1 REST Web Services

Overview

We are constantly looking at ways to make it possible (not to mention reliable and easy!) for users and developers to interact with and extend OpenClinica in a programmatic way. This can mean anything from data loading to more meaningful integrations of applications common to the clinical research environment.

As proponents of open, standards-based interoperability here at OpenClinica, our starting point is to develop interfaces using the protocols that power the World Wide Web (such as HTTP, SSL, XML, OAuth 2.0). They are relatively simple, extensively documented, widely understood, and well-supported out of the box. On top of this foundation, we rely heavily on the wonderful work of CDISC and the CDISC ODM to model and represent the clinical research protocol and clinical data.

This chapter describes a way to interact with OpenClinica using RESTful APIs and OAuth. The REST web services API relies on HTTP, SSL, XML, OAuth 2.0. This architecture makes the ODM study protocol representation for an OpenClinica study available and supports other interactions for study design.

Access the OpenClinica REST API

Why REST?

The OpenClinica RESTful architecture was developed to (initially) support one particular use case, but with the intention of becoming more broadly applicable over time. This use case is based on a frequent request of end users: for OpenClinica to support an visual method for designing, editing, and testing rules which define edit checks, email notifications, skip pattern definitions, and the like to be used in OpenClinica CRFs. Users have had to learn how to write rules in XML, which can be confusing and have a big learning curve for non-technical individuals. The OpenClinica Rule Designer is an application that allows end users to build cross field edit checks and dynamics within a GUI based application. It is a centrally hosted Software as a Service (SaaS) based application available for OpenClinica Enterprise customers at https://designer.openclinica.com.

To support interaction of the centrally hosted rule designer with any instance of OpenClinica Enterprise installed anywhere in the world, we needed to implement a secure protocol and set of API methods to allow exchange of study information between the two systems, and do so in a way where the user experience was as integrated as if these applications were part of the same integrated code base. In doing so, and by adopting the aforementioned web and clinical standards to achieve this, we have built an architecture that can be extended and adapted for a much more diverse set of uses.

This chapter specifies how 3rd party applications can interact with an OpenClinica instance via the REST API and OAuth security, and details the currently supported REST API methods. The currently

supported API methods are not comprehensive, and you may get better coverage from our SOAP API. However the OpenClinica team is continuing to expand this API and since it is open source anyone may extend it to add new methods to meet their own purposes. If you do use the API in a meaningful way or if you extend the API with new methods, please let others know on the OpenClinica Forums, and submit your contributions for inclusion back into the codebase - youll get better support, increased QA, and compatibility with future OpenClinica releases.

RESTful Representation, based on ODM

REST, an acronym for **RE**presentational **S**tate **T**ransfer, describes an architectural style that allows definition and addressing of resources in a stateless manner, primarily through the use of Uniform Resource Identifiers (URIs) and HTTP. A RESTful web service (also called a RESTful web API) is a simple web service implemented using HTTP and the principles of REST. It is a collection of resources, with three defined aspects:

- the base URI for the web service, such as http://example.com/resources/
- the Internet media type of the data supported by the web service. This is often JSON, XML or YAML but can be any other valid Internet media type.
- the set of operations supported by the web service using HTTP methods (e.g., POST, GET, PUT or DELETE).

In the context of REST for clinical research using OpenClinica, we can conceptually think of an electronic case report form (CRF) as a **resource** that is essentially a bunch of metadata modeled in CDISC ODM with OpenClinica extensions. An OpenClinica Event CRF is that same bunch of metadata with the corresponding item data, plus references to the study subject, event definition, CRF version, event ordinal, etc that it pertains to.

- The notion of a CRF version pertains to the representation of the CRF. It is not intrinsic to the event CRF (this is debatable but it is how OpenClinica models CRFs). Theoretically you should be able to address and view any Event CRF in any available version of the CRF (ie http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v1/edit and http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v2/edit both show you the same data represented in different versions of the CRF). Of course the audit history needs to clearly show which version/representation of the CRF was used for key events such as data capture, signature, etc.
- Rules are also part of the representation metadata as opposed to intrinsic metadata, even though you dont need to specify them on a version-by-version basis.
- Anything attached to the actual event CRF object or its item data discrepancy notes, audit trails, signatures, SDV performance, etc is part of that event data and should be addressable in the same manner (eg http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v1/GROUPOID/ORDINAL/ITEMOID/DN/1/view)

