SEAD 2.0 Publication API Walkthrough:

The Restful API to SEAD's Curation and Publication Services is focused on management (basic CRUD: Create, Read, Update, Delete) of repository, people, and research object publication request entries.

Matchmaking, which essentially requires create a 'test' research object publication request, is handles through additional endpoints for the Research Object service. The following sections describe the basic use of the services and the final section does a walk through of how to create new publication requests from the Beta Test Project Space.

Enunciate documentation of the rest calls can be found at http://seadva-test.d2i.indiana.edu/sead-c3pr/index.html (The links on this page are functional, but the links below in this document also point to our current test services and will help guide you to the most important endpoints.)

Repositories:

Repositories working with SEAD should POST a profile to the /api/repositories endpoint. The profile is a json-ld object containing descriptive terms about the repository (primarily from the re3data vocabulary) along with any additional descriptive terms repositories wish to add (i.e. other vocabularies). To participate in matchmaking, profiles must also contain terms that trigger rules (this list is available at /api/researchobjects /matchingrepositories/rules and is discussed further below). Repositories will also expected to identify preference terms they will respond to (and that users will see in the staging area). The format for identifying such preferences is not yet defined.

Repositories are required to include an "orgidentifier" value in their profile object. This should be a short term that is unique (e.g. sda, ideals, icpsr, bob) - it will be used to list existing repository profiles at /api/repositories. Individual profiles can be read by going to /api/repositories/<orgidentifier>, e.g. /api/repositories/sda.

Profiles may be updated by doing a PUT with a new profile to /api/repositories/<orgidentifier>, and can be deleted by doing a DELETE to the same URL.

Currently, the Bob repository has a profile that shows the entries required to trigger the existing rules (/api/researchobjects/matchingrepositories /rules):

People:

Personal profiles are available through the /api/people endpoint. Navigating to this endpoint lists the current set of profiles being managed. New profiles can be added by sending a POST to /api/people with a json object indicating the provider and identifier for the person at that provider. Currently ORCID, Google+, and local SEAD profiles are supported, so POSTs would contain, for example, "{"provider": ORCID", "identifier":" <numeric ORCID_ID> }" where "0000-0003-2164-8132" is one such identifier. Individual profiles can be retrieved from /api/people/<id>, e.g. /api /people/0000-0003-2164-8132. An empty PUT to a specific profile endpoint will cause a refresh (the profile will be retrieved from ORCID again), and DELETE will remove the profile.

Since people profiles are primarily for automated use during publication and matchmaking, it is expected that Project Spaces will automatically request profile harvesting for, for example, people who are creators of research objects, and, eventually, for all people who register with SEAD. The format of what is saved is expected to change as other providers are added.

Research Objects:

Research objects have a lifecycle that, from a curation and preservation service perspective, begins with a request for the object to be published, proceeds through a series of processing steps that result in status updates, and ends with successful publication, failure at some step, or revocation of the request. A request is made by POSTing to the /api/researchobjects endpoint. The object sent is a compound JSON-LD object consisting of an object describing the top level item (an ORE Aggregation) to be published, the list of user preferences (i.e. from the staging area), and the repository the request is for. A GET at /api/researchobjects shows the list of requests, and doing a GET on /api/researchobjects/<Identifier> will show the full request. /api/repositories/<repo_id>/researchobjects and /api/repositories/<repo_id>/researchobjects/new are convenience methods that return all of the publication requests for a given repository (regardless of status) or just the new requests (those for which the repository has not yet sent any status message) for a given repository, respectively. Within the full request is a URL from which the OREMap for the Aggregation can be retrieved. The Map includes a list of all sub-objects, their metadata, and their relationships in an ORE-compliant JSON-LD structure.

A DELETE to /api/researchobjects/<Identifier> causes the request to be revoked/deleted .

A PUT to /api/researchobjects/<Identifier>/status can be used to send a Status message to update the request

A GET to /api/researchobjects/<Identifier>/status will retrieve just the status information known about a request.

A GET to /api/repositories/<orgidentifier>/researchobjects will return a list of the research object publication requests to a given repository.

For Matchmaking, a POST can be made to /api/researchobjects/matchingrepositories using the same format as for a publication request - minus the key/value identifying a specific repository - and the score for the proposed research object at all repositories will be given. The total score is a sum of scores from all rules and the response details the individual scores as well as the total.

Walking Through:

From a 1.5 Space:

Create a collection and add subcollections and content as usual in the SEAD-Test Project Space (1.5). To create a new publication request, hit the "Submit for Publication" Button.

