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

IC/DoD Content Discovery & Retrieval Specification Framework

V1.0

9 May 2011

REVISION/HISTORY

Doc Revision	Revised By	Revision Date	Revisions
0.1	CDR IPT	18 December 2009	Initial draft
0.2	CDR IPT	24 December 2009	Updates to 2.2, 2.4, & 2.5
0.3	CDR IPT	31 December 2009	Updates to 2.3 & 2.5
0.4	CDR IPT	08 January 2010	0.3 comment adjudication
0.5	CDR IPT	15 January 2010	Updated all Sections
0.6	CDR IPT	22 January 2010	Minor updates across doc
0.7	CDR IPT	07 June 2010	Inserted Retrieve Section
0.8	CDR IPT	07 June 2010	Inserted Brokered Search Section
0.9	CDR IPT	07 June 2010	Inserted Deliver Section
0.91	CDR IPT	31 January 2011	Begin harmonization – consistent formatting, corrected typos
0.92	CDR IPT	06 February 2011	Attempt consistent structure across Sections
0.93	CDR IPT	23 February 2011	Additional harmonization changes prior to formal comments
0.94	CDR IPT	04 March 2011	Add Query Mgt. Section
0.95	CDR IPT	13 March 2011	After 1stadjudication
0.96	CDR IPT	13 March 2011	After 2nd adjudication
0.97	CDR IPT	15 April 2011	Reorganized component Sections structure
0.98	CDR IPT	23 April 2011	Resolve remaining issues, revised Query Mgt.
0.99	CDR IPT	27 April 2011	Edits corresponding to the V0.98 adjudication
1.0	CDR-IPT	3 May 2011	Tech Edits

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Purpose

This CDR Specification Framework document provides guidance for ensuring consistency and interoperability in the development of CDR Service Specifications. Generally, it describes the structure and content for CDR Service Specifications including the description of their key characteristics and a decomposition of key behaviors in the context of various environmental and technical considerations.

The following subsections describe the CDR Specification Framework's relationship to other CDR architectural elements, its intended uses, and its intended audience.

1.2 Relationship to Other CDR Architecture Elements[1]

The CDR Architecture Model shown in Figure 1 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 Content Discovery and Retrieval Reference Architecture (CDR RA)[2], illustrates this model.

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

[2] *CDR-RA, IC/DoD Content Discovery and Retrieval Reference Architecture V1.1, February 2011.*

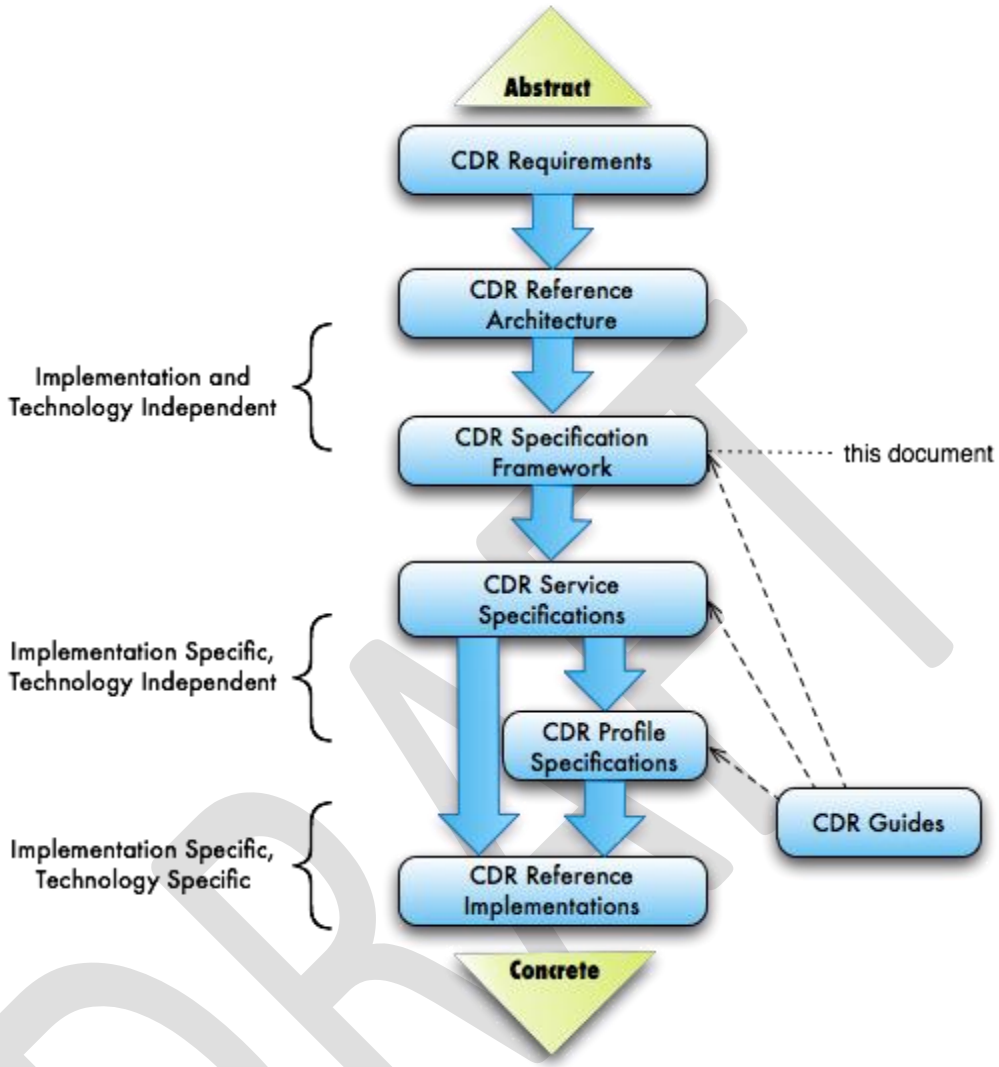

Figure 1 - CDR Architecture Model

As illustrated in Figure 1, this CDR Specification Framework represents one of the key tiers within the overall CDR Architecture. While this document provides more concrete levels of detail than the CDR RA, it does NOT provide implementation-specific guidance. That guidance is left to individual CDR Profile Specifications and CDR Service Specifications.

1.3 Intended Use and Audience

This Specification Framework document is intended to provide both CDR Service Specification developers/authors and CDR service developers/implementers guidance for developing and implementing CDR Service Specifications. Specifically, this Specification Framework describes the Interface models and related behavior for each Service Specification and how they should be codified. For CDR Service Specification developers/authors, the framework provides the structure and content guidance for how

CDR Service Specifications should be documented. For CDR service developers/implementers, the framework provides the common implementation and behavior guidance that, coupled with a specific CDR Service Specification, enables the realization of a CDR service.

1.4 Scope

This CDR Specification Framework describes in greater detail the CDR Components and capabilities presented in the CDR Reference Architecture. It is meant to provide guidance in enough detail to enable interoperability among independent implementations without otherwise constraining the implementation itself. In this vein, this document describes inputs and outputs to each component in the context of the expected behavior that clarifies what is needed as inputs, outputs, and other effects that are expected to be produced. It does not, however, specify the details of the internal implementation processing.

One of the important engineering aspects is to be able to compose the CDR Components to provide a full search and retrieval service. This Framework discusses the composable nature of the Components and describes the interfaces and interactions between components needed to compose them. It also identifies and provides guidance with respect to the Security and Messaging Components upon which CDR Components depend.

1.5 Guiding Principles

The CDR Specification Framework should:

1. Maintain consistency with the CDR RA and build on the CDR RA in a consistent manner.
2. Expand on concrete details and move towards implementation specifications.
3. Provide sufficient detail to enable its use in a consistent and unambiguous manner as a foundation for CDR Service Specifications. Thus, this framework will, as necessary, expand detail of elements within scope before expanding scope to other relevant aspects.
4. Reflect the needs of both the IC and DoD.

1.6 Conformance with CDR Specification Framework

The Specification Framework identifies and defines the interface and behavior of Core Components that underlie content discovery and retrieval capabilities. The identified components represent those within the current scope but this does not preclude additional components being identified and defined in the future.

Conformance with this specification does NOT REQUIRE that a particular use of this specification include all identified core components. However, conformance does REQUIRE that components which are used MUST conform to MANDATORY requirements for that component as identified in this specification.

1.7 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 Internet Engineering Task Force, Request for Comment (IETF RFC) 2119[3]. When these words are not capitalized, they are meant in their natural-language sense.

