EXCELERATE Deliverable D9.2

Project Title:	ELIXIR-EXCELERATE: Fast-track ELIXIR implementation and drive early user exploitation across the life sciences	
Project Acronym:	ELIXIR-EXCELERATE	
Grant agreement no.:	676559	
	H2020-INFRADEV-2014-2015/H2020-INFRADEV-1-2015-1	
Deliverable title:	Report on implementation of submissions interface and API	
WP No.	9	
Lead Beneficiary:	1: EMBL (EBI)	
WP Title	Use Case D: ELIXIR framework for secure archiving, dissemination and analysis of human access-controlled data	
Contractual delivery date:	31 July 2018	
Actual delivery date:	30 July 2018	
WP leader:	Thomas Keane (EMBL-EBI), Jordi Rambla (CRG)	1: EMBL (EBI) 8. CRG
Partner(s) contributing to this deliverable:	8 - CRG, 1 - EMBL (EBI)	

Authors and Contributors:

Sabela de la Torre (CRG – ES), Mauricio Moldes (CRG – ES), Jordi Rambla (CRG – ES)

Table of contents

Executive Summary 2

Impact 3

Project objectives 3

Delivery and schedule 4

Adjustments made 4

Background information 4

Appendix 1: Implementation of submissions interface and API 9

Introduction 9

The Submission Process – Common Aspects 9

Objects required 9

Identifiers 11

Internal identifiers 11

Submission and object status 11

The Submitter Portal 13

The Submission API 14

Submission API Response 16

HTTP response codes 17

How to use the API 18

Human and Process related aspects 30

Conclusions 30

Next Steps 31

Links 31

1. Executive Summary

Making submissions to EGA as easy and straight as possible is key to gather more data and better quality data to be shared with the community. We gathered feedback from users about the existing solutions to submit metadata to the EGA and have designed, developed and deployed a new set of tools to fulfil the requirements communicated by them.

The set of tools currently consist of the Submission API (a REST API) and a web based user interface, the Submitter Portal. The latter is leveraging the former, thus we can check

that the API is well suited for the features it aims to implement. The solution was designed with flexibility or extensibility in mind, thus, adding new features should be easy.

The Submission API is also meant for allowing advanced users and consortia to customize their own interfaces or tools to interact with the EGA. They can add any project-wise layer as a frontal or pre-process before actually doing the submission. Indeed, they can check the validity of the submission as often as required.

Additionally, the Submission ensemble provides better guidance and feedback along the process, thus this process is easier to accomplish.

The production phase started in January 2018 and it is the currently recommended channel for submitting to the EGA.

2. Impact

The Submitter Portal and API ensemble started production phase in January'18. Since then and until end of May'18, it has shown the following usage:

- 16% of analyses (916 submitted through the API in absolute numbers)
- 21% DACs (28)
- 12% datasets (32)
- 3% experiments (3,762)
- 21% policies (26)
- 3% runs (3,784)
- 8% samples (10,686)
- 13% studies (35)

The differences in the percentages are probably due that submission could be completed by different channels and by the fact some heavy users (e.g. The Sanger Institute) have existing mechanisms for managing highly repetitive parts, like the "run", "experiment" or "sample" objects. The goal is for new users to leverage these new tools and existing users to progressively migrate to them.

3. Project objectives

With this deliverable, the project has reached or the deliverable has contributed to the following objectives:

No.	Objective	Yes	No
1	To upgrade and make more portable -omics data collection and submission tools utilizing the European Genome-phenome Archive (EGA) as the core of an ELIXIR community secure data sharing network for - omics data.	X	
2	To enable value-added services at project specific, regional, or national resources by establishing ELIXIR-wide community facing tools that allow local resource owners and developers to add value to their systems through data and metadata services from the EGA.	X	

Horizon 2020 grant n. 676559

3	To extend and generalise the system of access authorization management and high volume secure data transfer developed in the EGA project to address the secure data access needs across ELIXIR resources and open new modes of secure data access such as through public and private clouds.	X

4. Delivery and schedule

The delivery is delayed: Yes ☑ No

5. Adjustments made

No adjustments were made

6. Background information

Background information on this WP as originally indicated in the description of action (DoA) is included here for reference.

Work package number	9	Start date or starting event:	month 1
Work package title	**Use Case D: ELIXIR framework for secure archiving, dissemination and analysis of human access-controlled data**		
Lead	1 - EMBL-EBI		
Participant number and person months per participant **1 - EMBL (34 PM)**, 5 - UTARTU (16 PM), 6 - NBIC (0 PM), UMCG (LTP to NBIC) (5 PM), 8 - CRG (22.3 PM), 14 - UPF (23.5 PM), 20 - CSC (24 PM), THL (LTP to CSC) (3 PM), 24 - UiO (6 PM), 45 - UU (2 PM), SU (LTP to UU) (4 PM).			
Objectives This Work Package has three main objectives: To upgrade and make more portable -omics data collection and submission tools utilizing the European Genome-phenome Archive (EGA) as the core of an ELIXIR community secure data sharing network for - omics data. Tools developed here will support submission of all types of -omics data from human samples consented for biomedical research from disease consortia such as International Cancer Genome			

Horizon 2020 grant n. 676559

Consortium (ICGC), Rare Diseases (Rd-Connect), national cohorts, and biobanks. Emphasis is given to supporting investigator and locally driven research projects with human data consented for biomedical research. To enable these projects, the data submission tool chain will be made more portable and user-friendly with the goal of distributing a common toolset “in-a-box” to enable local and national groups to collect -omics data and meta data in a distributed manner which is consistent across European groups through ELIXIR coordination.

To enable value-added services at project specific, regional, or national resources by establishing ELIXIR-wide community facing tools that allow local resource owners and developers to add value to their systems through data and metadata services from the EGA. For example, local research projects would be enabled to make their data discoverable and searchable, and linked with available -omics data from various sources, by leveraging stable unique EGA identifiers. Further, locally developed project specific data portals will be enabled through defined standard APIs using real time secure data links which allow -omics big data archived in the EGA to be presented in combination with biobanks or cohort data.

To extend and generalise the system of access authorization management and high volume secure data transfer developed in the EGA project to address the secure data access needs across ELIXIR resources and open new modes of secure data access such as through public and private clouds. For example, a trusted ELIXIR Cloud service can receive local copies of selected datasets through a secure data mirroring system and provide access to data and compute to those users that already have data access permissions available from appropriate Data Access Committees stored in the EGA system. The WP will partner first with 2-4 large resource owners to gain the required expertise, document the process in multiple ELIXIR member states and finally to propose a way to scale up these services to match wider European requirements. This WP will also be used to drive creation of the ELSI framework that supports the workflow (WP12).

