2017
annual INCOSE
international workshop
Los Angeles, CA, USA
January 28 - 31, 2017

Bill Chown, OASIS OSLC Steering Committee

OSLC Core 3.0 Overview

www.incose.org/IW2017

Members of OSLC can access this at
https://tools.oasis-open.org/version-control/svn/oslc-core/trunk/specs/oslc-core.html

Part 1

Overview

Goals of OSLC

The OSLC (Open Services for Lifecycle Collaboration) initiative supports integration between a heterogeneous set of tools and components from various sources using an architecture that is minimalist, loosely coupled, and standardized.

OSLC is based on World Wide Web and Linked Data principles, such as those defined in the W3C Linked Data Platform [*LDP*], to create a cohesive set of specifications that can enable products, services, and other distributed network resources to interoperate successfully [*LDP*].

OSLC Architecture

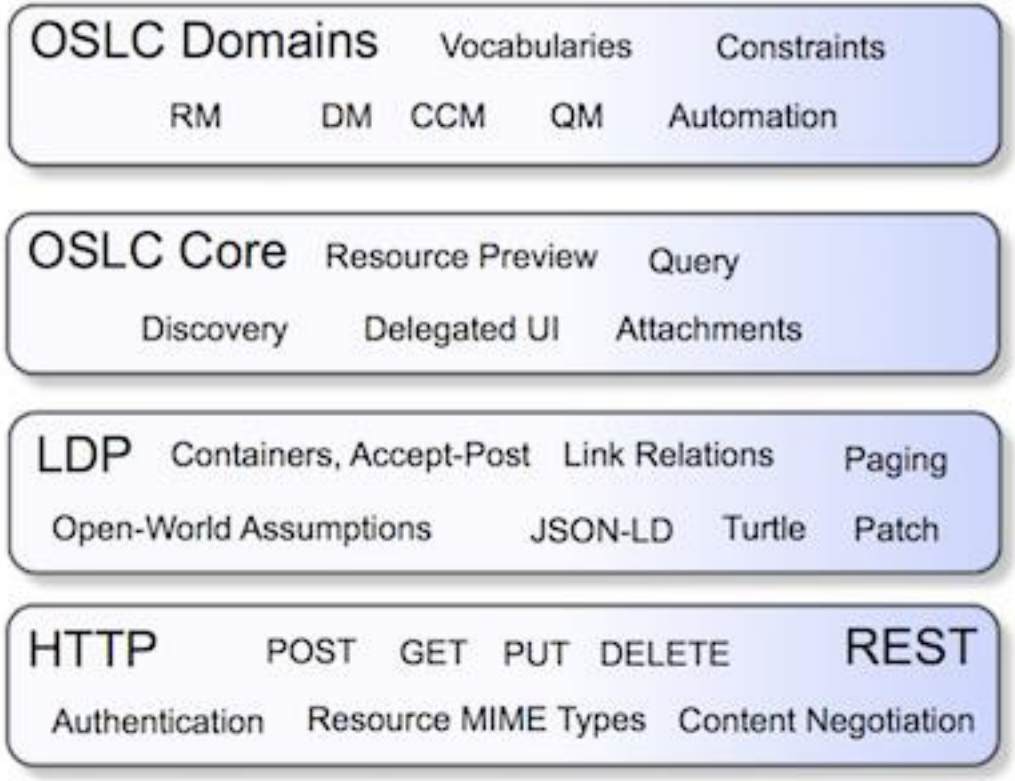
Layered architecture builds on Linked Data

Domains of interest that maintain separation of concerns and establish collaborative value streams through integration

Discoverability through Minimal, discoverable, self-describing capabilities to *enable* application integration

Reducing Variability through Self-describing, semantically rich, linked data resources leveraging HATEOAS

Address Complexity through HTTP and REST as the standard mechanism for distributed, loosely coupled APIs



OSLC Change Management 3.0 and OSLC Configuration Management 1.0 Specifications, OASIS