Throughout this document, system elements (functions and input/output), and inclusion information (Required, Optional, Recommended) are listed in tables similar to Table 1 below. These tables serve as the governing reference when it comes to determining whether or not a system element is required.

Table 1 - Convention for Stipulating System Element Inclusion

Element Name	Required/Optional
Name	Required

[3] *Internet Engineering Task Force (IETF) Request for Comment 2119, March 1997.*

2 Common Aspects of the CDR Component

2.1 Identification of the CDR Components

The CDR RA identifies four main component types as shown in Figure 1:

- Consumer Component
- Provider Component
- Core CDR Components
- CDR Dependency Components

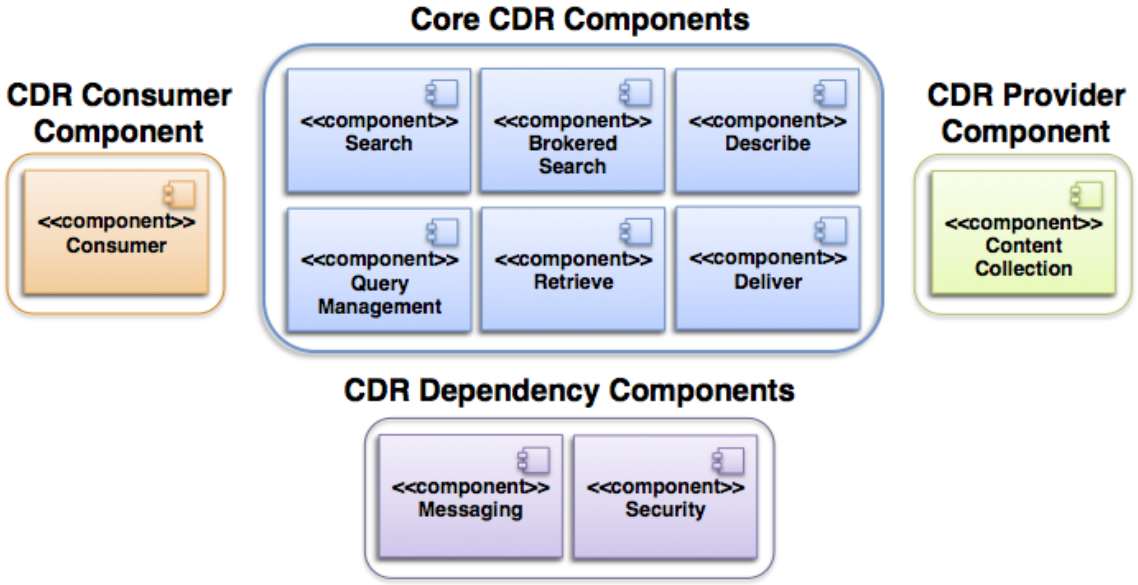

Figure 2 - CDR RA Components

The CDR Integrated Project Team (IPT) Requirements[4] are indicative of a diverse, heterogeneous and distributed information environment in which the CDR Services must operate. This specification elaborates on the Core CDR Components and provides context for the use of CDR Dependency Components that enable the realization of the CDR Functional Requirements. A summary of these functional requirements is:

- Enable standards-based searching, discovery, and retrieval of static and dynamic content and metadata, such as text, structured data in various ontologies and vocabularies, geospatial data, images, and binary documents. [5]
- Enable searches supporting a variety of query capabilities, including full text, wildcard, faceted, proximity, temporal, geospatial, field value based, natural language, and concept based queries.[6]
- Enable brokering of searches to multiple, distributed search service providers, constraining the search to particular content collections, and providing the aggregation and de-duplication of results[7]

[4] *CDR-IPT Requirements Specification, D2R-CDR Requirements FINAL 12-10-09-Mapping, available via CDR-IPT Technical Artifacts @ Unclassified Intelink Web Site:* https://www.intelink.gov/site/odni/cio/i2e/focus/iads/cdript/default.aspx.

[5] *CDR IPT Requirements addressed: 006, 025, 042, 043, 044, 054.*

[6] *CDR IPT Requirements addressed: 009, 010, 016, 017, 018, 026.*

- Enable the delivery of content via the most efficient mechanism given bandwidth, recipient type, security domain and other constraints.[8]
- Enable the generation of descriptions to inform consumers what data is available and the methods by which it can be discovered and retrieved, which supports the use of these descriptions in common registries.[9]
- Enable the storing of queries to be executed in the future, scheduled execution, subscribing to queries, and providing alerts to distributed consumers. [10]
- Enable security controlled access to all services, based on authenticated user identity, across different security domains and content of different classification levels.[11]

To satisfy these requirements, this CDR Specification Framework has been developed and is being managed incrementally based on prioritization of CDR capabilities. It describes the Interface models and related behavior of the CDR Search, Brokered Search, Retrieve, Deliver, Describe and Query Management (QM) Specifications to achieve the following capabilities:

Search:

- Enable searching through content and metadata in multiple formats as specified by the consumer, such as image files and textual documents.
- Enable searching through information content that is static, dynamic, structured and unstructured.
- Enable searching through and appropriately processing of information content and metadata at different classification levels, and with different handling caveats; information which could be located on different security domains. Metadata appears in many different formats, and could contain different parameters on which to search, (e.g., classification, date/time, country, National Intelligence Priority Framework (NIPF) category in the NIPF taxonomy, location, entity [person, organization, etc.]).
- Enable searching through natural language content (probably in many different languages) or highly formatted content such as geospatial or temporal content. (Data sources can be active [current, dynamically changing] or historic [static], each containing different data types).

Brokered Search:

- Facilitate the distribution of queries to applicable/relevant Search Components and content collections these Search Components expose.
- Aggregate the results returned individually into a single, uniform results set which is returned to the Consumer Component.

---

7 *CDR IPT Requirements addressed: 001, 004, 005, 014, 021.*

8 *CDR IPT Requirements addressed: 060.*

9 *CDR IPT Requirements addressed: 002, 011.*

10 *CDR IPT Requirements addressed: 012, 024.*

11 *CDR IPT Requirements addressed: 013, 051.*

Describe:

- Enable resource providers to expose information describing their content collections and content resources.
- Provide interested parties with a description of the resource and how it can be accessed or used.

Query Management:

- Enable service consumers to create, update, and store the search requests as Saved Searches.
- Enable service consumers to execute Saved Searches based on their specific request or on event triggers.

Retrieve:

- Enable retrieval of an identified content resource from the Content Collection in which it is stored.
- Initiate delivery of the retrieved resource to the requestor or to a designated alternate location using the Deliver Component.

Deliver:

- Enable a content resource to be delivered to a specified location which may or may not be the requesting component.
- Provide additional processing of the content to make it suitable for delivery to its destination and delivery path to be used.
- On behalf of the consumer, retrieve the requested content and then deliver to the specified location.

For each component, the Specification Framework denotes the component's externally visible behavior and consumer-facing interfaces for interacting with generalized external actors during certain basic scenarios, to include behavior during fault conditions. It is not meant to provide a fully exhaustive set of interactions. For example, many different alternative flows exist, particularly some of those that handle fault conditions.

The specification also provides an interface model, describing the parameters and resources that support information exchange described in the behavioral model. The interface model contains both mandatory elements that MUST be included in all implementations and optional elements that an implementation MAY choose to require, to leave as optional, or to ignore completely. The service specifications that derive from this Specification Framework will make such choices explicit for the implementations that will comply with that service specification.

This version of the Specification Framework includes a subsection in the discussion of each component that identifies relevant behaviors that were deemed out of scope at this time. In many cases, it was felt that such scoping limitations improved the feasibility of developing an initial implementation of the component; in some cases, it was felt that the behavior was beyond current experience to realistically define in sufficient detail. Given the iterative nature of this document, the fact that certain concepts are not included in the

current version in no way judges the potential applicability for a future version of the specification.

2.1.1 CDR Component Functions and Activities

The Specification Framework describes Core CDR Components in terms of the functions they realize and the internal activities that describe their behavior. For this specification:

- A CDR Component is a logical encapsulation of the processes and actions necessary to perform and realize the effects of a basic business task related to content discovery and retrieval.
- A CDR Component identifies a *function* as that processing which may be initiated by an external Consumer Component through an interface which defines the mandatory and optional inputs to be supplied by the Consumer Component and the outputs that will be returned per the Consumer Component's instructions.
- A CDR Component identifies an *activity* as that processing which is initiated internal to the component and is not directly accessed by a Consumer Component. Inputs to be used by an activity may be provided through the interface of a function that uses the activity and the output of the function may reflect processing provided by the activity.

