Search Web Services - searchRetrieve Operation: Binding for OpenSearch Version 1.0

Committee Draft 01

30 June 2008

Specification URIs:
This Version:
http://docs.oasis-open.org/search-ws/june08releases/opensearch-V1.0-cd-01.doc (Authoritative)
http://docs.oasis-open.org/search-ws/june08releases/opensearch-V1.0-cd-01.pdf
http://docs.oasis-open.org/search-ws/june08releases/opensearch-V1.0-cd-01.html

Latest Version:
http://docs.oasis-open.org/search-ws/v1.0/opensearch-V1.0.doc
http://docs.oasis-open.org/search-ws/v1.0/opensearch-V1.0.pdf
http://docs.oasis-open.org/search-ws/v1.0/opensearch-v1.0.html

Technical Committee:
OASIS Search Web Services TC

Chair(s):
Ray Denenberg <rden@loc.gov>
Matthew Dovey <m.dovey@jisc.ac.uk>

Editor(s):
Ray Denenberg rden@loc.gov
Larry Dixson ldix@loc.gov
Matthew Dovey m.dovey@jisc.ac.uk
Janifer Gatenby Janifer.Gatenby@oclc.org
Ralph LeVan levan@oclc.org
Ashley Sanders a.sanders@MANCHESTER.AC.UK
Rob Sanderson azaroth@liverpool.ac.uk

Related work:
This specification is related to:

- Search Retrieve via URL (SRU)

Abstract:
This is a binding of the Search Web Services - searchRetrieve operation – Abstract Protocol Definition. This binding is the specification of openSearch.

Status:
This document was last revised or approved by the OASIS Search Web Services TC on the above date. The level of approval is also listed above. Check the “Latest Version” or “Latest Approved Version” location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the

"Send A Comment" button on the Technical Committee's web page at http://www.oasis-open.org/committees/search-ws

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/search-ws/ipr.php.

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/search-ws/.

Notices

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The names "OASIS", [insert specific trademarked names and abbreviations here] are trademarks of OASIS, the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see http://www.oasis-open.org/who/trademark.php for above guidance.

Table of Contents

1 Introduction

This is a binding of the OASIS SWS (Search Web Services) searchRetrieve operation – ABSTRACT PROTOCOL DEFINITION.

This specification is intended to be fully compatible with http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_3

This binding is the specification of OpenSearch.

This binding is intended to be fully compatible with http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_3

This document defines the OpenSearch model, request parameters, response elements, and description document.

Search clients can use OpenSearch description documents to learn about the public interface of a search engine. These description documents contain parameterized URL templates that indicate how the search client should make search requests.

1.1 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “NOT RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in **[RFC2119]**. When these words are not capitalized in this document, they are meant in their natural language sense.

1.2 Normative References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

2 OpenSearch Binding Details

2.1 Model

2.1.1 Processing Model

A server provides a description document that a client reads to determine how to formulate a search/retrieve request and interpret the response. The client may send a request, including search terms, to the server, who replies with a response that includes results based on the search terms.

The server returns results either as a stream ("stream mode") or a page ("page mode"). A stream is an arbitrary range of results, for example, results 10 through 100. In page mode, the server groups the results into pages, and returns one page. The server will always return results as a stream or always as a page, and indicates one or the other in its description file.

If the server returns a page, the request may include the 'count' parameter, suggesting how many results there should be per page. The request may also include the 'startPage' parameter indicating which page is desired. (See note 1.) The server may ignore the 'count' parameter and determine the number of results per page itself. (See note 2.)

If the server returns a stream, the request may include the parameter 'startIndex' to indicate the desired position within the result set of the first result within the stream. For example if the value of the 'startIndex' parameter is 61, and if the server returns 30 results, the stream will consist of results 61 through 90. The request may also include the 'count' parameter (for example, a value of 30, if the client wants results 61 through 90) but the server may ignore it. (See note 3.)