OSLC Core 3.0 Specification, OASIS

LDP 1.0 Specification, LDP.next Working Group, W3C

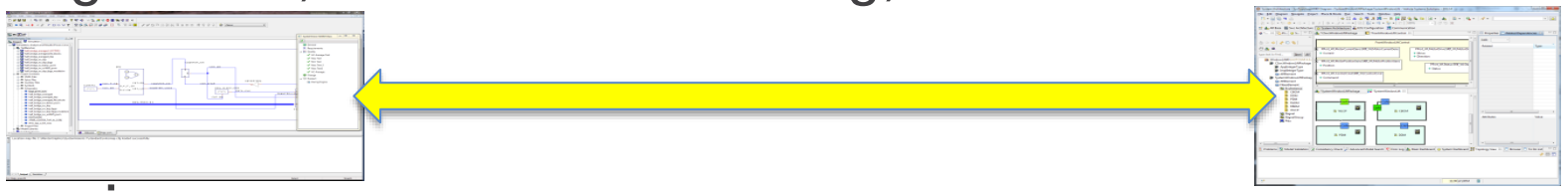
HTTP 1.1 Specification, IETF

Applicability of OSLC

- Domain-driven scenarios inspire standardization of common capabilities across disciplines
  - Disciplines include Change Management, Requirements Management, and Quality Management
  - Cross-domain scenarios such as Application Lifecycle Management (ALM) & DevOps, Product Lifecycle Management (PLM), and Integrated Service Management (ISM)
- The OSLC approach focuses on software lifecycle management to ensure it meets a core set of scenarios and requirements

Key Interactions in the Flow

- Data
  - E.g. netlist, schematic to cabling, etc. Bulk data transfer



Not the focus of OSLC today

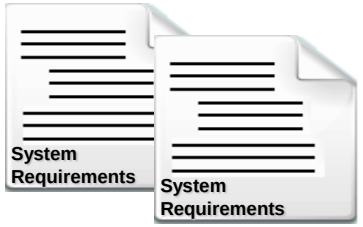
Netlist or Transform

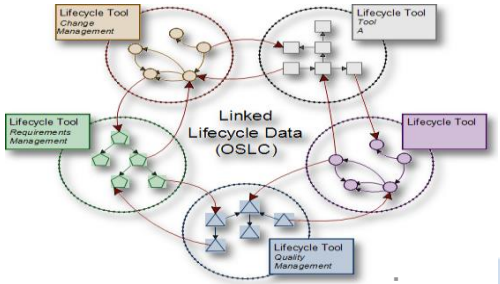
- Behavior
  - Executable models, run time code, functional co-simulation

