Extensible User Interface Framework for Faceted Browsing Applications

Master’s Thesis Defense

Eric Rozell, Tetherless World Constellation, RPI
Outline

• Motivation / Problem Statement
• Contributions Summary
• Historical Review
• Research Methodology
• Software Design
• Ontology Overview
• Web Services Implementation
• FacetOntology Implementation
• Evaluations
• Conclusions
Motivation

• Cruise Preparation Data Dashboard
  – Intended to address needs of multi sub-disciplines of oceanography

• Challenges for data managers:
  – Keeping up with rapidly evolving Web stds
  – Designing user friendly interfaces

• Challenges for scientists
  – Data service discovery
  – Learning curve for search interfaces/services
Problem Statement

• Semantic Web service vocabularies (e.g., OWL-S [1]) connect data and services layer
  – Enables automation of tasks such as composition, discovery [2], etc.

• Can similar vocabularies be developed, connecting the data and service layer with the UI layer?

• What levels of automation can this support?
  – E.g., discovery, invocation, composition, and re-use
Contributions Summary

1. Lightweight ontology to describe UI components that can be linked to compatible Web service inputs and outputs and FacetOntology descriptions
2. Framework to deploy faceted data dashboards from RDF descriptions of UI components, Web services and instances of the FacetOntology
3. Extensions to the FacetOntology to better leverage the full capabilities of the S2S framework
4. Discussion of the trajectory of S2S towards a more general application integration framework
5. Synthetic approach to Semantic Web Methodology and Technology Development Process, incorporating cross-cutting use cases
State-of-the-Art Applications with S2S
Historical Review

• Faceted Browse Frameworks
  – Longwell [3], Exhibit [4], Flamenco [5], /facet [6], mSpace [7]

• Semantic Web Service Vocabularies, Extensions, and Frameworks
  – OWL-S [1], WSMO [8], SWSO [9]
  – SAWSDL [10], OpenSearch [11]
  – SADI [12] and SSWAP [2]
Semantic Web Methodology & Technology Development Process

- Establish and improve a well-defined methodology vision for semantic technology based on application development
- Leverage controlled vocabularies, etc.

Open world: evolve, iterate redesign, redeploy

Evaluation

Leverage technology infrastructure

Adopt technology approach

Science/expert reviews and iteration

Use tools

Develop model ontology

Small team, mixed skills

Use case

Rapid prototype

[13]
Ontology Overview

• Applying SWM, designed an ontology to model the domain-independent task of building data dashboards

• Other UI ontologies did not fit the use case...
  – Semantic Hypermedia Design Method [16] and Rich Internet Applications ontology were too granular
  – Fresnel [17] based on RDF data, which was not desirable for initial use cases and scalability
Ontology Overview – Metrics and Criteria

• Description logic complexity: $ALQ(D)$
• Gruber’s Design Criteria [18]
  – Clarity: Ontology changes to improve understandability, e.g., parameter $\rightarrow$ input
  – Coherence: Ontology is logically consistent
  – Extendibility: Demonstrated by core extensions
  – Minimal encoding bias: no unnecessary use of DL constructs (e.g., range or cardinality constraints)
  – Minimal ontological commitment: very small core ontology, 65 triples
S2S Implementations for Faceted Browsing

• Undesirable to take a dependency on RDF data for faceted browsing framework
• Initial implementation for OpenSearch [11] (RDF compatibility through manually curated SPARQL queries in Web script)
• Proposed compatibility for SAWSDL [10]
• FacetOntology [19] implementation in latest iteration for automated SPARQL query generation
OpenSearch Faceted Browsing

• Based on URL templates, e.g.
  http://example.com/search?q={searchTerms}

• Two extensibility aspects used:
  – URL template parameters correspond to s2s:Inputs
  – “rel” attributes of XML tags correspond to s2s:Interfaces

• To support faceted browse, must define N+1 <Url> templates
  – N s2s:InputValuesInterfaces, 1 s2s:SearchResultsInterface
SAWSDL Faceted Browsing Proposal

