UNCLASSIFIED

Intelligence Community and Department of Defense Content Discovery & Retrieval Integrated Project Team (CDR IPT)

IC/DoD SOAP Interface Encoding Specification for CDR Search v1.1

12 May 2011

UNCLASSIFIED

REVISION/HISTORY

Doc Revision	Revised By	Revision Date	Revisions
0.1		20 November 2009	Initial draft for subgroup review.
0.2		18 December 2009	All sections updated.
0.4.1		01 January 2010	All sections updated. Changed the revision number to reflect the associated RA and Specification Framework documents, respectively.
0.4.4		15 January 2010	All sections updated.
0.9.0	CDR IPT	22 January 2010	First draft release for CDR SOAP Working Group review.
0.9.1	CDR IPT	28 January 2010	All sections updated.
0.9.2	CDR IPT	04 February 2010	All sections updated.
0.9.3	CDR IPT	16 February 2010	Minor updates.
0.9.4	CDR IPT	19 February 2010	Minor updates.
0.9.5	CDR IPT	26 February 2010	Minor updates.
1.0-Milestone 1	CDR IPT	09 March 2010	Minor updates.
1.0-Milestone 1	Farley	23 April 2010	Tech Edit

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Service Overview

The Search Component, as defined by the "IC/DoD Content Discovery and Retrieval (CDR) Specification Framework" [SF], serves as the primary content discovery mechanism to expose content collections for discovery and accessibility. This component provides a common interface and behavioral model for IC and DoD content collections, enabling consumers to discover relevant content resources from disparate collections across the IC/DoD Enterprise. Specifically, the Search Component provides a means to accept a uniform syntax and semantics that can be transformed, as needed, and applied to newly-developed or existing content collections, unambiguously conveying a query without knowing or setting requirements on the implementation of the underlying content collection.

This specification defines requirements and provides guidelines for the realization of the CDR Search Component as a web service using the SOAP messaging protocol, 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 function that is responsible for three activities that underpin Content Discovery capabilities: search, result presentation, and results paging. As discussed in 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[1]. Some of the information contained within each Search result may provide the information necessary for a consumer to retrieve or otherwise use a resource.

Search services may support SOAP 1.1 or SOAP 1.2. While SOAP 1.2 is RECOMMENDED, Search service providers should support the version of SOAP that best meets their business objectives, referencing standard registries such as DISR[2] and ICSR[3], as appropriate, but MAY support both versions, if desired.

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 specific security components and interactions defined by the Joint IC/DoD Security Reference

[1] *The Search Component returns metadata about a resource, which may sometimes describe the underlying resource (e.g., an image), while other times representing a sub-set of the data that 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.*

[2] *DISR: DoD IT Standards Registry*

[3] *ICSR: Intelligence Community Standards Registry*

Architecture (SRA), and MUST be in compliance with requirements and guidance for security outcomes as specified in the SRA and its associated specifications.

1.1.1 Relationship to Other CDR Architecture Elements[4]

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 [RA], illustrates this model.

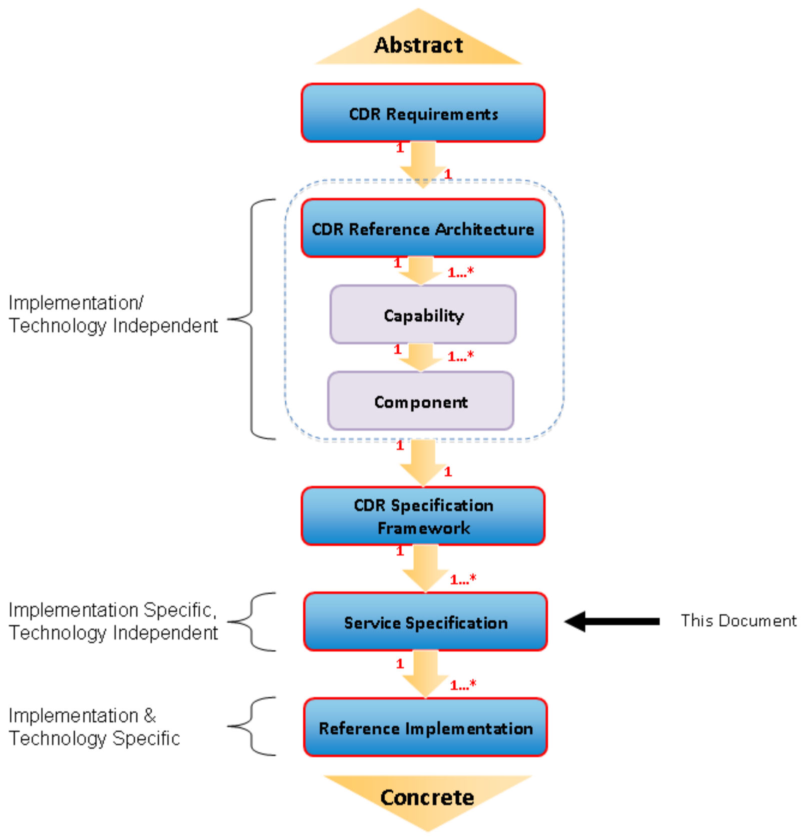


Figure 1 – CDR Architecture Model

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 then expands on the details of information flows and the information conveyed in those flows to provide a consistent basis for multiple Service Specifications to provide consistent interfaces, both in terms of the structure and semantics of the exchanged information. Service Specifications, such as this one, provide implementation-specific guidance.