Work Package Leads: Thomas Keane, EMBL-EBI (since 1/3/2017); Jordi Rambla, CRG EGA (since 1/3/2017); Justin Paschall, EBI (up to 1/3/2017); Arcadi Navarro, ES (up to 1/3/2017)

Description of work and role of partners

WP9 - Use Case D: ELIXIR framework for secure archiving, dissemination and analysis of human access-controlled data [Months: 1-48]

EMBL, UTARTU, NBIC, CRG, UPF, CSC, UiO, UU

This WP delivers the core ELIXIR workflow for long term archive and re-use of human data consented for biomedical research requiring access-control based on a data access agreement and approval process. The workflow supports data submitters and ELIXIR Node coordination on data deposition into the EGA archive in a manner that will maintain data ownership in the hands of the original research data owner, enable data release to authorised individual users from the archive and to partner with downstream secure ELIXIR data analysis platforms. This workflow ad supporting infrastructure will allow the data owners to focus on their unique areas of data generation and analysis expertise while being able to rely on EGA and the ELIXIR infrastructure for their common –omics big data storage, coordination and distribution needs under appropriate legal frameworks. The work described here will leverage the work of other ELIXIR-EXCELERATE Work Packages, for example WP10 to scale each service structure to cover all ELIXIR Nodes and with WP4 for technical service support, and relies on WP12 to establish the necessary legal framework that supports workflows.

The Workflow can be summarized as:

1. Data preparation, validation, and submission to the EGA making use of common supporting tools and data models (e.g. through Node data Network, WP10). Focus on providing software tools and remote APIs enabling local leadership and customisation within context of specific projects, supported by common ELIXIR coordinated tools and data models.

2. Bidirectional linking and secure data streaming between -omics data archived in EGA and local repositories or data portals that hold further information about the project and samples.

3. Management of user access-rights for release of archived data to authorized researchers under Data Access

Agreements using ELIXIR tools, such as the REMS (WP4), that allow resource owners to manage data access rights.

4. Expanded access through ELIXIR partner secure clouds that can host EGA datasets, requiring the provision of metadata and authorization APIs

5. Data synchronization between the main EGA archive and authorized project specific resources and access points, such as compute clouds.

Task 9.1: Enhanced secure data submission tools. (49PM)

This task will update the existing EGA submission tools and documentation to facilitate large-scale data submissions operations, emphasizing local leadership and customization within a common framework.

Partners: ES, EMBL-EBI, FI

Subtask 9.1.1: Support for large scale submission of -omics data and sample metadata to the EGA. (30PM)

Support for large-scale submission of -omics data and sample metadata to the EGA through improved online tools, automated verification, and tools for the application of standard vocabularies to phenotype collection. These tools will make use of table

"spread-sheet" based views of data for submitters less comfortable with technologies such as XML. Further tools and reports supporting global EGA stable identifier mappings will allow easier integration with local identifiers, in support of federated global tracking of submitted samples and their derived -omics data.

Subtask 9.1.2: Portable submission toolkit. (21PM)

This task is composed of data format definitions and software components, a "mini-EGA in-a-box" will allow increased local control and coordination of data collection, and allow early validation of standardized data and metadata formats.

This implementation provides the practical means for distributed projects to collect access-controlled human biomedical data in a manner that maintains a coordinated data model and dataset registry, enabling federated and a centralized single-point of discovery and access.

Task 9.2: Integrating centralized and distributed projects through transparent access to secure data: enabling local projects within a European wide framework. (40PM)

This task will enable local projects, such as study-specific data portals, local cohort resources, and national bioinformatics hubs by providing developer level APIs and services such that local efforts can efficiently build customized project branded solutions which make use of underlying ELIXIR and EGA tools and data archives.

Partners: ES, EMBL-EBI, FI, EE, NL, NO, SE

Subtask 9.2.1: Support secure integration of EGA data to downstream project client websites. (10PM)

Support secure integration of EGA data and metadata to downstream project client websites by providing new EGA programmatic interfaces that support standardized REST calls and provides results in ELIXIR endorsed formats (WP3 and WP6).

Subtask 9.2.2: Access management workflow support. (10PM)

Support access management workflows by data access committees through ELIXIR for EGA and other projects through developing applications of the Resource Entitlement Management Systems (REMS) expanding on an existing pilot project. This effort is focused on providing tools to delegate management to local projects and ELIXIR Nodes through new administrative roles.

Subtask 9.2.3: ELIXIR and EGA access integration. (20PM)

Specific efforts supporting controlled access-omics data infrastructure for use of partner national cohort studies in terms of submission, permissions management, and local and customized presentation of data under the cohort branding. Services will be tailored to respect the unique policy and data protection requirements of national cohorts, allowing single point of request and download from cohort branded web-pages. Support will be provided for distributed local hosting of datasets, within a common ELIXIR framework, where restrictions exist on the movement or hosting of data based on national borders.

Task 9.3: Federated authentication, large scale data management, and secure clouds in practice. (38PM)

This task is closely linked to the technically focused WP4 that provide the technical solutions required to deliver the outcomes of Task 9.1 and 9.2. In this task, technical components, including high volume secure data transfer and authentication and authorization management, are brought together to make -omics data from EGA and phenotypic data from cohort studies available for secure download, remote API access or from within public or private Cloud-based secure analysis environments. Cloud-based access to the EGA ecosystem provides a new access mode meeting a significant user need from research groups with limited local resources for compute and large-scale reference data storage.

Partners: EMBL-EBI, ES, FI, EE

Subtask 9.3.1: Large scale data mirroring support. (12PM)

Support for automated large scale data mirroring from the EGA archive to the authorized ELIXIR partner local services and cloud compute or HPC providers. This process instantiates concrete data flows based on data transfer technologies in WP4 to track domain specific files, versions of files, confirms transfer success, and tracks files available in different locations. Generic interfaces should provide transparent access to multiple underlying transfer and storage modules (e.g. gridFTP/irods/object store etc.)

Subtask 9.3.2: EGA data access authorization integration. (12PM)

Integrate EGA data access authorizations to local project data portals and Cloud access providers. This is a new service that allows authorized third-party services to programmatically check compliance with the current user data access authorizations from the ELIXIR coordinated repositories such as the EGA database each time user accesses a file in the cloud or other remote service. A first planned project using EGA data within the private, secure, cloud at CSC in Finland will provide our reference implementation.

Subtask 9.3.3: Data access APIs. (14PM)

Develop and implement standard data access APIs to be to used for inter and intra cloud communication and for secure remote REST API access in coordination with the Global Alliance for Genomics and Health (GA4GH).

For tasks 1-3 we expect to list a number of updates to the submission tools while we work with the first 2-4 chosen resources. These updates will be prioritised in the scope of this WP. WP4 will provide AAI support for WP 9, and vice versa WP9 will work with WP4 to set the requirements for ELIXIR AAI services. WP9 needs to information on service component availability and this information is expected to be available from technical services registry such as cloud resource allocation, valid EGA data access authorizations, and file mirroring status if data are not yet ready to be used in the cloud. WP12 will Create a set of Legal Frameworks for ELIXIR-related operations that will be integrated within WP9 with the technical solutions devised for particular EGA needs.

Horizon 2020 grant n. 676559

7. Appendix 1: Implementation of submissions interface and API

Introduction

EXCELERATE WP9 Task 9.1 is aimed to enhance the available options for submitting metadata to the EGA, especially for large submissions from large projects, but also for smaller submissions. Large projects usually require some level of harmonization between submitted objects or indeed being able to apply and validate custom rules before submitting to the EGA. Smaller submission usually require wider guidance during the submission process.

