Mica Documentation

OBiBa

Nov 01, 2018

Contents

Developped to meet the needs of individual epidemiological studies and study consortia, the OBiBa software stack (Opal, Mica etc.) provides an open-source solution for data management, analysis, harmonization and dissemination. Opal, OBiBa's core data warehouse application, provides all the necessary tools to import, manage, describe and transform data. The Mica application is used to build searchable web portals disseminating study and variable metadata, as well as research activities of both individual studies and study consortia. Content defined in Mica is published on a personalized web portal using Drupal, a popular open-source content management system used by websites worldwide. Mica works conjointly with Agate, OBiBa's authentication server which centralizes user related services such as profile management and email notification systems.

The OBiBa software stack is a core component of the Maelstrom Research program. For background information on Opal and Mica and their use for epidemiological research, see: Doiron, D., Marcon, Y., Fortier, I., Burton, P., & Ferretti, V. (2017). Software application profile: opal and mica: open-source software solutions for epidemiological data management, harmonization and dissemination. International journal of epidemiology.

Warning: Mica documentation is in the process of being rewritten. See also the `Mica Documentation Archive`

CHAPTER 1

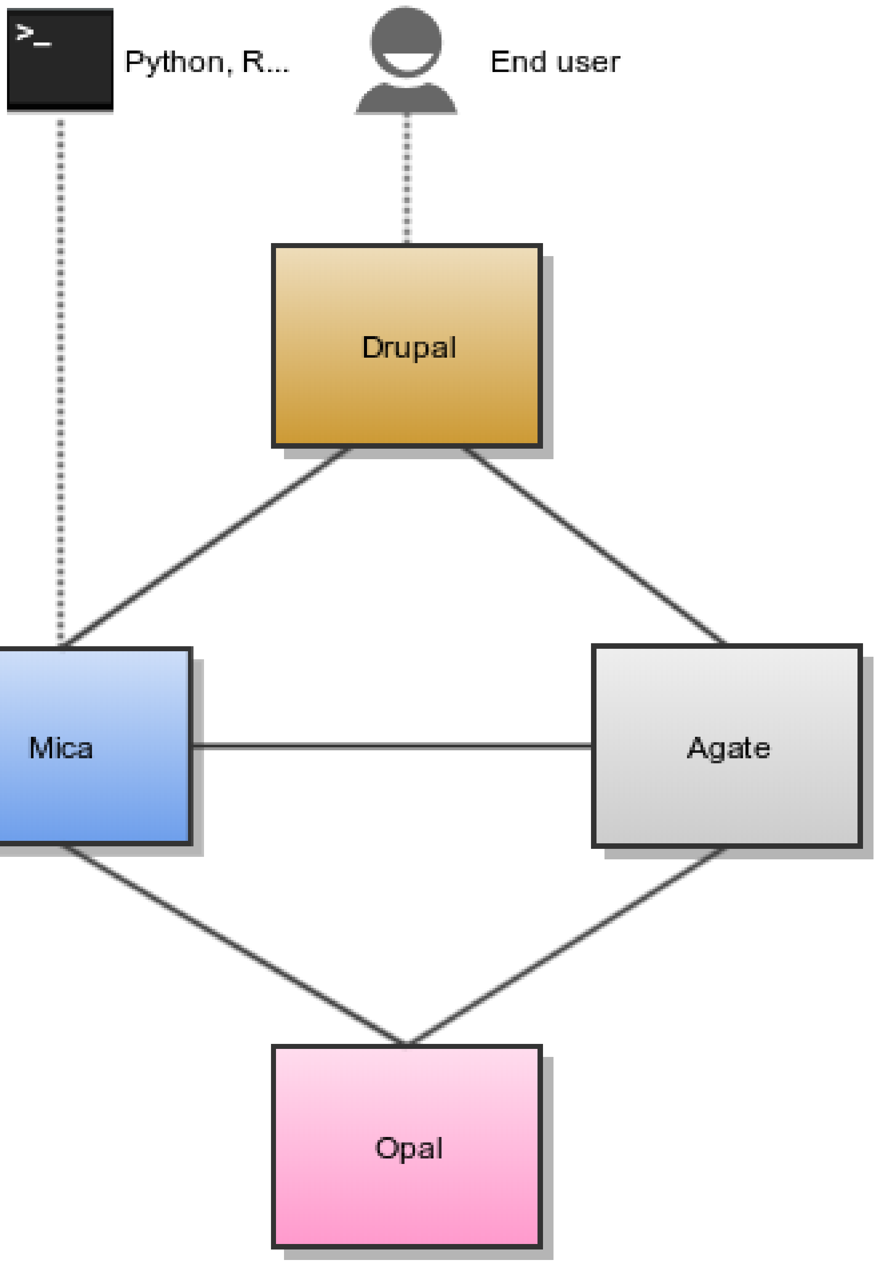
Introduction

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

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

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

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



1.1 Mica Server

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

Mica server is a client of Opal and Agate servers.

1.2 Opal Server

Opal application is used for:

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

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

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

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

1.3 Agate Server

Agate application is used for:

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

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

1.4 Drupal Server

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

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

CHAPTER 2

Documents

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

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

2.1 Types

2.1.1 Network

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

2.1.2 Individual Study

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

Population

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

Data Collection Event

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

2.1.3 Harmonization Study

A harmonization study is defined as a research project harmonizing data across individual studies to answer specific re-seach 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 an application form that researchers submit in order to gain access to the network/study data. There is a predefined workflow from submission to approval described in more detail in *Data Access Requests Management*.

2.1.8 Data Access Amendment

A data access amendment is an application form that researchers submit to request changes to a pre-approved data access request. The data access amendment workflow is identical to that of data access requests.