The response includes the element <totalResults>, the number of results found by the search. This element will be omitted only if the last of the available results is included in the response.

So the client can scroll through the results by issuing repeated requests until there is a response which omits the <totalResults> element, the omission signaling that there are no further results. Each request uses the same value for the parameter 'searchTerms', and :

- In stream mode: the value of the parameter 'startIndex' is the previous value plus the number of results included in the previous response.
- In page mode: the value of the parameter 'startPage' is the previous value plus one (1).

Notes:

1. The server returns one page only, contrary to the implication of the parameter name, 'startPage'.
2. If the server has ignored the count parameter, then the startPage parameter that the client has suggested will not retrieve the specific results that the client had in mind.
3. The 'count' parameter is defined as "desired number of results per page", but it applies not only in page mode, but also in stream mode: In stream mode the entire list of results is considered a single page.

2.1.2 Result Set Model

There are no explicit (named) result sets in openSearch. It is assumed that if multiple requests are issued to a search engine with the same value of parameter 'searchTerms' the results will be identical, that is, the same set of results in the same order. Therefore the parameter 'searchTerms' can be considered to represent a result set.

2.1.3 Data Model

The data model of the Abstract Protocol Model says that a "*datastore* is a collection of units of data. Such a unit is referred to as an *item…*"

```
In this binding:
```

- A data store is referred to as a *search engine.*
- For an openSearch response, the abstract element <item> corresponds to an element defined by the response schema, for example an <entry> or <item> in ATOM 1.0 or RSS 2.0 respectively.
- An item is sometimes referred to as a “result”.

The Abstract Protocol Model further notes that “associated with a datastore are one or more formats that may be used for the transfer of items from the server to the client. Such a format is referred to as an *item type.*.”

In this binding:

- There is no parameter equivalent to i*temType; the format is internally defined by the response format.*

The Abstract Protocol Model further notes that “The server may also partition the result set into *result groups*.”

In this binding:

- ‘groups are referred to as ‘pages’.

2.1.4 Diagnostic Model

OpenSearch does not include specific diagnostics. HTTP diagnostics are returned when a URL is badly formed or the server is unable to perform the search contained within the URL.

If the server is able to interpret but not process a request it can send back the OpenSearch Description Document that explains how to correctly construct a request.

2.1.5 Description and Discovery Model

OpenSearch mandates an OpenSearch Description Document that is consistent with the requirements of the Abstract Protocol Definition. There are six groups of data that may be included:

1. **General Description of the Server and its Capabilities.** The OpenSearch Description Document includes a *shortName*, and *longName* and also *tags* which are keywords that describe the server’s *content* (*datastore*).
2. **How to Formulate a Request.** The OpenSearch Description Document includes a mandatory URL element containing a mandatory request template.
3. **Query Grammar.** There is no explicit search grammar associated with OpenSearch.
4. **How to Interpret a Response.** The type attribute of the URL element indicates the MIME type (*format*) of the response.
5. **How to Process Results.** The OpenSearch Description Document may include extra elements explaining how to process and display the search results. These include an *image* and *attribution* for display against the results, an indication of *adultContent* and *syndicationRight*.
6. **Auto-Discovery Process.** An OpenSearch description documents may include a reference to other OpenSearch description documents.

2.2 OpenSearch Request

The OpenSearch URL template represents a parameterized form of the URL by which a search engine is queried. The client processes the template, replacing each instance of a template parameter, with the value for that parameter. The template parameters are the request parameters shown below.

2.2.1.1 Actual Request Parameters For this Binding

Table 1: Summary of Actual Request Parameters

Parameter Name	Description	Type/Value

searchTerms	keyword or keywords	string
startIndex	index of first search result desired by the client	positive integer
count	Number of search results desired by the client.	positive integer
startPage	page number of the set of search results desired by the search client.	positive integer
language	desired language for search results.	RFC 3066, or '*' to mean "any language"
inputEncoding	character encoding of the search request.	IANA Character Set Assignments, default UTF-8
outputEncoding	character encoding requested for the search results. The default is UTF-8	IANA Character Set Assignments, default UTF-8

107 #### 2.2.1.2 Abstract Vs. Actual Parameters

108 The following table lists the Abstract parameters defined inthe Abstract Protocol Definition, and the
109 openSearch actual parameters, in two columns, with corresponding parameters in the same row.