Not the focus of OSLC at all

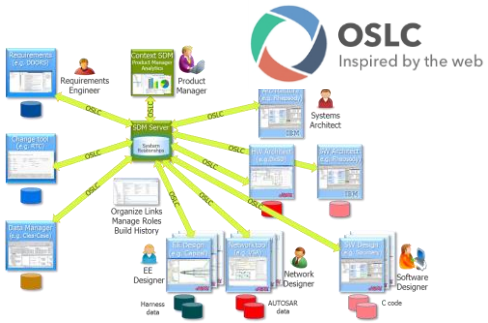
FMI, SVX, Codelink

- Intent
  - Requirements, work items, dependencies, meaning



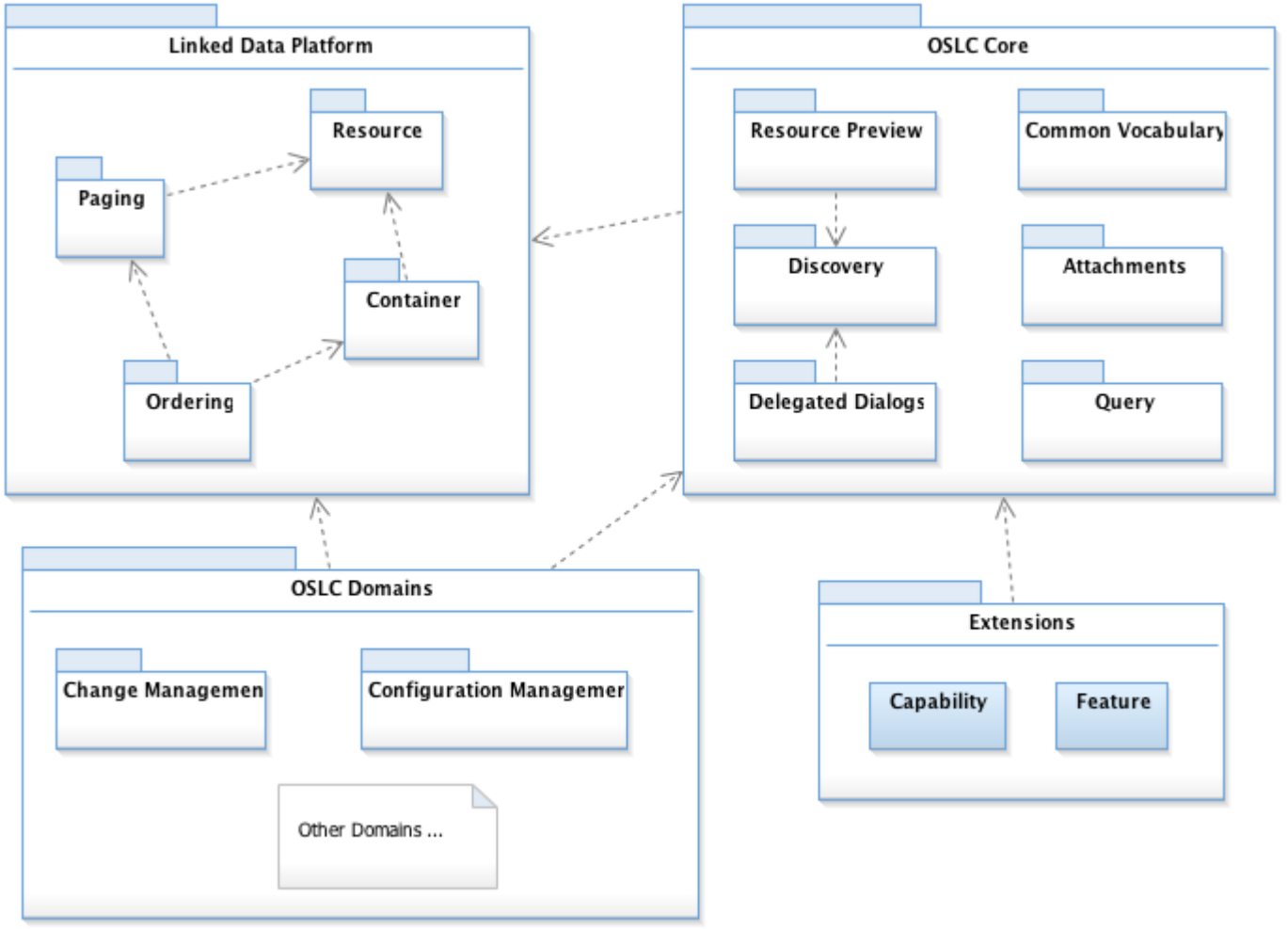


Linked Data, Shared Information



OSLC Core 3.0

- OSLC Core 3.0 builds on capabilities developed in different standards organizations, TCs and working groups



Goals for OSLC Core 3.0

The specific goals for OSLC Core 3.0 are to build on the existing OSLC Core 2.0 specifications to further facilitate the development and integration of domains and supporting tools that address additional integration needs. Specifically:

- Integration is based on an open standard, and not controlled by any single vendor
- OSLC 3.0 is based on the new W3C Linked Data Platform standard which provides a solid foundation for reading and writing linked data resources
- The specifications are simpler, more consistent and will potentially be more attractive to, and easier to consume by new integrations
- There are some new capabilities including Attachments, inverse link labels, traceability and impact types
- Domain vocabularies can be improved for data consistency and removing data gaps
- All Resource Shapes are provided in machine readable [*Turtle*] files

Terminology

- Previous revisions of OSLC-based specifications (OSLC Core2.0), used terminology that may no longer be relevant, accurate or needed any more.
  Some of those deprecated terms are:
  - Provider (deprecated)
    - See Server
  - Consumer (deprecated)
    - See Client
- Terminology uses and extends the terminology and capabilities of W3C Linked Data Platform, W3C's Architecture of the World Wide Web and Hyper-text Transfer Protocol
  - OSLC Server
    - LDP Server that also supports capabilities defined by at least on OSLC-based specification
  - OSLC Client
    - LDP Client that uses capabilities defined by some OSLC-based specifications.

