Querying Large SysML Models

Annelisa Sturgeon
Aleksander Przybylo
James Milstead
Presentation outline

• Presenters’ bios
• Overview of SLATE-FI implementation and use
• Migrating to SysML
  • What fits into SysML vs. what doesn’t
  • Occurrence modelling
• Query approach trade study
  • Basic assumptions and requirement
  • The three high level options being studied
• Recommendation
• Questions?
Authors’ bios

Global Product Data Interoperability Summit | 2018

• **Annie Sturgeon**
  Joined Boeing full time in 2017 after three internships in Production Engineering. Currently works on MBSE implementation in Everett, WA.

• **Alek Przybylo**
  Joined Boeing in 2010 as Process Engineer. Previous experience includes consulting for aerospace companies in the PLM and CAD fields. Currently works on MBSE development in Everett, WA.

• **Jim Milstead**
  Software developer with the BR&T Product Lifecycle Modeling & Simulation team in Huntsville, AL. Associate Technical Fellow since 2010.
Overview of SLATE-FI implementation and use

• SLATE-FI was built from out of the box SLATE and customized by Boeing
• Interface Control Design tool used to model logical architectures of systems
• First fully functional Model Based Systems Architecting tool
• Complex Data Schema
• Legacy programs generated large data base from SLATE-FI
SLATE-FI Occurrence Modeling

- Used to model Logical Architecture
- System Objects are modeled as Definitions
- These Objects are instantiated on higher level System Definitions Objects as Occurrence Objects
- This allows for re-use of System Objects
- Changes made to System Definition Objects are propagated to Occurrences through Occurrence links
Overview of SLATE-FI Occurrence Modeling

Malone, R., Friedland, B., Herrold, J., & Green, G. 2017 Figure 5. Occurrence Data Model
Migrating Model Data from Legacy Representations to SysML

Global Product Data Interoperability Summit | 2018

- Export SLATE-FI data schema to XML file
- Import SLATE-FI data objects with attributes into a UML Profile
  - Objects as Stereotypes
  - Attributes as Properties on corresponding Stereotypes
- Identify SysML representation of SLATE-FI data objects and relationships
  - Update Profile Elements with SysML representations
  - Use Profile in a Data Model to define allowed relationships between profile objects
- Import SLATE-FI data using Profile with SysML representations
Basic assumptions and user requirements

• Queried data needs to be live (i.e. we should not rely on daily batch replications)
• Query response time needs to be “reasonable” for all existing use cases
• SPARQL language must be supported
Current system architecture

- Teamwork Cloud
- MMS
- Postgres
- ElasticSearch
- MDK
  - Containment and Transclusion Tree repository
  - Node index
  - Branch information
- Cameo client
- Node data
High level solution concepts

• Option A: SPARQL parsing

• Option B: Additional graph repository

• Option C: MMS component replacement
Option A – SPARQL parsing

Global Product Data Interoperability Summit | 2018
SPARQL parsing (SPARQL -> SQL)

```sparql
SELECT ?o
{
  <SOME_ENTITY#12345> <SOME_ENTITY#FIELD> ?o .
}

SELECT FIELD AS o
FROM SOME_ENTITY
WHERE ID = 12345

SELECT ?s
{
  ?s <SOME_ENTITY#FIELD> Value .
}

SELECT ID AS s
FROM SOME_ENTITY
WHERE FIELD = Value
```
SELECT ?CRName ?ReqName
{
}

SELECT
  PACKAGE.NAME AS CRName,
  ELEMENT_CLASS.NAME AS ReqName
FROM
  PACKAGE
INNER JOIN ELEMENT_DEPENDENCY ON PACKAGE.ID = ELEMENT_DEPENDENCY._SOURCEIDS
INNER JOIN ELEMENT_CLASS ON ELEMENT_CLASS.ID = ELEMENT_DEPENDENCY._TARGETIDS
WHERE
  ELEMENT_CLASS.NAME="Employ SysML"
SPARQL parsing (SPARQL -> SQL+ES)

**Select:**
CRElement.NAME, ReqElement.NAME

**Filter:**
ReqElement.ID=DependencyLink._TARGETIDs

**Filter:**
CRElement.ID=DependencyLink._SOURCEIDs

**ReqElement**
TYPE="ELEMENT_CLASS" AND NAME="Employ SysML"

**CRElement**
TYPE="PACKAGE"

**Dependency Link**
TYPE="ELEMENT_DEPENDENCY"
SELECT ?s ?child ?child_name
{
}

SPARQL parsing (SPARQL -> SQL+ES)
Option B – Additional graph repository (Mark Logic)

