Home

Mica2 Documentation

Mica is a web application used to create web data portals for epidemiological studies or consortia (see also Overview section). The following guide intend to provide the information and instructions necessary to install, customize, operate and use Mica2.

Mica2 is the successor of Mica which was a custom Drupal *distribution*. Mica2 is built on a more scalable architecture allowing to manage a much larger number of variables with even more flexibility. Drupal is still used but for the web portal front-end only: see Architecture, Servers and Clients .

> Unless there is a risk of confusion, we will call Mica2 *Mica* in what follows.

Mica Server Documentation

The Mica Server Documentation consists of two different manuals intended for system administrators:

- Mica Server Installation Guide which provides detailed instructions on how to install Mica server,
- Mica Server Configuration Guide which provides post-install instructions.

Mica Web Application User Guide

The Mica Web Application User Guide details how to enter studies, networks and datasets into Mica; it is intended for users of Mica (with various permission levels).

Mica Drupal Client Documentation

The Mica Drupal Client User Guide consists of two different manuals:

- Mica Drupal Client Installation Guide which is intended for system administrator who wish to deploy Mica Drupal Client along with Mica server.
- Mica Drupal Client Configuration Guide which is intended for the administrator of the web portal.

Mica Python Client Documentation

The Mica Python Client User Guide also consists of two manuals, mainly intended for system administrators willing to automate tasks:

- Mica Python Client Installation Guide which provides the instruction to install this client,
- Mica Python Commands which provides details on the command line tools.

OBiBa acknowledgements

All study and/or consortium websites developed using the Mica software must exhibit the “Powered by Mica” link to the page http://www.obiba.org/pages/products/mica/.

Overview

Contents of this Guide

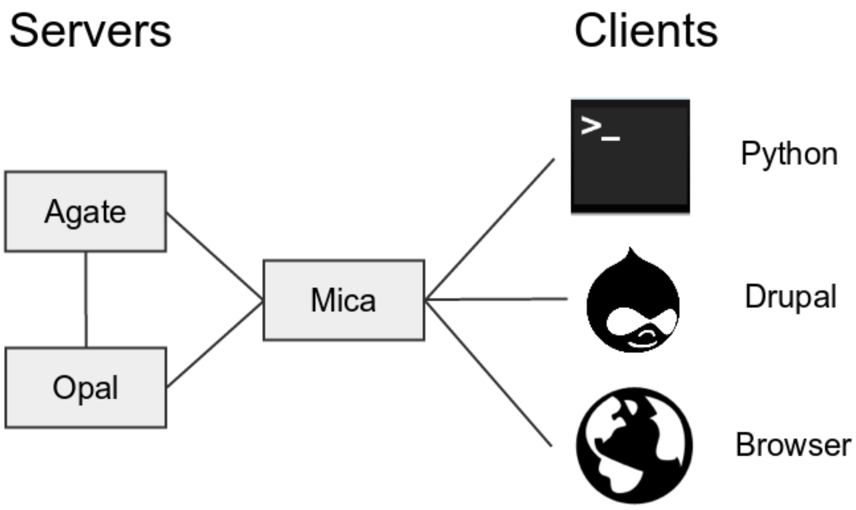
What is Mica?

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 :

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.

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



Documents

Overview

- Summary
- Types
    - Network
    - Individual Study
        - Population
        - Data Collection Event
    - Harmonization Study
        - Population
    - Collected Dataset

Summary

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 Requests (which is a form privately exchanged between a researcher and the study/consortium).

Types

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.

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.

Harmonization Study

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

Population

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

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.

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.

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.