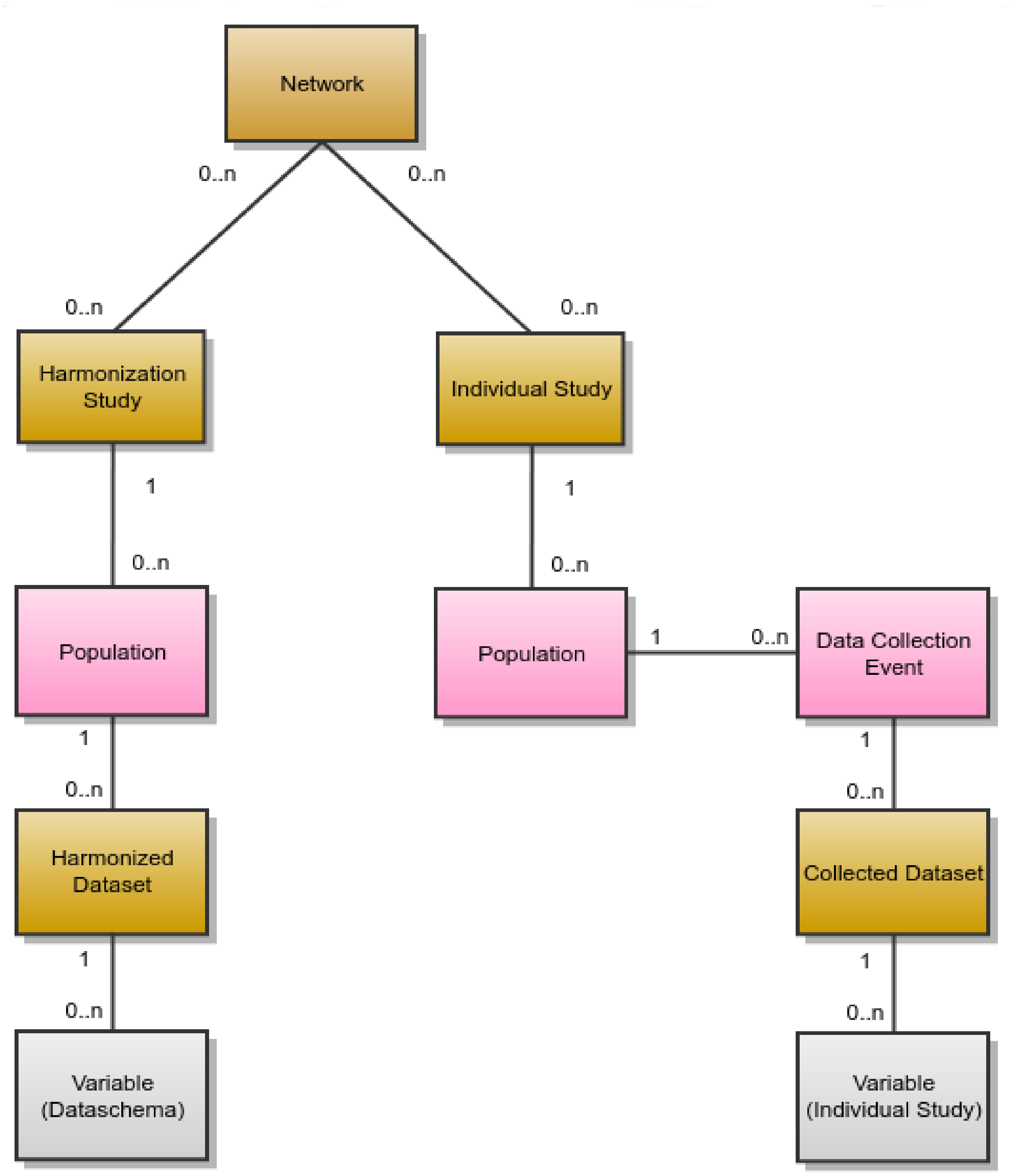
2.2 Search

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.

2.7 Cart

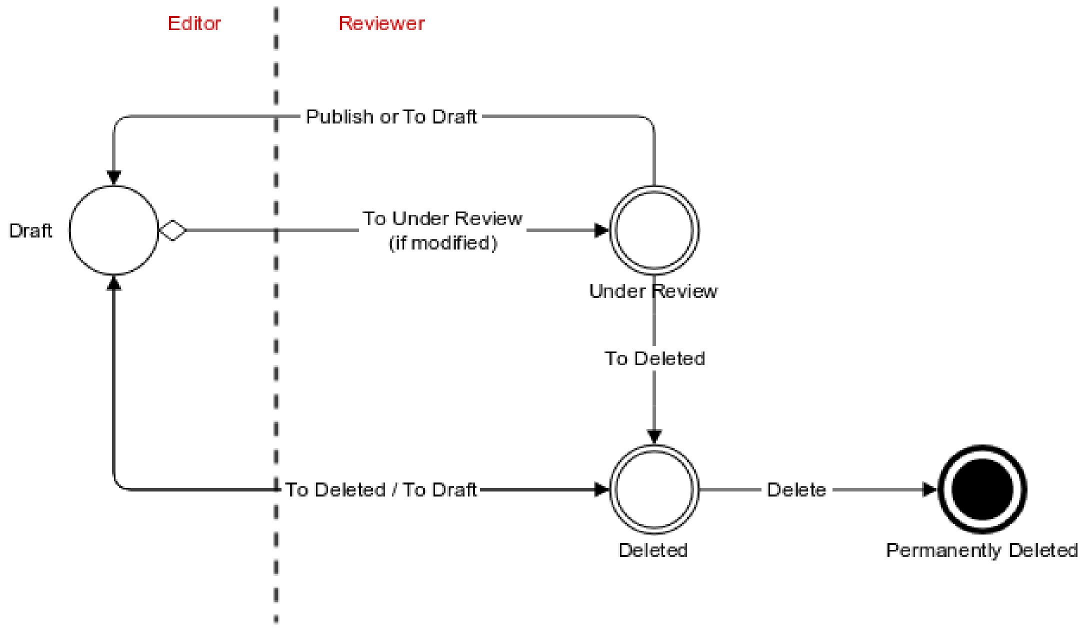
The cart groups a set of variables based on one or more search criteria. User can edit the cart by adding or removing variables, download its content (variables) as a CSV file and use it as a search criterion.

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`	Staged for	✗	✗	✓	• Draft • Under	• Draft

3.2 Transitions

The transitions between the different revision status are the following:

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

CHAPTER 4

Installation

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

4.1 Requirements

4.1.1 Server Hardware Requirements

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

4.1.2 Server Software Requirements

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

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

4.2 Install

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

4.2.1 Debian Package Installation

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

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

4.2.2 RPM Package Installation

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

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

4.2.3 Zip Distribution Installation

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

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

Set-up example for Linux:

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

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

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

4.3 Upgrade

The upgrade procedures are handled by the application itself.

4.3.1 Debian Package Upgrade

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

```
apt-get install mica
```

4.3.2 RPM Package Upgrade

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

```
yum install mica
```

4.3.3 Zip Distribution Upgrade

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

4.3.4 Upgrading to Mica 2.x

Mica 2.x is a major upgrade. Upgrade will affect the configuration files and database access.

Warning: As a general rule, always backup the configuration files and the databases before upgrading. Make sure also you can restore them!

Warning: To migrate to Mica 2.x, you need to have Mica >= 1.2.0

Mongodb access

For an automatic migration, Mica requires a user with full access to MongoDB. Make sure to create the role below and assign it to `micaadmin`:

```
use admin
db.createRole({
  role: 'obibauser',
  privileges:[{
    resource: {anyResource: true},
    actions: ['anyAction']
  }],
  roles: []
});

db.grantRolesToUser(
  "micaadmin",
  [ {role: "obibauser", db: "admin"} ]
);
```

Mica configuration file

Some configurations changed in file `/etc/mica2/application.yml`

Configurations with prefix "mongodb." are useless, you can delete them. Now, we need the property "spring.data.mongodb.uri". Exemple for the above user:

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

4.3.5 Upgrading to Mica 3.x

This is a major upgrade consisting of several model changes and the addition of the new Harmonization Study document.

Warning: To migrate to Mica 3.x, you need to first install 2.2.3, run the server once before you install the 3.x version.

In addition, the following taxonomy changes can affect the current installations. Before upgrading to Mica 3.x and to prevent conflicts or loss of data make sure your taxonomy vocabularies do not match the following:

New Study taxonomy vocabularies:

- className: denoting the type of study document (Individual or Harmonization)
- harmonizationDesign
- populations-id
- populations-name
- populations-description
- populations-dataCollectionEvents-id
- populations-dataCollectionEvents-name
- populations-dataCollectionEvents-start
- populations-dataCollectionEvents-end
- populations-dataCollectionEvents-description

New variable taxonomy vocabularies:

- studyId
- populationId
- dceId

After upgrading to Mica 3.0 make sure to clear all cache and re-index all documents in the administration module of the Mica server.

Note: A possible safeguard against taxonomy conflicts is to prefix custom vocabulary keys (omitting '.') so future updates do not override them.

4.4 Execution

4.4.1 Server launch

Service

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

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

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

```
service mica help
```

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

Manually

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

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

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

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

```
mica
```

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

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

4.4.2 Usage

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

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

4.4.3 Troubleshooting

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

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

CHAPTER 5

Configuration

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

5.1 HTTP Server Configuration

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

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

5.2 MongoDB Server Configuration

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

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

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

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

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

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

MongoDB User Creation Example

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

```
use admin

db.createRole({
  role: 'obibauser',
  privileges:[{
      resource: {anyResource: true},
      actions: ['anyAction']
  }],
  roles: []
});

db.createUser({
  user: "micaadmin",
  pwd: "micaadmin",
  roles: ['obibauser']
});
```

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

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

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

5.3 Opal Server Configuration

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

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

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

5.4 Mica Server Configuration

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

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

5.5 Shiro Configuration

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

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

5.6 Elasticsearch Configuration

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

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

Elasticsearch Cluster

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

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

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

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

An example of a configuration using the transport client:

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

Elasticsearch Server Configuration

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

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

5.7 User Directories

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

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

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

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

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

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

File Based User Directory

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

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

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

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

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

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


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

(continues on next page)

(continued from previous page)

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

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

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

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

5.8 Reverse Proxy Configuration

Mica server can be accessed through a reverse proxy server.

