RECAST RESTful API

Revision Information:

Version	Author	Description	Date
0.1	Randy Kolenko – Nextide Inc.	Initial Document release	November 4th, 2011
1.0	Randy Kolenko – Nextide Inc.	Updated Document to reflect completed initial phase development.	November 22nd, 2011
1.1	Randy Kolenko – Nextide Inc.	Updated document to include full specification for add parameter point for Request method	November 23rd, 2011

This document outlines the usage and functionality of the Recast RESTful web interface (also known as "The API").

Ths API document is broken in to each functional section according to data elements in the User Interface. In order to test the interface, you can use the Firefox browser with the Poster add-on: http://code.google.com/p/poster-extension/

Note: Please replace www.example.com with the appropriate URI to the server's web service.

ANALYSIS:

The recast-analysis resource contains the data associated to an analysis. Requests and eventual responses all relate back to an analysis and as such, this resource could be deemed as a root data element.

Resource	Method(s)	URI	Description
All Analyses (collection)	GET	http(s)://www.example.com/api/recast-analysis	Use this URI to get all of the analyses in the system. Returned in the stream are UUIDs for each Analysis in the system. The return is a collection (array) output of the analyses in the system. Not all data relating to an analysis is returned with this query and as such you should always

			query deeper in to a specific analysis to get all of its data.
Specific Analysis	GET	http(s)://www.example.com/api/recast-analysis/…uuid… e.g.: http://example.com/api/recast-analysis/a438ddf8-c681-4618-82ee-96df6052e490	Using a valid Analysis UUID (as returned by the All Analyses collection), this URI will fetch a specific Analysis from the system
Create an Analysis	POST	http(s)://www.example.com/api/recast-analysis Post body required parameters: username – string of portal username title – string, title of the anaysis collaboration – string, one of the following [Atlas, D0, CDF, CMS, ALEPH] e_print – string, URL for E Print journal – string, url to journal publication doi – string, DOI link/information inspire_url – string, url to Inspire description – string, describes the analysis	POST to the recast-analysis URI with the correct body parameters as noted, and a new Analysis will be created. The return will have a header called URI with the direct URI to the new analysis resource. The POST will also return an element with the new analysis UUID.
Add a Run Condition to an Existing Analysis	POST	http(s)://www.example.com/api/recast-analysis/…uuid…/add-run-condition Post body required parameters: username – string of portal username name – string, name of the run condition description – string, describes the run condition	Targeted POST action to add a run condition to the analysis. You must specify a valid UUID in the URI in order to add a run condition. Post body requires the username of the user who created the analysis as well as the run condition name and description.

REQUEST:

The recast-request resource contains the data associated to a request. Requests are associated to analyses and a response is associated to a request.

Resource	Method(s)	URI	Description
All Requests (collection)	GET	http(s)://www.example.com/api/recast-request http(s)://www.example.com/api/recast-request?page=[0..n]&pagesize=[x]&username=[usr]&type=[all|accepted] Use the page parameter to tell the system which page to view and how many requests per page to show. Default is page 0 with a pagesize of 10. Please note that the page numbering scheme starts at 0 and increases from there. To show 20 requests at a time starting at the first page of results: http(s)://www.example.com/api/recast-request?page=0&pagesize=20 To show 5 requests at a time on the 3rd page: http(s)://www.example.com/api/recast-request?page=2&pagesize=20 To fetch a specific user's requests (requests a user has made), use the username parameter: http(s)://www.example.com/api/recast-request?page=2&pagesize=20&username=bob To fetch all of *your* (if your username was 'bob') accepted requests: http(s)://www.example.com/api/recast-request?pagesize=100&username=bob&type=accepted	Use this URI to get all of the requests in the system. Returned in the stream are UUIDs for each request in the system. The return is a collection (array) output of the requests in the system. Not all data relating to a request is returned with this query and as such you should always query deeper in to a specific request to get all of its data. When fetching requests you've accepted, you will be returned a response_uuid field which will be your reference to update the response for the request.
Fetch a Specific Request	GET	http(s)://www.example.com/api/recast-request/…uuid… e.g.: http://example.com/api/recast-request/ a438ddf8-c681-4618-82ee-96df6052e490	Using a valid Request UUID (as returned by the All Requests collection), this URI will fetch a specific Request from the system
Update a Request	POST	http(s)://www.example.com/api/recast-request/…uuid…/update Plus additional POST body parameters (* denote mandatory): *analysis-uuid *username	The update of an existing Request will require the UUID of the request along with the targeted action of "update" added to the URI.

