** or **dataset**.
searchDatasetColumns	Show/hide and order the column names for the list of datasets. Possible values are **name**, **type**, **networks**, **studies** or **variables**.
searchStudyColumns	Show/hide and order the column names for the list of studies. Possible values are **name**, **type**, **study-design**, **data-sources-available**, **participants**, **networks**, **individual** or **harmonization**.
searchNetworkColumns	Show/hide and order the column names for the list of networks. Possible values are **name**, **studies**, **datasets** or **variables**.
searchVariableFields	List of the variable fields to be extracted from search results.
searchDatasetFields	List of the dataset fields to be extracted from search results.
searchStudyFields	List of the study fields to be extracted from search results.
searchNetworkFields	List of the network fields to be extracted from search results.
searchVariableSortFields	List of the variable fields to be used for sorting the search. Default is to sort by study, dataset, index (i.e. order in the dataset's data dictionary) and name.
searchDatasetSortFields	List of the dataset fields to be used for sorting the search. Default is to sort by study, population, data collection event and acronym.
searchStudySortFields	List of the study fields to be used for sorting the search. Default is to sort by acronym.
searchNetworkSortFields	List of the network fields to be used for sorting the search. Default is to sort by acronym.
searchCoverageEnabled	Logical to show/hide the **Coverage** search results tab.
searchGraphicsEnabled	Logical to show/hide the **Graphics** search results tab.
searchListEnabled	Logical to show/hide the **List** search results tab.
searchCriteria	Show/hide the search criteria in the sidebar by specifying their type (possible values are **variable**, **dataset**, **study**, **network**).

Variable page settings

Variable	Description
showHarmonizationSummary	For a dataschema variable, allow the possibility to display the summary statistics of a specific harmonized variable. Default is **true**.

Data Access pages settings

Variable	Description
dataAccessShowInstructionsPanel	Show/hide the instructions panel on the side of the data access form. Default is **true**.
dataAccessShowCollectionPanels	Show/hide the collection panels on the head of the data access pages. Default is **true**.
dataAccessShowTimeline	Show/hide the request timeline in the dashboard page when the data access is approved. Applies only when a project end date can be found. Default is **true**.
dataAccessShowArchived	Show/hide the **Archived** button, to users with appropriate permissions and when the data access request is completed. Default is **true**.

Charts settings

Variable	Description
barChartBackgroundColor	Background color of the chart elements (the bars or the countries for instance).
barChartBorderColor	Border color of the chart elements.
colors	List of colors to be used for a set of chart elements (portions of a pie chart for instance).

Files settings

Variable	Description
showFiles	Logical to show/hide the files that are associated to the documents (networks, studies, populations, data collection events, datasets). Default is **true**, but can be restricted to users within roles or groups. Note that the files can themselves require permissions.
showNetworkFiles	Logical to show/hide the files that are associated to the networks. Default is the same as what specified by `showFiles`.
showStudyFiles	Logical to show/hide the files that are associated to the studies. Default is the same as what specified by `showFiles`.
showStudyPopulationFiles	Logical to show/hide the files that are associated to the study populations. Default is the same as what specified by `showFiles`.
showStudyDceFiles	Logical to show/hide the files that are associated to the study data collection events. Default is the same as what specified by `showFiles`.
showDatasetFiles	Logical to show/hide the files that are associated to the datasets. Default is the same as what specified by `showFiles`.

Variables classifications charts settings

Variable	Description
`variablesClassificationsTaxonomies`	Enumerate the taxonomy names to render the charts of variables classifications coverage (count of variables annotated with each vocabulary). Default is **Mlstr_area**. If the list is empty, no chart will be displayed.
`networkVariablesClassificationsTaxonomies`	Enumerate the taxonomy names to render the charts of variables classifications coverage in the network page. Default value is `variablesClassificationsTaxonomies`.
`studyVariablesClassificationsTaxonomies`	Enumerate the taxonomy names to render the charts of variables classifications coverage in the study page. Default value is `variablesClassificationsTaxonomies`.
`datasetVariablesClassificationsTaxonomies`	Enumerate the taxonomy names to render the charts of variables classifications coverage in the dataset page. Default value is `variablesClassificationsTaxonomies`.