In this conceptual view of the world, CRFs (as well as CRF items, studies, study events, etc.) are RESTful resources with core, intrinsic properties and then some other metadata that has to do with how they are presented in a particular representation. We now have a model that allows us a great deal of flexibility and adaptability. We can support multiple modalities, with different representation metadata for rendering the same form, or perhaps the shared representation metadata but applied in a different way. We can address any part of the CRF in an atomic manner. This approach has been successfully applied in the Rule Designer, which takes the ODM study metadata and allows browse of the study CRFs and items, with the ability to drag and drop those resources into rule expressions. Some examples of additional future capabilities that could be easily realized on top of this

architecture:

- Multiple data entry modalities a user may need to deploy patient based data entry via web, a tablet, a thick client, or even paper/OCR, each with a very different presentation. Each of these may be part of OpenClinica-web or a separate application altogether, but all will rely on the same resource metadata to represent the CRF (according to the UI + logic appropriate for that modality), and use the same REST-based URL and method for submitting/validating the data.
- Apply a custom view (an XSL or HTML/CSS) to a patient event CRF or full casebook some uses of this could be to represent as a PDF casebook, show with all audit trails/DNs embedded in line with the CRF data, show a listing of data for that subject, provide (via an XSL mapping) as an XForm or HL7 CCD document for use by another application) - http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v1/view?renderer=somemapping.xsl
- The same path used in the URLs, eg http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v1/GROUPOID/ORDINAL/ITEMOID could be used as the basis for XPath expressions operating on ODM XML representations of CRFs and of event crf data
- Internationalization OpenClinica ought to allow our CRF representation metadata to have an additional sub-layer to render the form in different languages, and then automatically show the appropriate language based on client/server HTTP negotiation (like we do with the rest of the app). Currently internationalization of CRFs requires versioning the CRF.
- View CRF & Print CRF use the same representation metadata (form metadata) but apply slightly different rules on how the presentation works (text values instead of form fields, no buttons, turn drop down lists into text values)
- Discrepancy manager popup one requested use case would allow a user to update a single event CRF item data value directly from the discrepancy note UI point of view. In this case you could think of just updating that one item as addressing the resource http://oc/RESTpath/StudyA/Subj1234/VisitA/FormB/v1/GROUPOID/ORDINAL/ITEMOID/edit?mode=DN. In this model, whatever rules and presentation metadata need to get applied at presentation and save time happen automatically.
- Import of CDISC ODM XML files imported data would be processed through the same model, but only use the metadata thats relevant to the data import modality. Same for data coming in as raw ODM XML via a REST web service. A lot of times the import only populates one part of a CRF and the other parts are expected to be finished via data entry. This model would help us manage that process better that the current implementation of ODM data import.

There are many considerations related to user roles and permissions, workflows, and event CRF/item data status attributes that need to be overlaid on top of this REST model, but the model itself is conceptually a most useful way to think about clinical trials and the information represented therein. When implemented using CDISC ODM XML syntax it becomes even more powerful. As widespread support for ODM becomes the norm, the barriers to true interoperability - shared, machine readable study protocol definitions, and robust, real-time, ALCOA-compliant exchange of clinical data and metadata that aligns with users business processes - get eviscerated.

* This chapter frequently refers to ODM-based representations of study metadata and clinical data in OpenClinica. We strive as much as possible to implement ODM-based representations of OpenClinica metadata and data according to the generic ODM specifications (currently using ODM version 1.3). However, to ensure our representations support the full richness of information used in OpenClinica we often have to rely on ODMs vendor extensions capability. I have not always made distinctions in this chapter as to where we are using generic ODM versus OpenClinica extensions, but that is documented here. It is our goal as ODM matures and supports richer representations of

study information to migrate our extensions back into the generic ODM formats.

