Intelligence Community and Department of Defense
Content Discovery & Retrieval Integrated Project Team

IC/DoD REST Interface Encoding Specification for CDR Search

V2_20111214

14 December 2011

REVISION/HISTORY

Doc Revision	Revised By	Revision Date	Revisions
0.1		29 January 2010	Initial draft for subgroup review.
0.2	Dave Lemen, Dan Wigglesworth	1 February 2010	Second draft, limiting specification to format-agnostic OpenSearch implementation.
0.3	Dave Lemen	16 February 2010	
0.4	Dave Lemen	23 February 2010	
0.5	Dave Lemen	9 March 2010	
1.0	Mary Farley	14 April 2010	Technical Edits
1.1	Dave Lemen	27 May 2011	Updates based on specification framework changes
1.2	Dave Lemen	24 June 2011	Changes based on 23 June adjudicated comments
1.3	Dave Lemen	10 Aug 2011	Changes based on comments to v1.2
1.4	Dave Lemen	20 Oct 2011	Changes based on DMO comments on v1.3
1.4-20111128	Matt McKnight	28 Nov 2011	Changes based on SCA comments to v1.3
1.4-20111214	Wigglesworth	14 Dec 2011	Modified for ASEC Comments

TABLE OF CONTENTS

1 Introduction 5
1.1 Service Overview 5
1.2 Relationship to Other CDR Architecture Elements 5
1.3 Notational Convention 6
1.4 Conformance 7
1.5 Namespaces 7
1.6 License 7
1.7 Security 7
2 Search Service Behavior 8
2.1 Main Flow 8
2.2 Query 8
2.3 Paging 9
2.4 Result Presentation 10
2.5 Relevance of Search Results 10
3 Search Service Interface 12
3.1 Search Function 12
3.1.1 Preconditions 12
3.1.2 Input 12
3.1.3 Output 14
3.1.4 Post-conditions 15
3.2 Results Paging Function 15
3.2.1 Preconditions 15
3.2.2 Input 16
3.2.3 Output 16
3.2.4 Post-conditions 18
3.3 Fault Conditions 19
4 Search Service Implementation 19
4.1 Result Types 19
4.2 Sorting of Search Results 19
4.3 Security Considerations 19
5 Reference Documents 20

LIST OF FIGURES

Figure 1. CDR Architecture Documents 6
Figure 2. Search Sequence 8
Figure 3. Result Set Structure 14

LIST OF TABLES

Table 1. Referenced XML Namespaces 7
Table 2. Framework Input Variable Disposition 13
Table 3 – Framework Output Variable Disposition 14
Table 4 – OpenSearch Output Extensions 17
Table 5 – Fault Conditions and HTTP Responses 19

1 Introduction

1.1 Service Overview

The Search Component, as defined by the Intelligence Community/Department of Defense (IC/DoD) Content Discovery and Retrieval (CDR) Specification Framework [CDR-SF], serves as the primary content discovery mechanism to expose content collections for discovery and accessibility. This component provides a common service interface and behavioral model for IC and DoD content collections, enabling content consumers to discover relevant content resources from disparate collections across the IC/DoD enterprise.

This specification defines requirements and provides guidance for the realization of the CDR Search Component as a RESTful[1], OpenSearch [OS] web service, hereafter termed a Search service in this document. It describes a Search service's behavior, interface and other aspects in detail, providing enough information for Search service providers and implementers to create CDR-compliant Search services.

The Search service exposes a single Search operation that is responsible for three activities that underpin Content Discovery capabilities: Search, Results Presentation, and Results Paging. As discussed in the CDR Specification Framework, a Search service's results are resource metadata rather than actual content resources. In the context of Search, resource metadata generally refers to a subset of a resource's available metadata, not the entire underlying record[2]. Results normally contain information needed for a consumer to estimate relevance, retrieve or otherwise use the referenced resource.

1.2 Relationship to Other CDR Architecture Elements

The CDR Architecture prescribes an abstract-to-concrete model for the development of architecture elements and guidance for content discovery and retrieval. Each layer or tier of the model is intended to provide key aspects of the overall guidance to achieve the goals and objectives for joint DoD/IC content discovery and retrieval. The following graphic, discussed in detail within the CDR Reference Architecture [CDR-RA], illustrates this model.