Here there are some feedback from EGA users about the previous submission solution:

"It was very difficult to see what parts of the submission process still needed to be completed in each part."

"Getting the files with sample and phenotype information ready for upload took some time, because the system did not provide feedback about what was wrong."

"More clarity on submission status would be great - e.g. email verification at various checkpoints - data uploaded, data verified, data archived, data ready for release."

There are other reasons to look for an update of the submission solution, e.g.:

- Not possible to register analyses (*one of the EGA objects, see below*)
- Not possible to see the uploaded files by own
- Difficult to follow the steps that are required
- Difficult to see the errors generated during the submission process
- Duplication of objects while completing the submission

To solve these issues, we have developed:

1. The Submission API (a REST API)
2. The Submitter Portal, a web user interface (UI) that leverages that API.

The Submission API is meant to facilitate programmatic submissions, thus allowing the users to automatize the submission process.

The Submitter Portal UI guarantees that the Submission API is comprehensive and robust enough to manage submissions from regular EGA users.

The Submission Process – Common Aspects

Objects required

The metadata objects required for sequencing submissions are as follows:

- Study: information about the sequencing study
- Samples: information about the sequencing samples
- Experiments: technical information about the biological experiment
- Runs: references the raw files (FASTQ / BAM); associated with the biological experiment, samples and study

Horizon 2020 grant n. 676559

- Analysis: references the analysis (BAM / VCF / phenotype) files; associated with samples and study
- DAC: contains information about the Data Access Committee (DAC)
- Policy: contains the Data Access Agreement (DAA); associated with DAC
- Dataset: contains the collection of runs/analysis data files to be subject to controlled access; associated with Policy

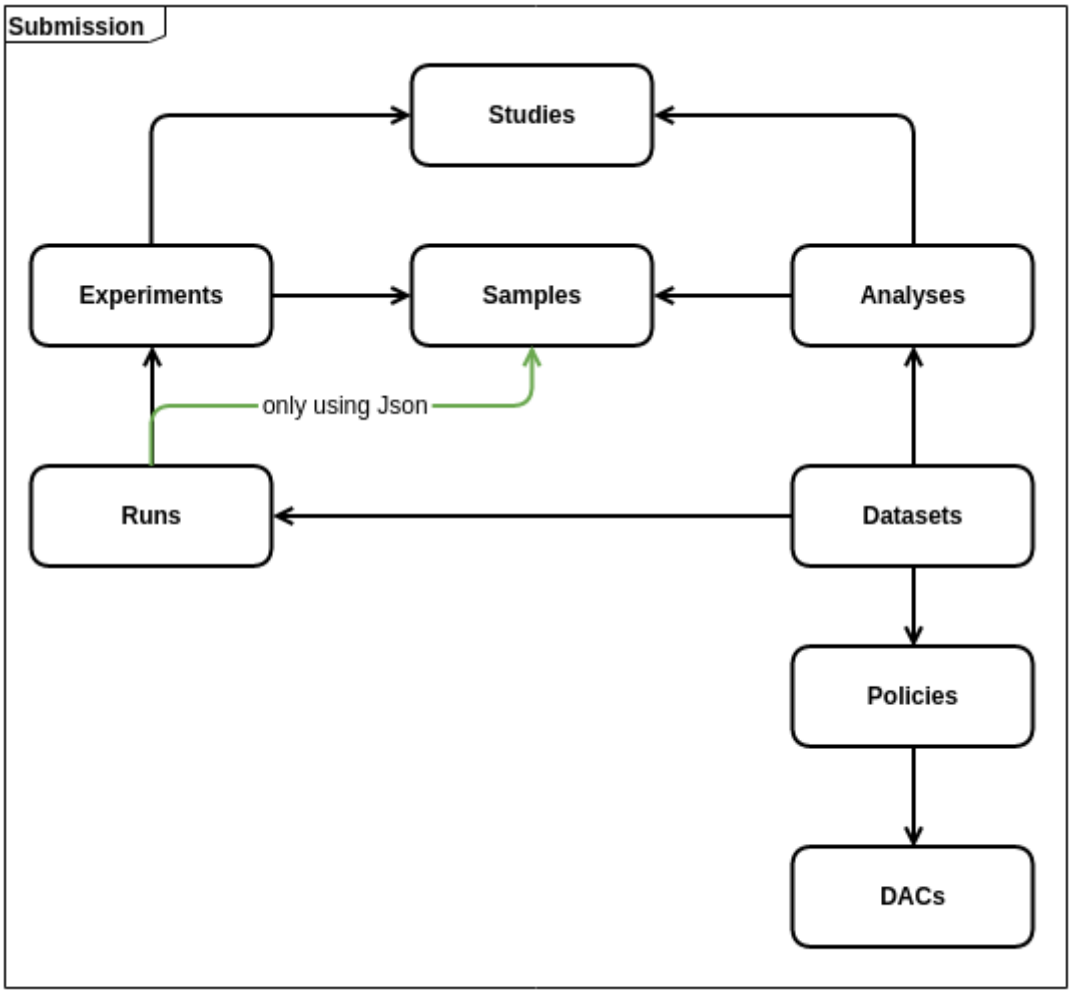

Figure 1: The Submission Objects

For Array-based submissions (raw and processed data obtained from microarrays), the Submitter Portal is partially used as of now. Study, Data Access Committee, Policy and samples are the metadata objects that can be registered via Submitter Portal. The actual linkage between samples and files within datasets is currently done parallelly.

Identifiers

EGA objects are identified by their unique accession. These are ID's displayed everywhere, shared among all EGA locations and specific for each data type (More information on the list below)

Horizon 2020 grant n. 676559

EGA Accession ID	EGA Object
EGAS	EGA Study Accession ID
EGAC	EGA DAC Accession ID
EGAP	EGA Policy Accession ID
EGAN	EGA Sample Accession ID
EGAR	EGA Run Accession ID
EGAX	EGA Experiment ID
EGAZ	EGA Analysis Accession ID
EGAD	EGA Dataset Accession ID
EGAB	EGA Submission ID
EGAF	EGA File Unique Accession ID

Table 1 EGA Accession ID nomenclature

Internal identifiers

The most important attributes of any EGA object are the identifiers. Each object is identified by four unique codes:

1. EGA_ACCESSION_ID: this is the EGA persistent ID, specific for each data type (more information on Table 1)
2. INTERNAL_EBI_ID: this is the internal object identifier used by the EBI which is shared across EBI's databases, e.g. ENA and EGA
3. INTERNAL_CRG_ID: same purpose as the INTERNAL_EBI_ID but applies only to the CRG instance of the object
4. ALIAS: unique name of each object defined by the user. NOTE: It is unique per user and object.

Submission and object status

In the new API (and Portal on top of , EGA submitted objects might pass through different statuses. When a new object is created its status is always DRAFT. This status provides complete freedom to the user since the only thing that is set is an internal ID, whereas the rest of the attributes can be blank. The object can also be deleted or edited with no concern.