Both a function and an activity may provide other effects, such as changes to an underlying database, which are not directly reflected in the component's output.

The CDR interfaces, in general, identify the minimal necessary set of inputs to support component functionality and additionally append a set of component-specific properties to configure implementation-specific options. For example, the Retrieve Component interface identifies the resource to be retrieved and the source from which the retrieval is to occur; the Retrieve Properties might include the content types the consumer is prepared to accept.

Note that the common definition of a software component is an encapsulated, reusable, and replaceable part of a software system. The CDR Component is consistent with the software definition when it is assumed that the corresponding software component automates the relevant processes and actions.

2.1.2 CDR Component Preconditions and Post-conditions

Successful realization of CDR Component behavior may depend on satisfying preconditions that are not reflected in the defined inputs and may result in changes that are not reflected in the defined outputs. For example, a precondition of Results Paging (Section 3.4.2) is that a previously executed query can be identified and an additional subset of the results can be accessed. If a sufficient time has passed since the query was executed, accessing the results of that query may no longer be possible. Similarly, the post-condition of Deliver (Section 0) is a specified resource should be resident with a specified recipient. Faults MAY result if pre- or post-conditions are not satisfied.

The interface definition for each component includes an explicit listing of preconditions and post-conditions. These are provided to contribute to a more complete understanding and facilitate a consistent set of expectations when interacting with a component

implementation. The listed preconditions and post-conditions are considered substantive to the purpose of the component but are not meant to be exhaustive, and the lists may be augmented or modified as experience in the use of the CDR capabilities expand and change over time.

2.1.3 CDR Component Composability

The CDR Component interfaces are designed to be stable and flexible while supporting evolution of the underlying functionality and how the individual components can be composed into more complex solutions. For example, the CDR Reference Architecture (RA) shows the interaction pattern where a resource accessed through the Retrieve Component can be redirected to another recipient through use of the Deliver Component. Indeed, the Brokered Search Component, as defined, heavily leverages multiple uses of the Search Component.

One mechanism by which the Core CDR Components building blocks enable composability is the use of an abstract properties bundle in the definition of the interfaces. The abstract properties allow for request embedding, where the information needed to invoke one service is included in the request to another service. A service can use this information to invoke another service in the process of generating its output. This allows request chaining, where the output of one service can serve as the input to another service.

As an example of request chaining, if a Retrieve Component implementation has the ability to engage a Deliver Component implementation, this can be enabled by supplying Deliver Properties as part of the Retrieve request. The retrieved resource that serves as the output of Retrieve becomes the input to Deliver.

As another example, security could specify that information necessary for user authentication be included as part of security properties.

The general use of the abstract properties bundle for each component is that the component-specific properties for one component can be included in a request to another component, and the component receiving the request will process the additional properties to the extent it is designed to do so. Thus, if a Retrieve implementation receives a request with Deliver properties and it does not support delivery instructions, then it will ignore those properties, possibly including a warning to the requester.

2.2 Common Security Concerns

The CDR RA discusses the dependencies between the CDR Core Components and a set of security components described in the Joint IC/DoD Security Reference Architecture (SRA),[12] Section 9.1 of the Specification Framework amplifies on this dependency and discusses the common security concerns for each of the CDR Core Components to aid specification writers, and component implementers in addressing these concerns. While

[12] *Joint IC/DoD Security Reference Architecture, v1.2, July 2008.*

the interaction diagrams for each component contain references to security-related concerns, the following should be noted:

- 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 SRA, and MUST be in compliance with requirements and guidance for security outcomes as specified in the SRA, its associated specifications and related security standards and guidance.
- The interaction diagrams in this Specification Framework include a notional step to authenticate the consumer's identity and authorization to use the CDR component functionality in the requested manner, as a placeholder for the security dependency.
- Properly securing CDR services is a team effort between engineers who are knowledgeable in implementing search and retrieval specifications, and security engineers who are knowledgeable in integrating security services into other IT enterprise services.

2.3 Common Definitions of Terms related to Search and Query

The concept of search is often discussed loosely in terms of queries and results, and variations of these may be combined with more advanced concepts dealing with saving and executing the search in the future. The following provides a consistent set of definitions for terms related to search and query as these terms are used throughout this document:

- Search: The process of (1) specifying (search) criteria against which matches to the criteria are to be found by a search capability that receives the criteria and (2) returning results of processing the criteria by the search capability; search may also specify parameters that control the search, e.g. how much time to wait for results, or the presentation of search results, e.g. how many results to return per page.
- Query: The criteria for the search expressed in a documented format, e.g. query language, identified by the query type; may also be referred to as query expression.
- Query Type: (deprecated input) Type of query that is represented in the Query input (e.g., keyword, XQuery, etc.)
- Query Metadata: (deprecated input) Information that assists in interpreting the query (see Section 3.4.1.1).
- Query Properties: Identified properties, e.g. query language, that enable determination of whether a search capability can process the query.
- Search Capability: An implementation that can accept, process, and return the results of a query.
- Search Request: All information sent as the request made by a Consumer Component (which may include other components acting in the role of consumer) that initiates a search.

- Search Results: The results of a search capability applying the (search) criteria to its internal corpus. The results may be metadata and indicate a mechanism to retrieve the indicated content or, if feasible and desirable, the entire content.
- Result: An individual match to a query; the aggregate of individual results comprises the search results. If available, result metadata MAY be returned with the result.
- Result Metadata: Information elaborating on the corresponding result. (See Section 3.3.)
- Result set: A subset of the search results that are returned in a response to a search request.
- Saved Search: A search request and related information appropriately identified and annotated, that is managed by implementations of the Query Management Component. Query Management may also submit a search request managed as part of a Saved Search to be executed by a search capability.
- Persistent Search: A Saved Search that is executed based on some defined trigger, where the trigger may be manual, time-based (e.g. every 3 days), or event-based (e.g. after 100 changes to a content collection).
- Query Management: A CDR Component that manages Saved Searches and may initiate search requests based on Saved Searches.

2.4 Use of Identifiers

This specification references identifiers in numerous places to uniquely identify a resource in a certain context. For example, queryID is used to identify a query and the corresponding results in the context of results paging. The Saved Search ID identifies a resource managed in a QM Collection.

This specification does not define whether specific identifiers are to be globally unique or unique within the domain of some managing authority. A Globally Unique Identifier (GUID) or a Universally Unique Identifier (UUID) uses a very large identifier space, on the order of 2^{128}, and algorithms that incorporate randomness to reduce the probability of identifier collisions. Uniform Resource Locators (URLs) and XML namespaces identify an authority to manage an address space and it is up to that authority to follow a scheme that ensures uniqueness.

In addition, identifiers may be unique in the context of a single system. In this case, the combination of a unique identifier for the system and the resource identifier provides a form of globally unique identifier, but the resource identifier is not guaranteed to be unique on its own.

The requirements for identifiers have not been defined in the context of CDR specifications. Such Guidance may be developed in future iterations of the CDR Specification Framework.

2.5 Common Properties Representation

Abstract properties are discussed in Section 2.1.3 in the context of CDR Component composability. The use of abstract properties also provides flexibility for configuring the use of a CDR Component without an incompatible change to the component interface.

While the generality of the abstract properties supports flexible composition, additional guidance for the structure and representation of these properties is needed to facilitate uniformity and consistency of use. Such Guidance may be developed in future iterations of the CDR Specification Framework.

2.6 Common Responses

The CDR components specify outputs and post-conditions that result from interacting through a defined component interface. The components also define faults that result if expected conditions do not occur or do not hold. In the abstract, there is a commonality of these responses among the CDR components. For example, a security fault is a type of fault that will be realized by all components; however, the uniformity of the response is among the components is not specified.

The Specification Framework does not provide guidance for standard confirmations or status responses that a typical Consumer Component may find useful in gauging the progress of their requests towards the defined outputs and post-conditions. Such Guidance may be developed in future iterations of the CDR Specification Framework.

3 Search Component

3.1 Component Overview

The Search Component 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 content consumers to discover relevant content resources from disparate collections across the IC/DoD Enterprise. Specifically, the Search Component provides a means to accept a well-defined 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.