• S2S analogies to WSDL
  – s2s:Interface $\rightarrow$ wsdl:operation
  – s2s:Input $\rightarrow$ wsdl:input
  – s2s:Output $\rightarrow$ wsdl:output

• SAWSDL annotation capabilities
  – sawsdl:modelReference
  – Embedded RDF in WSDL
  – liftingSchemaMapping and loweringSchemaMapping
FacetOntology Summary

Diagram:

- Product
  - TeraByteHardDrive
    - hasPrice: 80.00 (xsd:double)
  - hasManufacture: SeaGate
    - rdfs:label: Seagate Technology (xsd:string)
- Manufacturer
  - SeaGate
    - is a: Seagate Technology (xsd:string)
FacetOntology
Limitations

1. Tabular results structure.
3. Inability to filter out results and facet options that fall on a specified predicate path.
1. Tabular results structure.
3. Inability to filter out results and facet options that fall on a specified predicate path.
4. No support for common reasoning tasks, such as transitive and symmetric relationships.
5. No support for declaring named graphs in predicate paths.
6. No support for optional graph patterns.
FacetOntology
Limitations

1. Tabular results structure.
3. Inability to filter out results and facet options that fall on a specified predicate path.
4. No support for common reasoning tasks, such as transitive and symmetric relationships.
5. No support for declaring named graphs in predicate paths.
6. No support for optional graph patterns.
7. No mechanism to add context to the facet options.
FacetOntology
Limitations

1. Tabular results structure.
3. Inability to filter out results and facet options that fall on a specified predicate path.
4. No support for common reasoning tasks, such as transitive and symmetric relationships.
5. No support for declaring named graphs in predicate paths.
6. No support for optional graph patterns.
7. No mechanism to add context to the facet options.
8. No way to optimize for SPARQL queries over facets with shared predicate paths.
9. No support for co-dependent facet use cases.
FacetOntology

Limitations

Query With Implicit Variables

PREFIX : <http://example.com/>
SELECT ?goal WHERE {
  ?goal :hasManufacturer ?v11 .
  FILTER (?v11 = :Seagate)
  ?goal :hasManufacturer ?v21
  FILTER (?v22 = :USA)
}

Query With Explicit Variables

PREFIX : <http://example.com/>
SELECT ?goal WHERE {
  ?goal :hasManufacturer ?mfct.
  FILTER (?mfct = :Seagate)
  ?goal :hasManufacturer ?mfct.
  FILTER (?v22 = :USA)
}
## FacetOntology Extensions (foe:)

<table>
<thead>
<tr>
<th>URI</th>
<th>Domain</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foe:contextpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Adds predicate chains to context for results.</td>
</tr>
<tr>
<td>foe:sparqlSelectVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables must be selected from context.</td>
</tr>
<tr>
<td>foe:sparqlAggregateVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables should be included in a GROUP BY clause.</td>
</tr>
<tr>
<td>facet:labelpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Simple modification to support predicate chains to labels.</td>
</tr>
<tr>
<td>foe:Filter</td>
<td>N/A (owl:Class)</td>
<td>N/A (owl:Class)</td>
<td>Class added to contain information about filters for facets.</td>
</tr>
<tr>
<td>foe:filter</td>
<td>facet:Facet</td>
<td>foe:Filter</td>
<td>Relates a facet to filters that should be applied to it.</td>
</tr>
<tr>
<td>foe:filtervalue</td>
<td>foe:Filter</td>
<td>xsd:string</td>
<td>Specifies the actual constraint applied to filter results.</td>
</tr>
</tbody>
</table>
### FacetOntology Extensions (foe:)