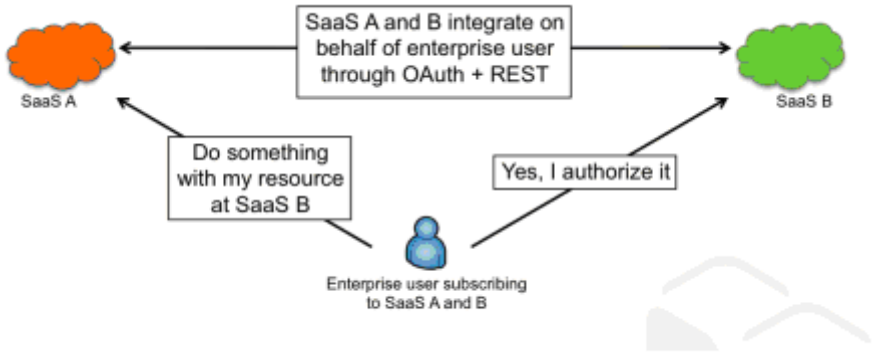
** Also note the RESTful URL patterns referred to above are conceptual. Refer to the technical subchapters of this REST API specification for the actual URLs.

Approved for publication by Cal Collins. Signed on 2016-03-11 3:42PM

Not valid unless obtained from the OpenClinica document management system on the day of use.

1.1 OAuth and OpenClinica

Use of the REST API is dependent on authentication via the OAuth protocol (version 2). OAuth is a security protocol that enables users to grant third-party access to their web resources without sharing their passwords. See http://oauth.net/2/ and http://hueniverse.com/2010/05/introducing-oauth-2-0/ for more information on OAuth 2.0.



(Source: http://cloud.dzone.com/news/enterprise-saas-integration)

The communication/authentication steps between the OAuth client and server are described below.

(D) = Designer - OAuth Client

(OC) = Authorized OpenClinica Instance OAuth service provider

1. (D) /access.....

2. (D) hits restTemplate call to OC

3. (D) throws exception - No OAuth 2 security context has been established.

Unable to access resource 'ocInstance'.

4. (D) As part of exception bubbling up OAuth2ClientContextFilter line 77 triggered results in

response.sendRedirect ::

http://localhost:8080/OpenClinica-web-SNAPSHOT/oauth/user/authorize?client_id=designer&redirect_uri=http%3A%2F%2Flocalhost%3A8080%2FDesigner-0.1.0.BUILD-SNAPSHOT%2Faccess%3Fhost%3Dhttp%3A%2F%2Flocalhost%3A8080%26app%3DOpenClinica-web-SNAPSHOT%26study_oid%3DS_DEFAULTS1%26provider_user%3Droot&response_type=code

5. (OC) redirect hits OC

6. (OC) initiates http://localhost:8080/OpenClinica-web-SNAPSHOT/oauth/confirm_access

7. (OC) the above url initiates a maybe just a normal request or REST call back to (D)

8. (D) in OAuth2ClientContextFilter the request URL looks like http://localhost:8080/Designer-0.1.0.BUILD-SNAPSHOT/access?host=http://localhost:8080&app=OpenClinica-web-SNAPSHOT&study_oid=S_DEFAULTS1&provider_user=root&code=2zLl3b

9. (D) hits /access .. controller code again

10. (D) hits restTemplate call to OC

11. (D) throws

org.springframework.security.oauth2.consumer.OAuth2AccessTokenRequiredException:

No OAuth 2 security context has been established. Unable to access resource 'ocInstance'.

12. (D) As part of exception bubbling up OAuth2ClientContextFilter line 77 triggered results in

13. (D) line 83 in OAuth2ClientContextFilter will trigger REST Call to

http://localhost:8080/OpenClinica-web-SNAPSHOT/oauth/authorize

14. (D) response from above call produces an accessToken

15. (D) returns back to /access ... controller code execution

16. (D) hits restTemplate call to OC

17. (D) call succeeds

Every time /access..... is invoked, Designer will receive the request and ask OpenClinica.com to check if the URL is an authorized instance (ie, an Enterprise instance or a Community instance that has registered here). If that check fails the user will be automatically directed to an error page. The above check will be continuously performed through out the OAuth handshake to check the validity of the URL.

1.2 RESTful URLs

In the context of REST for clinical research using OpenClinica, we can conceptually think of an electronic case report form (CRF) as a resource that is essentially a bunch of metadata modeled in CDISC ODM with OpenClinica extensions. Other OpenClinica objects (such as study definitions) can also be resources exposed in a RESTful manner.

About REST and Clinical Data Keys

In REST, resources are identified by logical URLs and are the key element of RESTful design. Interaction is stateless.