110

111

112 *Table 2: Abstract Vs. Actual parameters*

Abstract Parameter Name from APD	**openSearch Parameter**
responseType	(None. See type attribute of <url> element)
query	searchTerms
startPosition	startIndex
maximumItems	count
group	startPage
responseItemType	(None. See Data Model, fourth bullet.)
sortOrder	(None)
(None)	language
(None)	inputEncoding
(None)	outputEncoding

113 ## 2.3 openSearchResponse

114 ### 2.3.1 Response Elements

115 This section summarizes the openSearch response elements and compares them with the abstract
116 elements defined in the Abstract Protocol Definition.

117 #### 2.3.1.1 Actual Response Elements

118 The following table describes the actual XML response elements.

119 *Table 3: Summary of Actual Response Elements*

Element	Type	Occurence	Meaning
<totalResults>	xs:integer	zero or one	number of search results.
<startIndex>	xs:positiveInteger	zero or one	index of the first search result in the response.
<itemsPerPage>	xs:positiveInteger	zero or one	number of search results returned per page.
<query>	xs:string	zero or more	See “Query”.

120

121 ### 2.3.1.2 Abstract Vs. Actual Elements

122 The following table lists abstract elements from the Abstract Protocol Definition, and the openSearch
123 actual elements, in two columns, with corresponding elements in the same row.

Table 4: Abstract Vs. Actual elements

Abstract Element From APD	openSearch Element
<numberOfItems>	<totalResults>
<resultSetId>	(none)
<item>	defined by the response schema, for example an <entry> in ATOM 1.0 or <item>RSS 2.0.
<nextPosition>	*In page mode*: find the <link> element where the value of the ‘rel’ attribute is “next”. Within the corresponding query (‘href’ attribute) the value of the parameter corresponding to startPage is the number of the next page. *In stream mode:* <startIndex> + <itemsPerPage> - 1.
<diagnostics>	(none)
<echoedSearchRetrieveRequest>	the value of the ‘href’ attribute for the <link> element where the value of the ‘rel’ attribute is “self”.
(none)	startIndex
(none)	itemsPerPage
(none)	Query

124 ## 2.3.2 OpenSearch Response Examples

125 **Example 1:** A page of search results in Atom 1.0

The line numbers on the left are added for reference in the analysis below.

```
<?xml version="1.0" encoding="UTF-8"?>
 <feed xmlns="http://www.w3.org/2005/Atom"
       xmlns:OpenSearch="http://a9.com/-/spec/OpenSearch/1.1/">
   <title>Example.com Search: New York history</title>
   <link href="http://example.com/New+York+history"/>
   <updated>2003-12-13T18:30:02Z</updated>
   <author>
     <name>Example.com, Inc.</name>
   </author>
   <id>urn:uuid:60a76c80-d399-11d9-b93C-0003939e0af6</id>
1. <OpenSearch:totalResults>4230000</OpenSearch:totalResults>
2. <OpenSearch:startIndex>21</OpenSearch:startIndex>
3. <OpenSearch:itemsPerPage>10</OpenSearch:itemsPerPage>
   <OpenSearch:Query
4.      role="request" searchTerms="New York History" startPage="1" />
   <link
     rel="alternate" href="http://example.com/New+York+History?pw=3"
     type="text/html"/>
   <link
5.   rel="self"
     href= "http://example.com/New+York+History?pw=3&amp;format=atom"
     type="application/atom+xml"/>
   <link
6.   rel="first"
     href="http://example.com/New+York+History?pw=1&amp;format=atom"
     type="application/atom+xml"/>
   <link
7.    rel="previous"
     href="http://example.com/New+York+History?pw=2&amp;format=atom"
     type="application/atom+xml"/>
8. <link
      rel="next"
      href="http://example.com/New+York+History?pw=4&amp;format=atom"
      type="application/atom+xml"/>
9. <link
     rel="last"
     href="http://example.com/New+York+History?pw=4229991&amp;format=atom"
     type="application/atom+xml"/>
   <link
      rel="search" type="application/OpenSearchdescription+xml"
      href="http://example.com/OpenSearchdescription.xml"/>
   <entry>
     <title>New York History</title>
     <link
       href="http://www.columbia.edu/cu/lweb/eguids/amerihist/nyc.html"/>
     <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
     <updated>2003-12-13T18:30:02Z</updated>
     <content type="text">
       ... Harlem.NYC - A virtual tour and information on
       businesses ...  with historic photos of Columbia's own New York
       neighborhood ... Internet Resources for the City's History. ...
     </content>
   </entry>
```