		*title *audience *subscribers *activate *predefined-model *reason-for-request *model-type additional-information new-model-information e.g.: http://example.com/api/recast-request/ a438ddf8-c681-4618-82ee-96df6052e490/update POST body: analysis-uuid=a438ddf8-c681-4618-82ee-96df6052e490&username=admin&title=Request XX1&audience=authoritative&subscribers=admin,user2, userx&predefined_model=model2&new_model=this%20is%20a%20new%20model&activate=0&reason-for-request=	The POST body must contain the mandatory parameters. Parameters are defined as follows: analysis-uuid: in the format of (string) xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx username: (string) portal username title: (string) text of the request title audience: (string)(one of) `all|authoritative|selective` subscribers: (comma separated string) portal usernames e.g. user1,user2… activate: (integer/boolean) 0=Not active, 1=Active predefined-model: (string) one of the available model string IDs reason-for-request: (string) Text explaining this request model-type: (string) (one of) `defined|new` additional-information: (string) Text providing extra information new-model-information: (string) Text providing new model information
Create a Request	POST	http(s)://www.example.com/api/recast-request Plus additional POST body parameters (* denote mandatory): *analysis-uuid *username *title *audience *subscribers *status *predefined-model *reason-for-request	To create a request you must POST to recast-request all of the body parameters that are denoted as mandatory. See the Request Update method for parameter descriptions. Upon successful creation, a URI header is provided for you to use as the URI to the discreet information about the request itself.

		*model-type additional-information new-model-information e.g. e.g.: http://example.com/api/recast-request/ POST body: analysis-uuid=a438ddf8-c681-4618-82ee-96df6052e490&username=admin&title=Request XX1&audience=authoritative&subscribers=admin,user2, userx&predefined_model=model2&new_model=this%20is%20a%20new%20model&status=0&reason-for-request=reason	The return from this method is the actual UUID of the request.
Accept a Request	POST	http(s)://www.example.com/api/recast-request/…uuid…/accept?username=[username] e.g.: http://example.com/api/recast-request/ a438ddf8-c681-4618-82ee-96df6052e490/accept?username=bob Please note the return of the RESPONSE UUID. You must use that UUID to commit changes to via the API.	To accept a Recast Request, you must accept it based on a request UUID and provide your username to the system. The return stream has a URI header that points you to the response object created for this request. Also, the response UUID is returned. The API will reject an accept request when any of the following conditions exist: 1. User has already accepted the request 2. User is not on the subscribers list 3. Request is closed/canceled
Add a parameter point to a Request	POST (multipart /form-data)	http(s)://www.example.com/api/recast-request/…uuid…/add-parameter-point/?username=[usr]&parameter_point=[point]&number_of_events=[number of events]&cross_sections=[cross sections]&filename=[filename] POST Body:	To add a parameter point in one action, use the add-parameter-point targeted action to a specified UUID and include the appropriate parameters in the URI. The entire post body must be the LHE file being

		Data file for posting as the LHE file. Example with cURL: curl --data-binary @thefile.zip -H "Content-type: multipart/form-data" "http://example.com/api/recast-request/b9784458-a0b7-40ed-b66a-377a81f9edc8/add-parameter-point/?username=bob&parameter_point=x1,y2&number_of_events=2&cross_sections=2&filename=thelhefile.zip"	posted for this parameter point. The filename parameter is used to determine the extension of the file. The file will be renamed to the predefined file name convention once the upload has been completed. The content type must be set to multipart/form-data for the POST to work

SUBSCRIPTION:

The recast-subscription resource contains the data associated to subscriptions.