3.2 Component Scope

The following concepts are NOT included in the current draft; however, the exclusion of these concepts in no way judges the potential applicability of the items below or their inclusion in the future:

- Routing search requests beyond a basic request-response paradigm.
- The de-duplication of results returned from a given Search Component
- Batching multiple queries to a Search Component.
- Invoking a Search Component via multiple simultaneous query representations (e.g., keyword and XQuery, metadata or SPARQL).
- The publication of or subscription to query results from a given Search Component.

3.3 Component Behavior

The Search Component comprises three activities that underpin Content Discovery capabilities: search, result presentation, and results paging. It is important to note that a Search Component generally does not return the actual content resources, but rather metadata[13] about the content resources, in the form of search results, contained in the response.

Additionally, the Search function's results provide the information needed by the CDR RA's Retrieve Component to retrieve or otherwise use a resource. As described below, this may be in the form of information common to all results (see Result Set Retrieval Properties) or it may be information associated with a single result (see Result Retrieval Properties). If both methods are utilized in a single results set, the information associated with a single result MUST be considered an override of the common information. In particular, if the common information applies to many but not all of the results in a result set, then the results for which the common information does not apply SHOULD contain

[13] *In the context of Search, resource metadata generally refers to a* subset *of a resource's available metadata, not the entire underlying record. The Search Component generally 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.*

individual information that defines retrieval of that result. This specification does not state a preference for either method – thus, both are OPTIONAL – but retrieval information SHOULD be available through at least one.

From the perspective of information exchanges relevant to this specification framework, these activities provide the following:

- Search passes a recognizable, appropriately formatted query to a search capability and that generates search results in response to the query.
- Results presentation takes the search results, applies appropriate formatting, and returns the formatted search results to the search consumer. The presentation may include information about the search that provides context to the results, such as displaying the query that was executed or providing a timestamp of when the search was executed. Results presentation will provide a default sorting of results and may also sort the results per consumer instructions. The results and their presentation may be subject to other processing to implement business logic such as restricting results based on identity, but the details of such processing are out of scope.
- Results paging supports sequentially providing slices of resource metadata returned as results to the search consumer. Paging may respond to specific consumer instructions, e.g., using a specified starting point in a list of results, or may respond with a default next or previous slice.

This specification does not specify the processes involved in each activity, only the overall behavior that provides a context for understanding the effects of performing a search. For example, the search may be based on exact matches to the criteria in the query or may provide approximate matches based on fuzzy searches. The results paging may be implemented through a caching mechanism or a re-executing of the query, and may not guarantee continuity of search results while switching pages.

While the activities as described are sequential in nature, this specification does not preclude an overlap in processing by an implementation. For example, an implementation may begin formatting results before all results are returned from the search activity. Additionally, some degree of formatting and/or sorting will likely occur when the search results are generated.

3.4 Interface Model

The Search Component comprises two functions: Search and Results Paging. All Search Component implementations MUST implement the Search function; the Results Paging function is OPTIONAL.

Table 2 - Search Component Functions

Function Name	Required/Optional
Search	Required
Results Paging	Optional

3.4.1 Search Function (required)

3.4.1.1 Preconditions

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

- A Policy Decision Point has determined, based on the requestor's identity and authentication of that identity ("authenticated identity"), that the consumer is authorized to issue the search request, and a Policy Enforcement Point allows Search to be invoked for the consumer.
- A Policy Decision Point has determined, based on subject and object attributes that the consumer is authorized to query the specified content resource, and a Policy Enforcement Point allows the query to be processed.

3.4.1.2 Input

Table 3 - Search Function Inputs

Input Name	Required/Optional
Query	Required
Query Properties	Optional
Search Properties	Optional

Query – The query to execute. The framework does not mandate specific query language syntax. All CDR Search Specifications MUST include a Query input.

Query Properties – Identified properties, e.g. query language, that enable determination of whether a search capability can process the query; may specifically be used as part of Source Identification (see section 4.4.2.).

Query Properties replaces the previously separate inputs of Query Type and Query Metadata. There was not previously adequate distinction in the information that could be considered for the separate inputs, and the single input provides sufficient opportunity to attach information to assist at characterizing the query.

Search Properties – Information provided by the search requestor to specify and configure optional behavior supported by the Search Component implementation. The following query properties are commonly used in practice and serve to illustrate the utility of search properties: Example elements include but are not limited to Timeout, Result Metadata Format, Result Sorting Order, Results per page, and Start index.

> **Timeout** – The maximum time to take performing the search before results and/or a fault MUST be returned. While similar timeout functionality may be implemented by Service Level Agreements (SLAs) or the underlying transport protocol, this timeout value tells the provider how long they have to answer the query. This is especially useful for providers with slow or distributed underlying content repositories. CDR Search Specifications MAY choose not to support timeout or MAY define a default timeout. (Deprecated as individual input element).

Result Metadata Format – The metadata format to use for results. CDR Search Specifications MAY choose to define a single result metadata format, in which case this input would not be used. CDR Search Specifications that do use this input MAY define a default result metadata format. (Deprecated as individual input element).

Result Sorting Order – The order to sort the results. CDR Search Specifications MAY define a single result sorting order (e.g. relevancy). CDR Search Specifications that do use this input MAY define a default sorting order. (Deprecated as individual input element).

Results Per Page – The maximum number of results to return in the Search output. CDR Search Specifications that do not support this field MUST define a default value. (Deprecated as individual input element).

Start Index – The number of the first result to return. CDR Search Specifications that do not support this field MUST define a default value. Note: Start index must be between 1 and Total Result Count. The results that are returned for any paging request will be those between (Start index) and (Start index + Results per page - 1). This specification does not define the concepts of next page or previous page. These may be included in search service specifications that derive from this Specification Framework. (Deprecated as individual input element).

3.4.1.3 Output

Table 4 - Search Function Outputs

Output Name	Required/Optional
Result Set	Required

Figure 3 reflects the compositional hierarchical that exists between a Result Set and Results. This distinction is useful when processing search results.

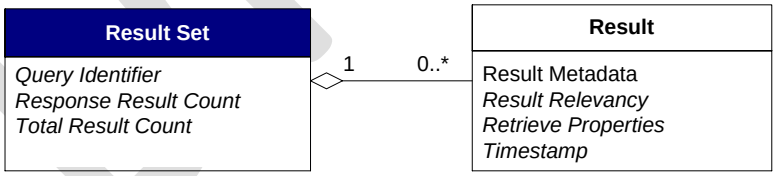

Figure 3 - Result Set Class Diagram

The single output of the Search Function is the Result Set.

Result Set – Required. The results of the input query; see the definition in Section 2.3. All CDR Search Specifications MUST contain a Result Set output. The Result Set MUST include 0 or more Results.

As part of the Result Set, the output MUST contain the REQUIRED elements that follow and MAY contain the RECOMMENDED and OPTIONAL elements.

Results Metadata – Required. A set of resource metadata properties describing the Result. CDR Search Specifications SHOULD support well-defined and community-accepted metadata formats when possible. CDR Search Specifications MAY define a required subset of properties if desired.

Result Relevancy – Optional. The relevancy of a Result in satisfying the criteria in the query. The framework does not provide any specific guidance on relevancy algorithms.

Retrieve Properties – Optional. A set of properties describing how to retrieve the associated Result.

Timestamp – Recommended. The time that the provider executed the query. This value can be used by Search consumers to determine the "freshness" of the results.

Query Identifier – Optional. A unique identifier for the query request that enables Results Paging. CDR Search Specifications MAY choose not to support this output.

Response Result Count – Recommended. The number of results in the Result Set that is currently being returned. The value MUST be less than or equal to the Results per Page input if the input is specified.

Total Result Count – Recommended. The total number of results that the provider has for the query.

3.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the search function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- Results correspond to defined outputs.
- Only the Search Results that the consumer is authorized to access will be passed back to the consumer. Authorization will be determined based on the consumer's attributes and the Search Result's Information Assurance metadata.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Search function or activities within the function.

3.4.2 Results Paging Function (optional)

3.4.2.1 Preconditions

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

- A valid query ID was generated during a previous search and the collection receiving the paging request recognizes and can respond for this query ID.

3.4.2.2 Input

Table 5 - Results Paging Function Inputs

Input Name	Required/Optional
Query identifier	Optional
Results per page	Optional
Start index	Optional

Query Identifier – An identifier for the query as defined in Section 3.4.1.3. This value MUST correspond to a "Query Identifier" output from a previously performed Search or Results Paging function, if applicable.