ClinicalData Objects (i.e. Study Subjects and their CRF data) in OpenClinica can be addressed by using ODM-based Clinical Data Keys (https://docs.openclinica.com/3.1/technical-documents/openclinica-and-cdisc-odm-specifications/cdisc-odm-representation-openclin-6#content-title-4523) as part of a URL path appended to /ClinicalData/, as follows:

```
GET /OpenClinica/ClinicalData/{format}/{mode}/ODM_XML_PATH?OPTIONS
```

Implementation in 3.1.3

The initial implementation of RESTful URLs in OpenClinica 3.1.3 supports a URL to return a read-only EventCRF with its associated FormData in HTML format. Only HTML format and view mode is implemented in OpenClinica 3.1.3. The general format of the URL is:

```
/OpenClinica/ClinicalData/html/view/{StudyOID}/{StudySubjectKey}/{StudyEventDefOID}[{StudyEventRepeatKey}]/{FormDefOID}?tabId={sectionNum}
```

The URL query string options supported for this URL are tabId & exitTo:

- tabId the CRF section number to show (HTML format only).
- exitTo URL for where the 'exit' button should take the user (optional, HTML format only, relative to the OpenClinica root URL).

An example would be:

```
GET
/OpenClinica/ClinicalData/html/view/S_CPCS/320999/SE_CPCS[1]/F_CPCS_1?tabId=1
&exitTo=ViewStudySubject?id=1
```

These URLs do not support OAuth security authentication.

Future Implementation Plans

The {format} component of the URL may be html or xml. XML provides CDISC ODM XML. If omitted use CDISC ODM XML

The {mode} component may be view, edit, or print (html format only).

The path builds an address to the resource. In the example above, the path ends at the Form OID level so we can determine the resource were looking at an EventCRF. It could go further down to the ItemGroup or Item level. The format of whats returned and the available options may vary based on the level of the resource addressing an EventCRF may be consumed differently than addressing an Item.

The full form of the URL would be:

```
GET
/OpenClinica/ClinicalData/{format}/{mode}/{StudyOID}/{StudySubjectKey}/{Study
EventDefOID}[{StudyEventRepeatKey}]/{FormDefOID}?tabId={sectionNum}&exitTo={e
xitURL}
```

This page is not approved for publication.

1.3 RestFul URL access to OpenClinica metadata and print Resources.

Starting OpenClinica 3.1.4, there will be restful access to OpenClinica's ODM metadata and print CRFs. The metadata can be obtained in json/xml formats, once the user is logged into the system.The printing of a CRF without any data will also be accessible via restful url.

These rest based urls do not yet support OAUTH yet and the user needs to be logged into OpenClinica.

The urls will be of the following format:

/host/{WEB-APP_CONTEXT}/rest/metadata/{format}/{mode}/ODM_XML_PATH.

As of 3.1.4, OpenClinica supports formats:html/xml/json

mode:view/print

The mode should be 'print' if the user intends to view a printed CRF,

ODM_XML_PATH: Consists of typically 3 variables.
{STUDY_OID}/{STUDY_EVENT_OID}/{FORM_VERSION_OID}

All the 3 variables are study OID,studyevent oid and form oid which are generated by OpenClinica and are unique across the system. They can be substituted by a * in order to mention a generic or include all as explained below.

For printing CRFs that do not belong to any study and/or are used in multiple events:

ODM_XML_PATH: */*/FORM_VERSION_OID

For printing crfs/viewing the json or xml of all of the study(The ODM in json and xml formats brings the whole ODM metadata even though we put a form_version_OID. This filtering at XML level will be worked on in near future.):

ODM_XML_PATH:STUDY_OID/*/FORM_VERSION_OID

Examples: To obtain the ODM of the entire study the following would be the restful path:

/host/{WEB_APP}/rest/metadata/xml/view/STUDYOID/*/*

To obtain the printed view of a CRF that belongs to a particular event in a study:

/host/{WEB_APP}/rest/metadata/html/print/STUDYOID/STUDYEVENTOID/FORMVERSIONOID

Starting OpenClinica 3.2, there will be restful access to OpenClinica's ODM clinical Data for populated print CRFs. The Clinical Data can be obtained in json/xml formats, once the user is logged into the system.The printing of a CRF with populated data will also be accessible via restful url.