[4] *For a detailed description of each of the layers, please reference the CDR RA Section 1.*

This Search Specification defines the specific guidance for implementing the CDR Search Component as a SOAP-based service. It describes base functionality, such as Query Types and Results Sets, and relies on additional sub-specifications to provide details for possible implementation. Over time, this approach will ensure a future-resilient specification that can easily adapt to changing inputs and outputs. For further guidance on which Query Types and Result Sets should be used in conjunction with this specification, please consult the *Search Service Policy Guide.*

This specification covers the following aspects of a SOAP-based Search Component:

- **Service Behavior** maps the Search interaction patterns defined in the Specification Framework to concrete SOAP constructs.
- **Service Interface** defines the base SOAP constructs to express inputs, outputs, and faults without defining concrete Query Types or Result Sets.
- **Implementation** provides additional implementation guidance beyond service behavior and interface guidance.
- **Reference Documentation** provides references to key sub-specifications (i.e, Atom Result Sets, DDMS Query and Result Set guidance) in addition to other CDR and community artifacts (i.e., Security Reference Architecture).

1.2 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). The use of {any} indicates the presence of an element wildcard (<xs:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>).

Examples in this text are distinguished by a black border and items that should be focused on are highlighted in yellow. These are meant to be illustrative and only one way that the described syntax can be used. Examples may include several lines derived from the sub-specifications, e.g. cdrss:Expression, and included solely to complete the example. These lines are presented in green.

1.3 Conformance

This specification defines an interface to a ***Search*** service to which an implementation and a subsequent deployment MUST conform. For an implementation to conform to this ***Search*** specification, it MUST adhere to all mandatory aspects of the specification do the following:

A ***Search*** service MUST support at least one query type.

A ***Search*** service MUST support returning at least one type of Result Set in its responses.

1.4 Namespaces

The namespace for the ***Search*** service is designated by the URI "**urn:cdr:1.0:soap:SearchService**". This namespace is indicated by the "**cdrss**" prefix in this document. Additional namespaces referenced in this document and the prefixes used to represent them are listed in Table 1.

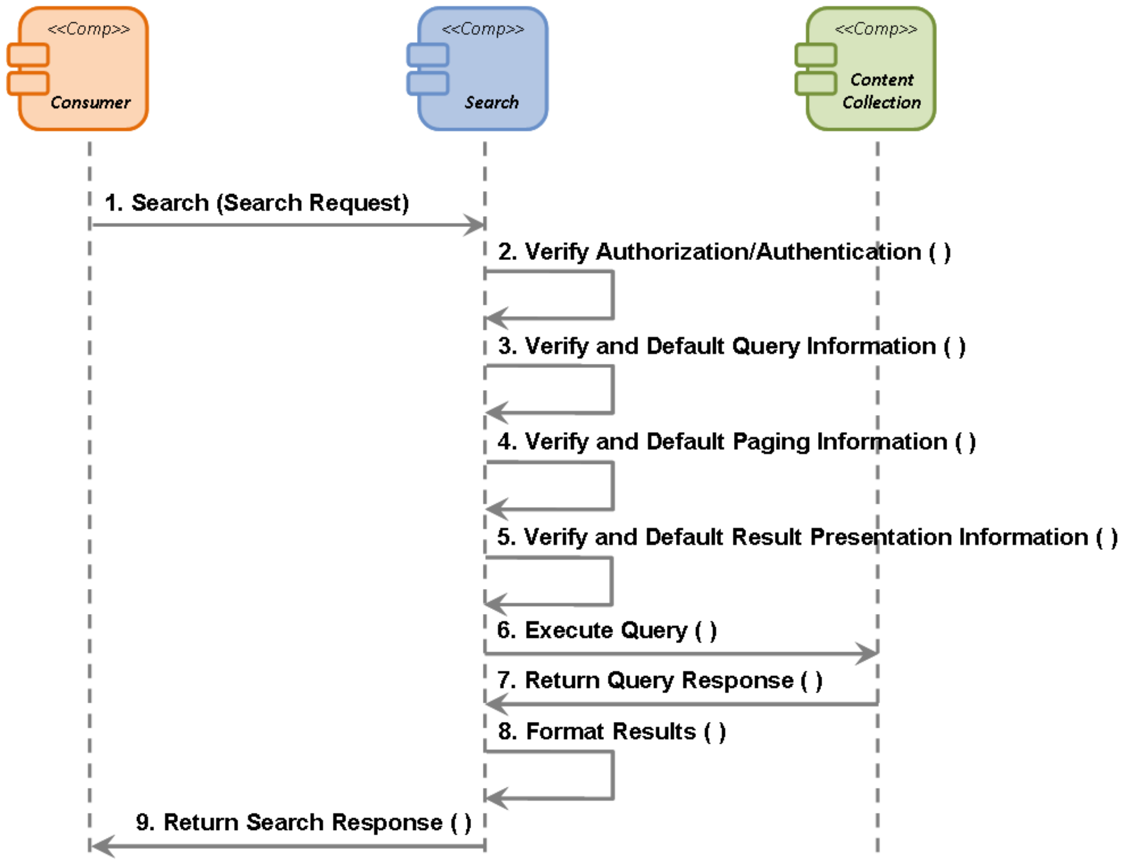
Table 1 only represents those XML Namespaces that are directly leveraged in this document. Additional namespaces are introduced in the set of specifications that compliment core search functionality, including those used in specific query types (e.g., Keyword, XQuery), response types (e.g., Atom), and data standards (e.g., DDMS, IRM.XML).