Results Per Page – As defined in Section 3.4.1.2.

Start Index – As defined in Section 3.4.1.2.

3.4.2.3 Output

Table 6 - Results Paging Function Outputs

Output Name	Required/Optional
Result Set	Required
Result Metadata	Required
Result Set Retrieval Properties	Optional
Result Relevancy	Optional
Result Retrieval Properties	Optional
Timestamp	Recommended
Query Identifier	Optional
Response Result Count	Recommended
Total Result Count	Recommended

Result Set – As defined in Section 3.4.1.3.

Result Metadata– As defined in Section 3.4.1.3.

Result Set Retrieval Properties – As defined in Section 3.4.1.3.

Result Relevancy – As defined in Section 3.4.1.3.

Result Retrieval Properties – As defined in Section 3.4.1.3.

Timestamp – As defined in Section 3.4.1.3.

Query Identifier – As defined in Section 3.4.1.3.

Response Result Count – As defined in Section 3.4.1.3.

Total Result Count – As defined in Section 3.4.1.3.

3.4.2.4 Post-conditions

The following post-conditions MUST be the end result of the Results Paging function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- Results correspond to defined outputs.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Search function or activities within the function.

3.4.3 Fault Conditions

Fault conditions as described below provide a brief description of faults that may occur during search component processing. Individual specifications SHOULD support these faults and MAY expand their description of fault conditions to address additional situations or provide additional detail on the fault.

Table 7 - Search Component Faults

Fault Name	Fault Description
Query Type Not Supported	A fault used when the provider does not support the type of query specified by the Query Type input parameter.
Invalid Query Syntax	A fault used when the contents of the Query input parameter are not valid.
Query Term Not Supported	A fault used if the query type is supported and the query conforms to the defined query syntax but a portion of the query is not supported by the provider.
Query Timeout	A fault used when the query cannot be executed in the amount of time specified by the Timeout input parameter.
Query Execution Fault	A fault used when an error occurs during query execution.
Query Metadata Fault	A fault used if the query metadata is not understood or contains an error. 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.
Security Fault	A fault used if the consumer is not authenticated or is not authorized to use the Search function.
Invalid Paging Value Fault	A fault used if a paging-related input parameter (e.g., start index, results per page) is not recognizable.

Out Of Range Fault	A fault used if the "Start Index" input if greater than the total number of results from the query.
Result Sorting Not Supported	A fault used when the provider does not support the result sorting mechanism specified by the Result Sorting input parameter.
Result Format Not Supported	A fault used when the provider does not support the result format specified by the Result Metadata Format input parameter.

606
607

4 Brokered Search Component

4.1 Component Overview

The Brokered Search Component serves as the primary mechanism to (1) facilitate the distribution of queries to multiple content collections and (2) aggregate the results returned individually into a single, uniform results set. A secondary mechanism is the identification of appropriate content collections to which a given query is to be distributed.

Inasmuch as the distributing of search requests is an internal activity of a Brokered Search Component implementation (see Section 4.3.3), these content collections are NOT REQUIRED to be accessed through use of CDR Search Components, but use of the CDR Search Component is RECOMMENDED. The remainder of Section 0 will discuss behavior in terms of the CDR Search Component where this facilitates ease and clarity of explanation.

4.2 Component Scope

The following concepts are NOT included in the current draft; however, the exclusion of these concepts in no way judges the potential applicability of the items below or their inclusion in the future:

- Mediation/Translation functions may be invoked by the Brokered Search Component but the specifics of how such functions are accomplished is outside the current scope.
- A Brokered Search Component federating to another Brokered Search Component raises the danger of infinite recursion, e.g., Broker A federates a search to Broker B which federates the search to Broker C, and Broker C federates to Broker A because it does not realize Broker A was already part of the search path. While a number of approaches may apply to avoid this problem, these will not be elaborated at this time.

4.3 Component Behavior

The Brokered Search Component comprises four activities that underpin Brokered Search capabilities: Brokered Search coordination, source identification, search component invocation, and federated search results processing. The coordination activity receives the search request and then invokes the source identification, search component invocation, and federated results processing in turn.

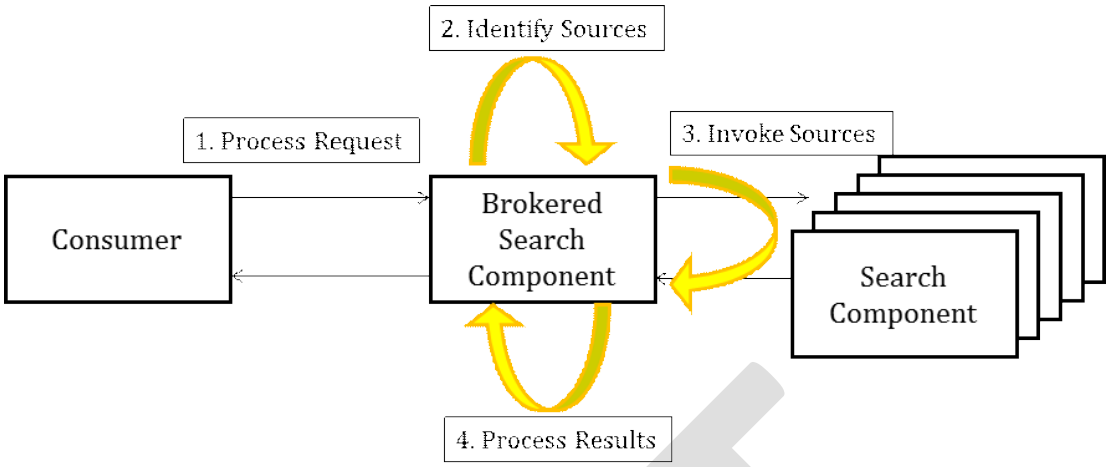

Figure 4 - Brokered Search Activity Flow

To satisfy the basic federated search use case, a Consumer Component would submit a search request to a Brokered Search Component that would then distribute the search to the applicable Search Components. The Brokered Search Component would conclude by compiling a list of search results[14] returned by the Search Components and process the results for return to the Consumer. An implementation of the Brokered Search Component may support results return by:

- Merging the results from all the Search Components into a consolidated set and then deliver the set to the originating consumer, or
- Returning the search results incrementally back to a consumer in grouped subsets as these become available from each Search Component.

The Consumer MAY use the Brokered Search Properties to express a preference for which return method to use; the Brokered Search Component will abide by the request if the method is available or will return a fault it the requested return method is not supported.

The following subsections provide additional information on behavior within each activity but do not specify a preferred mechanism by which the behavior is to be implemented.

4.3.1 Brokered Search Coordination Activity

The Brokered Search Coordination activity is the primary entry point to the Brokered Search function and provides coordination of the other activities that identify, invoke, and process results from the federation targets. In addition to managing internal communications among the activities, the Brokered Search Coordination activity MUST manage individual federation target invocations and respond to information exchanges with the federation targets. It may also be the point of invoking mediation to enable a larger number of targets to participate.

[14] *Results should be encoded in documented format, e.g., the IC/DoD supported format of Atom augmented with DDMS.*

4.3.2 Source Identification Activity

The Source Identification activity identifies Content Collection Components and the associated Search Components to which a search request will be distributed. The target Search Components may be identified through any combination of the following mechanisms:

(1) Chosen from an internal list (where the current specification does not detail how that list is created or maintained).
(2) Provided as input by the Consumer.
(3) Selected based on criteria specified by the Consumer and/or gathered through inspection of the query.

Additional information required from the Consumer to make use of the second mechanism SHOULD be provided through the Brokered Search Properties. The third mechanism assumes a search of a collection of content collection descriptions and is defined in terms of the CDR Search Component (see Section 0). As with other uses of search, the third mechanism may make use of translation/mediation capabilities, either internal or external, to interpret the search request and increase the number of content collections that can process the search request.

The Source Identification activity may make use of more than one collection of Content Collection or Search Component descriptions for any of the listed mechanisms. The Describe Component (Section 5) includes relevant discussion on description creation, maintenance, and use.

4.3.3 Search Component Invocation Activity

The Brokered Search Component will execute an instance of the Search Component Invocation activity for each federate target. This includes responding to requests and faults that may be sent by any federate target.