An object can be validated at any time, leading to two different scenarios: VALIDATED and VALIDATED_WITH_ERRORS. The first, validated, indicates that all mandatory attributes have been provided and are valid and also that references to other objects are correct, whereas the second, returns a list of errors that need to be fixed. Each scenario is a status by itself and the object can be edited, returning it to the DRAFT status.

From any status an object can be SUBMITTED. This pushes the object through a more exhaustive validation (e.g. references to files are verified) and if successful, generates a new identifier, the EGA_ACCESSION_ID, which will be unique and never re-used, and makes the object visible at all EGA locations.

There is an special status called PARTIALLY_SUBMITTED which only applies to submissions and experiments. In the first case, it means that some objects (but not all) that belong to this submission have already been submitted. In the second one, it means

that this experiment has been submitted but there are still some not submitted runs that point to it (see second figure in Functionality).

The figure below depicts the workflow we have just described.

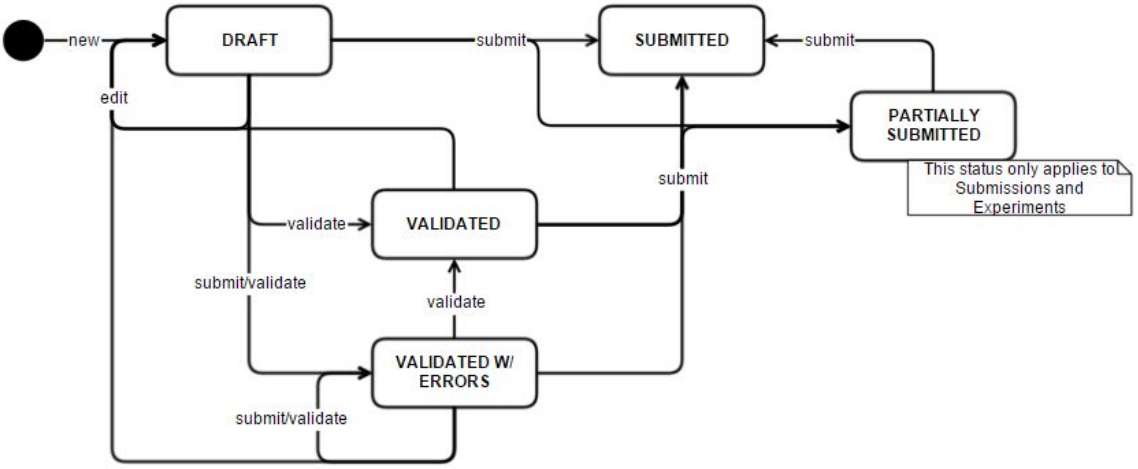

Figure 2 State transitions for objects

The Submitter Portal

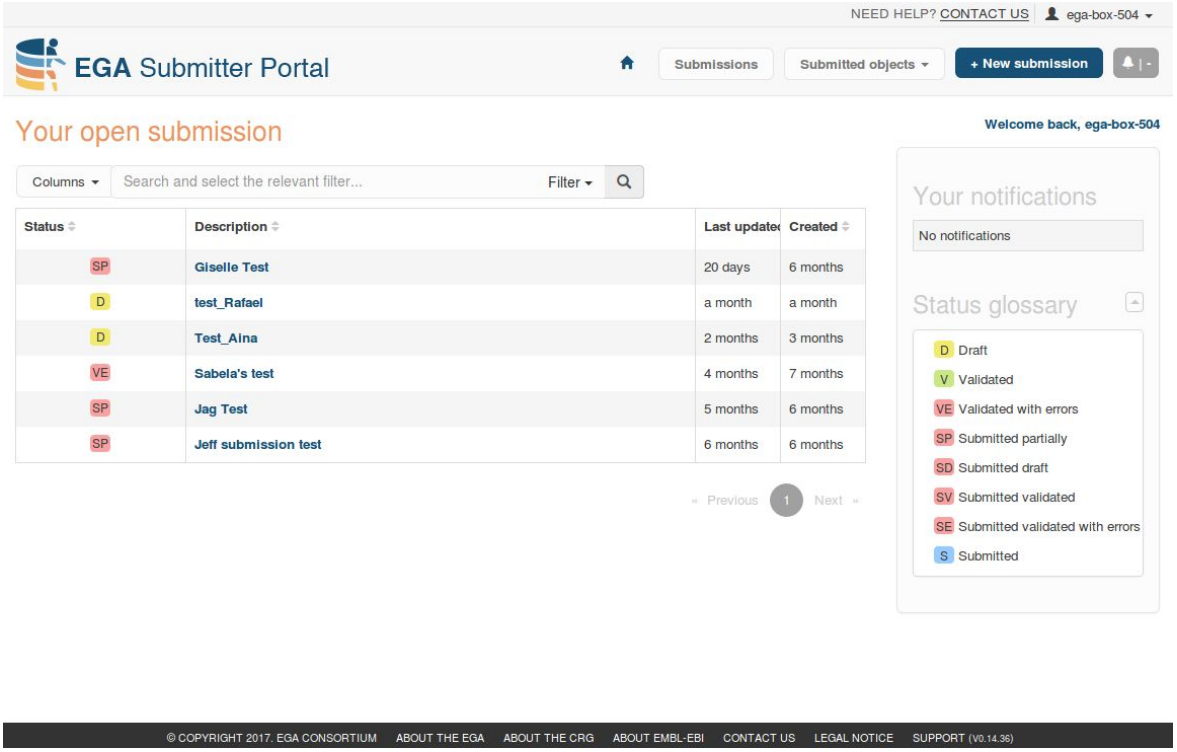

Figure 3 The Submitter Portal Home page

The Submitter Portal is a web user interface (UI) that leverages de Submission API to allow users to perform a full metadata submission or to review objects submitted by other

existing channels (i.e.: WebIn user interface, XML files upload or direct calls to the Submission API).

In summary, the Submitter Portal allows for:

- Creating a new (draft) submission, including all metadata object types supported by EGA for NGS studies,
- Having as many submissions “in progress” as the user wants, e.g. is indeed possible to use one as learning or testing and another one with the actual submission.
- List the raw data files, that should be submitted separately via specialized or typical transfer tools, and get some metadata required to perform the metadata submission. Listing of such files helps understanding how to link the metadata samples with the files themselves and avoids confusions
- Edit the (draft) submission as much as desired before doing the formal submission. This could be done in any order, as the user could fill in just those parts that they know at the editing time, leaving any unknown content for later.
- Validate the (draft) submission to check that all validation rules are met. That validation is equivalent to an actual submission that is just not confirmed at the latest step.
- Submit formally the objects to further publishing at the EGA

Some transversal features are:

- Richer error messages
- Possibility to check for errors pending to be solved

Describing the whole user interface in details is out of the scope of this document. Live documentation could be found at the corresponding EGA website page (https://ega-archive.org/submission/tools/submitter-portal)

The Submission API

The EGA API enables to tap into EGA and build custom EGA-powered applications for the user necessities.