Table 1 – Referenced XML Namespaces

Prefix	URI	Description
env	http://www.w3.org/2003/05/soap-envelope	W3C SOAP Version 1.2
wsa	http://www.w3.org/2005/08/addressing	WS-Addressing Definition
cdrss	urn:cdr:1.0:soap:SearchService	CDR v1.0 Search Service Specification for SOAP Implementations

2 Search Service Behavior

2.1 Main Flow


<<Comp>>
Consumer
<<Comp>>
Search
<<Comp>>
Content Collection
1. Search (Search Request)
2. Verify Authorization/Authentication ( )
3. Verify and Default Query Information ( )
4. Verify and Default Paging Information ( )
5. Verify and Default Result Presentation Information ( )
6. Execute Query ( )
7. Return Query Response ( )
8. Format Results ( )
9. Return Search Response ( )

1. The Consumer invokes the Search function on the Search Component with the parameters for a particular type of Search Request specified. The Search Request contains the query, paging information, and any other information required by the particular query type being sent.
2. The Search Component leverages a set of security components to verify that the Consumer is authenticated and authorized to perform the search query. The "Joint IC/DoD Security Reference Architecture" defines the specific security components and interactions needed to perform this verification.
3. The Search Component checks the type of the incoming Search Request (e.g., keyword, XPath, etc.) to verify that it supports that particular type and that its syntax is valid.
4. The Search Component checks the validity of the provided paging information and utilizes default values for any that are missing.
5. Because Search service result presentation is completely determined via SOAP binding, this step of the main flow is not applicable to SOAP-based Search services.

6. The Search Component executes the query against its Content Collection.
7. The Search Component receives the results for the query that was executed against its Content Collection.
8. The Search Component performs any necessary formatting of the results before returning a response to the Consumer.
9. The Search Component returns a formatted response back to the Consumer.

2.2 Result Presentation

The CDR Specification Framework identifies additional result presentation behaviors not discussed in the prior section:

- Result Sorting Order
- Result Metadata Style

Support for sorting functionality is OPTIONAL. If no sorting constraint is explicitly defined (e.g., temporal, geospatial), ***Search*** services SHOULD by default provide results sorted by relevance, if possible. The base search request does not include any parameters to request or control sorting, i.e., the Result Sorting Order variable described by the CDR Specification framework is ignored. Instead, individual Query Type specifications MAY add sorting parameters and behavior requirements.

Result Metadata Style is not provided as an input parameter for search requests. Although a ***Search*** service MAY support more than one type of Result Set in its response, ***Search*** service providers MUST associate a particular type of Result Set with a particular service binding. Therefore, the type of Result Set expected in the Response is implicit in the service being called. There will be a different binding for each supported Result Set type (e.g., one that returns results in an Atom Feed and a second that returns results in a custom Result Set type). Also, the resource metadata associated with each result is prescribed fully by the Result Set type, unless otherwise specified for a particular query type.

2.2.1 Paging of Search Results and Query ID

Paginated search results can be useful when the number of results is very large or indeterminate. Service consumers can page through the result sets, accessing a subset of the overall result set as necessary. This capability will prevent search requests with very large result sets from overloading the server, network, or client.