Analysis of the above example.

'pw' is the name of the parameter corresponding to the openSearch parameter 'startPage', for this server.

- Lines 1-3 indicate that there were 4,230,000 results associated with the search term "New York History". This response includes 10 results beginning with result 21 (thus results 21-30).

- Line 4 (<query role="request"…>) indicates how to regenerate the request from the beginning of the results (parameters searchTerms="New York History" and startPage="1")
- Line 5 indicates that the URL to generate the same request that generated this response (<link rel="self…>) with a response in Atom format (type="application/atom+xml"), is "http://example.com/New+York+History?pw=3&format=atom"
- line 6 (rel="first") indicates that the URL to get the first page of results, in atom, is href="http://example.com/New+York+History?pw=1&format=atom".
- line 7 (rel="previous") indicates that the URL to get the previous page of results is href="http://example.com/New+York+History?pw=2&format=atom".
- line 8 (rel="next") indicates that the URL to get the next page of results is href="http://example.com/New+York+History?pw=4&format=atom".
- line 9 (rel="last") indicates that the URL to get the last page of results is href="http://example.com/New+York+History?pw=4229991&format=atom".

Example 2: a page of search results in the RSS 2.0 format

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0"
     xmlns:OpenSearch="http://a9.com/-/spec/OpenSearch/1.1/"
     xmlns:atom="http://www.w3.org/2005/Atom">
  <channel>
    <title>Example.com Search: New York history</title>
    <link>http://example.com/New+York+history</link>
    <description>Search results for "New York history" at
Example.com</description>
    <OpenSearch:totalResults>4230000</OpenSearch:totalResults>
    <OpenSearch:startIndex>21</OpenSearch:startIndex>
    <OpenSearch:itemsPerPage>10</OpenSearch:itemsPerPage>
    <atom:link
      rel="search" type="application/OpenSearchdescription+xml"
      href="http://example.com/OpenSearchdescription.xml"/>
    <OpenSearch:Query
       role="request" searchTerms="New York History" startPage="1" />
    <item>
      <title>New York History</title>
      <link>http://www.columbia.edu/cu/lweb/eguids/amerihist/nyc.html</link>
      <description>
        ... Harlem.NYC - A virtual tour and information on
        businesses ...  with historic photos of Columbia's own New York
        neighborhood ... Internet Resources for the City's History. ...
      </description>
    </item>
  </channel>
</rss>
```

Example 3 a page of search results in the XHTML 1.0 format

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head profile="http://a9.com/-/spec/OpenSearch/1.1/" >
    <title>Example.com Search: New York history</title>
```

```
    <link rel="search"
          type="application/OpenSearchdescription+xml"
          href="http://example.com/OpenSearchdescription.xml"
          title="Example.com Web Search" />
    <meta name="totalResults" content="4230000"/>
    <meta name="startIndex" content"1"/>
    <meta name="itemsPerPage" content="10"/>
  </head>
  <body>
     <ul>
       <li>
         <a href="http://www.columbia.edu/cu/lweb/eguids/amerihist/nyc.html">
           New York History
         </a>
         <div>
           ... Harlem.NYC - A virtual tour and information on
           businesses ...  with historic photos of Columbia's own New York
           neighborhood ... Internet Resources for the City's History. ...
         </div>
       </li>
       <!-- ... -->
     </ul>
   </body>
</html>
```

3 Open Search Description Document

A server providing an OpenSearch interface provides a description document to describe the interface.