Apache

Example of Apache directives that:

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

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

(continues on next page)

(continued from previous page)

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

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

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

CHAPTER 6

Plugins

6.1 Repository

Mica plugins available are:

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

6.2 Installation

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

6.2.1 Automatic Installation

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

6.2.2 Manual Installation

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

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

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

6.3 Configuration

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

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

The layout of the plugin folder is as follow:

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

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

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

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

6.4 Backups

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

CHAPTER 7

Mica Web Application

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.

The following manuals are available:

- *Documents Management*: add, edit, publish documents and associated files
- *Data Access Requests Management*: approve/reject data access requests
- *Administration*: configure server settings and data access requests form

7.1 Requirements

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

CHAPTER 8

Documents Management

8.1 View, Edit, Publish

8.1.1 Summary

This guide provides a description of the web interface for viewing and managing documents .

8.1.2 Operations

Edit

Users with Editor or Reviewer permission can modify the properties of a document with status Draft. See *Permissions* and *Publication Flow* for more information.

Publish

Users with Reviewer permission can publish a document with status Under Review.

Unpublish

Users with Reviewer permission can unpublish an already published document.

Delete

Users with Reviewer permission can delete a document with status Deleted.

Status Change

Users with Editor or Reviewer permission can change the status of a document. See *Publication Flow* for more information.

8.1.3 Network Operations

Manage Contacts

Users with Editor or Reviewer permission can manage (add/remove/edit) the list of contacts.

Manage Investigators

Users with Editor or Reviewer permission can manage the list of investigators.

Manage Studies

Users with Editor or Reviewer permission can manage the list of studies.

Manage Networks

Users with Editor or Reviewer permission can manage the list of networks.

8.1.4 Individual Study Operations

Manage Contacts

Users with Editor or Reviewer permission can manage (add/remove/edit) the list of contacts. Manage Investigator-sUsers with Editor or Reviewer permission can manage the list of investigators.

Manage Populations

Users with Editor or Reviewer permission can manage the list of populations.

Manage Data Collection Events

Users with Editor or Reviewer permission can manage the list of data collection events of a population.

8.1.5 Harmonization Study Operations

Manage Contacts

Users with Editor or Reviewer permission can manage (add/remove/edit) the list of contacts.

Manage Investigators

Users with Editor or Reviewer permission can manage the list of investigators.

Manage Populations

Users with Editor or Reviewer permission can manage the list of populations.

8.1.6 Collected Dataset Operations

Manage Study Table

Users with Editor or Reviewer permission can manage (add/remove/edit) the associated study table.

8.1.7 Harmonized Dataset Operations

Manage Study Tables

Users with Editor or Reviewer permission can manage (add/remove/edit) the list of study tables. This operation only applies to harmonized datasets having a collection of study tables.

8.2 Revisions

8.2.1 Summary

Each time a document is edited a new history revision is added. The revisions are ordered from the most recent (current) to the oldest. If the document is published, a star indicates which revision is currently online.

8.2.2 Operations

View

Shows a read-only view of the network of the selected revision.

Restore

Restores the selected revision by replacing the current document. This operation is tracked as a new revision.

8.3 Files

8.3.1 Summary

Mica File System is a repository of files associated with all Mica domain documents . Similar to their associated documents, files have a publication flow and history revisions. The publication flow can apply to one or a group of files. Folders can be used to organize and group files into logical hierarchies but do not represent real data and therefore some operations such as searching do not apply to them.

8.3.2 Operations

Status Change

Refer to *Publication Flow* page for details.

Rename

Users with Editor or Reviewer permission on the containing document can rename a File and Folder. Names cannot contain the following characters: `$ / % #`.

Copy

Selected files and folders can be copied and pasted in other folders.

Move

Selected files and folders can cut and pasted in other folders.

8.3.3 File Specific Operations

Upload

A file from the local file system can be uploaded into the selected folder in the Mica file system.

Download

A file from Mica file system can be download into the local file system.

Search

Files can be searched in two ways:

- free-text where the keyword is matched against the file name , type or description .
- predefined searches listed in the search panel.

The predefined searches are all recursive in that the search query matches all files in all folder hierarchies. By toggling the Recursive button a free-text search can be recursive or applied to the current folder.

Type Edition

File types can be considered as tags and are a comma separated list of keywords associated with a file. They can be edited in the file detailpanel.

Description Edition

File description is localized and can be edited in the file detail panel.

8.3.4 Folder Specific Operations

Folder Creation

Folders can added as a single folder (baseline) or in form of a path (baseline/temp).

8.3.5 Batch Operations

Operations such as copy, move and publication flow can be performed in batch mode when they are applied to a group of selected files and/or folders.

8.3.6 Draft Permissions

The following tables describe file and folder draft permissions:

File Permissions

Role	Description
Reader	Can only view and download a file.
Editor	Can only edit, download and change the status of a file.
Reviewer	All operations are permitted.

Folder Permissions

Role	Description
Reader	Can only view and download a folder.
Editor	Can only edit, download, change the status and upload files into a folder.
Reviewer	All operations are permitted.

Note: File permissions are inherited from their parent folder unless specified.

8.3.7 Publication Access

Publication access makes folders and files available to everyone in *Mica Drupal Client*.

Note: Publication access is only available if Mica content is not configured for *open access*.

8.4 Comments

8.4.1 Summary

All Mica domain *documents* can be commented on by all users with the proper permissions. The content can be pure text or in Markdown format.

8.4.2 Operations

Comment

The entered text (markdown) will be associated with the current document.

Preview

A preview of the rendered markdown text is presented.

Edit

The comment text can be updated and previewed.

Delete

Deletes the selected comment.

8.4.3 Permissions

Role	Description
Reader	Can add, edit and delete own comment. Can view comments of others.
Editor	Can add, edit and delete own comment. Can view comments of others.
Reviewer	Can add, edit and delete own comment. Can view comments of others.
Administrator	Can view, add, edit and delete all comments.

8.5 Permissions

8.5.1 Summary

Access to each publishable *documents* can be controlled. There are actually two sets of privileges:

- **permissions** that apply to **draft** documents: only users having a permission on the draft document can see it,
- **accesses** that apply to **published** documents: by default published documents are open access, i.e. anyone (even a anonymous web portal visitor) can see the publications. This setting `Open access` can be changed in the *General Administration*.

8.5.2 Operations

Add Permission

Adding a permission gives a role to a named user or group of users on the draft document. The available roles are:

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.

Edit Permission

selected permission role can be modified.

Delete Permission

Delete selected permission.

Add Access

This operation is only available if the `Open access` general setting has been disabled. Adding an access gives the right to see the published document to a named user or group of users. As the *Files permissions* on the associated files can be managed independently, when adding an access there is an option for applying same access to all the files (selected by default). Note that user (or group) name * (star) is an alias for *Anyone* (or *Any group*).

Delete Access

Delete selected access.

CHAPTER 9

Administration

The *Administration* menu is available to users with the role `mica-administrator`. This menu gives access to server configuration and status listed below:

9.1 System Administration

9.1.1 General Administration

Summary

The *Administration* menu is available to users with the role `mica-administrator`. This menu gives access to server configuration and status.

Configuration

Properties

This section allows to define all general Mica configuration as *Server Identification*, *Content*, *Data Source* and *Documents Sections*.

Section	Name	Description
Server Identification	Name	The name of the organization using this instance of Mica server. It will be used when sending notification emails.
	Public URL	Public base URL of the server. It will be used when sending notification emails.
	Portal URL	Public base URL of the portal that will be used when sharing drafts.
Content	Languages	The languages in which the data pertaining to studies, networks and/or datasets will be entered in Mica.
	Default Character Set	The character set with which the data will be entered in Mica e.g. , UTF-8, iso-8859-1, etc.
	Open access	If checked, access to published documents will be opened to everyone.
Data Source	Primary Opal server Public URL	This Opal server is the primary source of variables and data summaries. (see below)
	Participant Privacy Threshold	No data summary will be returned from Opal if the number of participants is below this threshold.
Search Layout	Layout 1	Layout emphasizing a search bar that maps keywords to available taxonomies.
	Layout 2	Layout providing a search panel on the left-hand-side that displays the full taxonomy of various search categories (variable, study, and network).
Application Scope	Single study enabled	If checked, only one study is present in Mica
	Network section enabled	If checked, n etwork section is accessible.
	Single network enabled	If checked, only one network is present in Mica.
	Study datasets section enabled	If checked, study datasets section is accessible.
	Harmonization datasets section enabled	If checked, harmonization datasets section is accessible.