4.3.4 Federation Results Processing Activity

The Federation Results Processing activity performs the required processing necessary to combine individual Search Component Invocation activity outputs into a single uniform results set per any processing instructions provided in the Brokered Search Properties. The processing MAY include but is not limited to:

- Converging or normalizing results sets from each federation target, including format translation.
- Eliminating duplicates.
- Calculating a consistent relevance ranking.
- Caching results for further processing.
- Results paging.
- Providing incremental feedback to the requesting Consumer Component
- Responding with faults, as needed.

4.4 Interface Model

The Brokered Search Component comprises two functions: Brokered Search and Source Identification. All Brokered Search Component implementations MUST implement the Brokered Search function; the Source Identification function is OPTIONAL.

Table 8 - Brokered Search Component Functions

Function Name	Required/Optional
Brokered Search	Required
Source Identification	Optional

4.4.1 Brokered Search Function (required)

4.4.1.1 Preconditions

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

- A Policy Decision Point has determined, based on the requestor's identity and authentication of that identity ("authenticated identity"), that the consumer is authorized to issue the search request, and a Policy Enforcement Point allows Brokered Search to be invoked for the consumer.
- A Policy Decision Point has determined, based on subject and object attributes that the consumer is authorized to query the specified content resources, and a Policy Enforcement Point allows the query to be processed.
- The Consumer Component request must be consistent with the means the Brokered Search function has available for resolving source identification.

4.4.1.2 Input

Table 9 - Brokered Search Coordination Function Inputs

Input Name	Required/Optional
Search Function Inputs	Required/Optional per Table 3
Brokered Search Properties	Optional

Search Function Inputs – The Brokered Search Component search inputs conform to the Search Component inputs described in Section 3.4.1.1. These inputs may be used directly or processed as necessary for use in the Brokered Search Coordination function.

Brokered Search Properties – Properties that configure activities of the Brokered Search Component. These properties MAY include a Consumer-selected list of sources, maximum time to wait for each source (timeout), and requested result format (e.g., merged or grouped) for the Federation Results Processing activity.

As part of its input, the Brokered Search Component MAY also receive input from the Consumer Component for configuring the presentation of results. This input MAY be retained by the Brokered Search Component as configuration for final processing when the Brokered Search activities are complete and results are returned to the Consumer Component.

4.4.1.3 Output

Table 10 - Brokered Search Coordination Function Outputs

Output Name	Required/Optional
Merged Results Set and associated output	Required/Optional per the description of the Results Set data construct (Figure 3)
Total Results by Source	Optional

Merged Results Set and Associated Output – Results set after merging of results sets of individual Search Component Invocations and other federation results processing as indicated through Brokered Search Properties. This output will include required and optional outputs as indicated in Table 4 and available from each invocation.

Total Results By Source – The total number of results returned by a search request for a given source.

4.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the brokered search function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- Results correspond to defined outputs.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Brokered Search function or activities within the function.

4.4.2 Source Identification Function (optional)

4.4.2.1 Preconditions

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

- The consumer is already authorized to invoke Brokered Search.
- The consumer's security attributes are available for a Policy Decision Point and a Policy Enforcement Point to ensure that the Search Request only goes to content resources that the consumer is authorized to find.
- Searchable collection of content collection descriptions must be available.

4.4.2.2 Input

Table 11 - Source Identification Function Inputs

Input Name	Required/Optional
Search Function Inputs	Required/Optional per Table 3

Search Function Inputs – The Source Identification function inputs conform to the Search Component inputs described in Section 3.4.1.1. The query is constructed by the Brokered Search component based on the original Consumer query and information provided in the Brokered Search Properties.

4.4.2.3 Output

Table 12 - Source Identification Function Outputs

Output Name	Required/Optional
Identified Content Collections and access (as Results Set of Search Function Outputs)	Required

Identified Content Collections and Access – A set of identifiers and/or access mechanisms for Content Collections that are to act as federation targets.

The output of most interest from this function is the list of identified content collections to be used as federation targets. This information MUST be conveyed in the Results Set outputs are described in Section 3.4.1.3.

4.4.2.4 Post-conditions

The following post-conditions MUST be the end result of the source identification function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- A Policy Decision Point has determined, based on subject and object attributes that the consumer is authorized to query the identified content resources, and a Policy Enforcement Point allows the query to be sent to the appropriate Search component.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Brokered Search function or activities within the function.
- Results correspond to defined outputs.

4.4.3 Fault Conditions

Fault conditions as described below provide a brief description of faults that may occur during brokered search component processing. Individual specifications SHOULD support these faults and MAY expand their description of fault conditions to address additional situations or provide additional detail on the fault. The faults explicitly listed here are in addition to the faults described in Section 3.4.3.

Table 13 - Brokered Search Component Faults

Fault Name	Fault Description
Unsupported Required Brokered Search Properties Fault	A fault used if a required property of the Brokered Search Properties is not supported. This fault will terminate execution without proceeding to the other Brokered Search Component activities.

Unsupported Optional Brokered Search Properties Fault	A fault used if an optional element of the Brokered Search Properties is not supported. A Brokered Search Component MAY choose to continue the execution of the query but SHOULD provide some indication of the fault in the output's Result Metadata Properties.
Brokered Search Properties Error Fault	A fault used if an element of the Brokered Search Properties is supported but contains an error in its representation. In the case of a required element, the fault will terminate execution without proceeding to the other Brokered Search Component activities. In the case of an optional element, a Brokered Search Component MAY choose to continue the execution of the query but SHOULD provide some indication of the fault in the output's Result Metadata Properties.
Results Return Method Preference Fault	A fault used if the requested return method, e.g. a single merged results or incrementally grouped results, is not supported by the Brokered Search Component. This fault will terminate execution without proceeding to the other Brokered Search Component activities.
Source Invocation Fault	A fault used if the Brokered Search Coordination activity cannot invoke, monitor, or process results from the Source Identification activity.
Search Component Invocation Fault	A fault used if the Brokered Search Coordination activity cannot invoke, monitor, or process results from any Search Component Invocation activity. This fault MUST clearly identify which invocation manifested the problems.
Federated Results Processing Fault	A fault used if the Brokered Search Coordination activity cannot invoke, monitor, or process results from the Federated Results Processing activity.
Invocation Results Set fault	A fault used when federation results processing cannot process the results set of an individual Search Component Invocation. This indicates an error in the returned results set or an inconsistency in interpreting the results set specified format.
Invocation Results Optional Output fault	A fault used when federation results processing cannot process optional outputs as described in Table 4 for an individual Search Component Invocation.
Results Return Processing fault	A fault used when federation results processing supports the specified results return method, e.g. merged or incremental, but fails to cannot complete processing successfully.

802

5 Describe Component

5.1 Component Overview

The Describe Component serves as the primary mechanism for providers to expose information describing their resources. The resulting description may be used for numerous purposes, such as discovering that a resource exists, identifying policies applied to the resource, or finding mechanisms through which a content resource can be retrieved. In general, a description tells an interested party *what the resource is* and *how it can be accessed or used.*

The CDR RA introduces a broad set of components whose ultimate implementations and context will vary within a heterogeneous enterprise. Descriptions are used to explicitly characterize CDR-related resources, including CDR component implementations, and give resource providers the ability to reflect both the static and dynamic aspects of a resource. Static aspects may include the name and summary description of the component and the information model used by the component. Dynamic aspects change on a frequent basis and may describe concepts in terms of data, performance, or other fungible classifications. For instance, a consumer may only be interested in content collections that contain data less than 24 hours old. The Describe Component description model can be used to characterize both the static and dynamic aspects of a resource.

5.2 Component Scope

The following concepts are NOT included in the current draft; however, the exclusion of these concepts in no way judges the potential applicability of the items below or their inclusion in the future:

- Description Recipients.
- Alternative Description Access Mechanisms.
- Specific property sets and their corresponding property values.
- Specifics of using Deliver Component to provide description to Description Recipient.

The Describe Component is general enough to be applied to any resource. This Section will focus on the resources that are directly applicable to the CDR RA, including but not limited to, the following: Content Collections, Search Components, and Retrieve Components.

5.3 Component Behavior

The Describe Component enables the creation and maintenance of a consistent representation of description that supports multiple description uses in a uniform manner. The fundamental output of the Describe Component is *a metadata construct*, referred to as a Description.

5.3.1 Description

A description of a given resource comprises one or more description statements, as indicated below in Figure 5.