2. Override one of the page model templates

The model templates are to be found in the models folder. This allows to alter some portions of the pages, without affecting the general layout.

The override of the template is done by installing a file with same name, at the same relative location in the application's configuration folder.

```
MICA_HOME
└── conf
    └── templates
        └── models
            └── <template name>.ftl
```

This is the preferred approach when a document's model was modified (new fields added/removed to the network, study, dataset etc.).

3. Override the main page templates

These templates are located at the templates' root folder. This gives full control of the page content but may ignore enhancements or break when upgrading the application.

The override of the template is done by installing a file with same name, at the same relative location in the application's configuration folder.

```
MICA_HOME
└── conf
    └── templates
        └── <template name>.ftl
```

6.1.2 Adding Pages

It is possible to add new pages, for providing additional information or guidance to the regular user. This can be done as follows:

- Install a new page templates
- Add a new menu entry

1. Install custom page template

The new template page is to be declared in the configuration folder:

```
MICA_HOME
└── conf
    └── templates
        └── custom.ftl
```

You can check at the provided templates to make your template fit in the site theme and structure. The profile page template could be a good starting point.

FreeMarker will look at its context to resolve variable values. For a custom page the objects available in the context are:

Object	Description
`config`	The Mica configuration
`user`	The user object (if user is logged in)
`roles`	The list of user roles: `mica-administrator`, `mica-reviewer`, `mica-editor`, `mica-data-access-officer` or `mica-user` (if user is logged in)
`query`	The URL query parameters as a map of strings

This custom template page can load any CSS or JS file that might be useful. These files can be served directly by adding them as follows (there are no restrictions regarding the naming and the structure of these files, as soon as they are located in the **static** folder):

```
MICA_HOME
└── conf
    └── static
        ├── custom.css
        └── custom.js
```

The URL of this custom page will be for instance: `https://mica.example.org/page/custom`.

2. Custom menu entry

To link to a custom page (or an external page), some templates can be defined to extend the default menus: left menu can be extended on its right and right menu can be extended on its left. The corresponding templates are:

```
MICA_HOME
└── conf
    └── templates
        └── models
            ├── navbar-menus-left.ftl
            └── navbar-menus-right.ftl
```

Check at the default left and right menus implementation as a reference.

6.2 Theme and Style

6.2.1 Theme

The default theme is the one provided by the excellent AdminLTE framework. It is based on Bootstrap and JQuery. In order to overwrite this default theme, the procedure is the following:

- Build a custom AdminLTE distribution
- Install this custom distribution
- Change the template settings so that pages refer to this custom distribution instead of the default one

1. Build custom AdminLTE

This requires some knowledge in CSS development in a Node.js environment:

- Download AdminLTE source (source code or a released version)
- Reconfigure Sass variables
- Rebuild AdminLTE (see instructions in the README file, contributions section)

2. Install custom AdminLTE

The objective is to have the web server to serve this new set of stylesheet and javascript files. This is achieved by creating the folder **MICA_HOME/conf/static** and copying the AdminLTE custom distribution in that folder. Not all the AdminLTE are needed, only the **dist** and **plugins** ones. The folder tree will look like:

```
MICA_HOME
└── conf
    └── static
        └── admin-lte
            ├── dist
            └── plugins
```

3. Template settings

Now that the custom AdminLTE distribution is installed in the web server environment, this new location must be declared in the page templates. The default templates settings are defined in the libs/settings.ftl template file. See the **adminLTEPath** variable. This variable can be altered by defining a custom **settings.ftl** file as follows:

```
MICA_HOME
└── conf
    └── templates
        └── models
            └── settings.ftl
```

In this custom **settings.ftl** file the new AdminLTE distribution location will be declared:

```
adminLTEPath = "/admin-lte"/>
```

6.2.2 Style

As an alternative to theming, it is also possible to alter the style of the pages by loading your own stylesheet and tweaking the pages' layout using javascript (and JQuery). The procedure is the following:

- Install custom CSS and/or JS files
- Custom the templates to include these new CSS and/or JS assets

1. Install custom CSS/JS

The objective is to have the web server to serve this new set of stylesheet and javascript files. This is achieved by creating the folder **MICA_HOME/conf/static** and copying any CSS/JS files that will be included in the template pages. The folder tree will look like:

```
MICA_HOME
└── conf
    └── static
        ├── custom.css
        └── custom.js
```

2. Custom templates

For the CSS files, the **models/head.ftl** template allows to extend the HTML pages "head" tag content with custom content. For the JS files, the **models/scripts.ftl** template allows to extend the HTML pages "script" tags. The folder tree will look like:

```
MICA_HOME
└── conf
    └── templates
        └── models
            ├── head.ftl
            └── scripts.ftl
```

Where the **head.ftl** template will be:

```
<link rel="stylesheet" href="/custom.css"/>
```

And the **scripts.ftl** template will be:

```
<script src="/custom.js"/>
```

6.3 Translations

The translations are performed in the following order, for a given `locale`:

1. check for the message key in the messages_<locale>.properties (at different locations)
2. check for the message key in the <locale> JSON object as defined the **Administration > Translations** section of the administration interface

For the messages_* properties, the translations can be added/overridden as follows:

```
MICA_HOME
└── conf
    └── translations
        ├── messages_fr.properties
        └── messages_en.properties
```

Note that you can declare only the messages_* properties files that are relevant (locales available from the website) and the content of these files can contain only the translation keys that you want to override.

CHAPTER 7

Plugins

7.1 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

7.2 Installation

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

7.2.1 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.

7.2.2 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.

7.3 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.

7.4 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 8

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.

8.1 Requirements

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

CHAPTER 9

Python Introduction

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

9.1 Requirements

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

9.2 Installation

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

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

9.2.1 Debian Package Installation

Follow the OBiBa Debian Repository instructions and run:

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

9.2.2 RPM Package Installation

Follow the OBiBa RPM Repository instructions and run:

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

9.2.3 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)

9.3 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 10

Authorization Commands

Document authorization (on draft and published versions) management.

10.1 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]
```

10.1.1 Arguments

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

10.1.2 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`

10.1.3 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.

10.1.4 Extras

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

10.1.5 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
```

10.2 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]
```

10.2.1 Arguments

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

10.2.2 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`

10.2.3 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.

10.2.4 Extras

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

10.2.5 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
```

10.3 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]
```

10.3.1 Arguments

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

10.3.2 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`

10.3.3 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.

10.3.4 Extras

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

10.3.5 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 11

Document Commands

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

11.1 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]
```

11.1.1 Arguments

Argument	Description
ID	The collected dataset identifier

11.1.2 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.

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

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
```

11.2 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]
```

11.2.1 Arguments

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

11.2.2 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.

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

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'
```

11.3 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]
```

11.3.1 Arguments

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

11.3.2 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.

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

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
```

11.4 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]
```

11.4.1 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.

11.4.2 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.

11.4.3 Extras

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

11.4.4 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))'
```

11.5 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.

11.5.1 Arguments

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

11.5.2 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`

11.5.3 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.

11.5.4 Extras

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

11.5.5 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 12

Other Commands

Other commands for advanced users.

12.1 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]
```

12.1.1 Arguments

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

12.1.2 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.

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

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 13

REST API Introduction

The REST API allows to do all necessary operations for managing *Documents* and handling *Publication Flow*. As Mica manages both `draft` and `published` states of each document, there are distinct REST entry points, requiring different level of permissions.

13.1 Authentication

Mica supports Basic authentication, which consists of a HTTP request's header field in the form of `Authorization: Basic <credentials>`, where credentials is the Base64 encoding of user's (or application's) ID and password joined by a single colon `:`.

Using cURL, it is as simple as providing the –user option:

```
curl --user <id>:<password> [...]
```

13.2 Authorization

Authorizations are role based. The built-in roles are:

Role	Description
`mica-administrator`	Can edit/publish data and change system configuration.
`mica-reviewer`	Can edit draft data and publish them.
`mica-editor`	Can edit data draft data.
`mica-data-access-officer`	Can manage data access requests.
`mica-user`	Can view published data.