Data Access Request

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

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

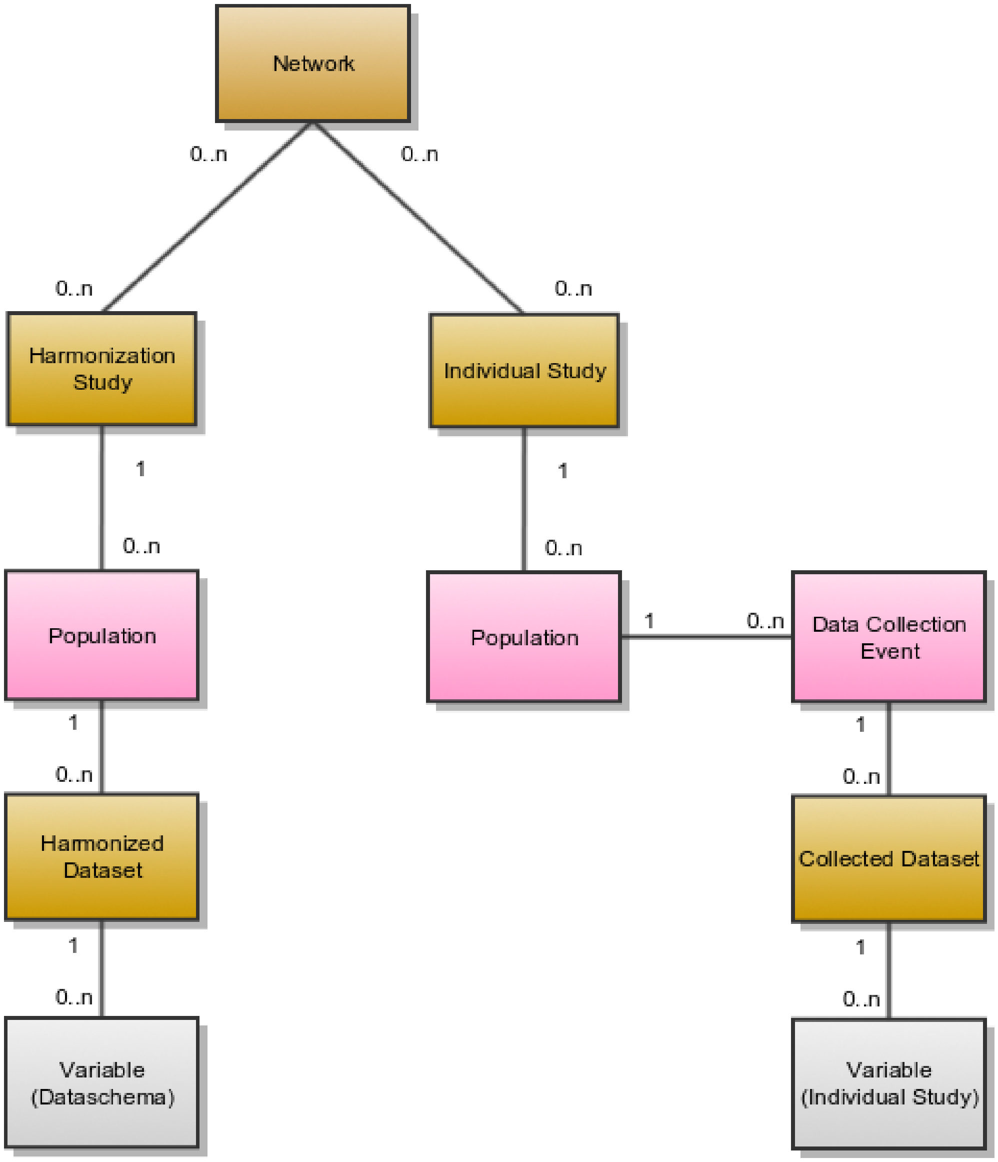
Search

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
- ...

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).



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.

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.

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.

Publication Flow

Overview

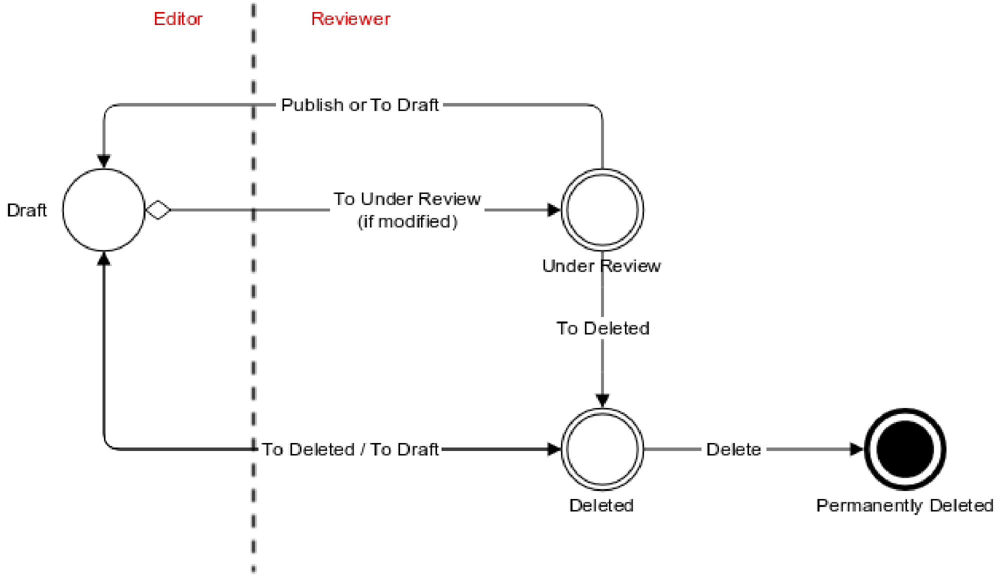
- Summary
- Revision Status
- Transitions

Summary

Documents (and their associated files) are all publishable documents (except Data Access Requests). 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:



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 permanent deletion. The document is not editable, nor publishable. Being published does not prevent a document from going into the `Deleted` state: the un-publication will be effective when the deletion is permanent. Note also that the document can be un-published at any time.	✗	✗	✓	• `Draft` • `Under Review`	• `Draft`

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`	-

Architecture, Servers and Clients

Overview

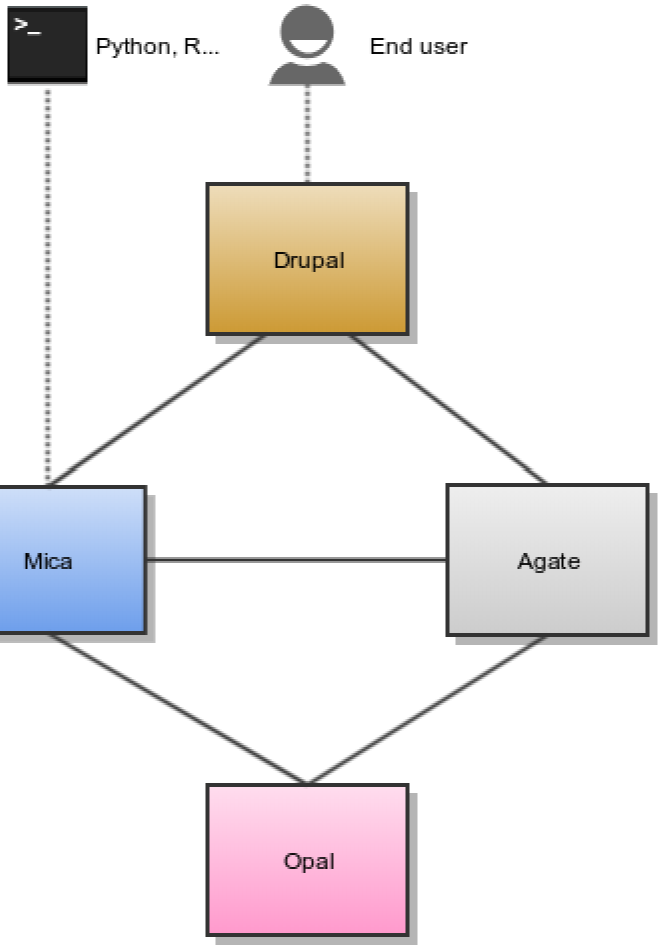
Summary

The architecture of Mica is split in several servers:

- Mica server: holds the domain and controls what is to be published,
- Opal server: holds the data with their dictionary and provide statistics services,
- Agate server: user directory for data access requests management.
- Drupal server: the web portal front-end using Mica server as its source of published documents and Agate server as its user directory.