To edit a field, click on “Edit” and edit or modify the content the fields therein.

Note: Terminology

By the *primary* Opal server , we mean the Opal server on which Mica looks for data if no other Opal is specified in a study definition. Initially, the primary Opal server is set in Mica Configuration Files , but that entry is overridden by what you enter in here.

Note: Notifications

Mica requires Agate to send email notifications, which should be configured in Mica Configuration Files .

Membership Roles

Networks and Studies can have a list of members associated with them, which are grouped in roles. The list of available roles can be edited and by default the roles contact and investigator are present.

Note: Removing a role from the system is permanent and affects all networks and studies in Mica, i.e. removing and role and recreating it again with the same identifier, won’t recover the list of members for that role. However you can still recover the list of members if you revert to an old revision where the role was defined in the system.

Encryption keys

This section presents the tool related to the encryption–through HTTPS–of transactions between Mica and its clients by means of a trusted or a self-signed certificate.

Note: In the instruction below, when you are told to cut and paste the content of the certificate, private key or of an *.pem file, make sure that you copy **all* content, that is **including** the lines containing " —–BEGIN XXXXXXXX—–" and " —–END XXXXXXXX—– ".

Create a (self-signed) certificate

Go to **Administration > Encryption Keys**, click on the *Add Keys* drop-down under the subsection title *Encryption Keys* and select *Create*

1. Click on the *Add Keys* drop-down under the subsection title Encryption Keys.
2. Select *Create*.
3. Fill in the form and click on *Save*.
4. Click on the *Download Certificate* button under the section title *Encryption Keys*.

Your certificate (*.pem file) should automatically be downloaded on your computer.

Import a certificate

Go to **Administration > Encryption Keys**, click on the *Add Keys* drop-down under the subsection title *Encryption Keys*"and select *Import*. Here you may use (1) certificate and (2) private key that you created using third party software e.g., OpenSSL. Note that:

1. Both the certificate and the private key must in PEM format i.e., you can read them and the file starts with a —– BEGIN [. . .].
2. You must copy the certificate (or the content of the *.crt file) in the public key box and the private key (or the content of the *.key file) in the private key box.

In either case, you finish by clicking on *Save*. Finally, in order for the changes to be taken in account you need to restart Mica with

```
sudo service mica2 restart
```

Opals Credentials

In order to establish a secured connection with an Opal server, you must create a user in Opal along with the proper permissions, tell Mica to communicate with that Opal using this user. To do so, there are various scenarios available: you may connect to Opal by means of an SSL certificate or via authentication, these methods are explained in the following three sub-sections. Finally, the last section is about the permission of the user you created in Opal.

Note: In any scenario and for security reasons, never let Mica connect to an Opal as Opal's administrator. You must configure a specific user with appropriate reading permissions.

In **Administration > Opal Credentials** When you click on the drop-down menu *Add Opal Credentials* under the subsection title "Opal Credentials", you are presented with three choices, each corresponding to one of the next three subsections.

Create a certificate

With this first option, you can create a certificate directly in Mica with which you can create a user in Opal. In order to proceed that way:

1. Select "Create" in the drop down menu *Add Opal Credential*.
2. Fill in the necessary information to create the certificate and click on "Save".
3. The Opal you described at point 2 should now appear in the list under the *Add Opal Credential* drop-down. At the end of the line for that Opal, click on the download button in the Action column to download the *.pem file which is the certificate created taking in account the information you entered for that Opal and which will be use to add a user with certificate below.

   Note: The URL for that Opal must begin with https:// if we are about to use a certificate as the authentication method.

4. Login Opal and go to **Administration > Data Access > Users and Groups**.
5. Click on the drop-down menu Add a User and select the option "Add a user with certificate…".
6. Fill in the info and paste in the content of the *.pem file.
7. Save the information.

The user should now be in the list. You'll be done after restarting Mica with

```
sudo service mica2 restart
```

Import a certificate

In the case that you have already have a pair of keys, you may import it here to secure the communication with Opal. You may select "Import" and:

1. Fill in the fields (Opal's URL, public and private keys) appropriately.

   Note: Restrictions on how to fill the public key and private key fields using *.pem , *.crt and *.key files are the same as in **Encryption Keys > Import a Certificate** above.

2. You can now proceed as in the instruction to Create a Certificate starting from point 4.

The user should now be in the list and you'll be done after restarting Mica server.

Opal Credentials

This last point is probably the easiest:

1. Go in **Opal Administration > Data Access > Users and Groups**
2. Click on the drop-down menu *Add a User* and select the option "Add a user with password…".
3. and you create a user filling the form.

With that user's credentials i.e. , username and password, you select the item "Username" in the "Add Opal Credential" Menu. You fill in the form using Opal's URL and the credentials of the user you created in Opal.

Last step: giving proper permissions to the Mica server

You must now give the user you just created the proper permissions on tables in Opal so that he can carry out his tasks. Here are the steps to do so:

Note: Recommended permission is View dictionaries and summaries. You can grant such a permission by

1. Going in Opal
2. In **Project > <some specific project> > <some specific table of that project>**
3. Click on the "Permissions" tab
4. Click on the "Add Permission" button and on "Add user permission" in the drop-down menu
5. In the pop-up window, add the name of the user to which you want to grant access and select "View dictionaries and summaries" permission
6. Click on save
7. Repeat steps 1-6 for any other table you want the user to have access to

9.1.2 Notifications Administration

Summary

When a document/file goes throw a publication flow , relevant users and groups are notified with a status change according to the permissions the y have on the document/file. A reviewer is notified when a document/file status changes to Under review or to Deleted since only a reviewer can publish or permanently delete a document/file. When a document/file status changes to Draft, the editor is notified in order to make the necessary changes on the draft.

Configuration

Notifications can be configured by checking one or many of the following options.

On network status changed notification	Send email notifications when network status changes to Deleted, to Under review or to Draft.
On study status changed notification	Send email notifications when study status changes to Deleted, to Under review or to Draft.
On study dataset status changed notification	Send email notifications when study dataset status changes to Deleted, to Under review or to Draft.
On harmonization dataset status changed notification	Send email notifications when harmonization dataset status changes to Deleted, to Under review or to Draft.
On file status changed notification	Send email notifications when files status changes to Deleted, to Under review or to Draft.
On comment notification	Send email notifications when a comment is added or updated.
On research project status changed notification	Send email notifications when research project status changes to Deleted, to Under review or to Draft.

9.1.3 Style Administration

Summary

The Mica default style can be override with a personalized stylesheet. The CSS classes are the ones of Bootstrap (Bootswatch Flatly theme).

9.1.4 Translations Administration

Summary

The Mica's interface can be translated in any language defined in the General Administration.

Definition

A translation is a key-Value pair that can be added, edited or specified for all configured languages. A translation list is composed of build-in translations and custom translations added through the interface. Note that Mica is already translated in French and English through the build-in translation stored as an application config file.

Operations

Adding translations can be done one key at a time (Add Entry) or by importing a translation file (Import Translation).

Add Entry

First, choose a target language by clicking on a language tab. Then, click on "Add Translation" then on "Add entry" to display the New Translation Entry form. Fill in the Name (key) and Value (Translation) and click on Ok. Scroll down and click on "Save", a new translation key is added. This key should be specified in all the other languages (cf. the "Search/Edit a translation" section).

Import Translation

Instead of being created once at a time, custom translations can be added to the application in one action. Click on "Add Translation" then on "Import Translations", click on "choose file" to choose a translation file (in JSON format). Once the file is uploaded, click on Import.