> Since 1.5 does not have a Staging Area, changing preferences or setting the repository target for publication require Admin privileges and setting the Project Space default values for these entries. Going to Admin Page/Config Tab/ 2.0 Beta Publication section, set the default repository and preferences as desired and hit "submit". New requests made will use these values.
>
> To revoke a publication request, click the "Publication Requested" button on your collection and it will revert to say "Submit For Publication" again.

From 2.0:

> Create a Dataset and add files/folders and content as usual in the SEAD2 Beta Server. To create a new publication request, hit the "Publish" Button and follow the process in the Staging Area to - make any final changes to your submission, match to repositories, select a repository and set preferences, submit the publication request for processing, and monitor the status of the request. Requests can be revoked until processing by the repository has begun.

After a "Submit For Publication" request, you should then be able to go to /api/researchobjects and see the new request.Your request will be deleted if you revoke it from the project space.

The request from the project space is augmented two ways during processing:

> If your object has metadata about its creators (dcterms:creator), which can be added in the project space, and if the value is an identifier from ORCID, Google, or SEAD and that person has listed employment or education affiliations in their profile, these affiliations are added to the request automatically. Multiple creators can be listed.
>
> An initial status message indicating that the request has been received by the curation and publication services is also added.

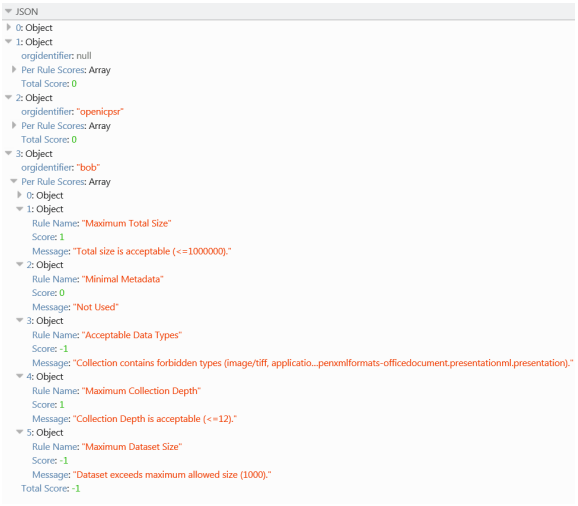
Copy the "Identifier" to the end of the URL (e.g. /api/researchobjects/<Identifier>) and you'll see the full request. Within this request, click on the link listed as the "@id" of the "Aggregation". This URL will return the full OREMap JSON-LD file for your request. This file can be used by repositories to process the request - it contains a full list of all aggregated objects (Collections and Datasets for v1.5), their metadata, and their relationships (including the collection/dataset hierarchy). These objects are listed with an ORE:similarTo URL pointing at the content itself (for 1.5, this is the restful endpoint to the live binary content for a dataset, which, since only metadata about a dataset can change, is also the content for the version being published).

A repository can find requests for it at /api/repositories/<orgidentifier>/researchobjects or /api/repositories/<orgidentifier>/researchobjects/new and can again retrieve the full request and OREMap as above. As a repository processes the request, it should add status messages using a POST to /api/researchobjects/<Identifier>/status with a message of the form: { "reporter": "<repo-ID>", "stage": "Receipt Acknowledged", "message": "request recorded and processing will begin"} , replacing the values as appropriate. The status, with a timestamp generated by the service will be added. The full status chain for a request can be seen doing a GET to /api/researchobjects/<Identifier> /status.

A message with "stage":"Success", and a "message" with the final identifier should be sent to finish processing. (The services will then notify the project space to add the DOI (to the live Collection and published data page in 1.5, to the Dataset and Curation Object in 2.0).

Matchmaking:

To test matchmaking, the simplest approach is to publish from a SEAD2 project space. To test programmatically, copy an existing publication request entry and remove the Affiliations, Repository, and Status entries (Nominally, a matching request would not include status or a repository choice, but these fields are ignored, so you don't have to delete them). The JSON object with the remaining Aggregation and Preferences entries can be POSTed to /api/researchobjects/matchingrepositories to receive a json document with the scores. Note that rules will only fire if the repository profile has the appropriate value as listed in /api/researchobjects/matchingrepositories/rules , i.e. to limit dataset size, a repository profile must include "Max Dataset Size":<long value in bytes>. The score document includes a message describing whether a rule fired or not and why the result is what it is. To make this work, be sure that the content type and accept type are both application/json. A sample response from late 2015 is shown below:



This result depends on the current content for that curation object, the current profile for the Bob repository, and the active current rule set.)

People Management

In this walkthrough, all management of people and their profile harvesting from ORCID occurs automatically. The endpoints described above can be used to view this information and/or to manually add/update/delete additional profiles. These will eventually be used to support profile harvesting when a user joins SEAD or adds their ORCID ID, and to enable profiles to be searched to enable matching of names as dc:creator metadata is types (as we used to do with VIVO IDs).