Query Syntax Evaluation

Sean Hardman

August 5, 2011

This document evaluates existing and proposed query syntaxes for use by the PDS Search Service.

# Requirements and Features

The Search Service design document contains the following requirements related to query syntax:

**L5.SCH.6** The service shall support searching by accepting criteria as a sequence of open text keywords.

**L5.SCH.7** The service shall support searching by accepting criteria as a series of values for constraints on specified indexes.

The above requirements refer to two common features of a query syntax:

**Term Search**

This feature is most commonly seen in a Google-like search. The user may provide a word or a phrase to be queried for against the contents of the underlying metadata store. The assumption here is that the query is performed against all fields or a defined default field.

**Fielded Search**

This feature allows the user to specify a specific field in which to query for the given term. The available fields are dependent on the underlying metadata store and must be known to the user prior to submitting the query.

Although the current requirements don’t elaborate on query syntax features, the following features are available in current PDS implementations:

**Wildcard Support**

This feature pertains to the term(s) in a query. Typically this includes single and multiple character substitution utilizing the “?” and “\*” symbols, respectively.

**Relational Operators**

All query syntaxes support “=” but the full set of relational operators includes: “!=”, “<”, “<=”, “>” and “>=”.

**Range Search**

This feature allows the user to specify a lower and upper bounds for a particular field in the query. This essentially involves the use of the “<” and “>” relational operators for what is considered an exclusive range search and the “<=” and “>=” relational operators for what is considered an inclusive range search.

**Logical Operators**

The logical operators include AND, OR and NOT. When not explicitly specified, the default operator may be AND or OR depending on the underlying implementation.

**Grouping**

This feature allows the user to group clauses together using parenthesis and the logical operators. A clause may be just a term, a field and a term, a range, etc.

# HTTP Protocol

The Search Service interface is a REST-based interface over the HTTP protocol. The HTTP protocol is pretty basic in nature with respect to passing parameters.

http://<host-name>/<path>[? <field>=<term>[ & <field>=<term>…]]

As shown above the “&” symbol separates multiple clauses and may be interpreted as AND or OR depending on the underlying implementation. With this basic protocol, many of the features detailed above are not inherently supported. Because of this, many query syntaxes lump their query expression into a single parameter (e.g., …?query=<query-expression>). Placing the query expression in a single parameter does not preclude the use of other parameters for dictating service behavior such as specifying the return type, maximum number of rows to return, starting index, etc.

# Query Syntax Options

This section compares existing and proposed query syntaxes by comparing their feature list against the features listed above and by providing a query example based on the following query criteria:

* Mission = Cassini-Huygens
* Instrument = RADAR
* Target = Titan
* Time Range = 2005-001 to 2005-365

## OODT

The OODT protocol [1][2] is currently used at the Engineering Node in its web service query interface to the catalog database. It is also utilized at the Nodes wherever Profile or Product Servers are deployed. The following table details the feature list:

|  |  |
| --- | --- |
| Term Search | Not inherently supported in the syntax but can be supported with a default field. |
| Fielded Search | Supported. |
| Wildcard Support | Not inherently supported in the syntax but can be achieved through the use of the LIKE and NOTLIKE relational operators. |
| Relational Operators | Supported. |
| Range Search | Not inherently supported in the syntax but can be achieved through the use of the relational and logical operators and grouping. |
| Logical Operators | Supported. |
| Grouping | Supported. |

The following is an un-encoded example of the query in OODT syntax:

<base-search-url>? q=msnname=CASSINI-HUYGENS AND instid=RADAR AND targname=TITAN AND strttime<=2005-365T23:59:59.999 AND stoptime>=2005-001T00:00:00.000 OR stoptime IS NULL

## Lucene/Solr

The Lucene/Solr protocol [3] is currently used at the Engineering Node for the main catalog query interface (<http://pds.nasa.gov/tools/data-search/>). The following table details the feature list:

|  |  |
| --- | --- |
| Term Search | Supported. |
| Fielded Search | Supported. |
| Wildcard Support | Supported. |
| Relational Operators | Supported. |
| Range Search | Supported. |
| Logical Operators | Supported. |
| Grouping | Supported. |

The following is an un-encoded example of the query in Lucene/Solr syntax:

<base-search-url>?q=mission:CASSINI-HUYGENS AND target\_name:TITAN AND instrument\_id:RADAR AND start\_time:[\*+TO+2005-12-31T23:59:59.999Z] AND stop\_time:[2005-01-01T00:00:00.000Z+TO+\*] OR -stop\_time:[\*+TO+\*]

## Planetary Data Access Protocol (PDAP)

The PDAP specification [4] is proposed by the International Planetary Data Alliance and is currently being used for certain search services developed by the European Space Agency. The following table details the feature list:

|  |  |
| --- | --- |
| Term Search | Not inherently supported in the syntax but can be supported with a default field. |
| Fielded Search | Supported. |
| Wildcard Support | Not supported. |
| Relational Operators | Supports all but “!=”. |
| Range Search | Supports inclusive range but not exclusive range. |
| Logical Operators | Not inherently supported in the syntax but some support does exist in the range feature. |
| Grouping | Not supported. |

The following is an un-encoded example of the query in PDAP syntax:

<base-search-url>?MISSION\_NAME=CASSINI-HUGYENS & INSTRUMENT\_NAME=RADAR & TARGET\_NAME=TITAN & START\_TIME=/2005-365T23:59:59.999 & STOP\_TIME=2005-001T00:00:00.000/

The main difference between this syntax and the others is that it uses distinct parameters with respect to the HTTP protocol.

## Feed Item Query Language (FIQL)

The FIQL protocol [5], although proposed as a standard, was never adopted as a standard. That said, it was incorporated into Apache’s CXF package, which is a web service framework. The following table details the feature list:

|  |  |
| --- | --- |
| Term Search | Not inherently supported in the syntax but can be supported with a default field. |
| Fielded Search | Supported. |
| Wildcard Support | Supported. |
| Relational Operators | Supported. |
| Range Search | Supported. |
| Logical Operators | Supported. |
| Grouping | Supported |

The following is an un-encoded example of the query in FIQL syntax:

<base-search-url>?q=mission==CASSINI-HUYGENS ; target\_name==TITAN ; instrument\_id==RADAR ; start\_time=le=2005-12-31T23:59:59.999Z ; stop\_time=ge=2005-01-01T00:00:00.000Z

# Conclusion

This document by no means contains an exhaustive look at query syntaxes but the examples above do offer a pretty good sampling of what is available in current implementations. One thing that is apparent in this sampling is that if advanced query features are desired they are most likely offered in a syntax where the query expression is contained in a single parameter with respect to the HTTP protocol. Although not required for all queries, the advanced features would be useful in satisfying some of the identified search scenarios. Additionally, there are existing services deployed in PDS that already support these advanced features, meaning that to select a syntax that does not support them, would be a step backwards.

Given the information contained in this document and development experience using Lucene and Solr, I propose that we base the query syntax for the Search Service on Lucene’s query syntax. As mentioned previously, this does not preclude specifying additional parameters for dictating specific service behavior outside of the query.

# References

[1] <http://oodt.apache.org/components/maven/xmlquery/tutorial/>

[2] <http://oodt.apache.org/components/maven/xmlquery/disquery/>

[3] <http://lucene.apache.org/java/2_9_1/queryparsersyntax.html>

[4] <http://planetarydata.org/projects/previous-projects/2009-2010-projects/pdap-specification/pdap-specification/at_download/file>

[5] <http://tools.ietf.org/html/draft-nottingham-atompub-fiql-00>