The urls will be of the following format:

/host/{WEB-APP_CONTEXT}/rest/clinicaldata/{format}/{mode}/ODM_XML_PATH.

formats:html/xml/json

mode:view/print The mode should be 'print' and the format should be 'html' if the user intends to view a printed CRF,

For Clinical Data the ODM_XML_PATH: Consists of typically 4 variables. {STUDY_OID}/{Study_Subject_OID}/{STUDY_EVENT_OID}/{FORM_VERSION_OID} plus parameters for additional options.

All the 4 variables are STUDY_OID , Study_Subject_OID , STUDY_EVENT_OID , FORM_VERSION_OID are generated by OpenClinica and are unique across the system. They can be substituted by an * in order to mention a generic or include all as explained below.

For printing CRFs for All Subjects per Study : ODM_XML_PATH: /Study_oid/*/*/*

For printing CRFs for One Subject per Study : ODM_XML_PATH: /Study_oid/Study_Subject_OID/*/*

For printing CRFs for One Subject and One event: ODM_XML_PATH: /Study_oid/Study_Subject_OID/Study_Event_OID/*

For printing an Event CRF : ODM_XML_PATH: /Study_OID/Study_Subject_OID/Study_Event_OID**%5B**1**%5D**/Form_Version_OID

%5B**1**%5E : Represents the Study Event Repeat #in brackets. [1]

%5B is the URL Escape code for Left Square Brackets and **%5D** is the URL Escape code for Right Square Brackets

Additional parameters will be needed to include Audit Logs and Discrepancy Notes and to filter Item or event status.

Add the following parameter '**?includeAudits=y**' (case sensitive) to the end of your URL to include all the Audit Log for SubjectData, StudyEventData, FormData and ItemData attributes that exist (not null)

Add the following parameter '**?includeDNs=y**' (case sensitive) to the end of your URL to include all the Discrepancy Note for SubjectData, StudyEventData, FormData and ItemData attributes that exist (not null)

The OC extension attributes and ItemData values will be always be incuded in the output whether parameters are added to the end of the URL or not. The OC extension attributes include for SubjectData, StudyEventData, FormData entities.

1.4 Read OpenClinica ODM Metadata REST

Service

The service will get ODM metadata about a single study in the OpenClinica. Standard authentication applies, and the user must have read privileges for the associated study.

This page is not approved for publication.

1.4.1 Calling Methods and Arguments

Reads is invoked as an HTTP GET method on a specific instance of a resource, qualified with a STUDY OID value.

GET pages/rule/studies/{study}/metadata

1.4.2 Responses

On success, a response with a 200 OK HTTP status code and a representation of the requested ODM object is returned. The response will be presented in the following schema.

- OpenClinica-ODM1-3-0-OC2-0-foundation.xsd
- OpenClinica-ODM1-3-0-OC2-0.xsd
- OpenClinica-ToODM1-3-0-OC2-0.xsd

1.5 Validate Rule REST Service

The service gets the Validity of a rule posted to it. Standard authentication applies, and the user must have privileges to conduct such an operation.

This page is not approved for publication.

1.5.1 Calling Methods and Arguments

This operation is invoked as an HTTP POST method on a specific instance of a resource qualified

with a STUDY OID value.

POST pages/rule/studies/{study}/validateRule

1.5.2 Responses

On success, a response with a 200 OK HTTP status code and a representation of the requested validation result is returned. The response will be presented in the following schema.

- response.xsd

1.6 Test Rule REST Service

Gets the Validity of a rule and test keys/values posted to this service. Standard authentication applies, and user must have privileges to conduct such an operation.

> This page is not approved for publication.

1.6.1 Calling Methods and Arguments

This operation is invoked as an HTTP POST method on a specific instance of a resource qualified with a STUDY OID value.

POST pages/rule/studies/{study}/validateAndSaveRule

1.6.2 Responses

On success, a response with a 200 OK HTTP status code and a representation of the result of saving the rule is returned. The response will be presented in the following schema.

- response.xsd

1.7 Save Rule REST Service

Saves the rule and returns a message. Standard authentication applies, and user must have privileges to conduct such an operation.

This page is not approved for publication.

1.7.1 Calling Methods and Arguments

This operation is invoked as an HTTP POST method on a specific instance of a resource qualified with a STUDY OID value.

POST pages/rule/studies/{study}/validateAndSaveRule

1.7.2 Responses

On success, a response with a 200 OK HTTP status code and a representation of the result of saving the rule is returned. The response will be presented in the following schema.

- response.xsd