OpenSearch description documents have the following mime type (pending IANA registration):

application/OpenSearchdescription+xml

OpenSearch description elements (table below) have the following XML Namespaces URI

http://a9.com/-/spec/OpenSearch/1.1/

3.1 Description Elements

Table 5: Description Elements

Element	Occurence	Description/ Restrictions
OpenSearchDescription	Must occur exactly once (as the root node of the document).	
ShortName	Must occur exactly once.	16 or fewer characters of plain text (no HTML or other markup).
Description	Must occur exactly once.	1024 or fewer characters of plain text (no HTML or other markup).
Url	Must occur exactly once.	See URL Element.
Contact	May occur zero or one time.	Email address for owner of the description document
Tags	May occur zero or one time.	keywords describing search content. One or more single words delimited by spaces. Total 1024 or fewer characters of plain text (no HTML or other markup).
LongName	May occur zero or one time.	An extended human-readable title that identifies this search engine. 48 or fewer characters of plain text (no HTML or other markup).
Image	May occur zero or more times.	URL for an image that can be used in association with this search content. Attributes: height, width, type (MIME); all optional
Query	May occur zero or one time.	See Query Element.

Element	Occurence	Description/ Restrictions
Developer	May occur zero or one time.	human-readable name or identifier for creator or maintainer of the description document. 64 or fewer characters of plain text (no HTML or other markup).
Attribution		a list of all entities to be credited for the content in the search feed. 256 or fewer characters of plain text (no HTML or other markup).
SyndicationRight		the degree to which search results provided by this search engine can be queried, displayed, and redistributed See table below.
AdultContent	May occur zero or one time.	boolean: true if the search results may contain material intended only for adults. "false", "FALSE", "0", "no", and "NO" will be considered boolean FALSE; all other strings will be considered boolean TRUE. Default: "false"
Language	May occur zero or more times.	one "Language" element for each language that the search engine supports. Values from RFC 3066. A value of "*" (default) signifies that the search engine does not restrict search results to any particular language.
InputEncoding	May occur zero or more times. (One for each character encoding that can be used to encode search requests.)	as specified by the IANA Character Set Assignments. Default: "UTF-8".

268

269

270 **Values for Parameter SyndicationRight**

right → **value** **|** **v**	**The search client may request search results**	**may display the search results to end users**	**client may send the search results to other search clients**
"open",	yes	yes	yes
"limited"	yes	yes	no
"private"	yes	no	no
"closed"	no	no	no

3.1.1 URL Element

The Url element has the form as shown in the following example:

```
<Url
     type= "application/xhtml+xml"
     indexOffset="0"
     template=
      "http://example.com/search?q={searchTerms}&amp;start={startIndex?}"/>
```

3.1.1.1 Attributes of the URL Element

indexOffset, pageOffset. The starting number for the first search result or first page of search results, for index-based and page-based results respectively. Defaults are "1"; the "indexOffset" and "pageOffset" attributes may be used to inform search clients of different starting values.

type. The MIME type of the search result format. The 'type' attribute of the <url> element is what the client uses to determine how to request a specific response format. There may be several <url> elements, each with a type attribute of a different value. The one with the desired value (mime type) is the one belonging to the template to use for that response format.

3.1.1.2 Template Syntax

The OpenSearch URL template represents a parameterized form of the URL by which a search engine is queried. The search client will process the URL template and attempt to replace each instance of a template parameter, generally represented in the form {name}, with a value determined at query time.