Every API endpoint is built on the top of a microservices architecture, which powers the web interface. The fact that EGA hosted interfaces use the same API that other EGA users guarantees the quality of the API. The API provides a simple, predictable, standardized, resource-oriented, RESTful interface with JSON-formatted responses to use EGA features, including creation, editing, deleting and the validation of objects / entire projects. It embraces built-in HTTP features, so the API is easily managed with off-the-shelf HTTP clients. Every request to the EGA API can be easily performed using, e.g., the *curl* Linux command line tool.

The EGA Submitter Rest API allows submitters to upload or request metadata from EGA, such as ANALYSIS, DAC, DATASET, EXPERIMENT, POLICY, RUN, SAMPLE, STUDY and SUBMISSION. The Figure 1 shows the dependencies between objects with arrows (i. e. in runs there is a mandatory reference to experiments, therefore runs depend on experiments) and how they are grouped under a submission.

Input data can be in XML or JSON format. The latter allows you to take advantage of a new functionality which is represented with the green arrow in the Figure 1 and that

simplifies the managing of runs, experiments and samples. Using the Json format runs and samples can be linked without having to use experiments to do this. So, instead of having to repeat the same experiment data as many times as runs you have to point to the corresponding samples (Figure 2), you can point runs to samples and to a common experiment that will be shared among many runs (Figure 3).

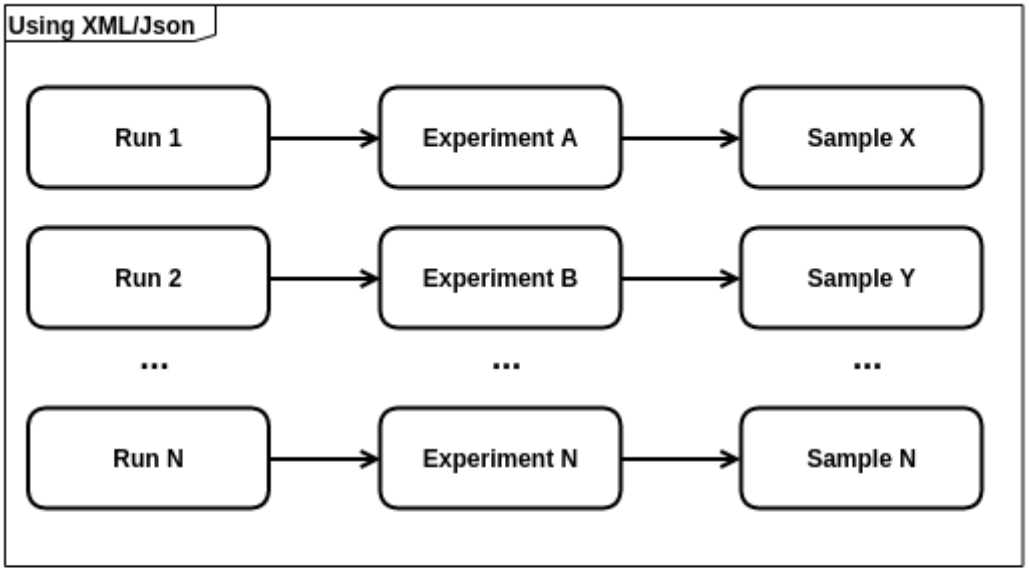

Figure 4 - Linking Runs and Samples using XML

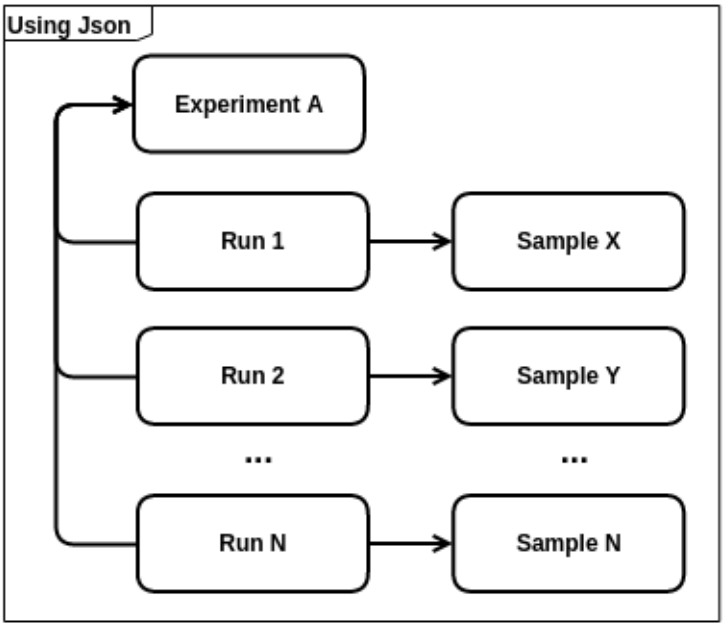

Figure 5 Linking Runs and Samples using Json

Horizon 2020 grant n. 676559

Submission API Response

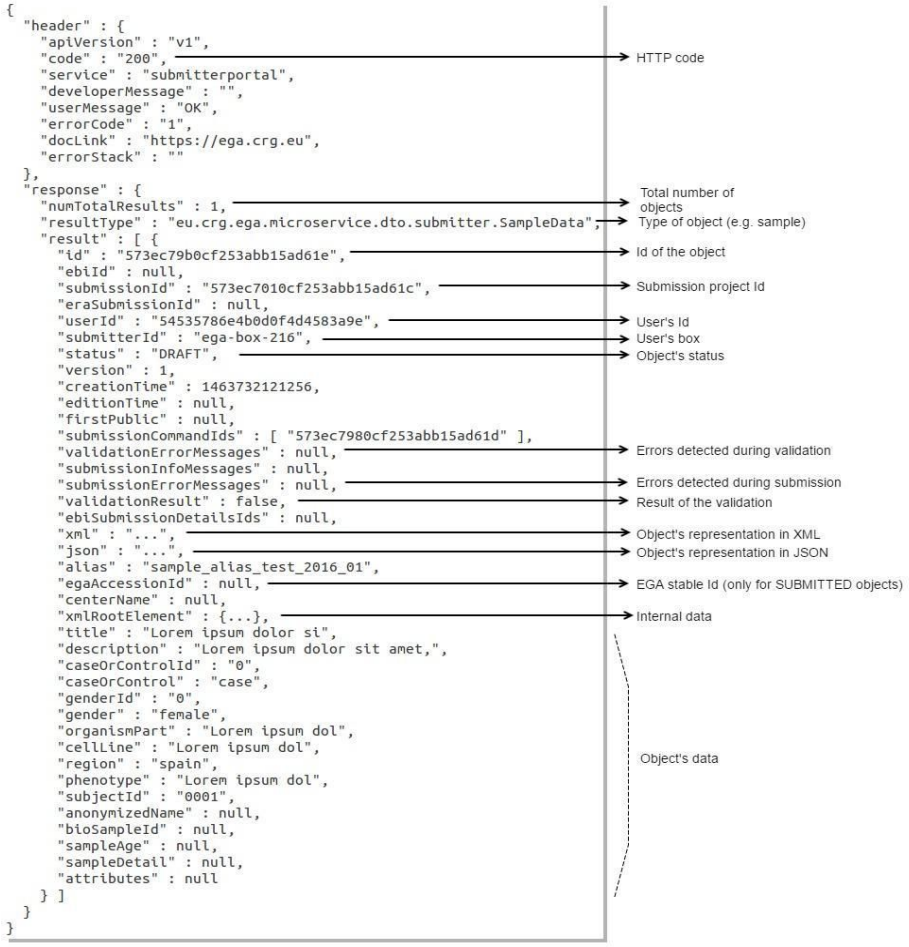

Figure 6 - The Submission API response

The server response is divided within 2 main sections they are the header and response.