Resource	Method(s)	URI	Description
All Subscriptions (collection) All Your Subscriptions (collection)	GET	http(s)://www.example.com/api/recast-subscription?page=[0..n]&pagesize=[0..x]&username=[usr] e.g.: To fetch all subscriptions: http://example.com/api/recast-subscription To fetch your subscriptions if your username is 'bob': http://example.com/api/recast-subscription/?username=bob To fetch 20 items from the 2nd page of results: http://example.com/api/recast-subscription/?page=1&pagesize=20 To fetch the top 100 of your subscriptions http://example.com/api/recast-	Use this URI to get all of the subscriptions in the system. Returned in the stream are UUIDs for each subscription in the system. The return is a collection (array) output of the subscriptions in the system. Not all data relating to a subscription is returned with this query and as such you should always query deeper in to a specific subscription to get all of its data.

		subscription/?username=bob&page=0&pagesize=100	
Specific Subscription	GET	http(s)://www.example.com/api/recast- subscription /…uuid… e.g.: http://example.com/api/recast-subscription/ a438ddf8-c681-4618-82ee-96df6052e490	Using a valid Subscriptions UUID (as returned by the All Subscription collection), this URI will fetch a specific Subscription from the system
Create a Subscription	POST	http(s)://www.example.com/api/recast- subscription POST Body Parameters: analysis-uuid: string, UUID of the analysis to subscribe to username: string, your username subscription-type: string, “provider” or “observer” requirements: string, text specifying requirements notifications: string, comma separated list of notifications of the following options: recast_requests|recast_responses|new_subscribers	Create a subscription for a specified Analysis UUID. You must provide the listed body parameters in order for a subscription to become active. You must use the web UI to request that your subscription is made authoritative or not.
Update a Subscription	POST	http(s)://www.example.com/api/recast- subscription/…uuid.. POST Body Parameters: analysis-uuid: string, UUID of the analysis to subscribe to username: string, your username subscription-type: string, “provider” or “observer” requirements: string, text specifying requirements notifications: string, comma separated list of notifications of the following options: recast_requests|recast_responses|new_subscribers	Identical to that of the create with the exception that you need to provide the UUID of the subscription to update in the URI.

RESPONSE:

The recast-response resource contains the data associated to responses.

Resource	Method(s)	URI	Description
Retrieve all Responses (collection) Your Responses (collection)	GET	http(s)://example.com/api/recast-response?page=[0..n]&pagesize=[0..x]&username=[usr] e.g.: To fetch all responses: http://example.com/api/recast- response To fetch your responses if your username is 'bob': http://example.com/api/recast- response/?username=bob To fetch 20 items from the 2nd page of results: http://example.com/api/recast- response/?page=1&pagesize=20 To fetch the top 100 of your responses http://example.com/api/recast- response/?username=bob&page=0&pagesize=100	Use this URI to get all of the responses in the system. Returned in the stream are UUIDs for each response in the system. The return is a collection (array) output of the responses in the system. Not all data relating to a response is returned with this query and as such you should always query deeper in to a specific response to get all of its data.
Update a Response's result file	POST	http(s)://example.com/api/recast-response/[uuid]/update?filename=[filename] Example with cURL: curl --data-binary @thefile.zip -H "Content-type: multipart/form-data" "http://example.com/api/recast-response/b9784458-a0b7-40ed-b66a-377a81f9edc8/update/?filename=theresponsefile.zip" You must specify to cURL that you are posting binary data with the –data-binary flag. You must specify a Content-type header of "Multipart/form-data" as shown in the example so that the RESTful controller accepts the post.	To update a specific response with a file, you must target your POST at a response's UUID and noting to the RESTful controller that you are updating the response with a filename denoted with the filename attribute. The POST body should be the file contents.

		The @thefile.zip is the actual file you are trying to POST. The filename attribute is what the eventual file will be called when submitted to the system combined with a numeric date stamp to keep filenames unique.	
Update a Response's data points	POST	http(s)://example.com/api/recast-response/[uuid]/update POST Body Parameters: total_luminosity: decimal value luminosity_efficiency: decimal value lower1_signal: integer upper1_signal: integer upper2_signal: integer lower2_signal: integer result_url : string. Prefix with http:// to ensure link created is an external link.	Use this method to update the data points associated to a response. NOTE: Omitting the filename URL parameter signals the service that this is a data point update vs. file update.

NOTE: A response is created by accepting a request. Once a request is accepted, a response node and UUID is generated allowing you to POST updates to it.