All parameter names are associated with a namespace; the OpenSearch 1.1 namespace is the default if no other is indicated. Parameter names are case sensitive.

A template parameter is designated as optional by using the "?" as shown in the two examples below.

The template parameters are the openSearch request parameters in table 1.

Examples

Example 1: a search URL template that contains a template parameter:

```
http://example.com/search?q={searchTerms}
```

In this example, the openSearch parameter 'searchTerms', in curly brackets, is an abstract parameter to be replaced by the actual parameter for this search engine, in this case 'q'. {searchTerms}" is required as indicated by the absence of "?"

Example 2: optional template parameter:

```
http://example.com/feed/{startPage?}
```

This example, the question mark, "?", is used to mean that the parameter startPage is optional.

3.1.2 Query Element

The Query element may appear in a description document or search response and is used to supply search requests that can be performed by a search client.

The Query element attributes correspond to the search parameters in a URL template. The core search parameters are explicitly defined as Query attributes, and custom parameters can be added via namespaces as needed.

At least one Query element with role="example" should be provided in each description document so that search clients can test the search engine. In addition a Query element with role="request" in each search response so that search clients can recreate the current search.

3.1.2.1 Attributes of the Query Element

The query element may contain the following attributes defined in the OpenSearch namespace, as well as attributes from external namespace.

- **role.** Required. Values:
  - "request" : the search query can be performed to retrieve the same set of search results.
  - "example"
  - "related" :thequery can be performed to retrieve similar but different search results.
  - "correction" : corrected query (e.g. a spelling correction) which can be performed to improve results set,
  - "subset": a query that will narrow the current set of search results.
  - "superset": a query that will broaden the current set of search results.
- **title. P**lain text string describing the search request. 256 or fewer characters. optional.
- **totalResults.** Expected number of results to be found if the search request were made. Optional.
- **searchTerms, count, startIndex, startPage, language, inputEncoding**, **outputEncoding**. The value representing these parameters. All are optional.

3.1.2.2 Query Element Examples

Example 1: Query element in a description document to provide an example search request

```
<Query role="example" searchTerms="cat" />
```

Example 2: Query element in a response to echo back the original search request

```
<Query role="request" searchTerms="cat" startPage="1" />
```

Example 3: Query element in a response to correct the spelling of "OpenSurch":

```
<Query role="correction" searchTerms="OpenSearch" totalResults="854000"
title="Spelling correction"/>
```

Example 4: An extended parameter

```
<Query xmlns:custom="http://example.com/OpenSearchextensions/1.0/"
       role="example"
       searchTerms="cat"
       custom:color="blue"
       title="Sample search" />
```

Example 5: an extended role

```
<Query xmlns:custom="http://example.com/OpenSearchextensions/1.0/"
       role="custom:synonym"
       title="Synonym of 'cat'"
       searchTerms="feline" />
```

Example 6: a set of Query elements used in the context of an Atom-based OpenSearch response

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:OpenSearch="http://a9.com/-/spec/OpenSearch/1.1/">
  <!--- ... --->
  <OpenSearch:Query
role="request" searchTerms="General Motors annual report" />

  <OpenSearch:Query
role="related" searchTerms="GM" title="General Motors stock symbol" />

  <OpenSearch:Query
role="related" searchTerms="automotive industry revenue" />

  <OpenSearch:Query
role="subset" searchTerms="General Motors annual report 2005"

  <OpenSearch:Query role="superset" searchTerms="General Motors" />
     ……
</feed>
```

3.2 Example Description Documents

Example 1: a simple OpenSearch description document

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/OpenSearch/1.1/">
       <ShortName>Web Search</ShortName>
       <Description>Use Example.com to search the Web.</Description>
       <Tags>example web</Tags>
       <Contact>admin@example.com</Contact>
       <Url type="application/rss+xml"
                template=
"http://example.com/?q={searchTerms}&amp;pw={startPage?}&amp;format=rss"/>
</OpenSearchDescription>
```