Search result pages may be traversed using the information from the original ***Search*** service request combined with the endpoint information provided by the Endpoint Reference (EPR) Self Link (see section 3.1.3.4.1) describing the ***Search*** service from which the current result set was generated. The ***Search*** service EPR allows a service consumer to issue a search request for the next "page" of data.

To invoke the ***Search*** service EPR, a service consumer MUST compose a SOAP message as defined by this ***Search*** service specification where the `cdrss:SearchRequest` in the message SHOULD be based on the one present in the search results. If a service consumer wished to get the next "page" of results, it would take the existing `cdrss:SearchRequest` and update the `cdrss:SearchRequest/@startIndex` element to the appropriate value. First, consider the following query, which will produce a set of results that support paging:

```
<cdrss:SearchRequest
  queryTypeURI="urn:cdr:1.0:soap:query:keyword"
  startIndex="1"
  resultsPerPage="10">
  <cdrss:Expression updatedMin="2010-01-26T00:00:00Z"
              updatedMax="2010-01-27T00:00:00Z">
    advanced conventional weapons
  </cdrss:Expression>
</cdrkw:SearchRequest>
```

Next, consider the corresponding Result Set containing the following `opensearch:totalResults` element and `cdrss:SearchRequest` element (a `cdrkw:KeywordSearchRequest` in this example; SOAP elements omitted):

```
<resultSet>
  …
  <opensearch:totalResults>492420</opensearch:totalResults>
  …
  <cdrss:SearchRequest
    queryTypeURI="urn:cdr:1.0:soap:query:keyword"
    startIndex="1"
    resultsPerPage="10">
    <cdrss:Expression updatedMin="2010-01-26T00:00:00Z"
              updatedMax="2010-01-27T00:00:00Z">
      advanced conventional weapons
    </cdrss:Expression>
  </cdrss:SearchRequest>
  …
</resultSet>
```

A subsequent request to get the next "page" of results starting with an offset of 11, would look like this (SOAP elements omitted):

```
<cdrss:SearchRequest
  queryTypeURI="urn:cdr:1.0:soap:query:keyword"
  startIndex="11"
  resultsPerPage="10">
  <cdrss:Expression updatedMin="2010-01-26T00:00:00Z"
              updatedMax="2010-01-27T00:00:00Z">
    advanced conventional weapons
  </cdrss:Expression>
</cdrss:SearchRequest>
```

Alternatively, service providers MAY return a `cdrss:QueryId` element as an extension to the result set in order to access cached results generated by the initial query. The length of time to retain a cached result set and the choice to provide a `cdrss:QueryId` in a result set is left to the individual provider implementations. This extension indicates the unique identifier that the service provider returned with the initial results of the executed search. Service consumers SHOULD include the value of the `cdrss:QueryId` extension as a property of subsequent requests for the other pages of the result set. Consider a Result Set that contains an `opensearch:totalResults` element, `cdrss:QueryId` element, and `cdrss:SearchRequest` element:

```
<resultSet>
  …
  <opensearch:totalResults>492420</opensearch:totalResults>
  <cdrss:QueryId>e86c698a-d922-445d-9785-85f2ba22a83</cdrss:QueryId>
  …
  <cdrss:SearchRequest
    queryTypeURI="urn:cdr:1.0:soap:query:keyword"
    startIndex="1"
    resultsPerPage="10">
    <cdrss:Expression updatedMin="2010-01-26T00:00:00Z"
                  updatedMax="2010-01-27T00:00:00Z">
      advanced conventional weapons
    </cdrss:Expression>
  </cdrss:SearchRequest>
  …
</resultSet>
```

A subsequent request to get the next "page" of results starting with an offset of 11, would look like this (SOAP elements omitted):

```
<cdrss:SearchRequest
  queryTypeURI="urn:cdr:1.0:soap:query:keyword"
  startIndex="11"
  resultsPerPage="10">
  <cdrss:QueryId>e86c698a-d922-445d-9785-85f2ba22a83</cdrss:QueryId>
</cdrkw:SearchRequest>
```

While the `cdrss:QueryId` is intended to supplant query terms in subsequent paging requests, service providers SHOULD gracefully handle cases where both query terms and a `cdrss:QueryId` are provided within the same request. For instance, the provider may choose to leverage the `cdrss:QueryId` instead of query terms, if the `cdrss:QueryId` is valid. Alternatively, if a `cdrss:QueryId` has expired, the service provider may choose to issue a new query based on the supplied query terms. Regardless of the chosen approach, a service provider SHOULD return an appropriate indication of how a conflicting request was handled in the "Results Metadata" of the Result Set (see 3.1.3.3 for more information). In particular, if a cdrss:QueryId is presented in a request but is no longer valid, an "Invalid Paging Value Fault" should be returned.

It is important to note that the paging mechanism supported by the ***Search*** service does not guarantee continuity of search results while switching pages in the sense that it may not be possible to guarantee that consumers will be able to reconstruct the contents of the entire result set at a particular time. Data assets 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, unless such continuity is explicitly supported.