<table>
<thead>
<tr>
<th>URI</th>
<th>Domain</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foe:contextpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Adds predicate chains to context for results.</td>
</tr>
<tr>
<td>foe:sparqlSelectVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables must be selected from context.</td>
</tr>
<tr>
<td>foe:sparqlAggregateVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables should be included in a GROUP BY clause.</td>
</tr>
<tr>
<td>facet:labelpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Simple modification to support predicate chains to labels.</td>
</tr>
<tr>
<td>foe:Filter</td>
<td>N/A (owl:Class)</td>
<td>N/A (owl:Class)</td>
<td>Class added to contain information about filters for facets.</td>
</tr>
<tr>
<td>foe:filter</td>
<td>facet:Facet</td>
<td>foe:Filter</td>
<td>Relates a facet to filters that should be applied to it.</td>
</tr>
<tr>
<td>foe:filtervalue</td>
<td>foe:Filter</td>
<td>xsd:string</td>
<td>Specifies the actual constraint applied to filter results.</td>
</tr>
</tbody>
</table>
## FacetOntology Extensions (foe:)

<table>
<thead>
<tr>
<th>URI</th>
<th>Domain</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foe:contextpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Adds predicate chains to context for results.</td>
</tr>
<tr>
<td>foe:sparqlSelectVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables must be selected from context.</td>
</tr>
<tr>
<td>foe:sparqlAggregateVariable</td>
<td>facet:Facet</td>
<td>xsd:string</td>
<td>Informs the SPARQL grounding that additional variables should be included in a GROUP BY clause.</td>
</tr>
<tr>
<td>facet:labelpredicate</td>
<td>facet:Facet</td>
<td>facet:Predicate</td>
<td>Simple modification to support predicate chains to labels.</td>
</tr>
<tr>
<td>foe:Filter</td>
<td>N/A (owl:Class)</td>
<td>N/A (owl:Class)</td>
<td>Class added to contain information about filters for facets.</td>
</tr>
<tr>
<td>foe:filter</td>
<td>facet:Facet</td>
<td>foe:Filter</td>
<td>Relates a facet to filters that should be applied to it.</td>
</tr>
<tr>
<td>foe:filtervalue</td>
<td>foe:Filter</td>
<td>xsd:string</td>
<td>Specifies the actual constraint applied to filter results.</td>
</tr>
</tbody>
</table>
### Facet Ontology Extensions

<table>
<thead>
<tr>
<th>URI</th>
<th>Domain</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foe:transitive</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Notifies the SPARQL grounding that the predicate should be treated (or queried) as a transitive property.</td>
</tr>
<tr>
<td>foe:symmetric</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Notifies the SPARQL grounding that the predicate should be treated (or queried) as a symmetric property.</td>
</tr>
<tr>
<td>foe:sparqlGraph</td>
<td>facet:Predicate</td>
<td>xsd:anyURI</td>
<td>Specifies that a predicate should be queried from a specific named graph.</td>
</tr>
<tr>
<td>foe:sparqlBinding</td>
<td>facet:Predicate</td>
<td>xsd:string</td>
<td>Specifies the variable that should be used for the predicate in a SPARQL query.</td>
</tr>
<tr>
<td>foe:codependent</td>
<td>facet:Facet</td>
<td>facet:Facet</td>
<td>Specifies that two facets share a predicate path.</td>
</tr>
<tr>
<td>foe:optional</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Specifies that a predicate should be queried as optional.</td>
</tr>
</tbody>
</table>
## FacetOntology Extensions

