Mapping Federated AEC Projects to Industry Standards using Virtual Views

Jeroen Werbrouck – Pieter Pauwels – Jakob Beetz – Erik Mannens
INTRODUCTION
Using a 3rd party Web BIM service, the Asset Owner localises a damage pattern on a picture of the Facility Manager, linking it to an existing building element via the as-built 3D model provided by the Architect, and referring to external regulation datasets provided by the government.
CONTEXT: SOLID

- **Web ID:** A URL that represents an actor on the Web ➔ “Web username”
- **Data Pod:** Personal data storage linked to a Web ID ➔ Linked Data Platform (LDP) + AUTH
- **Identity Provider:** Instance hosting your WebID (and Pod) ➔ Can be self-hosted

© Mads Holten Rasmussen, 2018

© Ruben Verborgh, 2018
RESERCH BASELINE

- “Semantics” should be managed in metadata resources rather than in URLs/containment triples
- Metadata can be used to containerise resources according to certain parameters
- A resource being contained in a Solid POD/LDP container does not mean it cannot be contained virtually elsewhere
- Separate “Storage” from “Discovery”

“Virtual views” generate particular views on federated data catalogs, based on metadata queries

⇒ URLs should not contain “meaning”, in order to keep them stable
⇒ Compliance with standard-imposed data structures can happen at presentation level (middleware)
The project registry of a stakeholder is no more than a specific “aggregator”. It is modelled as a DCAT Catalog, in this case one that groups projects this stakeholder participates in.

The “project access point” is an LDP container which aggregates relevant project data. It effectively contains (ldp:contains) the contributions of the stakeholder (/local/), and virtually contains relevant contributions of other stakeholders in the project.

* In Solid, “.meta” files are mapped to specific LDP containers. Dereferencing the container yields the .meta resource.
** By default, “.meta” files in Solid cannot be edited directly. In this case, this restriction was overruled in favor of project organisation.
DATASET STORAGE

```
<> a dcat:Dataset ;
  dct:title "the title of the dataset";
  dcat:keyword "topology" ;
  dcat:distribution <distribution1> , <distribution2> .

<distribution1> a dcat:Distribution ;
  dc:format <https://www.iana.org/assignments/media-types/text/turtle>;
  ex:status "shared"
  dcat:downloadUrl </distribution1> .
```

Query **UNION** to generate virtual views
Solid Pods use LDP (Linked Data Platform) to organise resources in a hierarchic, **tree-based** container structure.

Virtual view structured as an LDP Container

Virtual containers can organise resources in a **graph-based** “container” structure

- Solid compatibility: use LDP containment triples pointing to resources that are not effectively hosted on the Pod in this hierarchy

- (alternative: DCAT: use `dcat:dataset` to aggregate datasets into `dcat:Catalog` (`rdfs:subClassOf dcat:Dataset`)
API DISCOVERY

- DCAT services registered in “project access point”
- Endpoint URL
- Standard number

```turtle
# [...] Pointers to partial projects etc. ]
<> dcat:service <#f20dd53e>.
<#f20dd53e> a <http://www.w3.org/ns/dcat#DataService>;
    dcat:endpointURL <http://www.w3.org/ns/dcat#DataService>;
    dcterms:conformsTo <https://icdd.my-api.org/7712c210/>;
    dcterms:conformsTo “ISO 21597” .
```
USE CASES
ISO 19650 – STAGES OF PUBLICATION

- Trivial example: can be defined by a single property
- “Satellite”: SPARQL CONSTRUCT a “virtual container” which is served as a classic LDP container
- User Interface: SPARQL SELECT

```sparql
CONSTRUCT {?virtualContainer ldp:contains ?downloadURL }
WHERE {
    # Dataset discovery starting from Project Access Point
    # initial source: project access point
    # link traversed source: partial project
    ?partial lbds:hasDatasetRegistry ?dsr .
    # link traversed source: dataset registry
    # Subquery for dataset filtering
    # link traversed source: dataset
    # aggregate all resources with status "shared" (example ontology)
    ?ds ex:publicationStatus "shared";
    dcat:distribution/dcat:downloadURL ?downloadURL .
    BIND(UUID() as ?virtualContainer)
}
```

Trivial example: can be defined by a single property

“Satellite”: SPARQL CONSTRUCT a “virtual container” which is served as a classic LDP container

User Interface: SPARQL SELECT
ISO 21597: ICDD

1. “Payload Triples”