2.2.2 Relevance of Search 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 ***Search*** service specification defines a `cdrss:Relevance` element that communicates the relevance score:

```
<cdrss:Relevance>0.97</cdrss:Relevance>
```

This specification does not define the mechanism by which the relevance score is determined and comparing scores provided by different ***Search*** implementations may not indicate a true comparison of relevancy.

An example Result Set that includes `cdrss:Relevance` elements in its individual results follows:

```
<resultSet>
  ...
  <result>
    result metadata
    ...
    <cdrss:Relevance>0.97</cdrss:Relevance>
  </result>
  <result>
    result metadata
    ...
    <cdrss:Relevance>0.42</cdrss:Relevance>
  </result>
  ...
</resultSet>
```

Description of significant elements:

/cdrss:Relevance
This OPTIONAL value is intended to capture a relevancy score. The range of values allowed is any decimal between 0 to 1, inclusive, with 1 being the most relevant and 0 the least.

3 Search Service Interface

Web Service Description Language (WSDL) documents are provided as supplements to this service specification that provide the necessary WSDL bindings for the ***Search*** Service.

3.1 Search Function

Function	Input	Output	Fault
Search	*cdrss:SearchRequest*	cdrss:SearchResponse	Defined within CDR Framework

The ***Search*** service specification is REQUIRED to function as described by the Content Discovery and Retrieval (CDR) Specification Framework with any input, behavior, output, and fault condition extensions listed below.

3.1.1 Input

In addition to the requirements imposed by the CDR Specification Framework, the following describes further input constraints on the ***Search*** service Search function:

- A *`cdrss:SearchRequest`* element MUST be the only element present in the `/env:Body`

3.1.1.1 Search Request

The *cdrss:SearchRequest* element should be used directly.

The following example illustrates a search request message to a ***Search*** service, leveraging a notional keyword query (for more information on Query Types, please see section 5.1.1.1):

```
<env:Envelope>
  <env:Header>
    …
    <wsa:Action>
      urn:cdr:1.0:soap:action:request
    </wsa:Action>
    …
  </env:Header>
  <env:Body>
    <cdrss:SearchRequest
      queryTypeURI="cdr:1.0:soap:query:keyword"
      startIndex="1"
      resultsPerPage="50"
      timeout="20000">
      <cdrss:Expression
        updatedMin="2010-01-25T01:00:00Z"
        updatedMax="2010-01-26T12:00:00Z">
        UNMANNED AERIAL VEHICLE
      </cdrss:Expression>
    </cdrss:SearchRequest>
  </env:Body>
</env:Envelope>
```

Description of significant elements:

/cdrss:SearchRequest
This REQUIRED element, located directly inside the `env:Body`, encapsulates the search request, as shown in the example.

***/cdrss:SearchRequest*/@queryTypeURI**
This REQUIRED element identifies the type of query being provided in the search request.

***/cdrss:SearchRequest*/@startIndex**
This OPTIONAL element describes the desired start index of the search execution. Its value, if provided, MUST be greater than or equal to 1. The default value is 1. Support for this property is REQUIRED.

***/cdrss:SearchRequest*/@resultsPerPage**
This OPTIONAL element describes the desired number of search results per page. Its value, if provided, MUST be greater than or equal to 1. The default value is 10. Support for this property is REQUIRED.

***/cdrss:SearchRequest*/@timeout**
This OPTIONAL element describes the desired timeout period (in milliseconds) of a search request. If present, a ***Search*** Service must return results (even if partial) by the end of the given timeout period. If no results are available at the end of the timeout period and the search has not yet completed, a fault SHOULD be returned. If partial results are returned due to a timeout, a service provider SHOULD return an appropriate indication in the “Results Metadata” of the Result Set (see 3.1.3.3 for more information). Support for this property is OPTIONAL.

***/cdrss:SearchRequest*/cdrss:QueryId**
This OPTIONAL element, intended to leverage result caching, describes the internal processing ID assigned to a previously executed request.

3.1.1.2 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 function. Table 2 below relates the disposition of each variable defined in the Framework in this specification:

Table 2 – Framework Input Variable Disposition

Activity	*Framework Input Variable*	*Search Specification*
Search	Query (R)	*cdrss:SearchRequest* and the concrete search request types defined in associated query type specifications (R)
	Query type (O)	*cdrss:SearchRequest*/@queryTypeURI (R)
	Query metadata (O)	Specialized – query type definitions MUST specify all possible information (including metadata) that can be used as input for its type. In essence, query metadata is not specified in the common search request, rather it is located in the specific query types, to be defined and used as necessary on a per-type basis.