9.1.5 Caching Administration

Summary

The Administration menu is available to users with the role `mica-administrator`. This menu gives access to server configuration and status.

Definitions

All available caches are listed in the first table under Type . If one or many of these caches becomes desynchronized or do not behave as expected, you may trash the content of the cache so that it will have to be rebuilt. This is done using the trash can icon under the title Action .

There is also an option to build the cache for Dataset variables statistics by clicking the play icon at the right. This specific cache is, by far, the biggest of those used by Mica and may take seconds to hours to build depending on the size of the dataset and various other factors. Having a control over the build is convenient since, for instance, one may want to disconnect Opal from Mica, since the cache is built from the data contained in Opal, the cache has to be functional before the disconnection so that it may continue to work.

9.1.6 Indexing Administration

Summary

The **URL-NEEDED:search engine** is based on indexed documents. Each index can be rebuilt without affecting the history of changes and the publication status of the documents.

Index Type

Type	Description
All	Indices about any type of document: network, study, dataset (and associated variables), file, person.
Networks	Index that allows to search for networks.
Studies	Index that allows to search for studies.
All datasets	Index that allows to search for datasets of any type and associated variables.
Study datasets	Index that allows to search for study datasets and associated variables.
Harmonization datasets	Index that allows to search for harmonization datasets and associated variables.
Taxonomies	Index that allows to search for taxonomy terms (i.e. the search criteria).

9.1.7 Application Metrics Administration

Summary

This page provides a many metrics which intend to help the system administrator in his tasks. Information about HTTP requests, service statistics and Ehcache statistics are provided. In addition, one can find information on the JVM concerning memory usage (total, heap and non-heap), garbage collection and threads.

9.1.8 Logs Administration

Summary

This is to select what should appear in the log file typically located at `/var/log/mica2`. For each namespace, there are various level you can choose from such as: `trace` , `debug` , `info` , `warn` , and `error`. Each level is typically more informative than the next one.

9.2 Content Administration

9.2.1 Document Configuration

Summary

This section provides a description of the web interface for configuring the document types. The document type fields configuration is based on the schemaform framework that allows to define the form to collect the model of the document.

Operations

Form

Document fields configuration consists of providing the necessary information (in JSON format) to build a form:

- **Schema**: specifies the name and data type to be collected
- **Definition**: specifies how to layout the form (field positions, translations, section titles, help text)
- **Preview**: is the result of the interpretation of the *schema* and the *definition* by schemaform
- **Model**: displays the data collected (in the preview) according to the specified schema.

Note: Due to the structure of the study type, the form of the study is split in several pieces:

- study: general definition of the study
- population: each study can have one or more populations, this form applies to these only
- and in case of individual studies, data collection event : each population can have one or more data collection events, this form applies to these only.

For detailed documentation on how to use schemaform, see the schemaform documentation. The default schema and definition provided by Mica can also be a good starting point for getting into *schemaform* configuration.

Note that not all fields of a document type are configurable: there are some built-in fields such as name, description… that are necessary for Mica to operate. These fields will appear at the head of the form (when editing a document, not when having a preview of the form configuration). In addition to that other built-in fields are not handled by schemaform , such as the list of studies of a network, the Opal table(s) associated to a dataset, the persons that are members of a study or a network…

It is currently not possible to dynamically integrate *schemaform* addons to Mica. Please contact us if you have a specific need.

Search

Once the fields have been defined, the document search can be configured so that documents can be found by field value. The purpose of the search configuration is then to define a *taxonomy* specific to the considered document type.

The taxonomy is a classification of existing documents based or their fields. It describes the search criteria used to search documents. A criterion is defined on the document's fields and each criterion can (or not) consists in terms related to the values that a field can take. This leads to two criteria types

1. Criteria without terms based on string fields, which leads to a free text search, or based on numerical fields to search a number range.
2. Criteria with terms. The terms can be enumerated values for a string, pre-defined number ranges for a numerical field or the values true/false for a Boolean.

Add Criterion

A user with administrator rights can add a criterion to the existing taxonomy.

`Name` (identifier), `Title`, and `Description` are required fields. Under **Definition**, `Field` is required and represents a fully qualified field name under the document model. Check the `Repeatable` definition if search must be an exact match or contained value. Check the `Localized` definition if the field is multilingual.

Under **Display**, one can choose how and if the criterion will be shown. Checking `Hidden` will hide the criterion from the search while `Facet` controls whether or not it will be exposed in the list page.

Add Term

A user with administrator rights can add a term to an existing criterion. A term is described by a unique `Name` for string criterion or a range for numerical criterion, a `Title`, a `Description` and a set of `Keywords` useful when building search requests.

Dataset Specific Search

Despite the fact that study datasets and harmonization datasets can have different fields, there is only one dataset taxonomy that apply to both sub-type of datasets. See the className search criterion that allows to discriminate datasets by their specific type.

In addition to that, datasets in Mica have associated variables that are extracted from Opal. Variable model cannot be configured as it lives in Opal, but variable base taxonomy (i.e. that refers the variable properties) can be adjusted in this section.

Permissions

The permissions that apply to all the documents of the considered type can be specified in this section. See Permissions documentation which is still relevant (expect that is applies to all documents in a type instead of a specific one).

9.2.2 Data Access Request Administration

Definitions

Application Form

The web application form can be specified both in terms of schema and UI definition. See *Angular Schema Form* documentation for more details.

The *Preview* and *Model* tabs are informational only and can be used to preview the rendered form and the input data that will be collected.

Properties

Title Field	Specifies the field value to be used as the document's title.
Summary Field	Specifies the field value to be used as the document's summary.
Enable Amendments	Enables the capability of requesting amendments to an approved data access request. This option is disabled by default.

PDF Download

The data collected in the form can be downloaded in the following formats:

PDF Template, this option requires uploading a template (one for each configured languages) compatible with the custom *Angular Schema Form* model.

Printable Page, this option renders the form as a printable page in the browser so the user can either save the form as a PDF or send it to a printer.

Predefined Action Names

Predefined Action Names are common actions that *data access officers* perform while processing the data access requests. These actions can then be logged in the data access request *history section*.

The action names defined here are in fact translation keys that must be created in *Translations Administration*.

The following steps demonstrate how these names are added:

- add the action names without any spaces or any of these invalid characters: `.` ~ !
- save the configuration
- add the action keys in *Translations Administration*

Note: An action key is the action name preceded by its translation path: `data-access-request.action-log.config.label.<action-name>`.

Amendment Form

Same as the *Application Form*, it uses *Angular Schema Form* to define the schema and UI definition.

Under the **Properties** section one can specify the document's *Title Field* and it's *Summary Field* as defined in the custom *Angular Schema Form*.

Notifications

Email notifications can be sent, if configured, to applicant and data access officers when an event happens on the data access request. Events can be: status changes or comment additions or updates.

These configurations apply to both the data access requests and their amendments.

Settings

The data access request goes through several steps. Some minimum settings can be applied to control this workflow, i.e. enabling the *review* status and making the accepted and rejected status final. Also the pattern to generate identifiers for data access requests can be configured.

These configurations apply to both the data access requests and their amendments.

Permissions

Role	Description
Reader	Can view all data access request and their amendments.

There are also two additional (sub) permissions that can be granted to a *Reader*:

Apply the same permission to associated action logs	This permission grants viewing the action logs.
Give access to the private comments section	This permission grants viewing and editing of private comments.

Customizing Reports

Warning: This is an **experimental** functionality, make sure to backup your database beforehand.