Mica, Opal and Agate are applications developed by OBiba. OBiBa also provides extensions for the Drupal application. Each of these OBiBa servers expose web services to allow easy interconnection. The *Mica web portal* is the final application which leverages each server specific domain and functionalities in one.

The following diagram shows how these servers are linked together:



Mica Server

Installation and configuration guides can be found in the section Mica Server Administrator Guide.

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.

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 Server Administrator Guide.

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

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 Server Administrator Guide.

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.

Mica Server Administrator Guide

Contents of this Guide

Introduction

This guide is for whoever will set up Mica--typically, a system administrator.

The guide covers hardware and software requirements and includes procedures for deploying Mica on a server.

When requirements are met, administrators can follow:

1. the Mica Server Installation Guide,
2. and the Mica Server Configuration Guide.

Mica server package is available in different format:

Type	Package Repository
Debian	Debian Repository
RPM	RPM Repository
Zip	Zip Repository
Docker	Docker Repository

Requirements

Server Hardware Requirements

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

Server Software Requirements

Software	Suggested version	Download Link	Usage
Java	>= 1.8.x	Java Oracle Download	Java runtime environment
MongoDB	>= 3.2.x	MongoDB downloads	Database engine

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

Mica Server Installation Guide

Contents of this Guide

Introduction

The aim of this guide is to explain in details how to install Mica server application.

Installing Mica

Mica is distributed as a Debian package and as a zip file. Installing Debian package is recommended on Debian-like Linux systems.

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 Mica Server Configuration Guide.

Installation of Mica Debian package (recommended)

Mica is available as a Debian package from OBiBa Debian repository.

Download Mica Debian package

To proceed installation, do as follows:

1. **Install Debian package**. Follow the instructions in the repository main page for installing Mica.

2. **Manage Mica Service**: after package installation, Mica server is running: see how to manage the Service.

Installation of Mica RPM package (recommended on RPM-based Linux distributions)

Mica is available as a RPM package from OBiBa RPM repository.

Download Mica RPM package

To proceed installation, do as follows:

1. **Install RPM package**. Follow the instructions in the RPM repository main page for installing Mica.

2. **Manage Mica Service**: after package installation, Mica is running: see how to manage the Service.

Installation of Mica Zip distribution

Mica is also available as a Zip file.

Download Mica Zip package

To install Mica zip distribution, proceed as follows:

1. **Download Mica distribution**
2. **Unzip the Mica distribution**. Note that the zip file contains a root directory named `mica2-dist-` *x* . *y* . *z* (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.

3. **Create an MICA_HOME environment variable**
4. **Separate Mica home from Mica distribution directories (recommended)**. This will facilitate subsequent upgrades.

Set-up example for Linux

```
mkdir mica-home
cp -r mica-dist-x/conf mica-home
export MICA_HOME=`pwd`/mica-home
./mica-dist-x/bin/mica2
```

5. **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.

Upgrading Mica

There are two way of upgrading Mica, the one you use depends on the way you installed Mica in the first place.

Upgrading Debian Package

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

```
apt-get install mica2
```

Manually upgrade from Mica Zip distribution

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

Configuring Mica

See Mica Server Configuration Guide.

Executing Mica

Launching Mica Server

Service

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

Service Environment

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/mica2`.

Service Script

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

```
service mica2 help
```

Service Logs

The Mica service log files are located in `/var/log/mica2` 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.

Command Environment

Configure the following environment variables:

Environment variable	Required	Description
`MICA_HOME`	yes	Path to the Mica "home" (configuration and file system) directory. Note: `MICA_HOME` should point to the location (the parent) of the `conf` directory.

`JAVA_OPTS`	no	Options for the Java Virtual Machine.
		For example:
		`-Xmx4096m -XX:MaxPermSize=256m`
		To change the defaults update:
		`bin/mica2` or `bin/mica2.bat`
		Note: If there is not enough memory allocated to the JVM Mica may run slow and/or run out of memory running commands using large data sets.

Regular Command

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

```
mica2
```

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

Command Logs

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.

Using Mica

Accessing Mica from the Web

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

- `http://<servername>:8082` will provide a connection without encryption,
- `https://<servername>:8445` will provide a connection secured with ssl.

See Mica Web Application User Guide for more details.

Installation Troubleshooting

If you encounter an issue during the installation and you can't resolve it, please report it in our Mica2 Issue Tracker.

Reporting errors

Mica2 logs can be found in `/var/log/mica2`. If the installation fails, always refer to this log when reporting an error.

Upgrading to Mica 2.x

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

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

> 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*:

Mongo console

```
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
```

Upgrading to Mica 3.x

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

> 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.

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

Mica Server Configuration Guide

Contents of this Guide

Prerequisites

Mica server is installed, following the instructions described in Mica Server Installation Guide.

Mica Server Configuration Files

Mica has some configuration files that allows fine tuning of your Mica server. See Mica Configuration Files documentation.

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
- 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
    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):

/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>
```

Mica Configuration Files

Overview

Summary

The following configuration files are available in the **`MICA_HOME/conf`** directory.

File	Description
shiro.ini	The user authentication file.
application.yml	The main configuration file to be edited before starting Mica server.
logback.xml	The logs configuration file (for advanced settings).

Mica Server Main Configuration File

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.

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.

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

> 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:

MongoDB Console

```
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/mica2/application.yml* for the above user:

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

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.