- "header"
  - Contains information about the HTTP connection
  - "code"
    - Please see more about response codes on the table below

"response"

- Is the container for the response
- "numTotalResults"
  - By default, only 10 results are shown.
  - This field tells you if there are more results hidden (e.g. if the user asks for all the samples he/she created, there might be hundreds or thousands)
- "resultType"
  - The type of the objects returned (e.g. sample, DAC, datasets, etc.).

Horizon 2020 grant n. 676559

- "result"
    - The list of objects returned.
    - From "id" to "xmlRootElement" are common fields shared among all objects.
    - "status"
        - Valid values: DRAFT , VALIDATED, VALIDATED_WITH_ERRORS , PARTIALLY_SUBMITTED (only for submissions and experiments) and SUBMITTED .
    - "validationResult"
        - TRUE if the object passes the validation; otherwise FALSE.
        - By default its value is FALSE. User must validate or submit the object to get this value updated with the result of the validation.
            - NOTE: When submitting, the process automatically validates the object.
        - Users must look at this field to know if the object has passed the validation .
            - If it doesn't, user must look in "validationErrorMessages" or "submissionErrorMessages" to know what is wrong.
    - "egaAccessionId"
        - It is automatically set after the object has been successfully submitted.

HTTP response codes

API responses always return a status call. Below you have a summary on what these codes:

HTTP Status Code	Description	Resolution
200	OK	No error handling necessary
400	Bad Request	Request incorrectly formulated
401	UNAUTHORIZED/FORBIDDEN	You do not have permissions over the object you are trying to access. / The authorized user is not permitted to make the given request.
404	NOT FOUND	The object /resource you are trying to access does not exist.
500	INTERNAL SERVER ERROR	This is a server-side error. Contact the Heldpesk to notify the issue.

How to use the API

In the following sections we detail how to use the Submission API both for metadata retrieval and submission. The examples provided here use the Linux *curl* command to perform all the actions, but they can be easily adapted to be used on any preferred scripting language (e.g. Python) or in REST clients (e.g. Postman).

Login - How to obtain a session ID

An active session is required to work with the API. Each time you log in with your credentials a new session is started, which is identified by an X-Token. Below an example on how to obtain one session token using curl:

```
curl -X POST https://ega.crg.eu/submitterportal/v1/login -d
     username=ega-box-xxx --data-urlencode password="thepassword" -d
     loginType="submitter"
```

All responses of the API are in JSON format. If successful, the response should include a new Token to use during the session:

```
"session" : {
             "sessionToken" : "12c810d5-0974-4d11-937f-bd75a81de762",
             "expeditionTime" : 1429107047601,
             "expirationTime" : 1429110647601,
             "method" : "N/A",
             "ipAddress" : "192.168.110.14"
           }
```

NOTE: To save space, from this point onwards we will simplify the URL: We will write path instead of the common part of the URL (https://ega.crg.eu/submitterportal/v1) that is repeated in every call. We will leave out the header: -H “X-Token: 12c810d5-0974-4d11-937f-bd75a81de762” (but remember that it always must be present!)

How to log out

Simply use the following endpoint:

```
curl -X DELETE -H “X-Token:  12c810d5-0974-4d11-937f-bd75a81de762”
     path/logout
```

Retrieving metadata

The calls to the metadata service are basically identical for all EGA objects as well as the filters and methods available. Importantly, EGA objects can fall under a submission project. This concept is not very important when retrieving data, as it only provides another way to filter the results, however, it has a key role when submitting new objects. We will deal with this later on.

The following objects can be queried by the catalog, which correspond to the name of the method:

- analyses
- dacs

- datasets
- experiments
- policies
- runs
- samples
- studies
- submissions

Filtering by status: Status can be filtered by appending '`?status=[name of the status]`' at the end (see Enumerations for the complete list of values).

For already submitted objects it is necessary to use `'?status=SUBMITTED'`.

When no status is provided, all objects with status DRAFT, VALIDATED, VALIDATED_WITH_ERRORS, PARTIALLY_SUBMITTED are returned (that is, all except SUBMITTED). Therefore, only metadata objects registered via the REST API and not submitted yet are retrieved.

Below some examples:

Display all studies on draft, validated and validated with errors:

```
curl -X GET path/studies
```

Display all studies that have been validated and had errors:

```
curl -X GET path/studies?status=VALIDATED_WITH_ERRORS
```

Show all samples that have been submitted:

```
curl -X GET path/samples?status=SUBMITTED
```

All calls return a list of JSON objects, one for each EGA object, that contains all attributes of the object.

Filtering the Results

The identifiers described in section Identifiers can be used to filter out the results for a given object. If no parameter is specified, the filtering is done by either of the two internal IDs indistinctly:

```
curl -X GET path/samples/{sampleId}
```

Notice that "{sampleId}" must be an identifier of an already existent sample (in this example). To filter by EGA accession Id or alias, `'?idType=[name of the id]'` must be added at the end. Example:

```
curl -X GET path/samples/EGAN00001176841?idtype=ega_stable_id
```


Horizon 2020 grant n. 676559

```
curl -X GET path/samples/this_is_the_alias?idType=ALIAS
```

Filtering by submission

You can list any objects filtering by submission Id.

All analyses that belong to this submission:

```
curl -X GET path/submissions/{submissionId}/analyses
```

All experiments that belong to this submission:

```
curl -X GET path/submissions/{submissionId}/experiments
```

As when retrieving metadata, you can add '`?status=[name of the status]`' at the end to also filter by status. All analyses from this submission that are submitted:

```
curl -X GET path/submissions/{submissionId}/analyses?status=SUBMITTED
```

Remember that if no status is provided, an aggregation of DRAFT, VALIDATED, VALIDATED_WITH_ERRORS, PARTIALLY_SUBMITTED is returned (that is, all except SUBMITTED).

Retrieving information about files

Apart from metadata objects the API also provides a way to obtain information about the uploaded files. This may come from different sources, which must be specified in the call:

1. CRG_INBOX: Files uploaded to CRG's box which have not been archived yet.
2. EBI_INBOX: files that the user has uploaded to their box at EBI that have not been archived yet. A process runs every X hours and copies all files uploaded to CRG boxes to EBI boxes. So, all files that are seen using CRG_INBOX are also seen using EBI_INBOX (be aware of the delay).