13.3 Clients

Usage examples provided are based on cURL, a command line client, the Python command line tool (see *Python Introduction*) and the R package micar (published content only).

CHAPTER 14

Draft Networks

Manage operations on the list of networks in draft mode.

14.1 List

GET /draft/networks

List a summary of the networks in draft mode.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role, otherwise an empty list is returned.

Example requests

Using cURL

```
curl --user administrator:password https://mica-demo.obiba.org/ws/draft/networks
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪GET /draft/networks --json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json

[
  {
    "id": "bioshare-eu",
    "acronym": [
      {
        "lang": "en",
        "value": "BioSHaRE-EU"
```

(continues on next page)

(continued from previous page)


```
      }
    ],
    "name": [
      {
        "lang": "en",
        "value": "Biobank Standardisation and Harmonisation for Research␣
↪Excellence in the European Union"
      }
    ],
    "studyIds": [
      "finrisk-2007",
      "ship",
      "lifelines"
    ],
    "permissions": {
      "delete": true,
      "edit": true,
      "publish": true,
      "view": true
    },
    "published": true,
    "timestamps": {
      "created": "2021-04-12T06:38:00.904Z",
      "lastUpdate": "2021-04-12T06:38:04.547Z"
    },
    "obiba.mica.EntityStateDto.networkSummaryState": {
      "permissions": {
        "delete": true,
        "edit": true,
        "publish": true,
        "view": true
      },
      "publicationDate": "2021-04-12T06:38:04.571Z",
      "publishedBy": "administrator",
      "publishedId": "57c9b73f7eec70a4ea471e96c741ddd4c41737e9",
      "publishedTag": "3",
      "revisionStatus": "DRAFT",
      "revisionsAhead": 0
    }
  }
]
```


Query Parameters

- **study** (*string*) – List networks linked to the study with provided identifier.
- **query** (*string*) – RQL search query.
- **from** (*numeric*) – List offset. Default value is `0`.
- **limit** (*numeric*) – List maximum count of documents.
- **sort** (*string*) – Sorting field. Default value is `id`.
- **order** (*string*) – Sorting order. Default value is `asc`.
- **exclude** (*strings*) – Document identifiers to be excluded from the list.
- **filter** (*string*) – Document state filter which possible values are: `ALL`, `PUBLISHED`, `UNDER_REVIEW`, `IN_EDITION`, `TO_DELETE`. Default value is `ALL`.

Response JSON Array of Objects

- **id** (*string*) – The document unique identifier.
- **acronym** (*object*) – The acronym (short name) of the document, as an object that describes a localized string.
- **name** (*object*) – The name of the document, as an object that describes a localized string.
- **studyIds** (*strings*) – The identifiers of the studies that are part of the network.
- **permissions** (*object*) – The different actions that can be performed on this document.
- **published** (*boolean*) – Whether the document is published.
- **timestamps** (*object*) – The date times (format ISO-8601) at which the document was created and updated.
- **obiba.mica.EntityStateDto.networkSummaryState** (*object*) – The publication state of the document.

Request Headers

- Authorization – As described in the *Authentication* section
- Accept – `*/*`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – The documents list to which user has read access rights.
- 500 Internal Server Error – Server error.

14.2 Create

POST /draft/networks

Create a network in draft mode.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role.

14.3 Index

PUT /draft/networks/_index

Rebuild both draft and published indices for all networks.