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.

Agate Server Configuration

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

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

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.

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.

> 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 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
```

Other Configurations

Configurations not listed above are available in the main configuration. They are for internal use only.

Mica Server Logs Configuration File

The file is **MICA_HOME/conf/logback.xml** allows to specify how Mica server should log it's internal activity. Remember that outputing too much logs could affect the performance of your server. Default log file is **MICA_HOME/log/mica.log**. Mica server needs to be restarted for the changes to be effective. Default configuration is usually enough.

User Directories Configuration

Overview

- Summary
- User Directories
  - File Based User Directory

Summary

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`**.

More information about how to configure Shiro using the `ini` file can be found here.

Default configuration is a static user `'administrator'` with password `'password'` (or the one provided while installing Mica2 Debian package).

> Remember to change default administrator password if you did not installed Mica2 with the Debian package!

User Directories

By default Mica server has several built-in user directories:

- 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`**.

> 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.

conf/shiro.ini

```
# =======================
# Shiro INI configuration
# =======================
[main]
# Objects and their properties are defined here,
# Such as the securityManager, Realms and anything else needed to build the
SecurityManager
#securityManager.sessionManager.globalSessionTimeout = 10000
[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$500000$wwA43u/bAd6R/w0MYdJvYQ==$6Emj+fIAgo2I
JFV1SyDQTXcr9zVUuzJQha49XW4qM+I=,mica-administrator
anonymous =
$shiro1$SHA-256$500000$dxucP0IgyO99rdL0Ltj1Qg==$qssS60kTC7TqE61/JFrX/OEk0j
sZbYXjiGhR7/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):

Password encryption for Linux

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

Mica Plugins Installation Guide

Contents of this Guide

Introduction

From Mica 3.1.0, new services can be added as plugins discovered at runtime. This guide specifies how to install and configure Mica plugins.

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

Installation

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

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.

Manual Installation

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

1. **Download the plugin** of interest (zip file) from OBiBa Plugins Repository,
2. **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.
3. **Read the installation instructions** (if any) of the plugin to identify the system dependencies or any other information,
4. **Restart Mica**.

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.

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.

Mica Web Application User Guide

Contents of this Guide

Introduction

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

See the Mica Domain 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

Requirements

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

Documents Management

View, Edit, Publish

Overview

Summary

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

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.

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.

Individual 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.

Manage Data Collection Events

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

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.

Collected Dataset Operations

Manage Study Table

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

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.

Revisions

Overview

- Summary
- Operations
    - View
    - Restore

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.

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.

Files

Overview

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.

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.

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 detail panel.

Description Edition

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

Folder Specific Operations

Folder Creation

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

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.

Permissions

Permissions are applied to the file's associated document and not on the file itself. The following table describes each role the corresponding permitted operations:

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

Comments

Overview

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.

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.

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.

Permissions

Overview

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 page.

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 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.

Data Access Requests Management

Summary

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

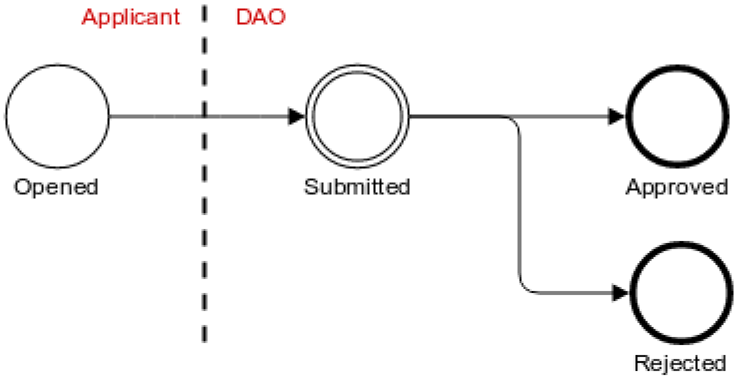
Request Workflow

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 the 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.

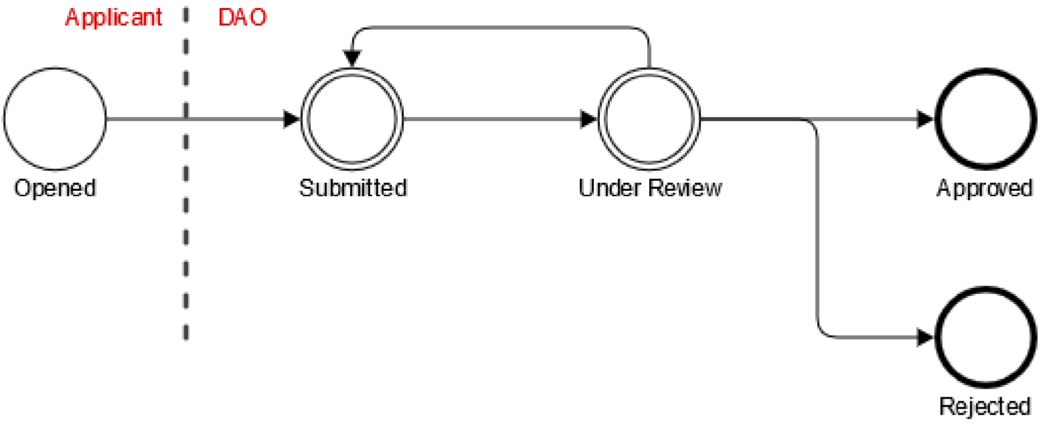
After the request is submitted, a *data access officer* will *review* it in order to *approve* or *reject* it. The data access officer can also *reopen* (or *conditionally approve*) the request if he considers that modifications are needed. This workflow, as well as email notifications that can be sent 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:

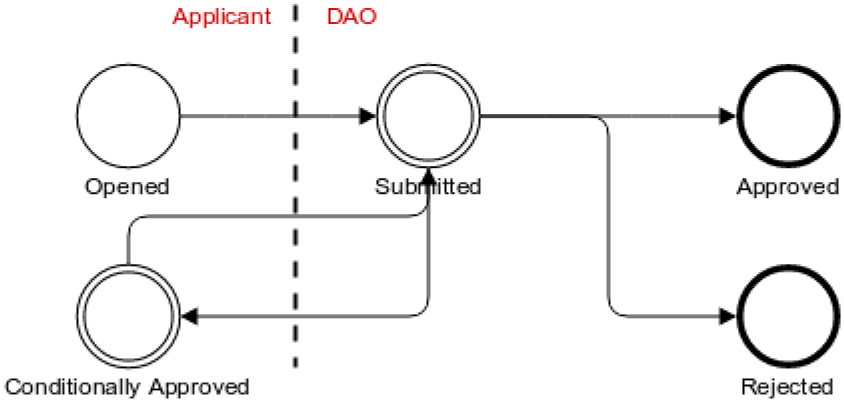
- no `Under Review` intermediate state
- no `Conditionally Approved` intermediate state
- `Approved` state is final
- `Rejected` state is final



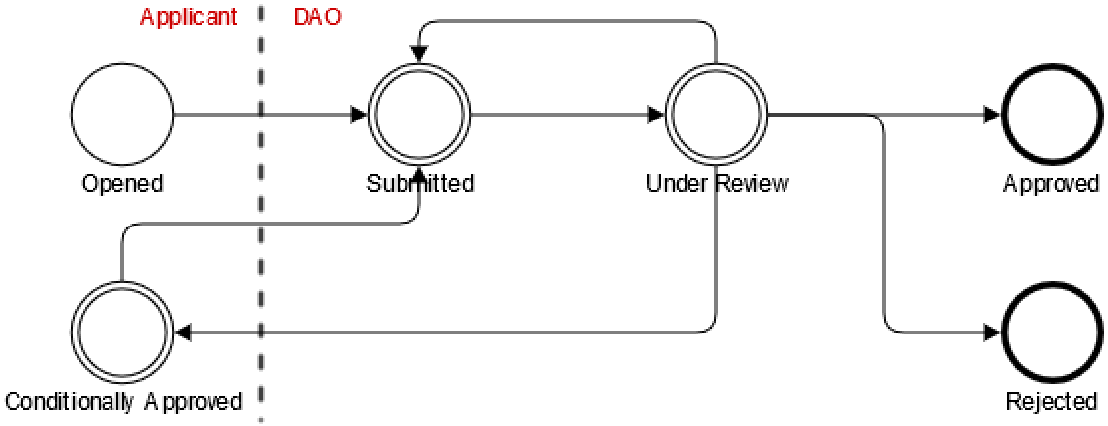
When the `Under Review` intermediate state is activated (which is the default configuration), the flow is:



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`

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).

Comments

At any step, the user and the data access officer can add comments in order to collaborate.

History

A *history* of all actions performed on the request is kept to track its status.

Administration

Overview

- Summary

Summary

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

System Administration

- General Administration
- Notifications Administration
- Style Administration
- Translations Administration
- Caching Administration
- Indexing Administration
- Application Metrics Administration
- Logs Administration

General Administration

Overview

- Summary
- Configuration
    - Properties
    - Server Identification
    - Content
    - Data Source
    - Sections
    - Roles
    - Encryption keys
        - Create a (self-signed) certificate
        - Import a certificate
    - Opals Credentials

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.
Sections	**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.

> **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.

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

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.

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.

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.

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.

> 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.

> 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:

> 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 step 1-6 for any other table you want the user to have access to.

Notifications Administration

Overview

- Summary
- Configuration

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.

Style Administration

Overview

Summary

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

Translations Administration

Overview

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

Add Translation

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.

Download

Translations can be downloaded into a JSON or Gettext format.

- **JSON**: click on "Download" then on "JSON (all languages)" to download custom translations in JSON format. If no custom key has been added, the file is empty.
- **Gettext**: click on "Download" then on "Gettext (<Language>)" to download all Mica's translations into a language-specific .PO file.

Search/Edit a translation

The search allows to find and modify any existing translation key and specify it in all language.

- **Edit a key value:** select a language tab, then in the search box, enter either a key name or a key value. Identify a key and change its value.
- **Translate a key in another languages**: select a language tab and search a translation using the search box. Copy the key name, select a language table, paste the key name in the search box and change its value.

Once all changes performed, scroll down and click on Save.

Delete custom key

All custom translation keys can be deleted. To do so, search a custom key using the search box, click in the Value field. A delete icon appears. Click on the delete icon, scroll down and save.

Caching Administration

Overview

- Summary
- Definitions
- Operations

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.

Operations

Indexing Administration

Overview

- Summary
- Index Type
- Operations

Summary

The 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).

Operations

Choose the type of index then click on the button "Build index" in the Action column to build the corresponding index.

Application Metrics Administration

Overview

- Summary
- Files
- Definitions
- Operations

Summary

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

Files

From the Administration page, the system administrator can browse Mica's file system.

Definitions

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. About the later, note that one can have the list of all threads by clicking on the "eye" icon at the right of the title *Th reads*.

Operations

Logs Administration

Overview

Summary

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

Definitions

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.

Operations

Content Administration

Document Configuration

Overview

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*.

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 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.

Study Specific Form

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.

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. A criterion is described by a unique name (identifier), a title, a description, a type, a document field it's based on and search characteristics ; Repeatable to Search for exact match or contained value, Hidden to exclude the criteria from the search and Localized for multilingual fields.

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).

Data Access Request Administration

Overview

- Summary
- Definitions
    - Application Form
    - Notifications
    - Settings
- Operations

Summary

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

Definitions

Application Form

The web application form can be specified both in terms of data structure (*Schema*) and data display and validation (*Definition*). See form schema 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.