[1] *REST is a style of software architecture that **is** built around the transfer of resource representations and is the predominant architectural style of the World Wide Web.*

[2] *The Search Component returns metadata about a resource, which may sometimes describe the underlying resource (e.g., an image), while at other times representing a sub-set of the data the makes up a resource (e.g., a collection of attributes). In some cases, the metadata returned from an instantiation of the Search function and the Retrieve function, which returns a resource itself, may happen to be the same, though this is considered an edge condition.*

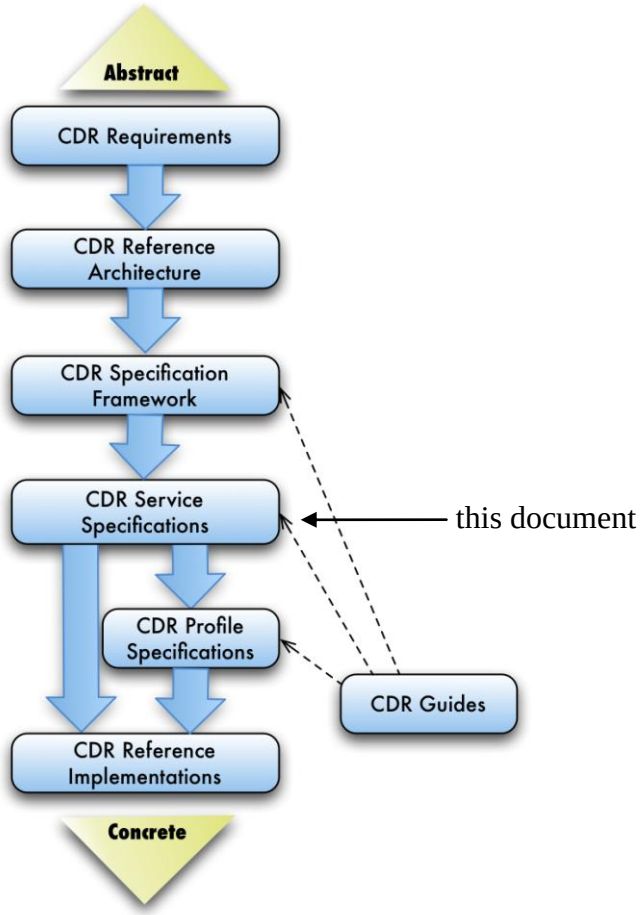

Figure 1. CDR Architecture Documents

As illustrated in Figure 1, the Specification Framework derives from the Reference Architecture (RA) and can describe behavior in terms of the capabilities, components, and usage patterns defined in the RA. The Specification Framework defines a common interface and corresponding behavior from which multiple Service Specifications may be derived for any CDR Component.

This specification provides guidance for implementing the CDR Search Component using the RESTful OpenSearch [OS] standard. It is intended to provide minimal requirements for implementing OpenSearch. Additional sub-specifications will provide further guidance for implementation profiles that include specific query types and result formats.

1.3 Notational Convention

The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" in this specification are to be interpreted as described in the IETF RFC 2119. When these words are not capitalized, they are meant in their natural-language sense.

When describing concrete XML schemas and example XML documents, this specification uses XPath as the notational convention. Each member of an XML schema is described using an XPath notation (e.g., /x:RootElement/x:ChildElement/@Attribute). Examples in this text are distinguished by a black border. These are meant to be illustrative and only one way that the described syntax can be used.

```
<atom:entry>
<atom:title>This is an example.</atom:title>
</atom:entry>
```

1.4 Conformance

Search services must support OpenSearch 1.1 Draft 4 [OS].

This specification defines an interface to a Search service to which an implementation and a subsequent deployment MUST conform. A deployment is an instance of an implementation. For an implementation to conform to this Search specification, it MUST adhere to all mandatory aspects of the specification.

1.5 Namespaces

Namespaces referenced in this document and the prefixes used to represent them are listed in the following table.

Table 1. Referenced XML Namespaces

Prefix	URI	Description
opensearch	http://a9.com/-/spec/opensearch/1.1/	OpenSearch 1.1 (Draft 4)[3]
atom	http://www.w3.org/2005/Atom	Atom 1.0

1.6 License

This specification is licensed under the Creative Commons Attribution-ShareAlike 2.5 Generic License (http://creativecommons.org/licenses/by-sa/2.5/), because it builds on the OpenSearch [OS] standard, which is licensed with the share-alike clause.