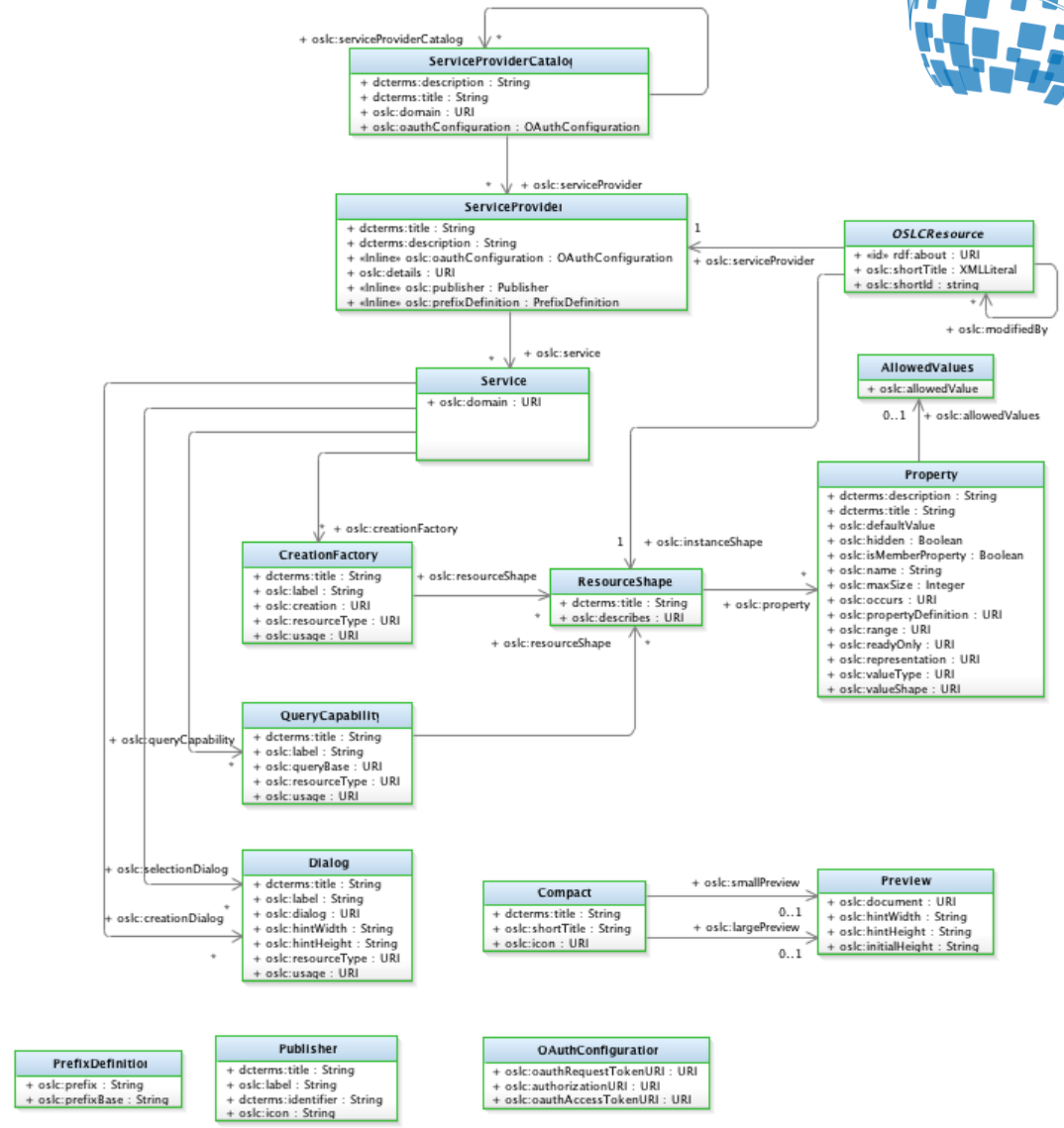
    A particular software component or application could be an OSLC Server supporting a set of domains, and an OSLC Client of other domains depending on its needs.
  - Domain
    - A topic area of a specific focus area and/or collection of disciplines. Often OASIS OSLC-affiliated TCs are organized around a domain.

Part 2

Discovery

Resource Discovery

- A common approach for HTTP/LDP-based servers to be able to publish their RESTful API capabilities and how clients can discover and use them.
- *Static Upfront* and *Dynamic Incremental* options to discovery approaches are defined



Part 3

Resource Preview

Resource Preview

- Resource Preview specifies a technique to get a minimal HTML representation of a resource identified by a URL

- Applications often use this representation to display a link with an appropriate icon, a label, or display a small or large preview when a user makes some gesture over a link
- Servers can provide both small and large previews. A client might show a small preview first, then if the user gestures, show additional details from the large preview. Servers suggest sizes for previews, and previews can ask to be resized after they are displayed.
- A client only needs to know the URI of a resource to display a link and a preview. It doesn't need to know anything else about the resource. Clients don't need to copy, synchronize, or cache any data.