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.

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.

Operations

Data Discovery

A search and query interface that allows investigators to quickly and easily identify studies of interest and available data items.

- Building Queries
- Exploring Results

Building Queries

- Summary
- Operations
  - Create a search criterion
    - Search box
    - Search properties
    - Classifications
  - Update a search criterion

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**

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.

Update 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

Exploring Results

- Summary
  - List
  - Coverage
  - Graphics
  - Download 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.

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 Mica 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.

Tutorials

- Create and Publish a new Study
- Document Form Configuration

Create and Publish a new Study

Overview

Summary

This page contains a tutorial to create, publish and manage a given study**.** Each study in Mica can be in **"Draft Mode"** where the study can be edited and **"Review Mode"** which is an intermediate phase required to validate and publish the study. See the Publication Flow for more details.

Create a new study

The following steps summarize how to create a new study.

1. Sign in Mica as a document Reviewer. See Document Permission for further details.
2. In the main menu, click on "Studies" then on "+Add Study".
3. Fill in all important information. Then, choose wisely the Acronym which will be used as a study identifier. Note that the study acronym, name and objective are mandatory fields and should be filled-in in all defined languages.
4. Scroll down and click "Save" to create your study. The created study is now in "Draft Mode".

Once created, a study can be managed through five tabs: "View", "Revisions", "Files", "Comments" and "Permissions". More details are given in the section Documents Management.

Add Populations and Data Collections Events

The participants targeted by the study can be described by creating populations and specifying all required information as number, sex and age of participants, selection criteria or countries of residence. The study can collect participants information on one or several waves of collect called Data Collection Events. Specifying the data collection event allows Mica to create a Study TimeLine.

Add Population

In the study "View", scroll down and click on "+Add Population". Fill-in needed information: ID, name, description, sex, age of participants, etc. and click on Save.

Add Data Collection Event

After creating all the study populations, each population can have several data collection events. Choose and existing population, click on "+Add Data Collection Event", specify all information and click on save.

Add and manage Files

Files can be attached to an existing study through the tab "Files" in a study page. Refer to the Files Management Section for more details.

Edit and publish the study

Each study may go through several stages before the final publication on the web site. See the Publication Flow for more details.

Add Study Members

A list of members can be associated to a given study grouped in Roles. The roles are configured throw the Mica's General Administration, section Roles.

1. In order to specify roles, click on "Administration" then "General" in Mica's main menu. Scroll down and click on "+Add Role" to add as many roles as needed.
2. Once the roles are defined, click on Studies menu, choose a study, scroll down to the Members section and add as many members as required.

Document Form Configuration

Overview

Summary

Mica provides a user experience to configure any mica document (Network, Study, Dataset, Research Project, Data Access Request). It is possible to fully customize any document form by adding, modifying or removing fields. In addition, the search module allows to search any document according to added/altered fields if the search taxonomy is well configured. Therefore, here's some test scenario to experiment these functionalities.

> **Taxonomy**
>
> A taxonomy is a two-level hierarchical classification of existing documents based or their fields. The first level specifies the vocabularies and the second-level contains corresponding terms.

Scenario 1: Add a number field to the Network form

Below is the required steps to add the field "Publications" to Network, to create its corresponding search criterion in the Network taxonomy as a range value: *:5, 5:10, 10:* and to search Mica's content according to the added field.

Step 1: Add the new field in the schema form

In the Network administration Form section, apply the following actions.

1. Add the code below in the Schema TAB. This code defines two fields, "Website" and our new field "Publications".

Network Schema

```
{
  "type": "object",
  "properties": {
    "publications": {
      "title": "Publications",
      "type": "integer"
    },
    "website": {
      "title": "t(website)",
      "type": "string"
    }
  },
  "required": []
}
```

2. Add the following code in the Definition TAB

Network Definition

```
[
  {
    "type": "fieldset",
    "items": [
      "website",
      "publications"
    ]
  }
]
```

3. Click on the Preview tab, you can see that a new "Publications" field is added. Save the configuration

Step 2: Define a search criterion on the added field

In the Network administration Search section, perform the actions below.

1. Click on "+ Add Criterion" and fill in the form with the vocabulary required information. The criterion should be linked to the correct field in the schema.Therefore, in the section Field, start typing "publications" and select the correct name "model.publications". Click on "Apply".
2. In the Criteria list, click on "publications" and add three terms ; "Less than 5", "Between 5 and 10" and "More than 10", by specifying the good number rang for each term.

Step 3: Populate and search

In order to search documents according to the new added field, edit some Networks and populate the field "Publications" for each, modify two networks with Publication = 4 and one network with Publications=7, save and publish. After that, go to the Mica search, search by the "Publications" field. The search results displays two networks with the correct counts for each publications category (term).

Scenario 2: Add a single choice field to the Study form

Below is the required steps to add the field "Status" to the Study document, to create its corresponding search criteria with the values "Ongoing" and "Completed" and to search Mica's content according to the added field.

Step 1: Add the new field in the schema form

In the Study administration Form section, apply the following actions.

1. In the Schema TAB, in the "properties", add the following code that defines the "Status" field.

Study Status Schema

```
"status": {
 "type": "string",
 "enum": [
     "ongoing",
        "completed"
    ],
    "title": "Status"
},
```

2. In the Definition TAB, add the following code.

Study Status Definition

```
{
 "key": "status",
 "type": "radios"
},
```

3. Click on the Preview tab, you can see that a new "Status" field is added. Save the configuration.

Step 2: Define a search criterion on the added field

In the Study administration Search section, perform the actions below.

1. Click on "+ Add Criterion" and fill in the form with the criterion required information. The criterion should be linked to the correct field in the schema form.Therefore, in the section Field, start typing Status and select the correct name "model.status". Click on "Apply".
2. In the criteria list, click on the status criteria and add two terms, "Ongoing" and "Completed".

