OSLC Configuration Management Deliveries and delivery history

2023-08-17 Version 2

David Honey, IBM

Contents

1 Introduction

The OSLC Configuration Management 1.0 specification defines the RDF representation of components, streams, baselines, and change sets. It defines how new versions of concept resources can be created in the context of a stream or a change set. However, it currently does not define any means for a client to:

1. Request the delivery of a change set to a target stream. See https://github.com/oslc-op/oslc-specs/issues/528.
2. Get the history of deliveries of one or more change sets into a target stream.

The first issue became apparent as we tried to map SysML v2 APIs to OSLC equivalent operations. The SysML APIs were modelled on Github, where the focus of delivering changes is commit. A Github user can make changes to multiple files, and perform an atomic commit to the Github repository. The collection of changes for a commit is analogous to the OSLC Configuration Management concept of a change set.

This discussion paper presents a design proposal to address both these gaps together.

2 New delivery type and resource shape

I propose we introduce a new `oslc_config:Delivery` resource type with the resource shape described below. A delivery represents the delivery of a single change set to a single target stream. If multiple change sets are to be delivered, each would be done as a separate delivery.

Prefixed Name	Occurs	Read-only	Value-type	Representation	Range	Description
dcterms:created	Zero-or-one	true	dateTime	N/A	Unspecified	Timestamp when the delivery was made.
dcterms:creator	Zero-or-one	true	AnyResource	Either	foaf:Agent, foaf:Person	Creator of the resource. The link target is usually a `foaf:Person` or `foaf:Agent`, but could be any type.
dcterms:identifier	Zero-or-one	true	String	N/A	Unspecified	A unique identifier for this resource.
dcterms:title	Zero-or-one	false	XMLLiteral	N/A	Unspecified	An optional title of the delivery.
oslc_config:sourceConfiguration	Exactly-one	false	AnyResource	Reference	oslc_config:ChangeSet	The change set that was delivered. This cannot be modified after creation.
oslc_config:targetStream	Exactly-one	false	AnyResource	Reference	oslc_config:Stream	The stream to which the change set was delivered. This cannot be modified after creation.
rdf:type	One-or-more	True	AnyResource	Reference	rdfs:Class	The resource MUST include the type oslc_config:Delivery.

3 Supported operations on delivery objects

Servers MUST support a GET of a delivery object URI. The response should include an `Etag` header.

Servers SHOULD support a HEAD on a delivery object URI. The response should include an `Etag` header.

Delivery objects may be immutable, or a server might permit the title to be changed. If a server allows the title to be changed, it should support a PUT with `If-Match` header. Servers may silently ignore any data that cannot be changed, such as read only properties, or `oslc_config:sourceConfiguration` or `oslc_config:targetStream`.

POST should not be supported. A new object can only be created through a creation factory – see Delivering change sets through a delivery creation factory.

DELETE may be supported, but if a server does support it, the semantics of that deletion are undefined. For example, does deleting a delivery remove the change set from the target stream?

4 Delivering change sets through a delivery creation factory

4.1 Key features

There are several key aspects of change set delivery that OSLC Configuration Management should support:

1. Delivery is atomic. Either all the changed/new/removed resources are delivered, or none are.
2. A delivery might be a long-running operation.
3. Delivery of a change set should not result in an earlier change being lost. A conflict arises when a change set applies changes on top of version X of an object, whereas the target stream has changes on top of version X+N of that object.

4.2 Creating a delivery

An OSLC server would declare an OSLC Creation Factory for `rdf:type oslc_config:Delivery` in some OSLC service that was discoverable from an OSLC Service Provider Catalog. The declaration of that creation factory SHOULD reference a resource shape that is compatible with the shape described in New delivery type and resource shape.

A client would perform a POST with an RDF body conformant with the resource shape. The server would respond one of:

1. 201 Created, with the response including a `Location` header of the new `oslc_config:Delivery` resource that was created. This might be the case for a server where the operation is quick to execute.
2. 400 Bad request, if the request was invalid. For example, missing source configuration or target stream in the POSTed content.
3. 409 Conflict, if the delivery could not be made because of a conflict between the change set and the target stream.
4. 303 See other, including a `Location` header of an existing delivery if the change set was already delivered to the target stream.
5. 202 Accepted, with the response including a `Location` header of an `oslc_config:Activity` resource indicating the progress or result of that activity. Clients SHOULD poll the Activity resource periodically until the state indicates it has finished.

4.3 Conflicts

If the delivery of a change set would result in the loss or overwriting of a later change to the same object, the delivery MUST fail and report a conflict. The corresponding `oslc:Error` should provide sufficient information for a client to understand the nature of the conflict. For each conflict, the corresponding `oslc:Error` should provide a statement using `oslc_config:deliveryConflict` to an inline resource of implied type `oslc_config:DeliveryConflict` with the following resource shape:

Prefixed Name	Occurs	Read-only	Value-type	Representation	Range	Description
oslc_config:sourceVersionResource	Exactly-one	true	AnyResource	Reference	oslc_config:VersionResource	The version resource in the source configuration that is in conflict.
oslc_config:targetVersionResource	Exactly-one	true	AnyResource	Reference	oslc_config:VersionResource	The version resource in the target stream that is in conflict.
rdf:type	Zero-or-more	True	AnyResource	Reference	rdfs:Class	The resource MAY include the type oslc_config:DeliveryConflict. If not specified, then that type is implied.

OSLC Configuration Management will not specify any means to resolve such conflicts, or to preview a delivery to identify potential conflicts without committing the delivery. Conflicts must be resolved using the tool's UI and/or non OSLC defined APIs.

5 Getting delivery history through a query capability

There are several common uses cases for finding information about deliveries:

1. When was a specified change set delivered to a specified stream, and by whom?
2. Which streams has a specified change set been delivered to?
3. What the change sets that have been delivered to a specified stream?

The obvious OSLC pattern to use here is an OSLC Query Capability. Servers SHOULD provide a query capability for `rdf:type oslc_config:Delivery` in some OSLC service that is discoverable from an OSLC Service Provider Catalog. The declaration of that query capability SHOULD reference a resource shape describes the LDP query results container (as per the OSLC Query specification), which in turn provides a value resource shape for its members that is compatible with the shape described in New delivery type and resource shape.

Examples:

Description	Unencoded `oslc.where` expression
Find the delivery for a specified change set and target stream	`oslc_config:sourceConfiguration=<`*`changeSetUri`*`> and oslc_config:targetStream=<`*`streamUri`*`>`
Find the deliveries for a specified change set to any stream	`oslc_config:sourceConfiguration=<`*`changeSetUri`*`>`
Find the deliveries for all change sets to a specified target stream	`oslc_config:targetStream=<`*`streamUri`*`>`
Find the deliveries for all change sets to a specified target stream since 1st January 2023	`oslc_config:targetStream=<`*`streamUri`*`> and dcterms:created >= "2023-01-01'T'00:00:00"^^xsd:dateTime`

Clients that want to find the files changes associated with a delivery would do so through the referenced change set. Query capabilities MAY support nested query properties in `oslc.select`. For example, an unencoded `oslc.select` value of `dcterms:title,dcterms:created,oslc_config:sourceConfiguration, oslc_config:sourceConfiguration {dcterms:title,oslc_config:selections{oslc_config:selects}}` might return the delivery title, when it was delivered, the URI of the change set, the title of the change set, and the URIs of the files changed in the change set.

6 Persistence of delivery information

The persistence of delivery information is a server implementation choice. In servers where there is a corresponding persisted object, a server might map such objects to delivery objects. In servers where there is no first-class object representing a delivery, the delivery might be constructed from querying internal data.