	Timeout (O)	*cdrss:SearchRequest*/@timeout (O)
Results paging	Results per page (O)	*cdrss:SearchRequest*/@resultsPerPage (O)
	Start index (O)	*cdrss:SearchRequest*/@startIndex (O)
	Query identifier (O)	*cdrss:SearchRequest*/cdrss:QueryId (O)
Results presentation	Result metadata format (O)	Implicit – each Search service binding MUST be associated with a specific result metadata format (a.k.a., Result Set). Therefore this input variable is not needed.
	Result sorting order (O)	Default sorting by relevancy is RECOMMENDED. Individual query types MAY define input variables to control custom sorting.Otherwise, the sorting order input is not supported and no variable is defined to support it in *cdrss:SearchRequest*.

3.1.2 Behavior

An implementation of the ***Search*** service MUST follow the behavior defined in the CDR Specification Framework.

Additional requirements on behavior MAY be defined by query type specifications. ***Search*** services MUST implement the behaviors required by the query types they support.

3.1.3 Output

In addition to the requirements imposed by the CDR Specification Framework, the following describes the additional output constraints on the ***Search*** service Search function:

- A `cdrss:SearchResponse` element MUST be the only element present in the `/env:Body`.

The ***Search*** service MAY support multiple Result Set types within the `cdrss:SearchResponse`.

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

```
<env:Envelope>
  <env:Header>
  ...
  <wsa:Action>
    urn:cdr:1.0:soap:action:response
  </wsa:Action>
  ...
  </env:Header>
  <env:Body>
    <cdss:SearchResponse>
      <resultSet>
        <result>...</result>
        <result>...</result>
        ...
      </resultSet>
    </cdss:SearchResponse>
  </env:Body>
</env:Envelope>
```

Description of significant elements:

/env:Envelope/env:Body/cdrp:Response
This REQUIRED element contains a Result Set that, in turn, holds the resource metadata. It may also optionally contain status information and a description of the response contents (not shown).

3.1.3.1 Relation to Outputs Defined in the Specification Framework

The IC/DoD CDR Specification Framework defines a number of required (R) and optional (O) outputs from the Search function. The Table 3 below 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 `cdrp:Response` that contains a Result Set (or other object). The ***Search*** service MUST be associated with a particular return type on binding. (R)
	Results metadata (R)	CDR Result Set specifications MAY require certain types of data to be returned as part of the collection or the collection's individual 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 Set MUST follow the syntax and processing rules defined by that type. (R)
	Result set retrieval properties (R)	A cdrss:link[@ref="alternate"]/`wsa:EndpointReference`, providing bindings information to the ***Retrieve*** service, MUST be included in the Result Set. (R)

	Result relevancy value (O)	`cdrss:Relevance` (or, alternatively, a relevance variable defined by the Result Set specification) in each result (O)
	Result retrieval properties (O)	A cdrss:link[@ref="alternate"]/`wsa:EndpointReference`, providing binding information to a ***Retrieve*** service MAY be included in both the Result set and each individual result within the Result Set (which, if present, would override the same reference at the Result Set level). (O)
	Timestamp (O)	*`cdrss:SearchRequest`*/@`cdrss:timestamp` (O)
	Query identifier (O)	`cdrss:QueryId` in the Result Set (O)
	Response result count (O)	`opensearch:itemsPerPage` SHOULD be included in the Result Set (O).
	Total result count (O)	`opensearch:totalResults` SHOULD be included in the Result Set (O).

3.1.3.2 Including the Search Request in the Result Set

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

```
<resultSet>
  …
  <cdrss:SearchRequest
    queryTypeURI="urn:cdr:1.0:soap:query:keyword
    startIndex="1"
    resultsPerPage="7">
    <cdrss:Expression updatedMin="2010-01-26T00:00:00Z"
                 updatedMax="2010-01-27T00:00:00Z">
      advanced conventional weapons
    </cdrss:Expression>
  </cdrss:SearchRequest>
  …
</resultSet>
```

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 or the Result Set, beyond the metadata required by the Result Set 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 Set in its response MUST follow the requirements in the associated Result Set specification.

3.1.3.4 WS-Addressing Endpoint References

To support the paging of search results and the retrieval of data resources, the result MAY include a `wsa:EndpointReference` (EPR). An EPR, if present, MUST be included for the ***Search*** service itself and any associated ***Retrieve*** service endpoints within the results. The EPR to the ***Search*** service facilitates features such as paging while the EPRs to associated ***Retrieve*** services allow service consumers to retrieve data resources described in the results. The required format of these EPRs is described below.

3.1.3.4.1 Search Service Endpoint Reference

The ***Search*** service EPR must describe the interface that it implements, the qualified name of the service offered, the qualified name of the service endpoint, and the corresponding endpoint of the ***Search*** service. ***Search*** service implementations MUST include one EPR, self-describing the ***Search*** service. The Search service EPR should be within the returned Result Set. More detailed information can be found in ResultSet guidance, such as the Atom 1.0 Result Set Specification.