Step 3: Populate and search

In order to search documents according to the new added field, edit and publish some Studies with the status Ongoing an others study with a status Completed. After that, go to the Mica search, search ongoing key to display corresponding studies.

Mica Drupal Client User Guide

Contents of this Guide

Introduction

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.

When the requirements are met, administrators can follow:

- the Mica Drupal Client installation Guide,
- and the Mica Drupal Client Configuration Guide.

Requirements

Server Hardware Requirements

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

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

Mica Drupal Client Installation Guide

Contents of this Guide

Introduction

The *Mica Drupal Client* is not an application. It is an extension of the Drupal server application. This documentation describes how to install Mica related Drupal modules and themes. See also a general presentation about Drupal modules-themes.

Supported OS :

- Debian 9
- CentOS 7

Dependencies

This documentation assumes that PHP 5.6 is installed

Required software

System dependencies

For Debian-based 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
```

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
source .bashrc

# Verify Drush install
drush status

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

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).

> **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.

Drupal Modules and Theme Installation

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

Name	Type	Drupal link	Usage
`obiba_mica`	module	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/
```

These extra steps are needed to have a functional drupal on CentOS

Activate mod_rewrite in drupal

Add at the end of the file "/etc/httpd/conf/httpd.conf" :

```
<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 of ssl, 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

Drupal Modules and Theme 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

# 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.

Mica Drupal Client Configuration Guide

Contents of this Guide

Introduction

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

> If you decide to disable one of OBiBa Drupal module, make sure you know exactly what it does. As a general rule, *all* modules should be enabled in order to make Mica Client works. There are, however, two notable exceptions to this rule:
>
> - You may disable both the "Data Access Request" and the "OBiBa Auth" modules in the case you don't intend to use the Data Access Request feature provided by Mica.
> - Not an OBiBa module *per se*, but one which Mica Client use extensively is the Google Chart module (in the Chart section). If you intend to use Highcharts in your portal, you may want to activate the module there and disable the Google Chart modules.

Configuration

This section deals with the various configuration options provided in the section **Configurations** of the administration panel and enabled by

OBiBa Drupal modules.

OBiBa Mica settings

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

OBiBa Study Server (MICA)

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

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

Data Access Request

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

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

Statistics Settings

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

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.

Networks, Studies, Datasets and Variable Search

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

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

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

Study, Dataset and Variable Content

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

> At this time, there is no way to configure the display of the information pertaining to a network.

Taxonomies

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

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

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

Translation

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

Mica Drupal Client Templating

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

Dependencies

First of all, you need to get:

- **Drupal theme**. Bootstrap which you can get here.
- **Base theme.** OBiBa Bootstrap a sub-theme of Drupal which you can get here.

Further, the documentation related to sub-theming in Drupal can be found here.

Overriding templates via a new sub-theme

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

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

> 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:

1. First, create a sub-template as decribed in the documentations hyperlinked above

2. Define obiba_bootstrap as the base theme in the .info file of that sub-theme.

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

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

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:

1. Copy the template that you want to override in the folder "Template" of the custom module,
2. 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;
}
```

Mica Python Client User Guide

Contents of this Guide

Summary

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

Installation

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

- use a Python package

- use the Debian package manager

Please read Mica Python Client Installation Guide for more details.

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.

Commands

Permission Commands

Permission commands allow managing permissions on these Mica document **draft** versions:

Command	Description
perm-network	Set/remove network related permissions.
perm-individual-study	Set/remove individual study related permissions.
perm-harmonization-study	Set/remove harmonization study related permissions.
perm-collected-dataset	Set/remove collected dataset related permissions.
perm-harmonized-dataset	Set/remove harmonized dataset related permissions.
perm-project	Set/remove project related permissions.

Access Commands

Access commands allow managing access on these Mica document **published** versions:

Command	Description
access-network	Set/remove network access.
access-individual-study	Set/remove individual study access.
access-harmonization-study	Set/remove harmonization study access.
access-collected-dataset	Set/remove collected dataset access.
access-harmonized-dataset	Set/remove harmonized dataset access.
access-project	Set/remove project access.
access-file	Set/remove file access.

Other Commands

Command	Description

update-collected-dataset	Update and/or (un)publish a collected dataset.
update-collected-datasets	Update and/or (un)publish a set of collected datasets.
file	Upload, download and manage publication of files.
import-zip	Import data from zip file(s) that have been extracted from Mica 1
rest	Request directly the Mica REST API, for advanced users
search	Search for documents and list them in CSV format.

Requirements

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

Mica Python Client Installation Guide

Contents of this Guide

Introduction

There are two ways to install the Mica Python Client package on your computer:

- either from the Debian package (Debian-base Linux only),
- or from the Python package (Linux or Windows).

Installing Mica Python Client

Installation of the Debian package (recommended)

Mica Python client is available as a Debian package from OBiBa Debian repository.

Download Mica Python Client Debian package

Then Install the package:

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

> Some Debian-based systems (Debian Wheezy or Ubuntu Precise releases) only provide an older version of *python-protobuf* package dependency. When executing the mica python client the following error is reported:
>
> `ImportError: cannot import name enum_type_wrapper`
>
> This can be fixed manually using the commands:
>
> `apt-get purge python-protobuf`

```
apt-get install python-pip

pip install protobuf>=2.5.0

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

Installation of the RPM package (recommended for Fedora-based systems)

Mica Python client is available as a RPM package from OBiBa RPM repository.

Download Mica Python Client RPM package

Then Install the package:

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

Installation of the Python package

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

Install on Linux or Mac

1. Download the most recent version from:

   Download Mica Python Client package
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
   ```

> 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 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 http://download.obiba.org/mica/stable/mica-python-client-X.XX.zip 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)

Testing the Installation

Test the installation by performing a REST call to your Mica server:

```
mica rest /studies --mica <YOUR-MICA-SERVER-URL>  --user <YOUR-USER>
--password <YOUR-PASSWORD> --json
```

Mica Python Commands

The following commands target power users who are comfortable with command-line tools and wish to batch process and automate tasks. Below is an example of a task using the Mica Python REST command to backup all individual study documents:

```
mica rest /draft/individual-studies -m GET -mk http://localhost:8082 -u
administrator -p password -a application/json >
/var/backups/mica/studies/individual-studies.json
```

To comply with command-line conventions the documentation uses:

- <SOMETHING> to imply a *placeholder* or *mandatory* arguments.
- [SOMETHING] to imply *optional* arguments.