1.7 Security

This specification does not directly address security concerns. It will be necessary for any implementation of this specification to address security concerns relevant to the systems with which they interact and the governance bodies. Several aspects of search, to include returning only the results for which the requesting entity is authorized, should be addressed in the detailed security plan of an implementation, but are out of scope for this document. The Web Services Security Working Group is addressing these concerns.

3 *The OpenSearch specification can be found at http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_4.*

2 Search Service Behavior

2.1 Main Flow

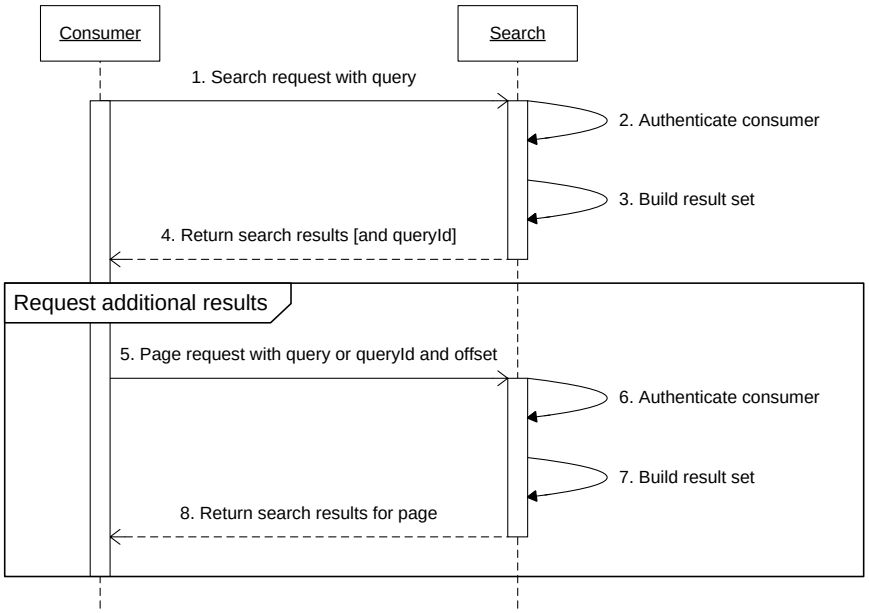

Figure 2. Search Sequence

1. A consumer sends a search request[4], which includes a query and any other information required.
2. The search component authenticates the consumer. The search component may at this point also authorize the consumer for access to search results.
3. The search component executes the query and builds a result set. If authorization for individual results is required, the search component checks authorization at this point.
4. The search component returns the initial result set.
5. The consumer requests an additional result set by sending the query and paging parameters.
6. The search component authenticates the consumer.
7. The search component builds the result set.
8. The search component returns the result set.

2.2 Query

The OpenSearch specification does not define a syntax for its primary query parameter, searchTerms, but it is generally used to support simple keyword queries. There are a

[4] *Terms such as search request, search results, result, and result set are defined in section 2.3 of the Specification Framework [CDR-SF] and used consistently with those definitions throughout this document.*

number of OpenSearch extensions that support query types beyond simple keyword queries as well. These include the OpenSearch Geo Extension [OS-GEO] for geospatial query terms, and the OpenSearch Time Extension [OS-TIME] for temporal query terms. Use of these extensions is discussed in more detail in section 3.1.2.1. Additional query types may be developed by the CDR IPT in the future.

2.3 Paging

Paginating the return of search results can be useful when the number of results is very large or indeterminate. Service consumers can page through the search results, accessing a subset of the overall search results as desired. This capability will prevent search requests that generate a very large number of results from overloading the server, network, or client.

Search results can be traversed using the information from the original Search service request combined with the endpoint information provided by the OpenSearch Description Document (OSDD) describing the Search service from which the current result set was generated. The Search service OSDD allows a service consumer to issue a search request for the next "page" of data.

CDR implementations SHOULD support paging.

Description of significant parameters which can be included in the URL template within an OSDD:

opensearch:count
The desired number of results to return per page.

opensearch:startIndex*
The starting index number of the returned results. The default value is 1.

opensearch:startPage*
The starting page number of the returned results. The default value is 1.