Follow these steps to customize the data access request and amendment reports:

- stop Mica server
- get the current report configuration files as templates:

```
mongo mica --eval 'db.dataAccessForm.find({}, {csvExportFormat: 1, _id: 0})'
mongo mica --eval 'db.dataAccessAmendmentForm.find({}, {csvExportFormat: 1, _id:␣
↪0})'
```

- make sure `/etc/mica2/config/data-access-form/` and `/etc/mica2/config/data-access-amendment-form/` directories exist
- copy your templates `export-csv-schema.json` under the previously created directories
- clear the *csvExportFormat* field in dataAccessForm and dataAccessAmendmentForm:

```
mongo mica --eval 'db.dataAccessForm.update({_id: "default"}, {$set:
↪{csvExportFormat: ""}})'
mongo mica --eval 'db.dataAccessAmendmentForm.update({_id: "default"}, {$set:
↪{csvExportFormat: ""}})'
```

- edit your templates:
  - `/etc/mica2/config/data-access-form/export-csv-schema.json`
  - `/etc/mica2/config/data-access-amendment-form/export-csv-schema.json`
- to make sure that these files can be accessed by Mica server run the following shell command:

```
``sudo chown -R mica:adm /etc/mica2``
```

- start Mica server

The snippet below shows a report configuration file:

```
{
  "headers": {
    "title": {
      "en": "<Organization> Access Office",
      "fr": "<Organisation> Bureau d'accès"
    },
    "subtitle": {
      "en": "Access Requests Report",
      "fr": "Rapport sur les demandes d'acces"
    },

  },
  "table": {
    "generic.accessRequestId": {
      "en": "ACCESS REQUEST ID",
      "fr": "ID DE LA DEMANDE D'ACCES"
    },
    "projectTitle": {
      "en": "TITLE",
      "fr": "TITRE"
    },

  }
}
```

Where fields under `headers` are fixed (built-in) but their translations can be modified. Fields under `table` can be fully customized (removed, re-ordered, added, etc).

The `table` properties can be inferred from the document's schema. They are the fields found in the model.

Note: Properties prefixed by *generic.* are internal and not part of the data access request or amendment form schemas and are considered *built-ins*. They can be removed, however.

Excluding Legacy Data Access Requests IDs

To make sure legacy data access request IDs won't be used by Mica follow the following steps:

- stop Mica server
- make sure `/etc/mica2/config/data-access-form/` exists
- create the file `data-access-request-exclusion-ids-list.yml` under the folder `/etc/mica2/config/data-access-form/`.
- add each ID on a separate line as the example below.
- run the command below to make sure the folder and the file have the proper permission:

```
sudo chown -R mica:adm /etc/mica2
```

- restart Mica server so the changes take effect.

Here is an example of the exclusion file:

```
exclusions:
  - "LEGACEY_ID_001"
  - "LEGACEY_ID_002"
  - "LEGACEY_ID_003"
```

9.3 Data Discovery

9.3.1 Building Queries

Summary

The search module allows users to filter and explore Mica documents by building taxonomy based queries. There are several types of criteria where each criterion is associated with one vocabulary. Below are some of the different types of criteria:

- Matching Criterion is used on vocabularies of type string.
- In Criterion are used on vocabularies of type string with several terms each term describing a possible value.
- Range Criterion are used on vocabularies of numeric type having several terms each term describing a bounding tuple [from, to).
- Numeric Criterion are used on vocabularies of numeric type without terms in a given range [from, to).

There are four main taxonomies each of which is used for searching a corresponding Mica document:

- Variable taxonomy Dataset taxonomy Study taxonomy Network taxonomy
- Dataset taxonomy
- Study taxonomy
- Network taxonomy

The Variable taxonomy is a generalisation of four specific taxonomies three of which are described and imported from Opal and are developed by Maelstrom Research to allow annotating study and harmonized variables:

- **Areas of information**: classification developed by Maelstrom Research
- **Source & Target**: information about the collected variable

- **Scales/Measures**: constructs for cognitive functioning and mental health

The only variable related taxonomy defined in Mica is **Variable properties** that is used to describe the Opal variable attributes.

Operations

Create a search criterion

Below are the three possible ways of creating query criteria:

Search box

The search box is a quick way of searching taxonomy vocabularies/terms to create a criterion. Typing keywords displays a type-ahead suggesting all matching terms and their corresponding vocabularies in all found taxonomies.

Search properties

The Search properties is a taxonomy specific browser that allows the selection of vocabularies and terms. The left-most column contains the list of all vocabularies, the middle column lists the terms of a selected vocabulary and the last column is the selected term. A criterion can be created by adding a selected vocabulary or one of its terms to the query.

Classifications

The classification section is a more descriptive view of the taxonomies grouped by Mica document types, namely, Variable, Dataset, Study and Network. Similar to Search properties, a criterion can be created either by selecting a vocabulary or one its terms.

Create a search criterion

Criterion values can be changed in the following ways:

- any: search all terms of the vocabulary.
- none: search all vocabularies except this vocabulary.
- in: choose all or a sub-set of terms of this vocabulary.
- Search box: search a text in the vocabulary value.
- From and To: search a range

9.3.2 Exploring Results

Summary

After building a search query as explained in the Building Queries page, corresponding search result can be visualized in the same search page. Query results are organized in three main tabs: *List*, *Coverage* and *Graphics*.

Operations

List

The List tab displays the results of the search query inventoried in the documents catalogue and organized in four different tabs; Networks, Studies, Datasets and Variables. Each tab describes in details the corresponding search result and contains links that redirect either to a page (e.g variable page, study page, network page) or to a count-search page (e.g variable count, study count).

Coverage

The Coverage displays the results of the search query in a tabular format, giving an overview of which document covers at least one of the search criteria, where the documents are Study, Dataset (Study Dataset and Harmonized Dataset) and Network. The table reports the total number of variables linked to each document and for each search criterion. Click on any number to obtain the corresponding list of variables. More advanced results can be obtains using the following functionalities.

- **Full coverage**: The study and Dataset list can be filtered by clicking on the "Full Coverage" button to keep only documents that collected information on all search criteria covered by the query. Select documents individually using the check-boxes to the left of the table and click on the button "Filter" the keep a document sub-set.
- **Data Collection Event (DCE)**: Check "Data Collection Event (DCE)" box to obtain the breakdown of variable numbers by data collection events. Click on the button "Full Screen" to better visualize the results. For more information about the Data Collection Event, refer to Mic a documents page, section Study.

Graphics

The tab Graphics summarizes the characteristics of the studies meeting the search criteria in terms of geographical distribution, study design, number of participants and collected biological samples. Click on any count displays the corresponding search page.

Download Results

The search result of any tab can be downloaded and stored in a csv file by clicking on the button Download.

CHAPTER 10

Data Access Requests Management

10.1 Summary

Before requesting access to data, a researcher needs to register by sending a *data access request*.

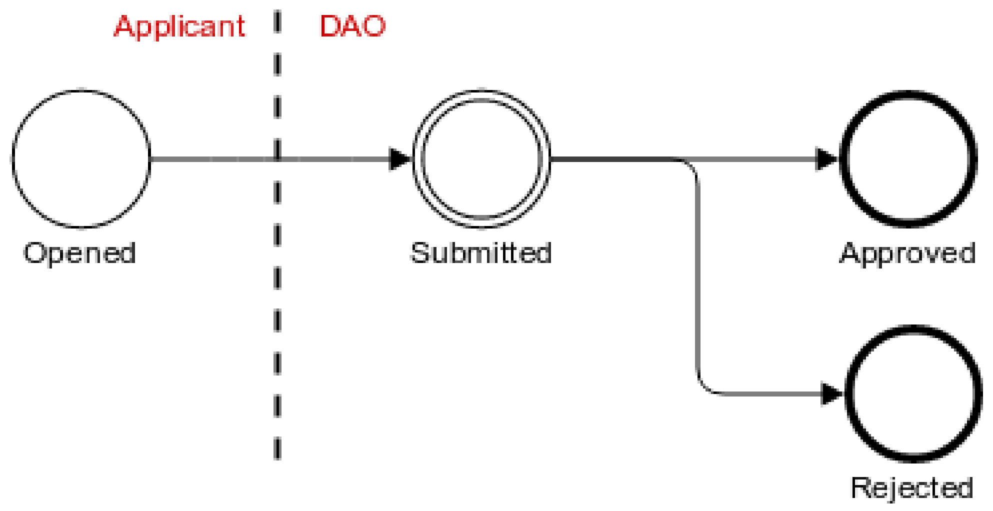
10.2 Request Workflow

To create a data access request, users have to fill out the application form and *submit* it once it is completed. The *validate* button can be used to check that the form contains the required information. Before submitting he request, the form can be edited and saved (for future edits) as needed. After the request is submitted, the form is frozen and can no longer be edited unless the request is reopened.

