Fundamental OSLC Concepts for Distributed Link Creation

OSLCFest 2021

Axel Reichwein
November 4, 2021
Overview

Distributed Link Creation Strategy

Key Ideas of OSLC APIs to enable distributed link creation

Impact of distributed linking on achieving interoperability faster and cheaper
Benefits of Linking Engineering Data

Knowing exactly how a requirement has been tested (vs believing what is written in a document)

Knowing what is impacted if something changes

Knowing when having to run again simulation models

Knowing how to run simulation models in the right sequence with the right parameters

Linked Engineering Data is more meaningful than siloed Engineering Data
What is a Link?

Each linked artifact needs to have a unique global/universal identifier (e.g. URL)

Each link has a type so that we understand the meaning of the relationship

Each link has a direction from source to target

---

Link Source
Identifier
https://rm.koneksys.com/api/oslc/project1/requirement/FAA23.155

Link Type
Identifier
http://open-services.net/ns/rm#:validatedBy

Link Target
Identifier
https://rm.koneksys.com/api/oslc/project1/testcase/23.155
Certain applications describe many different aspects of a system, so they provide support for data integration (PLM, ALM, MBSE).

Many useful capabilities provided by integration apps: visualization, analysis, etc.

Should ALL the links be created in such a central application?

Until now, links are created within a “central” application.
Traditional Centralized Link Creation Strategy

Problems:

- **Not user-friendly** for engineers to have to switch to a different application
- **Importing data not always possible** due to vendor lock-in (lack of integrations)
- **Schema often not flexible enough** to describe new link types
- **No single database suitable for all purposes!** No one size fits all! Example: data lake for IoT data vs data warehouse for “traditional” PLM data

“Central” Data Store (e.g. PLM/ALM/MBSE)

Centralized Link Creation does NOT SCALE
Distributed Link Creation Strategy

Links are created from **within ANY application**. Advantages:

- **User-friendly**: Links created only as needed by engineers within their familiar applications
- **Central registry for linking rules** to ensure that only meaningful links of a certain type are used between artifacts of a certain type
- **Multiple smaller and more manageable domain-specific schemas** instead of multidomain monolithic schemas
Link Creation Decoupled from Link Management

Links can be persisted in multiple storage solutions at time of creation.

Data and Links are accessible to API clients and can be collected and saved in another storage solution for:

- Query purposes
- Visualization
- Analysis
- Triggering automated workflows

Link Management not constrained to the capabilities of a single application.
Finding Link Targets Requires Application-specific Search Dialogs

Application-specific search dialogs exposed by OSLC APIs

OSLC

Requirements

OSLC

Simulation Models

OSLC

3D Models/PLM

OSLC

SysML Models
Traditional API Adopting Closed World Assumption

Example of a valid API update request

Before HTTP Request

API Client  →  HTTP PUT Request

API

Data Source

New links to “expected” artifacts

After HTTP Request

API Client  →  200 OK

API

Data Source

Example of a valid API update request
Traditional API Adopting Closed World Assumption

Before HTTP Request

API Client

HTTP PUT Request

API

New links to “unexpected” artifacts

Data Source

Example of an invalid API update request

CWA: what is not currently known to be true, is false (or only what is known to be true, is true)

After HTTP Request

API Client

X 400 Bad Request

Data Source

API
OSLC API Adopting Open World Assumption

Example of an valid API update request

**OWA:** what is not currently known to be true **may be true** (or only what is known to be false, is false)
OSLC API Adopting OWA decoupled from Linking Rules adopting CWA

Before HTTP Request

API Client

[Diagram Showing Before HTTP Request]

HTTP PUT Request

New links to “unexpected” artifacts

Data Source

Central Registry for Linking Rules

New links valid?

After HTTP Request

API Client

200 OK

[Diagram Showing After HTTP Request]

OSLC API and Linking rules are decoupled to support the reuse of an OSLC API
## Link Lifecycle Phases & Key OSLC Concepts

<table>
<thead>
<tr>
<th>To find link targets</th>
<th>Embeddable <strong>OSLC application-specific search dialogs</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>To find OSLC search dialog</td>
<td>OSLC API resources and services self-discoverable by API clients through <strong>hypermedia</strong></td>
</tr>
<tr>
<td>To create the link</td>
<td><strong>OSLC API adopts Open World Assumption</strong> and complies with linking rules</td>
</tr>
<tr>
<td>To persist the link</td>
<td><strong>OSLC agnostic</strong>: New link is created and persisted at specific location(s)</td>
</tr>
<tr>
<td>To access the link</td>
<td>Links exposed as <strong>properties of OSLC resources</strong> to be consumed by other applications</td>
</tr>
<tr>
<td>To review the status of a link</td>
<td><strong>no OSLC standard</strong></td>
</tr>
<tr>
<td>To expose changes to links (and associated artifacts)</td>
<td>Change events (deltas) exposed using <strong>OSLC TRS</strong></td>
</tr>
</tbody>
</table>
Successful Schemas for Data Exchange/Interoperability

Suitable for exchanging data between applications within the same domain with a clearly defined scope.
The larger the domain scope...

- the more likely the schema needs to be updated to stay relevant,
- the harder it is for implementations of the schema to support all its concepts and to stay up-to-date with schema changes
- the less likely the schema will be adopted,
- and the less likely the schema will provide value

Multi-domain schemas getting bigger and bigger...
In an ideal world from my perspective:

- Standards defined similarly as schema.org vocabularies (used for interoperability at Web scale)
- **Small domain-specific standards**
- **Cross-domain standards** defining link types for cross-domain relationships
- Domain-independent standards like for global configuration management etc.
- **No multidomain “monolithic” schemas**
- No semantic overlaps between different schemas
- Defining semantics using applied category theory?

We can achieve more interoperability by defining smaller more manageable standards
Data Integration: Old vs New (OSLC)

Old

- Single “central” application to define links
- Data accessible as files
- Data having to be transformed into a neutral “central” data format before being linked
- Links described in a vendor-specific format
- Links can only be accessed, managed, analysed, visualized within a single “central” application

New

- Any application can define links
- Data directly accessible as API resources (“objects”) at any level of granularity
- Data is linked by linking API resources (simple HTTP PUT operation to update an API resource representation)
- Links described in neutral open format
- Decoupling of concerns: Links can be accessed, managed, analysed, visualized in different applications
Thanks and get in touch!
axel.reichwein@koneksys.com