3.1.3.4.2 Retrieve Service Endpoint Reference

The ***Retrieve*** service EPR must describe the interface that it implements, the qualified name of the service offering, the qualified name of the service endpoint, and the endpoint of the ***Retrieve*** service. ***Search*** service implementations SHOULD include at least one EPR describing each data resource returned in the search results. To facilitate both single provider and brokered scenarios, ***Retrieve*** service EPRs can be specified in several ways:

- ***Search*** service implementations MAY include one or more EPRs at the Result Set level. These EPRs SHALL be considered the default ***Retrieve*** service EPRs for the entire Result Set. Data resources described by result elements that do not themselves contain ***Retrieve*** service EPRs MUST be retrievable through the ***Retrieve*** services described by the Result Set level ***Retrieve*** service EPRs.
- ***Search*** service implementations MAY include one or more EPRs at the individual result level. More detailed information can be found in the Result Set documentation.

3.1.4 Fault Conditions

An implementation of the ***Search*** service MUST allow for the Fault Conditions defined in the CDR Specification Framework. Although fault SHOULD use the existing Fault conditions, query type specifications MAY create additional Fault Conditions, when necessary. The following are Fault Conditions defined by the CDR Specification Framework:

Security

- During execution of the Search Function, one or more of the security components determines that the Consumer is either not authenticated or not authorized to perform the search query. As a result, the Search Component returns an appropriate security fault back to the consumer as defined by the "Joint IC/DoD Security Reference Architecture".

Query Type Not Supported

- During execution of the Search Function, the Search Component determines that it does not support the Query Type that was specified., and returns a "Query Type Not Supported" fault back to the consumer.

Invalid Query Syntax

- During execution of the Search Function, the Search Component determines that the Search Request syntax is not valid in accordance with the specified Query Type, and returns an "Invalid Query Syntax" fault back to the consumer.

Query Term Not Supported

- During execution of the Search Function, the Search Component determines that it cannot understand/support one or more elements of the search request, and returns a "Query Term Not Supported" fault back to the consumer.

Query Timeout

- During execution of the Search Function, the Search Component determines that the query cannot be executed in the amount of time specified by the Timeout input parameter, and returns a "Query Timeout" fault back to the consumer.

Query Execution Fault

- During execution of the Search Function, the Search Component encounters an error during query execution, and returns a "Query Execution" fault back to the consumer.

Query Metadata Fault

- During execution of the Search Function, the Search Component determines that query metadata is not understood or contains an error, and returns a "Query Metadata"fault" back to the consumer. In the case of Query Metadata not being understood, a Search Component MAY choose to continue the execution of the query. In this case, some indication SHOULD be provided in the output's Result Metadata Properties.

Invalid Paging Value Fault

- During execution of the Search Function, the Search Component determines that the Start Index and/or Results Per Page values are not valid values (e.g., non-integer, negative), and returns an "Invalid Information" fault back to the consumer.

Out of Range Fault

- During execution of the Search Function, the Search Component determines that the Start Index and/or Results Per Page values are not in a valid range, and returns an "Out Of Range"fault back to the consumer.

Result Format Not Supported

- During execution of the Search Function, the Search Component determines that it does not support the result format specified by the Result Metadata Format input parameter, and returns a "Result Format Not Supported" fault back to the consumer.

Result Sorting Not Supported

- During execution of the Search Function, the Search Component determines that it does not support the result sorting mechanism specified by the Result Sorting input parameter, and returns a "Result Sorting Not Supported" fault back to the consumer.

3.1.4.1 Fault Handling in SOAP

Different versions of SOAP may have different fault handling syntaxes. ***Search*** services MUST use the primary fault handling mechanism for the version of SOAP they support and to which the service is bound. In the following example, a "Unsupported Query Type" fault is returned using the SOAP 1.2 syntax:

```
<env:Fault>
  <env:Code>
    <env:Value>env:Sender</env:Value>
  </env:Code>
  <env:Reason>
    <env:Text>Unsupported Query Type Fault</env:Text>
    <env:Text>Service does not support the XQuery Query Type</env:Text>
  </env:Reason>
</env:Fault>
```

4 SOAP Search Specification Usage

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

4.1 Policy

This specification defines the technical requirements and guidelines for implementing a ***Search*** service. Policy for ***Search*** service implementations is described in auxiliary documents. See the Reference Documents section for a listing of relevant policy documents. Implementers MUST follow the guidance in those policy documents, as appropriate for their organization. For instance, DoD consumers should follow DoD policy guidance when sharing data across organizational boundaries, whereas IC consumers would follow IC policy, if differences between the two exist.