This entry point requires *Authentication* of a user with `mica-administrator` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT https://mica-demo.obiba.org/ws/draft/
↪networks/_index
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT /draft/networks/_index --json
```

Request Headers

- Authorization – As described in the *Authentication* section

Status Codes

- 200 OK – The indices rebuild task is scheduled.
- 401 Unauthorized – User does not have the permission to perform this operation.
- 500 Internal Server Error – Server error.

CHAPTER 15

Draft Network

Manage operation on a single network in draft mode.

15.1 Get

GET /draft/network/(string: id)?[key=(string: *key*)]
Get a network in draft mode.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role. Other users may provide an temporary access key.

Example requests

Using cURL

```
curl --user administrator:password https://mica-demo.obiba.org/ws/draft/network/
↪bioshare-eu
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪GET /draft/network/bioshare-eu --json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": "bioshare-eu",
  "name": [
    {
      "lang": "en",
      "value": "Biobank Standardisation and Harmonisation for Research Excellence␣
↪in the European Union"
```

(continues on next page)

(continued from previous page)


```
    }
  ],
  "acronym": [
    {
      "lang": "en",
      "value": "BioSHaRE-EU"
    }
  ],
  "description": [
    {
      "lang": "en",
      "value": "<p> </p>\r\n\r\n<p>BioSHaRE is a consortium of European␣
↪leading biobanks and international researchers from all domains of biobanking␣
↪science. The overall aim of the project is to build upon tools and methods␣
↪available to achieve solutions for researchers to use pooled data from␣
↪different cohort and biobank studies. This, in order to obtain the very large␣
↪sample sizes needed to investigate current questions in multifactorial diseases.
↪ This aim is achieved through the development of harmonization and␣
↪standardization tools, implementation of these tools and demonstration of their␣
↪applicability.</p>\r\n\r\n<p> </p>\r\n\r\n<p>As part of its mission,␣
↪BioSHaRE will document information collected by participant biobanks and␣
↪harmonize, integrate and co-analyse biobank-specific data to answer key␣
↪research questions on chronic diseases.</p>\r\n"
    }
  ],
  "content": "{\"website\":\"http://www.bioshare.eu\",\"maelstromAuthorization\":
↪{\"authorized\":false,\"authorizer\":null}}",
  "studyIds": [
    "finrisk-2007",
    "ship",
    "lifelines"
  ],
  "studySummaries": [],
  "logo": {
    "id": "9e000dc1-564b-4561-92f2-9484fb07054b",
    "fileName": "Bioshare.png",
    "type": "logo",
    "lang": "en",
    "size": 42718,
    "md5": "0c64fd3fb833e28abf8c97e6a8678615",
    "timestamps": {
      "created": "2021-04-12T06:38:04.502Z"
    }
  },
  "memberships": [],
  "permissions": {
    "view": true,
    "edit": true,
    "delete": true,
    "publish": true
  },
  "published": true,
  "timestamps": {
    "created": "2021-04-12T06:38:00.904Z",
    "lastUpdate": "2021-04-12T06:38:04.547Z"
  },
  "obiba.mica.EntityStateDto.state": {
```


(continues on next page)

(continued from previous page)


```
    "publishedTag": "3",
    "revisionsAhead": 0,
    "revisionStatus": "DRAFT",
    "publicationDate": "2021-04-12T06:38:04.571Z",
    "publishedBy": "administrator",
    "publishedId": "57c9b73f7eec70a4ea471e96c741ddd4c41737e9",
    "permissions": {
      "view": true,
      "edit": true,
      "delete": true,
      "publish": true
    }
  }
}
```


Query Parameters

- **key** (*string*) – Optional temporary access key.

Response JSON Object

- **id** (*string*) – The document unique identifier.
- **acronym** (*object*) – The acronym (short name) of the document, as an object that describes a localized string.
- **name** (*object*) – The name of the document, as an object that describes a localized string.
- **description** (*object*) – The description of the document, as an object that describes a localized string.
- **content** (*string*) – The document's model content, as a stringified JSON object.
- **logo** (*object*) – The associated logo file.
- **studyIds** (*strings*) – The identifiers of the studies that are part of the network.
- **studySummaries** (*array*) – The list of the associated studies' summary object.
- **memberships** (*array*) – The list of the associated members object.
- **permissions** (*object*) – The different actions that can be performed on this document.
- **published** (*boolean*) – Whether the document is published.
- **timestamps** (*object*) – The date times (format ISO-8601) at which the document was created and updated.
- **obiba.mica.EntityStateDto.state** (*object*) – The publication state of the document.

Request Headers

- Authorization – As described in the *Authentication* section
- Accept – `*/*`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – The document.

- 401 Unauthorized – Unauthorized access.
- 500 Internal Server Error – Server error.

15.2 Upate

PUT /draft/network/(string: id)?[comment=(string: *comment*)]
Update a network.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT -H "Content-Type: application/json" --
↪data-binary "@network.json" https://mica-demo.obiba.org/ws/draft/network/
↪bioshare-eu
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT --content-type "application/json" /draft/network/bioshare-eu < network.json
```

Query Parameters

- **comment** (*string*) – Optional revision comment.