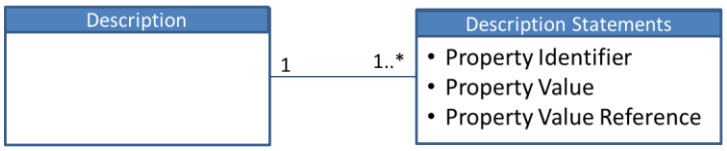

Figure 5 - Describe Logical Data Model

The Description Statement includes *the property and associated value* that convey individual static or dynamic informational concept. It is composed of a Property Identifier, Property Value, and a Property Value Reference, and this information constitutes the input to the Describe function.

Property Identifier – The common name of a property used to describe a resource. Property Identifiers MUST have an associated definition. This definition MAY include guidance on assigning values to the Property. The Property Identifier SHOULD be chosen to be applicable across communities where a description with this Property may be used.

Property Value – The value or set of values assigned to a Property Identifier.

Property Value Reference – An optional element that SHOULD be used to define the semantics or other source descriptions of the values assigned as Property Values. Multiple Property Value References MAY be identified for any Property Value. In addition, separate Property Value References MAY be identified for each Property Value when more than one Property Value is assigned to a Property Identifier.

5.3.2 Description Process

Figure 6 shows the actors and the basic process that generates description.

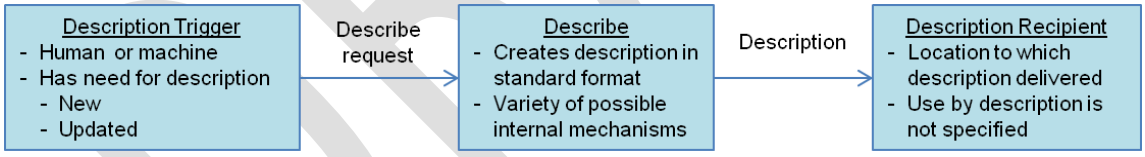

Figure 6 - Describe Process

The Description Information encompasses the information that constitutes the Description as shown in Figure 5**.** This specification does not prescribe a preferred concrete data model, and other documents will elaborate on recommended description content for specific resources. However, the specification of the elements of a description and the relationships among these elements is meant to provide a consistent structure upon which the description of any resource class can be specified. It is expected that this model will be the basis for specific describe specifications and implementation guides.[15]

[15] *The Service Description Framework (SDF) and Content Collection Description Framework (CCDF), available on the unclassified Intelink Web Site, provide examples of recommended description content tailored to a specific use of properties.*

5.3.2.1 Description Trigger

The Description Trigger is a Consumer Component that initiates using a Describe Component implementation to create or update a Description. The trigger may be a human or machine provider of information from which the Description will be created or updated. Alternately, the trigger may be an automated response to an event (e.g., a notification) or an automated periodic refresh of existing descriptions. This specification does not impose any limits on what may constitute the Description Trigger.

5.3.2.2 Describe

In the context of Figure 6 and as elaborated below, the Describe Component accepts Description Information as input and generates a Description that it provides to the Description Recipient. A Describe Component implementation may encompass a wide range of activities depending on the composition and form of the Description Information and the complexity of getting the description to the Description Recipient. The following are examples of activities which a Describe Component implementation may perform; however, this list is meant to be illustrative and neither requires nor restricts actual implementations.

1. In the simplest case, a new description is to be created using Description Information that conforms to the logical model in Figure 5, and the created Description is to be sent to a single, known recipient. In this case, a Describe Component implementation may validate the input representation is correct and send it to the recipient.

2. The Describe Component implementation may provide a Graphical User Interface (GUI) or other interface through which it receives Description Information and will transform the input into the required representation. The input may be values to associate with identified Property Identifiers or may be information sources that the Describe Component implementation is prepared to use for directly accessing or invoking other processes to create such values.

3. The Describe Component implementation is to provide the description to a Description Recipient for which the delivery requires special processing which may be better handled using an appropriate Deliver Component implementation. In this case, the Describe Component implementation will act as the Initiating Consumer of the Deliver Component.

5.3.2.3 Description Recipient

The Description Recipient refers to the location to which descriptions are delivered and the entity at that location that is prepared to accept and invoke necessary local activities related to the description. The following are examples of how the Description Recipient may process or use the description; however, this list is meant to be illustrative and not limit the actual recipient.

1. The Description Recipient may be a searchable collection where numerous descriptions are stored.

2. The Description Recipient may not support direct search but may be used as a source from which a separate searchable collection can aggregate descriptions as its content.

3. The Description Recipient also acts as the Description Trigger and triggers the creation of the most recent description for immediate local use without any long term storage.

If the output of the Describe Component is to be used as an update of an existing description, it is up to the Description Recipient to process or otherwise use the updated description as appropriate for their processes and needs. For example, the Description Recipient may simply overwrite the existing description; alternately, the Description Recipient may archive the existing description and identify the updated description as the current one.

Although the specifics of the Description Recipient are beyond the scope of the Describe Component, what is essential to the overall describe capability is that *descriptions must be available to support resource discovery and resource use.*

5.4 Interface Model

The Describe function is the single function of the Describe component. All Describe Component implementations MUST implement the Describe function.

Table 14 - Describe Component Functions

Function Name	Required/Optional
Describe	Required

5.4.1 Describe Function (required)

5.4.1.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity"), that they are authorized to provide a description and a Policy Enforcement Point allows Describe to be invoked for the provider.
- The provider has determined the security attributes associated with the content resource and its metadata.
- Property Identifiers should be identified for a resource class of interest.
- Controlled vocabularies should exist and be available as sources of Property Values and as the indicated Property Value Reference.
- Other source material, such as policies, requirements statement, etc., should be available to be identified as Property Value References.

5.4.1.2 Input

Table 15 - Describe Function Inputs

Input Name	Required/Optional
Description Information	Required

Description Information – This includes any information needed by the Describe Component to generate a Description and transfer the resulting Description to the Description Recipient. An example of one item in the Description Information could be the location(s) of the Description Recipient(s).

The Description Information may consist of values provided by the Description Trigger to be associated with indicated Property Identifiers or may identify information sources from which such values can be accessed or derived. In addition, if a Describe Component implementation is being used to update an existing description, the Description Information MUST include an identifier or other unique information that allows the Describe Component implementation to retrieve the existing description. The identifier or other retrieval information may be specified through any number of means, to include but not be limited to: being known by the Description Trigger, being accessible from a list of previously created descriptions (possibly maintained by a Describe Component implementation), or being included as part of the results set of a search.

5.4.1.3 Output

Table 16 - Describe Function Outputs

Output Name	Required/Optional
Description	Required

Description – Description of a resource conforming to the Logical Data Model as prescribed in (Section 5.3.1).

5.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the describe function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- The security attributes of the content resource, its metadata and its environment are stored in a database to support a Policy Decision Point used to control access to the content or its metadata.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Describe function or activities within the function.
- Description generated and resident with identified Description Recipient(s).

5.4.2 Fault Conditions

Fault conditions as described below provide a brief description of faults that may occur during describe component processing. Individual specifications SHOULD support these

997 faults and MAY expand their description of fault conditions to address additional
998 situations or provide additional detail on the fault.

999 **Table 17 - Describe Component Faults**

Fault Name	Fault Description
Insufficient Description Information	This fault occurs if the Description Information is insufficient for the Describe Component to generate a description. This fault should indicate which of the required input was not supplied.
Invalid Property	This fault occurs when a property that has not been defined in a vocabulary is added
Invalid Property Value	This fault occurs when a property value is found to be invalid per the vocabulary.
Invalid Property Value Reference	This fault occurs when a property value reference is found to be invalid.

1000

1001

6 Query Management Component

6.1 Component Overview

The Query Management (QM) Component serves as the primary mechanism to enable:

- Data management (Create, Read, Update, Delete) associated with a persistent collection, referred to as a QM Collection of Saved Searches[16].
- Searching the QM Collection.
- Execution of Saved Searches.

This Section is focused on the interface to this collection of QM functionality and makes no assertions regarding component implementation.

6.2 Component Scope

The following concepts are **NOT** included in the current version; however, the exclusion of these concepts in no way judges the potential applicability of the items below or their inclusion in the future:

- The alerting pattern of QM.
- Transactional aspects and versioning of the Saved Search.
- Support for viewing Saved Search Execution History.
- Management of Search Results.
- External Subscribers that are not managed by the QM Component.[17]
- For updates, providing only fields to be changed rather than entire Search Request.