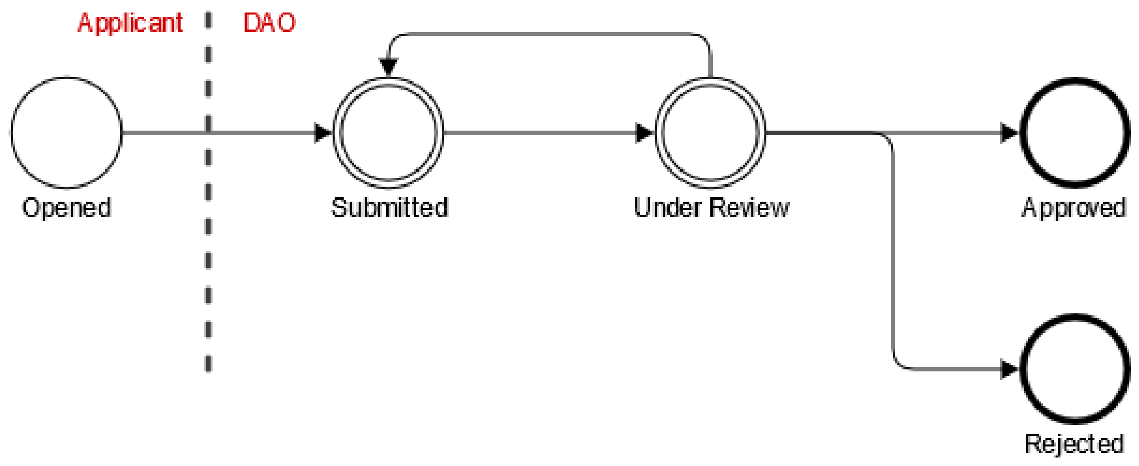
After the request is submitted, a data access officer will *review* it in order to *approve* or *reject* it. The data access officer can also *reopen* (or *conditionally approve*) the request if he considers that modifications are needed. This workflow, as well as email notifications that can be sent on request status changes, can be configured by the administrator, e.g. to omit the *review* step or make the *approve*/*reject* steps final.

The most simple workflow is:

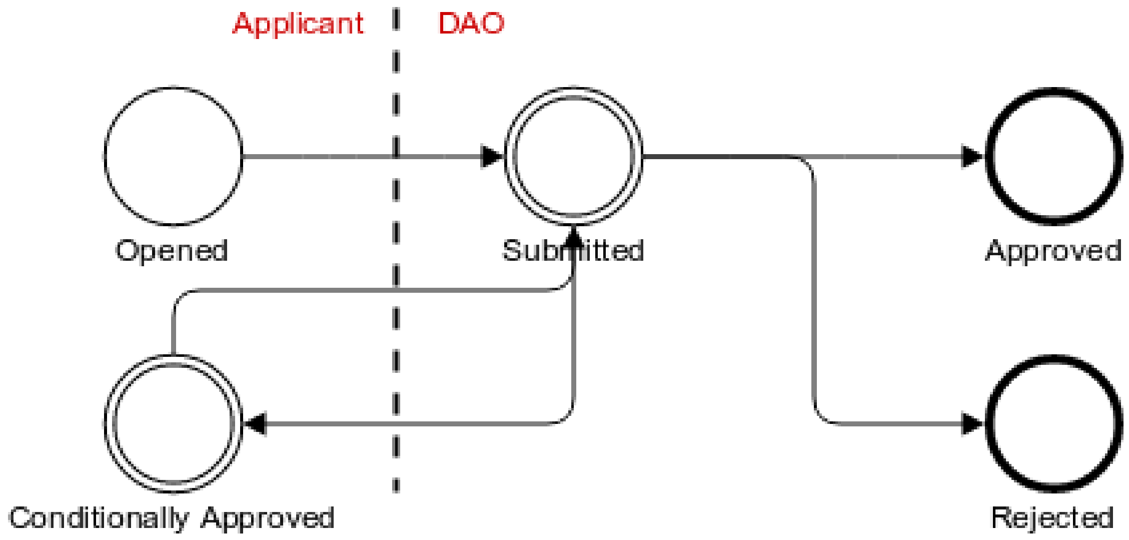
- no `Under Review` intermediate state
- no `Conditionally Approved` intermediate state
- `Approved` state is final
- `Rejected` state is final



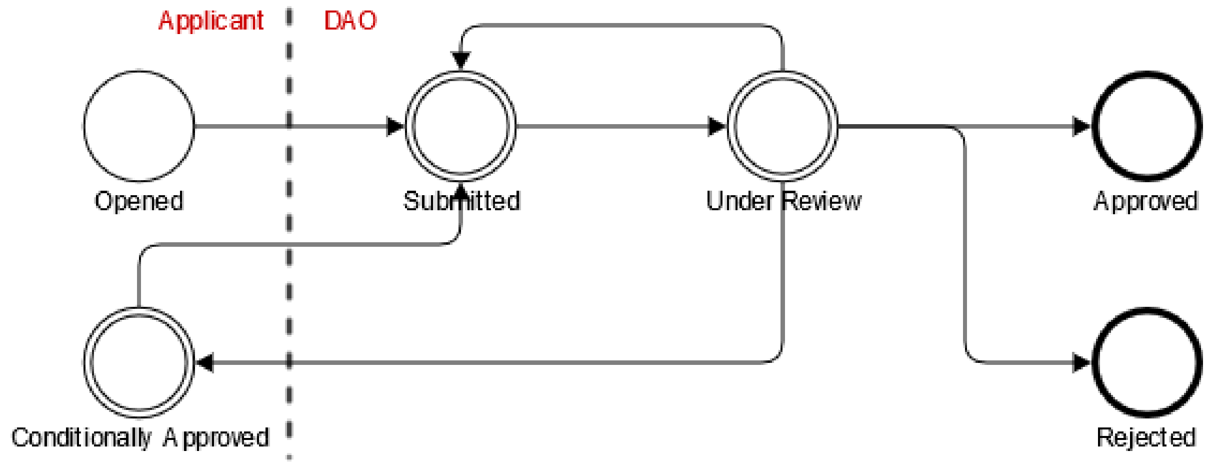
When the `Under Review` intermediate state is activated (which is the default configuration), the flow is:



When the `Conditionally Approved` intermediate state is activated, the flow is:



When both `Under Review` + `Conditionally Approved` intermediate states are activated, the flow is:



The workflow is resumed in the following table:

Status	Description	From Status	To Status
Opened	The request is in an editable state.		Submitted
Submitted	The request is submitted to the application officer and is not editable by the applicant. The application officer can (conditionally) approve or reject it or, if configured, go to an intermediate under review status.	Opened	Under Review or • Approved • Rejected
Under Review	The request is being reviewed by the application officer. He can reopen, conditionally approve (if this state is activated), approve or reject the request.	Submitted	• Opened • Approved • Rejected • Conditionally Approved • Submitted
Approved	The request has been approved. If configured, the request can go back to the submitted or reviewed status.	Submitted or Under Review	if state is not final: Submitted or Under Review
Rejected	The request has been rejected. If configured, the request can go back to the submitted or reviewed status.	Submitted or Under Review	if state is not final: Submitted or Under Review
Conditionally Approved	The request was submitted, has been reviewed and some adjustments are required before a final approval. The applicant can edit its request and will re-submit it.	Submitted or Under Review	Submitted

10.3 Application form

The application form is configured by the administrator, who can also define the PDF template used to create a printable copy of the form (available by clicking on the *Download* button).

10.4 Amendments

Researchers can submit amendments to request changes (e.g. additional data, new research collaborators) to a pre-approved data access request.