* *A Search service SHOULD NOT use both startIndex and startPage, since their functions overlap.*

Example:

```
http://example.com/test?q={searchTerms}
&startIndex={startIndex?}&format=atom
```

Description of significant elements that a Search service SHOULD return in the response:

/opensearch:totalResults
The total number of results available for the query

/opensearch:startIndex
The start index of this result set

/opensearch:itemsPerPage
The number of search results returned per page.

```
<atom:feed>
  ...
  <opensearch:totalResults>492420</opensearch:totalResults>
  <opensearch:startIndex>1</opensearch:startIndex>
  <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
  ...
</atom:feed>
```

The paging mechanism supported by the Search service does not guarantee continuity of search results while switching pages. Content resources may be added, updated, or removed in the period of time between which the different pages of the result set are accessed -- without the consumer being aware of these changes. Therefore, service consumers SHOULD NOT present paged result sets as coherent or complete or make assumptions to that effect.

2.4 Result Presentation

The CDR Specification Framework identifies additional result presentation behaviors:

- Result Sorting Order
- Result Metadata Style

Support for sorting functionality is OPTIONAL, however Search services SHOULD provide results sorted by relevance by default, if possible. Implementations MAY add sorting parameters or allow sort order to be expressed in the query.

Each supported response content type SHOULD have a separate URL template in the OSDD (see section 3.1.2).

2.5 Relevance of Search Results

The OpenSearch Relevance Extension [OS-RELV] provides a mechanism for communicating the relative importance of each result. Result relevance is generally a measure of how well a specific result matched the original query. Providing a result relevance measure allows better matched results to be prioritized relative to other results. A Search service implementation MAY provide relevance scores for individual search results with respect to the particular search with which it is identified. The OpenSearch Relevance extension defines a single element:

/relevance:score

The range of values allowed is any decimal between 0 to 1, inclusive, with 1 being the most relevant and 0 the least.

Usage:

```
<relevance:score>0.97</ relevance:score>
```

This scheme does not define the mechanism by which the relevance score is determined. In addition, comparing scores calculated under this scheme by different Search service instances may not provide a true comparison of relevancy.

An example Atom feed that includes opensearch:Relevance elements in its individual results follows:

```
<atom:feed>
  ...
  <atom:entry>
    result metadata
    ...
    <relevance:score>0.97</relevance:score>
  </atom:entry>
  <atom:entry>
    result metadata
    ...
    <relevance:score >0.42</ relevance:score>
  </atom:entry>
  ...
</atom:feed>
```

3 Search Service Interface

3.1 Search Function

3.1.1 Preconditions

The following preconditions MUST be satisfied if the search function is to correctly process input and generate results and post-conditions as described.

1. The requester is authenticated according to applicable policy requirements for auditing search activity and authorizing access.
2. The authenticated requester is authorized to access the search service.

3.1.2 Input

The URL template within the OSDD defines the syntax for how to call the Search service. The URL element requires two attributes to be present:

/@type
Contains the MIME type of the query result format

/@template
Contains the query URL to be processed according to the syntax rules

The template syntax rules define how values can be placed into the URL to create a valid query. For example:

```
<Url type="application/atom+xml"
template="http://example.com/?q={searchTerms}&amp;amp;pw={startPage}"/>
```

The type parameter determines the return format of the query. Guidance on return formats is discussed in Section 4.1.

Both {searchTerms} and {startPage} are variables that are to be replaced by a user or application using the URL. Variables without a question mark are required to be replaced with an actual value. Variables with question marks are required to be replaced by an empty string or by an actual value. An empty string indicates that the parameter should not be used for the query.

For further guidance regarding the URL element and the URL template syntax, refer to the OpenSearch specification.

3.1.2.1 OpenSearch Query Type Extensions

The OpenSearch specification defines one parameter, searchTerms, for the query. OpenSearch does not specify a syntax for the contents of the searchTerms parameter.

Extensions to OpenSearch have been developed to support query types that go beyond keyword-style queries. These include:

- OpenSearch Geo Extension [OS-GEO] – Defines parameters for supporting bounding box, point-radius, polygon, and other geospatial query types.