Example:

```
curl -X GET path/files?sourceType=CRG_INBOX
```

Output formatting

Additional data like parameters for pagination (`skip` and `limit`) can be added to all data retrieval queries in order to format the output. They must be specified as query parameters (i. e. `skip=1&limit=5`).

IMPORTANT! By default, only 10 results are shown and no result is skipped (`skip=0&limit=10`). If you want to retrieve ALL data set both skip and limit parameters to 0 (i. e. `skip=0&limit=0`).

Enumerations

Horizon 2020 grant n. 676559

Enumerations contain list of terms that are used as available options at different points of the API. They are available under the `/enums` path and are public, so you don't need to be authenticated to call them. Some of them refer to the metadata whereas other provide parameters for calls or methods:

APi details

```
path/enums/actions: Action values. EDIT/VALIDATE/SUBMIT.
     /submission_statuses: Submission status values.
     /id_types: List of the different ids accepted at the EGA.
     /entity_statuses: Object status values.
```

Metadata

```
path/enums/analysis_file_types: Analysis file type accepted values.
     /analysis_types: EGA elegible analysis types.
     /case_control: Case/control accepted values.
     /dataset_types: Dataset type accepted values.
     /experiment_types: Experiment types.
     /file_types: Run file type accepted values.
     /genders: Gender accepted values.
     /instrument_models: Platform accepted values
     /library_selections: Library Selection accepted values.
     /library_sources: Library source accepted values.
     /library_strategies: Library Strategy accepted values.
     /reference_chromosomes: Chromosome values.
     /reference_genomes: Genome values.
     /study_types: Study type accepted values.
```

For instance, this call lists all the experiment types available when registering a new experiment object:

```
curl -X GET path/enums/experiment_types


"response" : {
         "numTotalResults" : 5,
         "resultType" :
     "eu.crg.ega.microservice.dto.submitter.AttributeData",
         "result" : [ {
           "tag" : "4",
           "value" : "Curation",
           "label" : null,
           "unit" : null
         }, {
           "tag" : "1",
           "value" : "Exome sequencing",
           "label" : null,
           "unit" : null
         }, {
           "tag" : "2",
```


Horizon 2020 grant n. 676559

```
            "value" : "Genotyping by array",
            "label" : null,
            "unit" : null
          }, {
            "tag" : "0",
            "value" : "Whole genome sequencing",
            "label" : null,
            "unit" : null
          }, {
            "tag" : "3",
            "value" : "transcriptomics",
            "label" : null,
            "unit" : null
          } ]
        }
```

Submitting metadata

In a similar way as the catalog service, the submission service has a set of instructions that are basically identical for all objects. New objects can be defined using:

- XML files, which must follow SRA 1.5 schema. In every call, it is necessary to add the header:

```
-H "Content-type: application/xml"
```

- JSON format. In every call, it is necessary to add the header:

```
-H "Content-type: application/json"
```

Creating Objects

Submission project

All new EGA objects must fall under a submission project. This means that after the generation of a new project, the id project must be included every time a new object is added. In order to register a new submission project we must use the following endpoint:

```
curl -X POST path/submissions -d '{JSON Object }'
```

Analysis

Using JSON:

```
curl -X POST path/submissions/{submissionId}/analyses -d '{Json
     Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/analyses/xml -d 'Analysis
     XML to be included.
```

Data Access Committee

Using JSON:

Horizon 2020 grant n. 676559

```
curl -X POST path/submissions/{submissionId}/dacs -d '{Json Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/dacs/xml -d 'XML to be
     included.'
```

Dataset

Using JSON:

```
curl -X POST path/submissions/{submissionId}/datasets -d '{Json
     Objectt}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/datasets/xml -d 'XML to
     be included'.
```

Experiment

Using JSON:

```
curl -X POST path/submissions/{submissionId}/experiments -d '{Json
     Object}'
```

Using XML:

```
curl -H POST path/submissions/{submissionId}/experiments/xml -d 'XML
     to be included'
```

Policy

Using JSON:

```
curl -X POST path/submissions/{submissionId}/policies -d '{Json
     Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/policies/xml -d 'XML to
     be included'
```

Run

Using JSON:

```
curl -X POST path/submissions/{submissionId}/runs -d '{Json Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/runs/sequencing/xml -d
     'XML to be included'
```

Sample

Using JSON:

Horizon 2020 grant n. 676559

```
curl -X POST path/submissions/{submissionId}/samples -d '{Json
    Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/samples/xml -d 'XML to be
    included'
```

Study

Using JSON:

```
curl -X POST path/submissions/{submissionId}/studies -d '{Json
    Object}'
```

Using XML:

```
curl -X POST path/submissions/{submissionId}/studies/xml -d 'XML to be
    included'
```

Editing Objects

EGA objects can be edited once have been created or validated (edition of submitted objects is not supported yet). To edit the object the edited XML/JSON must be provided. Don't forget to add the corresponding header:

```
-H "Content-type: application/xml" for XML input data.

-H "Content-type: application/json" for JSON input data.
```

Edit a sample using XML as input data:

```
curl -X PUT path/samples/xml -d 'Edited XML'
```

When using XML format, there is no need to reference the object in the call; the system will automatically update all objects that match with the ALIAS used.

IMPORTANT! Be very careful when editing object. If the alias is not unique, all objects with the same alias will be updated.

Edit a sample using JSON as input data:

```
curl -X PUT path/samples/{sampleId}?action=EDIT -d '{Edited JSON
    object}'
```

Notice that when using JSON, you must explicitly specify the object Id in the URL and you must add '`?action=EDIT`' at the end.

Deleting Objects

Only objects in status DRAFT, VALIDATED or VALIDATED_WITH_ERRORS can be deleted. Below there are some examples.

Delete a sample:

```
curl -X DELETE path/samples/{sampleId}
```

Delete a run:

```
curl -X DELETE path/run/{runId}
```

Validating and submitting Metadata

Validating/Submitting an entire submission project

When all objects of a given submission project have been completed, the entire project can be validated. Only the submission project must be used for this:

VALIDATE

```
curl -X PUT path/submissions/{submissionId}?action=VALIDATE
```

NOTE: As stated before, "`{submissionId}`" must be an identifier of an already existent submission.

As explained before, validation can be done at any time and as many times as needed, since it does not make any change on the objects. On the contrary, when the action SUBMIT is used, the objects are submitted, assigned unique IDs and cannot be deleted (without manual intervention from our Helpdesk). Therefore, be careful, and make sure your objects are finished before using SUBMIT submission action.

SUBMIT

```
curl -X PUT path/submissions/{submissionId}?action=SUBMIT
```

Validating/Submitting a subset of objects of a submission project

You can add a Json object as a parameter to the call to specify which objects of the submission project you want to validate/submit. This can be very useful if you don't want to validate/submit all objects of that submission but you don't want to do it one by one either.

SUBMIT

```
curl -X PUT path/submissions/{submissionId}?action=SUBMIT -d '{JSON
    Object }'
```

Example:

```
curl -X PUT path/submissions/{submissionId}?action=SUBMIT -d '{
      "submissionSubset" : {
      "dacIds" : ["dac_id_01"],
```


Horizon 2020 grant n. 676559

```
        "policyIds" : ["policy_id_01", "policy_id_02"]
    }
    }'
```

Validating/Submitting only one item

Objects can be validated and submitted also one by one. The syntax is quite easy and uses the same method described in the previous section. Find below an example for analysis objects, that can be extrapolated for the rest of objects.