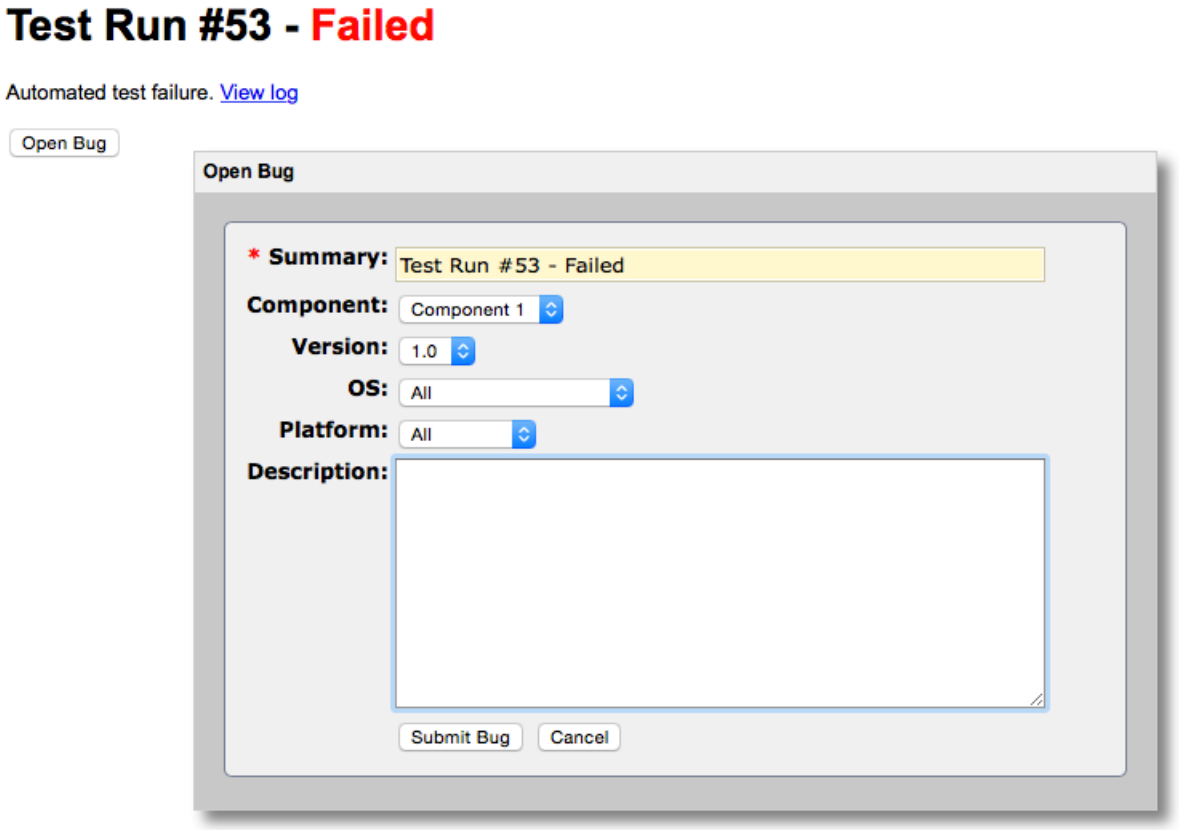
Part 4

Delegated Dialogs

Delegated Dialogs

- Delegated Dialogs allow one application to embed a creation or selection UI into another using HTML iframe elements and JavaScript code

- The embedded dialog notifies the parent page of events using HTML5 postMessage



Part 5

Attachments

Attachments

A minimal way to manage attachments related to web resources using LDP-Containers and Non-RDF Source

- Each resource that supports attachments has an attachment container, which is an LDP container.
- Clients discover the attachment container through an HTTP OPTIONS method and HTTP Link header.
  Clients look at the `ldp:contains` property on the container for the attachments, & get the attachment by simply making an HTTP GET request to the attachment URI

Part 6

Resource Shape

Resource Shape

- The shape of an RDF resource is a description of the set of triples it is expected to contain and the integrity constraints those triples are required to satisfy.
- Applications of shapes include validating RDF data, documenting RDF APIs, and providing metadata to tools that handle RDF data such as form and query builders.
- OSLC servers should describe their resources using resource shapes if they wish to integrate with OSLC Core3.0 clients or servers that are expecting resource shapes.
- OSLC domain specifications use shapes to specify the constraints on their domain vocabulary elements that must be supported by servers.

Part 7 (no more details included in this presentation)

Vocabulary

OSLC Summary

- Core 3.0 is close to publication after final review
  - Available now to OASIS-OSLC members
  - Will be public and available soon
- Domains TC getting established
- Change and Configuration Management TC is active and working on next revision of CCM specification, which will then separate in Change Management and Configuration Management for future work

Join OSLC to participate!

Contact me at bill_chown@mentor.com with any questions, etc.

OSLC Domains

A new Domains WG (TC) is forming within OASIS-OSLC

- Combined attention to the various domains, rather than separate domain groups
- Migrate existing finalized OSLC v2.0 domain specifications
  - Tracked Resource Set 2.0,
  - Architecture Management 2.0
  - Asset Management 2.0
  - Automation 2.1
  - Performance Monitoring 2.0
  - Quality Management 2.0
  - Requirements Management 2.0
- Capture requirements and use cases for other related domains

2017
annual INCOSE
international workshop
Los Angeles, CA, USA
January 28 - 31, 2017

www.incose.org/IW2017