<table>
<thead>
<tr>
<th>URI</th>
<th>Domain</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foe:transitive</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Notifies the SPARQL grounding that the predicate should be treated (or queried) as a transitive property.</td>
</tr>
<tr>
<td>foe:symmetric</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Notifies the SPARQL grounding that the predicate should be treated (or queried) as a symmetric property.</td>
</tr>
<tr>
<td>foe:sparqlGraph</td>
<td>facet:Predicate</td>
<td>xsd:anyURI</td>
<td>Specifies that a predicate should be queried from a specific named graph.</td>
</tr>
<tr>
<td>foe:sparqlBinding</td>
<td>facet:Predicate</td>
<td>xsd:string</td>
<td>Specifies the variable that should be used for the predicate in a SPARQL query.</td>
</tr>
<tr>
<td>foe:codependent</td>
<td>facet:Facet</td>
<td>facet:Facet</td>
<td>Specifies that two facets share a predicate path.</td>
</tr>
<tr>
<td>foe:optional</td>
<td>facet:Predicate</td>
<td>xsd:boolean</td>
<td>Specifies that a predicate should be queried as optional.</td>
</tr>
<tr>
<td>FacetOntology Extensions Evaluation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>FacetOntology Performance</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>(without extensions)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>(with extensions)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Programs Query</strong></td>
<td>0.19529828</td>
<td>0.19026822</td>
<td>0.20863098</td>
</tr>
<tr>
<td><strong>Projects Query</strong></td>
<td>0.90006838</td>
<td>0.28167599</td>
<td>0.32459245</td>
</tr>
<tr>
<td><strong>People Query</strong></td>
<td>1.65916089</td>
<td>N/A(^1)</td>
<td>1.82002423</td>
</tr>
<tr>
<td><strong>Deployments Query</strong></td>
<td>3.11063311</td>
<td>2.83033155</td>
<td>3.03049117</td>
</tr>
<tr>
<td><strong>Instrument Categories Query</strong></td>
<td>0.18230436</td>
<td>N/A(^2)</td>
<td>1.47551705</td>
</tr>
<tr>
<td><strong>Instruments Query</strong></td>
<td>1.14335615</td>
<td>0.3970559</td>
<td>0.71716915</td>
</tr>
<tr>
<td><strong>Instruments Query Given Instrument Category</strong></td>
<td>0.17712639</td>
<td>N/A(^3)</td>
<td>0.37238441</td>
</tr>
<tr>
<td><strong>Parameters Query</strong></td>
<td>0.07652801</td>
<td>1.3722061</td>
<td>1.83360435</td>
</tr>
<tr>
<td><strong>Awards Query</strong></td>
<td>0.06315582</td>
<td>0.33312342</td>
<td>1.76174197</td>
</tr>
<tr>
<td><strong>Platforms Query</strong></td>
<td>0.43843655</td>
<td>N/A(^4)</td>
<td>0.36344442</td>
</tr>
</tbody>
</table>

\(^1\) Without proposed extensions for filtering results along predicate path, FacetOntology grabs a superset of the expected facet values for the BCO-DMO people facet.

\(^2\) Cannot capture hierarchy structure of the instrument categories facet without contextual predicate paths proposed for FacetOntology extensions and also the ability to notify the query engine of the transitive property for the instrument categories hierarchy.

\(^3\) Relationship between facet results (i.e., datasets) and instruments are one-to-many. As instrument categories are filtered, without declaration of co-dependent facets, the instrument facet has a superset of the expected facet results. This is made possible both through the co-dependent declaration capability and the ability to assign specific variables in the FacetOntology extensions.

\(^4\) The BCO-DMO use case requires that the facet label for platforms be constructed from the title for the type of platform (e.g., R/V for research vessel) followed by the platform name (e.g., R/V Atlantis). This is not possible without the ability to declare a custom SPARQL select header as provided by the extensions. The results without the platform title are provided.

**Other Notes:**

The BCO-DMO use cases specify that a link to information about facet values should be provided (see Facet Values Explained from Polowinski’s [Widgets for Faceted Browsing](#)). This is not possible for any of the facets without the proposed extensions.

The intent was to show that there was a significant performance difference in the queries. There are numerous variables that affect query performance, including system load and as well as various SPARQL implementation specific parameters. Thus, it suffices to say that, for the most part the queries are relatively close in performance, so there is no gain to using one implementation over the other except for applicability to specific use cases.
### Faceted Browsing Evaluation [20]

<table>
<thead>
<tr>
<th>Domain</th>
<th>Environment</th>
<th>Filtering</th>
<th>Display of Results</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2S</td>
<td>Independent</td>
<td>Web</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Arbitrary Restriction Order</td>
<td>Yes</td>
<td>No¹</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Result Cardinalities Shown</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multiple Selection</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Range Selection</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Standard Classification Integration</td>
<td>No¹</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String Search Integrated</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Facet Values Explained</td>
<td>Yes</td>
<td>No²</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Special Widgets</td>
<td>No²</td>
<td>No²</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sorting of Results Configurable</td>
<td>No³</td>
<td>No³</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Grouping of Results</td>
<td>No³</td>
<td>No³</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Search for Similar Items</td>
<td>No³</td>
<td>No³</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Display Paradigm</td>
<td>No³</td>
<td>No³</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Facet Set Context Configurable by User</td>
<td>Yes</td>
<td>No⁵</td>
</tr>
</tbody>
</table>

1. Implementing standard classification integration for data services with multiple result types is left for future work.