Request Headers

- Authorization – As described in the *Authentication* section
- Accept – `*/*`

Response Headers

- Content-Type – `application/json`

Status Codes

- 204 No Content – Successful update of the document.
- 401 Unauthorized – Unauthorized access.
- 404 Not Found – The document does not exist.
- 500 Internal Server Error – Server error.

15.3 Get Model

GET /draft/network/(string: id)/model
Get the model part of a network.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role.

Example requests

Using cURL

```
curl --user administrator:password https://mica-demo.obiba.org/ws/draft/network/
↪bioshare-eu/model
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪GET /draft/network/bioshare-eu/model --json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "website": "http://www.bioshare.eu",
  "maelstromAuthorization": {
    "authorized": false,
    "authorizer": null
  }
}
```

Request Headers

- Authorization – As described in the *Authentication* section
- Accept – `*/*`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – The document's model content.
- 401 Unauthorized – Unauthorized access.
- 500 Internal Server Error – Server error.

15.4 Update Model

`PUT /draft/network/(string: id)/model`
Update the model part of a network.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` or `mica-editor` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT -H "Content-Type: application/json" --
↪data-binary "@model.json" https://mica-demo.obiba.org/ws/draft/network/bioshare-
↪eu/model
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT --content-type "application/json" /draft/network/bioshare-eu/model < model.
↪json
```

(continues on next page)

(continued from previous page)

15.5 Index

PUT /draft/network/(string: id)/_index
Rebuild both draft and published indices for a network.

This entry point requires *Authentication* of a user with `mica-administrator` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT https://mica-demo.obiba.org/ws/draft/
↪network/bioshare-eu/_index
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT /draft/network/bioshare-eu/_index --json
```

15.6 Update Status

PUT /draft/network/(string: id)/_status?value=(string: *status*)
Update the edition status of a network: `DRAFT` when the document is being edited, `UNDER_REVIEW` when the document is to be reviewed for publication, `DELETED` when the document is marked for permanent removal.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT https://mica-demo.obiba.org/ws/draft/
↪network/bioshare-eu/_publish
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT /draft/network/bioshare-eu/_publish --json
```

Query Parameters

- **status** (*string*) – The edition status which can be one of `DRAFT`, `UNDER_REVIEW`, `DELETED`.

15.7 Publish

PUT /draft/network/(string: id)/_publish
Publish a network.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` role.

Example requests

Using cURL

```
curl --user administrator:password -X PUT https://mica-demo.obiba.org/ws/draft/
↪network/bioshare-eu/_publish
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪PUT /draft/network/bioshare-eu/_publish --json
```

15.8 Unpublish

DELETE /draft/network/(string: id)/_publish

Unpublish a network.

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` role.

Example requests

Using cURL

```
curl --user administrator:password -X DELETE https://mica-demo.obiba.org/ws/draft/
↪network/bioshare-eu/_publish
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪DELETE /draft/network/bioshare-eu/_publish --json
```

15.9 Remove

DELETE /draft/network/(string: id)

Remove a network (unpublish it if necessary).

This entry point requires *Authentication* of a user with `mica-administrator` or `mica-reviewer` role.

Example requests

Using cURL

```
curl --user administrator:password -X DELETE https://mica-demo.obiba.org/ws/draft/
↪network/bioshare-eu
```

Using the *Web Services* Python command line tool

```
mica rest -mk https://mica-demo.obiba.org -u administrator -p password --method␣
↪DELETE /draft/network/bioshare-eu --json
```

CHAPTER 16

Partners and Funders

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

CHAPTER 17

Support

Please visit OBiBa support page.

/draft