4.2 Query Types

The CDR Specification set includes a number of predefined Query Types definitions that IC/DoD organizations can leverage in their ***Search*** service implementations. See the Reference Documents section for a list of Query Type specifications distributed with this service specification. Also, consult the policy documents to determine and requirements or recommendations concerning the use of particular Query Types.

4.3 Result Sets

The CDR Specification set includes at least one Result Set definition that IC/DoD organizations can leverage in their ***Search*** service implementations. See the Reference Documents section for a list of Result Set specifications distributed with this service specification. Also, consult the policy documents to determine requirements or recommendations concerning the use of particular Result Sets.

4.4 Security Considerations

The "Joint IC/DoD Security Reference Architecture" [S1] and its associated specifications define the specific security components and interactions needed to perform authorization and authentication. ***Search*** implementations MUST follow the guidance in those documents.

5 Reference Documents

The documents in this section provide the foundation for, define extensions to, or include implementation guidance for the ***Search*** service. They define additional specifications, including those provided as part of the greater CDR specification set, and guidance documents that communicate current policy or implementation details. Each document is assigned a reference identifier, which is cited when the document is referenced within this Search Service Specification.

In some cases, documents have been referenced with a version and data of "future" in order to track the iterative development of some of these extensions.

5.1 Specifications

5.1.1 Content Discovery and Retrieval Specifications

The following documents provide a foundation and guidance for the development of this Search Specification document. Search service implementers should have a thorough understanding of the concepts and guidance in these documents. This Search Specification represents a realization of the Search Component defined therein.

Ref.	Title	Version	Date
SF	IC/DoD Content Discovery and Retrieval Specification Framework	DRAFT 0.6.2	29 Jan 2010
RA	IC/DoD Content Discovery and Retrieval Reference Architecture	DRAFT 0.4	16 Dec 2009

5.1.1.1 Query Type Specifications

This Search Specification enables maximum flexibility for handling different types of queries by defining an extensible search request type. To provide a baseline capability, the CDR specification set includes a number of Query Type specifications. See the associated policy documentation for guidance on which query types organizations should support and see Community Data Specifications for guidance on how to leverage data standards (e.g., DDMS) within Query Types.

The following CDR Query Type specifications and distributed with this CDR Search Specification:

Ref.	Title	Version	Date
Q1	IC/DoD Content Discovery and Retrieval Keyword Query Type Specification	DRAFT 1.0-Milestone 1	09 Mar 2010
Q2	IC/DoD Content Discovery and Retrieval XQuery Type Specification	DRAFT 0.9.6	09 Mar 2010
Q3	IC/DoD Content Discovery and Retrieval OpenGIS	DRAFT 1.0-	09 Mar 2010

	Filter Query Type Specification	Milestone 1	

5.1.1.2 Result Set Specifications

This Search Service Specification mandates the use of a `cdrss:SearchResponse` type, but that type can in turn contain any Result Set (or other object). See Community Data Specifications for guidance on how to combine data standards (e.g., DDMS) into Result Sets.

The following documents define the expected format and content of a particular type of collection returned from a CDR Search function --beyond that specified by the underlying Result Set type itself, if the Result Set is based on an industry standard:

Ref.	Title	Version	Date
C1	IC/DoD Content Discovery and Retrieval Atom 1.0 Result Set Specification	DRAFT 1.0-Milestone 1	09 Mar 2010

5.1.2 Other Specifications

5.1.2.1 Security Specifications

Ref.	Title	Version	Date
S1	Joint IC/DoD Security Reference Architecture	1.0	25 Jul 2008

5.1.2.2 Service Discovery Specifications

Ref.	Title	Version	Date
D1	Joint IC/DoD Service Discovery Architecture	DRAFT 1.2	28 Sep 2007

5.1.2.3 Community Data Specifications

Ref.	Title	Version	Date
C1	DDMS Data Query Type and Result Type Guidance	1.0-Milestone 1	09 Mar 2010
C2	IRM.XML Data Query Type and Result Type Guidance	Future	Future
C3	UCore Data Query Type and Result Type Guidance	Future	Future

5.1.2.4 Industry Specifications

Ref.	Title	Version	Date
I1	The Atom Syndication Format	1.0	Dec 2005

I2	OpenSearch 1.1 (Draft 4)	1.1	N/A

5.2 *Policy and Guidance*

5.2.1 Content Discovery and Retrieval Policy and Guidance

This specification primarily addresses the behavioral and interface aspects common to all CDR ***Search*** service implementations. The following documents provide additional requirements and expectations set by policy:

Ref.	**Title**	**Version**	**Date**
P1	IC/DoD Content Discovery and Retrieval Search Service Policy for SOAP Implementations	DRAFT 1.0-Milestone 1	09 Mar 2010