- OpenSearch Time Extension [OS-TIME] – Defines parameters for specifying a time range. This specification does not describe how to interpret the temporal query terms. For example, it does not specify that the temporal terms apply to the date a resource was created or posted, or that it may apply to a historical event described by the resource. Guidance related to interpretation can be found in implementation guidance, such as the DoD Discovery Metadata Specification Implementation Guide [DDMS].

3.1.2.2 Relation to Inputs Defined in the Specification Framework

The IC/DoD CDR Specification Framework defines a number of required (R) and optional (O) inputs to the Search operation. The following table relates the disposition of each variable defined in the Framework as it relates to this specification:

Table 2. Framework Input Variable Disposition

Activity	*Framework Input Variable*	*Search Specification*
Search and Results Paging	Query (R)	The query is an aggregate of the parameters in the request (R).
	Query Properties (O)	Query type for OpenSearch requests is assumed to be a general keyword search. Additional parameters may be supported using OpenSearch extensions, as indicated in the URL template parameters in an OSDD.
	Timeout (O)	*Not supported*
	Results per page (O)	opensearch:count (O)
	Start index (O)	opensearch:startIndex or opensearch:startPage (O)
	Query identifier (O)	*Not supported – OpenSearch services typically use opensearch:searchTerms and other parameters defined in the URL template to identify a result set for paging purposes.*
	Result metadata format (O)	Implicit – each Search service binding MUST be associated with a specific result metadata format. Therefore this input variable is not needed.
	Result sorting order (O)	Default sorting by relevancy is RECOMMENDED. Individual query types MAY define input variables or query syntax to control custom sorting; otherwise sorting order input is not supported and no variable is defined.

The CDR Specification Framework allows for *Query Metadata* as an optional Search input. This Search Service Specification does not enable input of metadata independent of a particular type of query. Individual query types MAY define metadata inputs and

other kinds of inputs, as desired, as part of their search syntax. In the RESTful implementation specified by this document, this information is implicit in the type of parameter(s) sent in the Search Service request, therefore a separate query metadata input is not necessary.

3.1.3 Output

In addition to the requirements imposed by the CDR Specification Framework, the Search service Search function output is additionally constrained by the requirements specified in the OpenSearch Specification.

The following example illustrates the high level components of a response message (containing a result set of unspecified type) from a Search service:

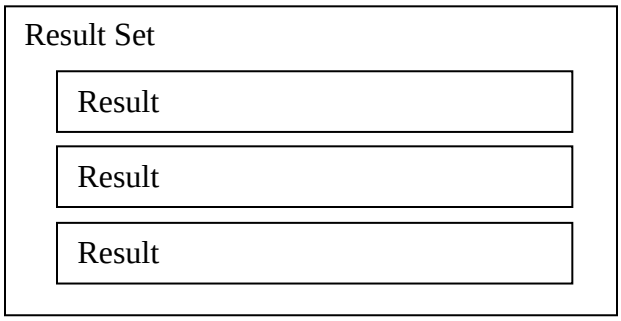

Figure 3. Result Set Structure

3.1.3.1 Relation to Outputs Defined in the Specification Framework

The IC/DoD CDR Specification Framework [CDR-SF] defines a number of required (R) and optional (O) outputs from the Search operation. The following table relates the disposition of each variable defined in the Framework in this specification:

Table 3 – Framework Output Variable Disposition

Activity	*Framework Output Variable*	*Search Specification*
Search and Results Paging	Result set (R)	A ***Search*** service MUST return a formatted set of results. Search service implementations SHOULD support HTML and Atom response formats.
	Results metadata (R)	CDR Result Type specifications MAY require certain types of data to be returned as part of the result set or individual result entries. Those specifications MAY also allow other types of metadata to be included and describe the mechanism for doing so. A Search service that supports a particular result type MUST follow the syntax and processing rules defined by that type. (R)
	Result relevancy value (O)	relevance:score in each result
	Result retrieval properties (O)	An element describing the linkage to the ***Retrieve*** service SHOULD be included in the results.

	Timestamp (O)	CDR Result Type may require timestamp element (ie feed/updated).
	Query identifier (O)	*Not Supported – OpenSearch services typically use opensearch:searchTerms and other parameters defined in the URL template to identify a result set for paging purposes.*
	Response result count (O)	opensearch:itemsPerPage SHOULD be included in the response.
	Total result count (O)	opensearch:totalResults SHOULD be included in the response.