10.5 Comments

To enhance the collaboration between users and data access officer, each member can add a comment to a data access request. Mica can be configured to send email notifications when a comment is added or updated.

10.6 Private Comments

Administrators, data access officers and users with proper permissions (see *data access request permissions*) can add private comments while processing a data access request or an amendment.

10.7 History

A list of all status changes of a data access request and its amendments. In addition, administrators and data access officers can log actions that they have taken while processing a request or an amendment. These actions are either *pre-defined* or free text. Only users or groups with proper permissions (see *data access request permissions*) can view action logs.

CHAPTER 11

Mica Drupal Client

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

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

11.1 Requirements

11.1.1 Server Hardware Requirements

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

11.1.2 Server Software Requirements

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

11.2 Dependencies

11.2.1 System dependencies

For Linux systems the following dependencies need to be installed:

- Debian

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

- CentOS

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

11.2.2 Drush and Composer

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

Install Composer:

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

Install Drush via Composer tool:

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

# Verify Drush install
drush status

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

11.2.3 Drupal Server

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

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

11.3 Installation

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

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

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

```
# Go to Drupal installation directory
cd DRUPAL_DIR

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

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

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

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

# Download Obiba Javascript dependencies
drush download-mica-dependencies

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

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

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

- Debian

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

- CentOS

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

To enable the mode_rewrite on Debian:

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

On CentOS the rewrite_mode is enabled by default.

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

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

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

11.4 Upgrade

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

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

```
# Go to Drupal installation directory
cd DRUPAL_DIR

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

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

(continues on next page)

(continued from previous page)

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

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

# Install Obiba Agate module new dependency
drush en autologout

# Clear all caches
drush cc
```

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

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

CHAPTER 12

Drupal Configuration

Mica Drupal Client is a set of modules that can be enabled/disabled in Drupal's `Administration > Modules` configuration page.

Note: Before disabling an OBiBa Drupal module make sure you know exactly what it does, disabling an important module can render Mica Client unusable.

The following modules can be considered as extensions and disabling them does not affect the core functionality of Mica Drupal Client:

OBiBa Mica Data Access	Adds data access request and amendments capability (the latter must be enabled in Mica server).
OBiBa Mica Analysis	Adds entity counts, variable cross-tabulation and statistics capability.
OBiBa Mica Research Project	Adds research project management capability.
OBiBa Mica Sets	Adds variable document set capability enabling users to save their search results in an anonymous set (cart).

12.1 OBiBa Mica Settings

The settings under `Administration > Configuration > OBiBa Mica` are required to make Mica Drupal Client communicate with Mica Server and enables the administrator to customize the Mica Drupal interface and module functionality.

12.1.1 Graphic Settings

Administrators can enable/disable Graphics in Drupal blocks or Mica Drupal pages as well as several interface related settings such as colors and captions.

12.1.2 Settings

The settings listed below are essential to make the communicate between Mica Drupal Client and Mica Server possible:

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

Cache Image settings

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

12.1.3 Content Settings

Each content setting has specific configuration related to their web page interface and functionality. Visit each setting section for detailed usage.

12.1.4 Analysis Settings

The only setting here is the title of the Entities Count page.

12.1.5 Search Settings

Configuration related to the Search web page interface and functionality. Visit each setting section for detailed usage.

12.2 Mica Drupal Client Template Overriding

We will examine two distinct ways: sub-theme and custom module.

12.2.1 Dependencies

First of all, you need to get:

- Bootstrap theme
- OBiBa Bootstrap sub-theme

Further, see the Drupal Bootstrap Documentation.

12.2.2 Overriding Templates via a new Sub-Theme

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

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

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

You may do template overriding as follow:

- First, create a sub-template as described in the links provided above.
- Define `obiba_bootstrap` as the base theme in the `.info` file of that sub-theme.

Once the sub-theme is set, you can override each view generated by a module as shown below:

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

12.2.3 Overriding Templates via Custom Module

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

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

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

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

CHAPTER 13

Mica Python Client

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

13.1 Requirements

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

13.2 Installation

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

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

13.2.1 Debian Package Installation

Follow the OBiBa Debian Repository instructions and run:

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

13.2.2 RPM Package Installation

Follow the OBiBa RPM Repository instructions and run:

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

13.2.3 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)

13.3 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 14

Authorization Commands

Document authorization (on draft and published versions) management.

14.1 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]
```

14.1.1 Arguments

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

14.1.2 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`

14.1.3 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.

14.1.4 Extras

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

14.1.5 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
```

14.2 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]
```

14.2.1 Arguments

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

14.2.2 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`

14.2.3 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.

14.2.4 Extras

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

14.2.5 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
```

14.3 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]
```

14.3.1 Arguments

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

14.3.2 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`

14.3.3 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.

14.3.4 Extras

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

14.3.5 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 15

Document Commands

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

15.1 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]
```

15.1.1 Arguments

Argument	Description
`ID`	The collected dataset identifier

15.1.2 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.

15.1.3 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.

15.1.4 Extras

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

15.1.5 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
```

15.2 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]
```

15.2.1 Arguments

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

15.2.2 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.

15.2.3 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.

15.2.4 Extras

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

15.2.5 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'
```

15.3 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]
```

15.3.1 Arguments

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

15.3.2 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.

15.3.3 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.

15.3.4 Extras

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

15.3.5 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
```

15.4 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]
```

15.4.1 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.

15.4.2 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.

15.4.3 Extras

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

15.4.4 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))'
```

15.5 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.

15.5.1 Arguments

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

15.5.2 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`

15.5.3 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.

15.5.4 Extras

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

15.5.5 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 16

Other Commands

Other commands for advanced users.

16.1 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]
```

16.1.1 Arguments

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

16.1.2 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.

16.1.3 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.

16.1.4 Extras

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

16.1.5 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 17

Mica R Client

Mica R client, available as a R package, enables data exploration of a Mica server published content.

17.1 Installation

Requirements: R 3.x must be installed on the system. See more about R .

You can install Mica R package using devtools devtools:

```
# Install dependencies
if (!require("httr")) {
  install.package(c("httr"), dependencies=TRUE)
}
# Install from source code repository (see releases at https://github.com/obiba/micar/
↪releases)
devtools::install_github("obiba/micar", ref="1.0.0")
```

17.2 Usage

All the R commands that perform search return data frames. The query parameter that can be passed as an argument is the one that can be copied from the search page.

Example of usage:

```
# Load library
library(micar)

# Open connection
m <- mica.login(url="https://mica-demo.obiba.org")

# Get networks
```

(continues on next page)

(continued from previous page)

```
mica.networks(m)
mica.networks(m, query="network(in(Mica_network.studyIds,clsa))")
mica.networks(m, query="variable(in(Mlstr_area.Lifestyle_behaviours,Drugs))", locale=
↪"en", from=0, limit=10)

# Get studies, populations and DCEs
mica.studies(m)
mica.studies(m, query="study(in(Mica_study.methods-design,cohort_study))")
mica.studies(m, query="variable(in(Mlstr_area.Lifestyle_behaviours,Drugs))", locale=
↪"en", from=0, limit=10)
mica.study.populations(m)
mica.study.dces(m)

# Get datasets
mica.datasets(m)
mica.datasets(m, query="dataset(in(Mica_dataset.className,HarmonizationDataset))")
mica.datasets(m, query="variable(in(Mlstr_area.Lifestyle_behaviours,Drugs))")

# Get variables
mica.variables(m)
mica.variables(m, query="variable(in(Mlstr_area.Lifestyle_behaviours,Drugs))")
mica.variables(m, query="dataset(in(Mica_dataset.className,HarmonizationDataset))")

# Get taxonomies, vocabularies, terms
mica.taxonomies(m,target="variable")
mica.taxonomies(m,target="variable", query="sex", locale="en", taxonomies = list(
↪"Mlstr_area", "Mlstr_additional"))
mica.taxonomies(m,target="study")
mica.vocabularies(m,target="variable", query="cancer", locale = "en")

# Close connection
mica.logout(m)
```