VALIDATE

```
curl -X PUT path/analyses/{analysisId}?action=VALIDATE
```

SUBMIT

```
curl -X PUT path/analyses/{analysisId}?action=SUBMIT
```

Json messages format

The alias field is optional. If omitted or left empty, the system will automatically assign it a unique code

Analysis

```
{
       "alias": "",
       "title": "",
       "description": "",
       "studyId": "",
       "sampleReferences": [
         {
           "value": "",
           "label": ""
         }
       ],
       "analysisCenter": "",
       "analysisDate": "",
       "analysisTypeId": "", → /enums/analysis_types
       "files": [
         {
           "fileId ": "",
           "fileName": "",
           "checksum": "",
           "unencryptedChecksum": ""
           "fileTypeId":"" -> /enums/analysis_file_types
         }
       ],
       "attributes": [
         {
           "tag": "",
           "value": "",
           "unit": ""
```


Horizon 2020 grant n. 676559

```
        }
      ],
      "genomeId": "", → /enums/reference_genomes
      "chromosomeReferences": [ → /enums/reference_chromosomes
        {
          "value": "",
          "label": ""
        }
      ],
      "experimentTypeId": [ "" ], → /enums/experiment_types
      "platform": ""
}
```

In files, either `fileId` or `fileName` is mandatory. You can set both but `fileId` will have priority.

DAC

```
{
      "alias": "",
      "title": "",
      "contacts": [
        {
          "contactName": "",
          "email": "",
          "organisation": "",
          "phoneNumber": "",
          "mainContact": ""
        }
      ]
    }
```

Dataset

```
{
      "alias": "",
      "datasetTypeIds": [ "" ], → /enums/dataset_types
      "policyId": "",
      "runsReferences": [ "" ],
      "analysisReferences": [ "" ],
      "title": "",
      "description":"",
      "datasetLinks": [
        {
          "label": "",
          "url": ""
        }
      ],
      "attributes": [
        {
          "tag": "",
          "value": ""
        }
      ]
    }
```


Horizon 2020 grant n. 676559

At least, one run or one analysis must appear inside `runsReferences/analysisReferences`.

Experiment

```
{
       "alias": "",
       "title": "",
       "instrumentModelId": "", → /enums/instrument_models
       "librarySourceId": "", → /enums/library_sources
       "librarySelectionId": "", → /enums/library_selections
       "libraryStrategyId": "", → /enums/library_strategies
       "designDescription": "",
       "libraryName": "",
       "libraryConstructionProtocol": "",
       "libraryLayoutId": "", → 0 (paired), 1 (single)
       "pairedNominalLength": 0,
       "pairedNominalSdev": 0,
       "sampleId": "",
       "studyId": ""
     }
```

Policy

```
{
       "alias": "",
       "dacId": "",
       "title": "",
       "policyText": "",
       "url": ""
     }
```

Run

```
{
       "alias": "",
       "sampleId": "",
       "runFileTypeId": "", → /enums/file_types
       "experimentId": "",
       "files": [
         {
        “fileId”: “”,
           "fileName": "",
           "checksum": "",
           "unencryptedChecksum": "",
           "checksumMethod": ""
         }
       ]
     }
```

In files, either `fileId` or `fileName` is mandatory. You can set both but `fileId` will have priority. The full filename should be referenced in the field "`fileName`" followed by the extension "`.gpg`". If the data has been deposited in a folder within your ega-box, the complete path to the file should be referenced in the "`fileName`" field

Horizon 2020 grant n. 676559

Sample

```
{
       "alias": "",
       "title": "",
       "description": "",
       "caseOrControlId": "", → /enums/case_control
       "genderId": "", → /enums/genders
       "organismPart": "",
       "cellLine": "",
       "region": "",
       "phenotype": "",
       "subjectId": "",
       "anonymizedName": "",
       "biosampleId": "",
       "sampleAge": "",
       "sampleDetail": "",
       "attributes": [
         {
           "tag": "",
           "value": ""
         }
       ]
     }
```

Use attributes field if you have any custom attribute that does not match with the predefine ones (phenotype, organismPart, genderId, etc.).

Study

```
{
       "alias": "",
       "studyTypeId": "", → /enums/study_types
       "shortName": "",
       "title": "",
       "studyAbstract": "",
       "ownTerm": "",
       "pubMedIds": [ "" ],
       "customTags": [
         {
           "tag": "",
           "value": ""
         }
       ]
     }
```

Submission

```
{
       "title" : "",
       "description" : "",
       "submissionSubset" : { JSON object·  }
     }
```


Horizon 2020 grant n. 676559

SubmissionSubsetData

```
{
     "analysisIds" : [ "" ],
     "dacIds" : [ "" ],
     "datasetIds" : [ "" ,
     "experimentIds" : [ "" ],
     "policyIds" : [ "" ],
     "runIds" : [ "" ],
     "sampleIds" : [ "" ],
     "studyIds" : [ "" ]
     }
```

Human and Process related aspects

During the development and production phase of both the Submitter API and Submitter Portal, the following steps have been performed:

- Survey with users about existing tools
- Review and feedback from the EGA Helpdesk team, that manages submissions
- Training of the Helpdesk Team
- Beta testing with some users familiar with the submission process
- Testing with users new to the EGA
- Gather of feedback from the early users
- Improvement of the tools

Conclusions

The initial production phase has proven a success as it could be seen by some feedback received from early Submitter Portal users:

- *"It worked. This new submission portal is muuuuch more user-friendly."*
- *"Ok, thanks. This portal does seem a lot easier - especially with that video. I will let you know if I have other questions about it, but seems good for now."*
- *"Nice that there is now an overarching "Submission"; From experience, it is usually a long-winded process to assemble/gather all the different parts. Having a way to keep multiple concurrent submissions separate and nicely bundled is a definite improvement!"*
- *"I like the accordion layout of the different steps, how it guides you through the correct order of things."*
- *"I absolutely LOVE the new "Submission files list". This feedback on what has transferred is lovely, and that you also import the checksums from the various accompanying .md5 and .gpg.md5 files is a great feature."*

Next Steps

The Submission API and Portal will extended with improvements coming from the feedback that the EGA Helpdesk is getting from current users. Additionally we are exploring the deprecation of the current microarray submission solution and extending the Submission API and Portal to allow such type of submission.

In a further future, we would evaluate how easy is to extend the solution to accept new sequencing solutions like single cell sequencing, long reads or metagenomics.

Links

1. Submission API endpoints base path: https://ega-archive.org/submission-api/v1/
2. Submitter Portal UI: https://ega-archive.org/submitter-portal/
3. Submitter Portal documentation: https://ega-archive.org/submission/tools/submitter-portal
4. Submitter Portal Video: https://www.youtube.com/watch?v=_7sH7koHlJY