3.1.3.2 Including the Search Request in the Response

To facilitate paging capabilities and to provide service consumers the ability to re-execute their queries, the search request that produced the output MAY be included in the results response. The mechanism for doing this will depend on the response format being returned. In the following example, the Search Request is inserted directly under the root of a notional response:

```
<atom:feed>
  ...
   <atom:link rel="self" href="http://example.com/test?q=tanks&amp;page=1&amp;format=atom"
type="application/atom+xml"/>
  ...
</atom:feed>
```

3.1.3.3 Including Metadata in the Results

Depending on the underlying data resources and the type of search request being executed, Search services MAY return metadata about each resource beyond that required by the Result Type specification. That specification controls the mechanism and syntax for including any additional metadata and whether or not such inclusion is permitted. Search services that support a particular Result Type in its response MUST follow the requirements in the associated Result Type specification.

3.1.4 Post-conditions

The following conditions MUST be met upon completion of a search.

1. The results returned to the requester are relevant to the input query.
2. The results are in the correct format.
3. The authenticated requester has been authorized to receive each result in the response.
4. The search function has been audited according to applicable policy.[5]

3.2 Results Paging Function

3.2.1 Preconditions

The following preconditions MUST be satisfied if the search function is to correctly process input and generate results and post-conditions as described.

[5] *The search function may be audited according to applicable policy regardless to the success or failure of the function.*

1. The requester is authenticated according to applicable policy requirements for auditing search activity and authorizing access.
2. The authenticated requester is authorized to access the search service.
3. Sufficient information from the original request is available to support building the result set for the desired page (e.g., the query and other search properties).

3.2.2 Input

The Search service specification is REQUIRED to function as described by the Specification Framework with any input, behavior, output, and fault condition extensions listed below.

opensearch:count
The desired number of results to return per page.

opensearch:startIndex*
The starting index number of the returned results. The default value is 1.

opensearch:startPage*
The starting page number of the returned results. The default value is 1.

** A Search service SHOULD NOT use both startIndex and startPage, since their functions overlap.*

3.2.3 Output

Search service implementations SHOULD use the elements and parameters from the OpenSearch specification for paging purposes. The exact mechanism and syntax for including the OpenSearch extensions are defined by the specification for the Result Type being used. The requirements for the inclusion of OpenSearch extensions are listed in the table below.

331 **Table 4 – OpenSearch Output Extensions**

Element	Description
`opensearch:totalResults`	The actual or estimated number of resources that match the current query. In the absence of a totalResults element or a next link (//link[@rel='next']), the search client should consider the current page to be the last page of search results. • Restrictions: The value must be a non-negative integer. • Default: The default value is equal to the offset index of the last search result on the current page. • Requirements: The element may appear zero or one time. **Recommendation:** The Search Service implementation SHOULD include the opensearch:totalResults in the response. **Example:** <opensearch:totalResults>42</opensearch:totalResults>
`opensearch:startIndex`	The offset of the first search result in the current set of search results. The OpenSearch specification allows for any integer offset for the first search result (0, 1, -5, etc.) The index of the first search result MAY be indicated in the OSDD's "Url" element, using the indexOffset attribute. If the startIndex element does not appear on the page then the search client should consider the current page to be the first page of search results. • Restrictions: The value MUST be an integer. • Requirements: The element MAY appear zero or one time. **Recommendation:** The Search Service implementation SHOULD include the opensearch:startIndex in the response. A Search implementation SHOULD use an indexOffset of 1. **Example:** <opensearch:startIndex>11</opensearch:startIndex>

`opensearch:itemsPerPage`	The number of search results returned per page. If the itemsPerPage element does not appear on the page then the search client SHOULD use the number of items of the current page as the default page size. • Restrictions: The value must a non-negative integer. • Default: The default value is equal to the number of search results on the current page. • Requirements: The element may appear zero or one time. **Recommendation:** The Search Service implementation SHOULD include the opensearch:itemsPerPage in the response. **Example:** `<opensearch:itemsPerPage>10</opensearch:itemsPerPage>`

The following XML illustrates one possible representation of these properties in a notional response (the Result Type specification will define the exact representation):

```
<atom:feed>
  ...
  <opensearch:totalResults>12</opensearch:totalResults>
  <opensearch:startIndex>11</opensearch:startIndex>
  <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
  ...
</atom:feed>
```