Example 2: a detailed OpenSearch description document

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/OpenSearch/1.1/">
        <ShortName>Web Search</ShortName>
        <Description>Use Example.com to search the Web.</Description>
        <Tags>example web</Tags>
        <Contact>admin@example.com</Contact>
        <Url type="application/atom+xml"
              template=
"http://example.com/?q={searchTerms}&amp;pw={startPage?}&amp;format=atom"/>
        <Url type="application/rss+xml"
              template=
"http://example.com/?q={searchTerms}&amp;pw={startPage?}&amp;format=rss"/>
        <Url type="text/html"

template="http://example.com/?q={searchTerms}&amp;pw={startPage?}"/>
       <LongName>Example.com Web Search</LongName>
        <Image height="64" width="64"
type="image/png">http://example.com/websearch.png</Image>
        <Image height="16" width="16"
type="image/vnd.microsoft.icon">http://example.com/websearch.ico</Image>
        <Query role="example" searchTerms="cat" />
        <Developer>Example.com Development Team</Developer>
        <Attribution>
   Search data Copyright 2005, Example.com, Inc., All Rights Reserved
        </Attribution>
        <SyndicationRight>open</SyndicationRight>
```

```
        <AdultContent>false</AdultContent>
        <Language>en-us</Language>
       <OutputEncoding>UTF-8</OutputEncoding>
        <InputEncoding>UTF-8</InputEncoding>
</OpenSearchDescription>
```

3.3 Extensibility

OpenSearch description documents can be extended provided that all foreign elements and attributes are associated with an explicit XML namespace. Clients that encounter unrecognized foreign elements should ignore them and continue to process the document as if these elements did not appear.

3.4 Autodiscovery

An OpenSearch description documents may include a reference to other OpenSearch description documents by including "link" elements on search results, with the following attributes/values:

- type ="application/OpenSearchdescription+xml".
- rel="search".
- href= [URI of an OpenSearch description document].
- title= [human-readable plain text string describing the search engine].

And in addition, for HTML and XHTML documents:

- The HTML <head/> element should include the attribute/value pair: profile="http://a9.com/-/spec/OpenSearch/1.1/".

Autodiscovery Examples

Example 1: Atom-based search results with an OpenSearch autodiscovery link element

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:OpenSearch="http://a9.com/-/spec/OpenSearch/1.1/">
……..
  <link rel="search"
        href="http://example.com/OpenSearchdescription.xml"
        type="application/OpenSearchdescription+xml"
        title="Content Search" />
  ……..
</feed>
```

Example 2: RSS-based search results with an OpenSearch autodiscovery link element

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0"
     xmlns:atom="http://www.w3.org/2005/Atom">
  <channel>
        …….
    <atom:link rel="search"
               href="http://example.com/OpenSearchdescription.xml"
               type="application/OpenSearchdescription+xml"
               title="Content Search" />
                        ……..
  </channel>
</rss>
```

Example 3: An HTML document that includes OpenSearch autodiscovery link elements

```
 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
 <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">
   <head profile="http://a9.com/-/spec/OpenSearch/1.1/">
     <!--- ... --->
     <link rel="search"
           type="application/OpenSearchdescription+xml"
           href="http://example.com/content-search.xml"
           title="Content search" />
     <link rel="search"
           type="application/OpenSearchdescription+xml"
           href="http://example.com/comment-search.xml"
           title="Comments search" />
     <!--- ... --->
   </head>
   <body>
     <!--- ... --->
   </body>
 </html>
```

A. Acknowledgements

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Participants:

Kerry Blinco, Australian Department of Education, Employment and Workplace Relations

Ray Denenberg, Library of Congress
Larry Dixson, Library of Congress
Matthew Dovey, JISC
Janifer Gatenby, OCLC/PICS
Ralph LeVan, OCLC
Farrukh Najmi, Wellfleet Software Corporation
Ashley Sanders, University of Manchester
Rob Sanderson, University of Liverpool