CONSTRUCT {?virtualContainer ldp:contains ?download }
WHERE {
    # [...dataset discovery patterns starting from Project Access Point (cf. Listing 4)]
    # [link traversed source: Dataset registry]
    ?dist dcat:downloadURL ?download;
    dcat:mediaType ?mt .
    FILTER(?mt IN ( <https://www.iana.org/assignments/media-types/text/turtle>,
                    <https://www.iana.org/assignments/media-types/application/rdf+xml>,
                    <https://www.iana.org/assignments/media-types/application/ld+json>
                # [... other RDF serialisations ...]
                ))
    BIND(UUID() as ?virtualContainer)
}

2. “Payload Documents”

FILTER (?mt NOT IN(…))
3. Ontology resources

CONSTRUCT {?virtualContainer ldp:contains ?vocabulary } 
WHERE {

  # [...reference registry discovery patterns (lbds:hasReferenceRegistry)]
  # link traversed source: Reference registry

  BIND(UUID() as ?virtualContainer)
}
ISO 21597: ICDD

4. Links.rdf

CONSTRUCT {
    ?concept a ls:Link ;
    ls:hasLinkElement ?le .
    ?le a ls:LinkElement ;
    ls:hasDocument ?distribution ;
    ls:hasIdentifier ?id .
    ?id ls:identifier ?identifier .
} WHERE {
    # [...]reference registry discovery patterns (lbds:hasReferenceRegistry)
    # link traversed source: Reference registry
    ?concept a lbds:Concept ;
    lbds:hasReference ?le .
    ?le lbds:hasIdentifier ?id ;
    lbds:hasDocument ?distribution .
    ?id lbds:hasIdentifier ?identifier .
}
CONSTRUCT {
    ct:creator ?creator ;
    ct:description ?projectDescription .
    ?dist a ct:InternalDocument ;
    ct:description ?dsDescription ;
    ct:format ?format ;
    ct:filename ?filename .
} WHERE {
    # [...] dataset discovery patterns starting from Project Access Point (cf. Listing 4)
    # link traversed source: Dataset registry
    ?ds dct:creator ?creator ;
    rdfs:comment ?dsDescription ;
    BIND(UUID() as ?index)
    BIND(replace(str(?mt), str("https://www.iana.org/assignments/media-types/"), str("")) as ?format)
}

ISO 21597: ICDD
Standardised API responses as JSON

JSON-LD = JSON + context

Example: BCF API

- Bundle federated LBDserver projects and present as (one or multiple) BCF-API endpoints

(https://github.com/BuildingSMART/BCF-API)
BCF API

GET /bcf/{version}/projects/{project_id}/topics

LBDserver Query

CONSTRUCT {  
?ds dct:identifier ?identifier ;  
dc:creator ?creator ;  
dct:title ?title ;  
rdfs:label ?label ;  
dct:created ?creationDate .  
} WHERE {  
# [...dataset discovery patterns starting from Project Access Point]  
?ds a bcfOwl:Topic ;  
dct:identifier ?identifier ;  
dc:creator ?creator ;  
dct:title ?title ;  
rdfs:label ?label ;  
dct:created ?creationDate .  
}

Standard BCF API Response

[...otherTopics,  
{
  "guid": "A211FCC2-3A3B-EAA4-C321-DE22ABC8414",
  "server_assigned_id": "ISSUE-00078",
  "creation_author": "Architect@example.com",
  "title": "Example topic 2",
  "labels": ["Architecture", "Heating", "Electrical"],
  "creation_date": "2014-11-19T14:24:11.316Z"
}
]

JSON-LD context

{  
"@context": {
  "guid": "http://purl.org/dc/terms/identifier",
  "creation_author": {"@id": "http://purl.org/dc/terms/creator", "@type": "@id"},
  "title": "http://purl.org/dc/terms/title",
  "labels": "http://www.w3.org/2000/01/rdf-schema#label",
  "creation_date": "http://purl.org/dc/terms/created"
}
}
CONCLUSION
Metadata-based, “virtual views” avoid the need to re-allocate LDP resources on a Solid Pod and modify their URL.

LBDserver storage patterns are discipline/topic-agnostic – this is covered semantically in the metadata documents.

Virtual views allow compatibility with existing industry standards, in a federated LD-environment.

The end user / client uses Pod-external services to see project data through a specific lens.
The “dynamic views” are read-only; data manipulation happens directly at storage level

Ontological mappings – e.g. base IFC (or ifcOWL) file, but satellite presents a view “as if LBD”

Complex configurations – e.g. viewing BCF Topics as discussion threads

“Blurry” differences between metadata and actual knowledge graph

Dynamic access-rights management

FUTURE WORK


**Main Research**
Jeroen Werbrouck  
PhD Researcher (FWO Flanders)  
Ghent University | RWTH Aachen  
jeroen.werbrouck@ugent.be

**Promotors**
Erik Mannens (Ghent University | IMEC)  
Pieter Pauwels (Ghent University | TU Eindhoven)  
Jakob Beetz (RWTH Aachen)
THANK YOU!