The response MAY include paging links to the previous and next results. In addition, the service MAY include the first results page, current results page, and last results page. For instance, if the response format is Atom, the paging links can be included as follows:

```
<atom:feed>
   ...
   <atom:link rel="previous" href="http://example.com/test?q=tanks&page=2&format=atom"
type="application/atom+xml"/>
   <atom:link rel="next" href="http://example.com/test?q=tanks&page=4&format=atom"
type="application/atom+xml"/>
   <atom:link rel="self" href="http://example.com/test?q=tanks&page=3&format=atom"
type="application/atom+xml"/>
   <atom:link rel="first" href="http://example.com/test?q=tanks&page=1&format=atom"
type="application/atom+xml"/>
   <atom:link rel="last" href="http://example.com/test?q=tanks&page=447&format=atom"
type="application/atom+xml"/>
   ...
</atom:feed>
```

3.2.4 Post-conditions

The following conditions MUST be met upon completion of this function.

1. The results returned to the requester are relevant to the input query.
2. The results are in the correct format.
3. The authenticated requester has been authorized to receive each result in the response.

4. This function has been audited according to applicable policy.[6]

3.3 Fault Conditions

An implementation of the Search service MUST allow for the Fault Conditions defined in the CDR Specification Framework.

Query type specifications MAY create additional Fault Conditions, as necessary. Any new fault types SHOULD derive from existing fault types, if possible.

Table 5 maps the CDR Specification Framework fault conditions to the HTTP status that SHOULD be returned for each.

Table 5 – Fault Conditions and HTTP Responses

CDR Framework Fault Condition	HTTP Status	HTTP Description
Unauthorized Access	403	Forbidden
Unsupported Search Request Syntax	400	Bad Request
Unsupported Search Element	400	Bad Request
Invalid Paging Value	400	Bad Request
Paging Value Out of Range	404	Not Found
Service Execution Error	500	Internal Service Error

4 Search Service Implementation

This section provides additional implementation guidance beyond the behavior and interface guidance provided in the previous sections.

4.1 Result Types

The CDR Specification set includes a single predefined Result Type definition that IC/DoD organizations can leverage in their Search service implementations, the IC/DoD Content Discovery and Retrieval Atom 1.0 Result Set Specification [CDR-ATOM]. Implementers SHOULD consult appropriate policy and implementation guidance to determine requirements or recommendations concerning the use of particular Result Types.

4.2 Sorting of Search Results

Sorting is OPTIONAL for Search services. Search services that do implement sorting SHOULD return results sorted by relevance.

4.3 Security Considerations

Any resource may have associated policies for use, especially as applies to authentication and authorization. These policies may be asserted by both the resource owner and those responsible for governance and management of the enterprise. The implementation of policies related to security considerations should leverage the currently applicable IC and DoD policies.

[6] *Afunction may be audited according to applicable policy regardless to the success or failure of the function.*

5 Reference Documents

The documents in this section provide the foundation for the Search service. Each document is assigned a reference identifier, which is cited when the document is referenced within this Search Service Specification.

Ref.	Title	Version	Date
CDR-SF	IC/DoD Content Discovery & Retrieval Specification Framework	1.0 DRAFT	9 May 2011
CDR-RA	IC/DoD Content Discovery and Retrieval Reference Architecture	1.1	25 Feb 2011
ATOM	The Atom Syndication Format http://www.ietf.org/rfc/rfc4287	1.0	Dec 2005
CDR-ATOM	IC/DoD Content Discovery and Retrieval Atom 1.0 Result Set Specification	1.0	March 2010
OS	OpenSearch http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_4	1.1, Draft 4	2009
OS-RELV	OpenSearch Relevance Extension http://www.opensearch.org/Specifications/OpenSearch/Extensions/Relevance/1.0	1.0, Draft 1	2007
OS-GEO	OpenSearch Geo Extension http://www.opensearch.org/Specifications/OpenSearch/Extensions/Geo/1.0/Draft_1	1.0, Draft 1	2009
OS-TIME	OpenSearch Time Extension http://www.opensearch.org/Specifications/OpenSearch/Extensions/Time/1.0/Draft_1	1.0, Draft 1	2010
DDMS	DoD Discovery Metadata Specification	3.0	2010