Access Commands

Document Access Commands

Contents of this page

Synopsis

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]
```

Document

Options	Descriptions
`DOCUMENT`	Mica document: network, individual-study, harmonization-study, collected-dataset, harmonized-dataset

ID

Options	Descriptions
`ID`	Identifier of the document

Credentials

Authentication is done either by username/password.

Options	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.

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

Extras

Options	Description
--help, -h	Displays the command's help message
--verbose, -v	Verbosely executes the command

Examples

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
```

File Access Command

Contents of this page

Synopsis

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]
```

PATH

Options	Descriptions
`PATH`	Path to the file in the Mica file system

Credentials

Authentication is done either by username/password.

Options	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.

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

Extras

Options	Description
--help, -h	Displays the command's help message
--verbose, -v	Verbosely executes the command

Examples

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
```

Permission Commands

Contents of this page

Synopsis

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]
```

Document

Options	Descriptions
`DOCUMENT`	Mica document: network, individual-study, harmonization-study, collected-dataset, harmonized-dataset

ID

Options	Descriptions
`ID`	Identifier of the document

Credentials

Authentication is done either by username/password.

Options	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.

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 permission will be granted
--type TYPE, -ty TYPE	Subject type: user or group

Extras

Options	Description
--help, -h	Displays the command's help message
--verbose, -v	Verbosely executes the command

Examples

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
```

Update Collected Dataset Command

Overview

Synopsis

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]
```

Id

Options	Descriptions
`ID`	The collected dataset identifier

Credentials

Authentication is done either by username/password or by public/private key files.

Options	Description
`--mica MICA, -mk MICA`	Mica server base url (default: http://localhost:8082).
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

Options

Mutually exclusive actions that can be performed on the file system. If omitted, the JSON representation of the file is returned.

Options	Descriptions
`--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.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose outut

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
```

Update Collected Datasets Command

Overview

Synopsis

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]
```

Id

Options	Descriptions
`ID`	A regular expression to filter the collected dataset identifiers.

Credentials

Authentication is done either by username/password or by public/private key files.

Options	Description
`--mica MICA, -mk MICA`	Mica server base url (default: http://localhost:8082).
`--user USER, -u USER`	User name. User with appropriate permissions is expected depending of the REST resource requested.
`--password PASSWORD, -p PASSWORD`	User password.

Options

Mutually exclusive actions that can be performed on the file system. If omitted, the JSON representation of the file is returned.

Options	Descriptions
`--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 dataset.
`--unpublish, -un`	Unpublish the collected dataset.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose outut

Example

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

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

File System Command

Overview

Synopsis

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

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

Path

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

Credentials

Authentication is done either by username/password or by public/private key files.

Options	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.

Options

Mutually exclusive actions that can be performed on the file system. If omitted, the JSON representation of the file is returned.

Options	Descriptions
`--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.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose outut

Example

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

```
mica file /study/foo/bar.pdf -mk http://localhost:8082 -u administrator -p
password -j
```

Download file `/study/foo/bar.pdf`

```
mica file /study/foo/bar.pdf -mk http://localhost:8082 -u administrator -p
password --download > bar.pdf
```

Upload a file to `/study/foo`

```
mica file /study/foo -mk http://localhost:8082 -u administrator -p password
--upload ~/bar.pdf
```

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

```
mica file /study/foo/bar.pdf -mk http://localhost:8082 -u administrator -p
password --status UNDER_REVIEW
mica file /study/foo/bar.pdf -mk http://localhost:8082 -u administrator -p
password --publish
```

Import Zip Command

Overview

Synopsis

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
```

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

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

Credentials

Authentication is done either by username/password or by public/private key files.

Options	Description
`--mica MICA, -mk MICA`	Mica server base url.
`--user USER, -u USER`	User name. User with study edition permission is required.
`--password PASSWORD, -p PASSWORD`	User password.

Options

Options	Descriptions
`path`	Path to the zip file or directory that contains zip files to be imported.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message.
`--verbose, -v`	Verbose output.
`--publish, -p`	Publish imported study.

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
```

REST API Command

Overview

Synopsis

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

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

Web Service

Options	Descriptions
`ws`	Web service path, see below for some examples.

Credentials

Authentication is done either by username/password or by public/private key files.

Options	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.

Options

Options	Descriptions
`--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.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message

`--verbose, -v`	Verbose outut

Examples

Get all the published studies visible to an anonymous user.

```
mica rest /studies -m GET -mk http://localhost:8082 -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 http://localhost:8082 -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
http://localhost:8082 -u administrator -p password -a application/json -j
```

Search Command

Overview

- Synopsis
    - Credentials
    - Options
    - Extras
- Examples

Synopsis

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]
```

Credentials

Options	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.

Options

Options	Descriptions
`--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.

Extras

Options	Descriptions
`-h, --help`	Show the command help's message
`--verbose, -v`	Verbose outut

Examples

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_stu
dy.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_stu
dy.methods-design,cohort_study))'
```

Mica R Client User Guide

Contents of this Guide

Summary

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

Installation

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

You can install Mica R package using 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")
```

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
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)
```