Global Product Data Interoperability Summit | 2018

Diagram:
- Teamwork Cloud
- MMS
- Postgres
- Elasticsearch
- Cameo client
- MDK
- NiFi
- FetchElasticSearchHttp Processor
- ExecuteSQL Processor
- PutMarkLogic Processor
- SPARQL templates
- Mark Logic
- Query plugin
Option B – Additional graph repository (GraphDB)
Option C – MMS component replacement (GraphDB)
Option C – MMS component replacement (Neo4j)

- Similar approach to GraphDB
  - GraphAware Neo4j replicates graph into ElasticSearch
- Relies on proprietary graph schema and query language (Cypher)
- Similar visual graph capabilities to GraphDB
Option D – Bypass MMS

OV-5b Operational Activity Model

1. Custom Cameo Plugin Registers Listener for Server Save Event
2. User Checks in Change to Teamwork Cloud
3. Cameo Plugin Sends List of Checked Out Entity IDs to REST Service
4. Rest Service Retrieves List of Element IDs (Artifact List) for latest revision from Teamwork Cloud
5. Rest Service Compares Element ID list with Previous Revision to get list of Added Elements
6. Generate RDF for Modified/Added Elements
7. Update GraphDB with Modified RDF
Other options (not explored)

- Cameo Collaborator OSLC
  - Allows for real-time query without synchronization
  - Allows exposing part of model as OSLC endpoint
- 3DX Platform – EXALEAD
  - White paper has few technical details
- Anzo
  - Claims ability to handle 1T triples
  - Nice dashboard capability – visual query builder
- Sparql plugin
  - Developed at GeorgiaTech
  - Appears to run a full translate to RDF for each query
**SPARQL UI**

Global Product Data Interoperability Summit | 2018

- **Visual SPARQL Builder**
  - Code available at [https://github.com/leipert/vsb](https://github.com/leipert/vsb)
    - Code not touched for 3 years
    - MIT License (good for commercial)
  - Configuration needed to connect to custom endpoint
    - Could not find list of verified supported endpoints

- **SPARQLGraph**
  - Code available at [https://github.com/tadKeys/sparqlgraph](https://github.com/tadKeys/sparqlgraph)
    - Code not touched for 4 years
    - GPL 2.0 License (changes must also be open source)
  - Highly customized to medical application
    - Open source could be modified
Evaluation criteria

- **Performance** (<1s/1-5s/5-15s/>15s)
- **Impact to commit performance** (none/low/medium/high)
- **Data duplication** (none/low/medium/high)
- **Data transformation** (none/low/medium/high)
- **Coding effort required** (none/low/medium/high)
- **License cost** (none/low/medium/high)
- **Future-proofness** (low/medium/high)
Benchmark query
## Evaluation criteria

### Global Product Data Interoperability Summit | 2018

<table>
<thead>
<tr>
<th>Option</th>
<th>Performance</th>
<th>Impact to commit</th>
<th>Data duplication</th>
<th>Data transformation</th>
<th>Coding effort</th>
<th>License cost</th>
<th>Future-proofness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cameo report</td>
<td></td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>Option A – Query parser</td>
<td></td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>High</td>
<td>None</td>
<td>Medium</td>
</tr>
<tr>
<td>Option B – External RDF store</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>High</td>
</tr>
<tr>
<td>MarkLogic</td>
<td>None</td>
<td>None</td>
<td>Medium</td>
<td>Low</td>
<td>None</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>GraphDB (Free)</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>Low</td>
<td>None (*)</td>
<td>Medium</td>
</tr>
<tr>
<td>Option C – MMS component replacement</td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td>GraphDB (Enterprise) (***)</td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td>Neo4j (Enterprise)</td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Option D – Bypass MMS (GraphDB)</td>
<td>Medium</td>
<td>High</td>
<td>High</td>
<td>Medium</td>
<td>None (*)</td>
<td>Medium</td>
<td>Medium</td>
</tr>
</tbody>
</table>

(*) – has limitations on parallel query (2) => Would potentially need Enterprise version anyways

(**) – license needed for ElasticSearch connector
Recommendations

Global Product Data Interoperability Summit | 2018

• Two recommended approaches
  • Bypass MMS and integrate Teamwork Cloud directly with GraphDB (no license cost but high development cost)
  • Mark Logic solution (lowest development cost but high license cost)
Questions