2. Sorting and grouping, although not currently implemented, can be implemented using a specialized result widget and a sorting/grouping parameter.

3. We seek use cases and algorithms for finding “similar” items in data management systems.

4. Various kinds of result widgets can be implemented; lists and tables are currently implemented, and mapped results and others are planned.

5. We have use cases for context-dependent facets, which will be implemented in future work.
S2S as an Application Integration Framework

- Faceted browsing was a useful paradigm for fleshing out the core concepts in S2S
- Additional applications for other paradigms, e.g., workflows, would be a useful extension
- Novelty of such an S2S workflow platform is the “human-in-the-loop”
- Another important extension would be for modeling scientific intent, to further automate and personalize the deployment of UIs
Contributions Summary

1. Lightweight ontology to describe UI components that can be linked to compatible Web service inputs and outputs and FacetOntology descriptions
2. Framework to deploy faceted data dashboards from RDF descriptions of UI components, Web services and instances of the FacetOntology
3. Extensions to the FacetOntology to better leverage the full capabilities of the S2S framework
4. Discussion of the trajectory of S2S towards a more general application integration framework
5. Synthetic approach to Semantic Web Methodology and Technology Development Process, incorporating cross-cutting use cases
Concluding Remarks

• What levels of automation are supported by S2S?
  – **Invocation:** Through abstract descriptions of JavaScript widgets (including source locations and encapsulating interface) enables invocation.
  – **Composition:** In specific paradigms, such as faceted browse, composition of UI components is made possible.
  – **Re-use:** Encapsulating widgets allows for re-use.
  – **Discovery:** Connecting abstract widget metadata with data and service metadata allows for run-time discovery.
References


5. http://flamenco.berkeley.edu/


Thank you!

• Any questions?
• Demos
  – BCO-DMO
  – IOGDS
Software Design – Overview

• JavaScript faceted browsing core
  – Uses chain-of-responsibility pattern
  – JavaScript and jQuery widgets

• Java Servlet
  – Proxy endpoint
    • Uses adapter pattern for extensibility
  – Metadata endpoint
    • Uses Jena and Ripple singletons to provide access to and caching of S2S RDF
    • Uses abstract factory pattern to support different sources of RDF (linked data, SPARQL, etc.)
Software Design – Adapter Pattern for Proxy Servlet

ProxyServlet
+doPost()

SearchService
+engine: WebServiceEngine

WebServiceEngine
+getInputs()
+getInterfacess()
+runQuery()

OpenSearchServiceEngine
+getInputs()
+getInterfacess()
+runQuery()

FacetOntologyServiceEngine
+getInputs()
+getInterfacess()
+runQuery()

FacetCollection
+buildQuery()

QueryableSource
+sparqlSelect()

... query = facetCollection.buildQuery(interface, inputs);
result = queryableSource.sparqlSelect(query);
...
Software Design – Chain-of-Responsibility Pattern for UI Panels

**Panel**
- `notify()`
- `update()`
- `setInputData(input, f)`
- `getInputData(input)`

**SearchService**
- `notify()`
- `setInputData(input, f)`
- `getInputData(input)`

**StaticInputsPanel**
- `successor: Panel`
- `notify()`
- `update()`
- `setInputData(input, f)`
- `getInputData(input)`

**ResultsPanel**
- `successor: Panel`
- `notify()`
- `update()`
- `setInputData(input, f)`
- `getInputData(input)`

**WidgetPanel**
- `successor: Panel`
- `notify()`
- `update()`
- `setInputData(input, f)`
- `getInputData(input)`