6.3 Component Behavior

The QM Component comprises three activities that underpin Content Discovery capabilities for Saved Searches: manage, search, and execute. The QM Manage Activity provides mechanisms to manage the search requests that may be used to discover relevant content resources, the QM Search Activity provides a mechanism to investigate the contents of the QM Collection, and the QM Execute Activity provides a mechanism to use Saved Searches as future search requests. The activities leverage the resource structure shown in Figure 7 to identify the parts and relationships of the information resources associated with a Saved Search.

The Saved Search Bundle shown in Figure 7 is the information resource stored in the QM Collection and includes a search request and the target search capability (where the search is to be executed). The search request conforms to the inputs for the Search Function as defined in Section 3.4.1.2. Each Search Bundle has an associated Saved Search ID which is used to reference the Saved Search Bundle within a QM Collection. The Description shown in Figure 7 is that characteristic metadata that can make the Saved Search discoverable.

---

16 *Search request, Saved Search, and other terms related to search are defined in Section 2.3.*

17 *"External Subscribers" refers to Consumer Components that make requests of the Query Management Component to execute a Saved Search at regular intervals. This kind of subscription makes no additional demands on the interface of the Query Management Component.*

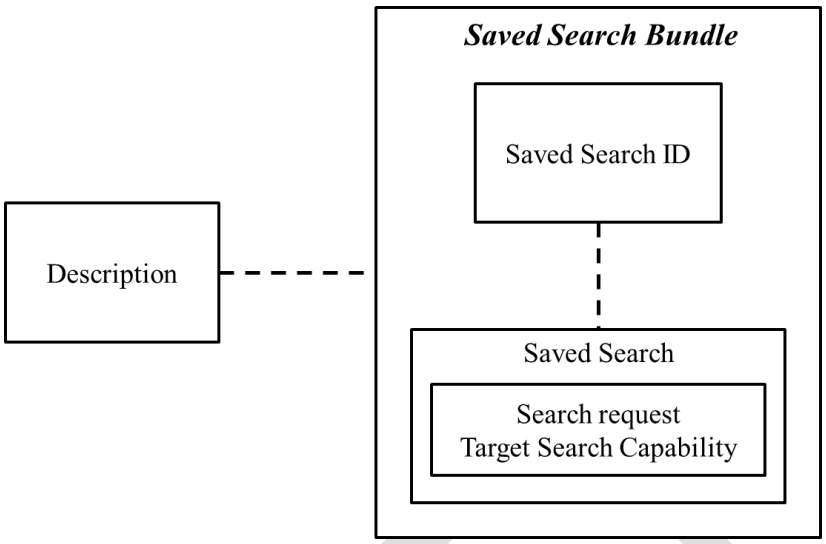

Figure 7 - Saved Search Bundle

The QM Manage Activity is realized through four functions: QM-Create, QM-Read, QM-Update, and QM-Delete. While these correspond to the traditional create, read, update, and delete (CRUD) functions, the Saved Search analogs should not be confused with proprietary CRUD functions that correspond to a particular implementation of the QM Collection. Rather, the QM CRUD functions are tailored to provide generic interfaces specifically to manage Saved Searches, and the relationships of QM CRUD functions to proprietary collection interfaces are not defined in this specification. While the QM CRUD functions may be generalized in the future, such generalization is beyond the current scope.

The QM Search Activity enables a prospective consumer to inspect the QM Collection for Saved Searches of interest. It leverages the CDR Search Component but the query would be in terms focused on the Saved Search, such as the Description as shown in Figure 7. In addition, the target search capability would be directed to a QM Collection and the results would be metadata corresponding to Saved Searches. In its simplest form, the query can be one of a set of predefined list requests, such as list Saved Searches identified with a particular owner.

The QM Execute Activity enables a search consumer to execute a Saved Search. It leverages a modified version of the CDR Search Function, substituting the Saved Search ID for the query in the Search Function inputs. This requires the identified Saved Search to be retrieved from the QM Collection or the corresponding search request to be otherwise derived from the identifier information so it can be used by the target search capability.

By defining the Saved Search in terms of the CDR Search inputs, the Saved Search can be used directly with any compatible target search capability that supports the CDR Search interface. This lessens the processing required of the QM Component. In addition, the corresponding inputs of a QM Execute MAY be used as overrides for values in the Saved Search.

A Saved Search becomes a Persistent Search when it is associated with information that controls when it is to be executed. Execution is caused by a trigger, where the trigger may be manual, time-based (e.g. every 3 days), or event-based (e.g. after 100 changes to a content collection). The Saved Search may be executed any number of times as a Persistent Search, or it may simply be saved as documentation with no intent to execute.

The search capability acting as the target of the search SHOULD NOT be able to differentiate between a Saved Search and an identical search generated as a new search by a consumer. Conversely, the consumer receiving a response SHOULD NOT be able to differentiate between the response resulting from a Saved Search and the response resulting from a new search by a consumer. While metadata returned with the results based on a Saved Search may carry additional information that explicitly refers to the Saved Search, there SHOULD NOT be anything inherently different between a new search and a Saved Search.

6.4 Interface Model

The QM Component comprises the six functions shown in Table 18.

Table 18 - Query Management Interface Functions

Function Name	Required/Optional
QM-Create	Required
QM-Read	Required
QM-Update	Required
QM-Delete	Required
QM-Execute	Required
QM-Search	Optional

6.4.1 QM-Create Function (required)

The QM-Create function enables the user to construct a Saved Search. In response to a QM-Create request, the QM component implementation will assign a Resource Identifier referred to as a Saved Search ID.

6.4.1.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity"), that they are authorized to create a Saved Search Bundle and a Policy Enforcement Point allows QM-Create to be invoked for the provider.
- The QM Collection is known/identified.

6.4.1.2 Input

Table 19 - QM-Create Function Inputs

Input Name	Required/Optional
Search Request	Required
Target Search Capability	Required
QM Properties	Optional

Search Request – Request that initiates a search (see Section 2.3). The search request MUST be composed of Search Function inputs as defined in Section 3.4.1.2.

Target Search Capability – Reference to the Search Component or Brokered Search Component implementation that is to process the search request. The reference MUST provide or be able to be transformed to an address through which a Consumer Component can later initiate a search by sending the search request to that address.

QM Properties – Information provided by the QM consumer to specify and configure optional behavior supported by the QM Component implementation.

6.4.1.3 Output

Table 20 – QM-Create Function Outputs

Output Name	Required/Optional
Saved Search ID	Required
Saved Search	Optional
Saved Search Description	Optional

Saved Search ID – A unique identifier provided by the QM Component that identifies the result of a QM-Create or QM-Update request.

Saved Search – Identified and managed search information as defined in Sections 2.3 and further described in 6.3.

Saved Search Description – Metadata that describes the Saved Search. The Saved Search Description MAY be returned in addition to the Saved Search ID in order to make the Consumer Component immediately aware of any data elements that were set by the QM Component, such as “date created” or other metadata.

6.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the search function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- Information describing the security policy regarding access to (CRUD, Search and Execute) for the Saved Search Bundle is stored in a Policy Decision Point database.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the QM function or activities within the function.

6.4.2 QM-Read Function

The QM-Read function SHOULD utilize the definition of the Retrieve Function described by the Retrieve Component. The Retrieve Function is used as a composable service definition to support reading Saved Search resources managed by a QM Component implementation. It does not execute the Saved Search, but MAY facilitate a scenario where a Consumer Component uses the information contained in the Saved Search to execute a search on a Search Component.

6.4.2.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity") that they are authorized to use QM-read and a Policy Enforcement Point allows QM-Read to be invoked for the provider.
- A Policy Decision Point has determined, based on subject and the Saved Search Bundle security attributes, that the consumer is authorized to read the Saved Search, and a Policy Enforcement Point allows the consumer to read the Saved Search Bundle.
- Saved Search is under management of QM and may be retrieved through reference to its Saved Search ID.

6.4.2.2 Input

Table 21 - QM-Read Function Inputs

Input Name	Required/Optional
Saved Search ID	Required
QM Properties	Optional

Saved Search ID – As defined in Section 3.4.1.3.

QM Properties – As defined in Section 3.4.1.2.

6.4.2.3 Output

Table 22 - QM-Read Function Outputs

Output Name	Required/Optional
Saved Search	Required
Saved Search Description	Optional

Saved Search – As defined in Section 3.4.1.3.

Saved Search Description – As defined in Section 3.4.1.3.

6.4.2.4 Post-conditions

- Saved Search Bundle is not affected by read.

- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the QM function or activities within the function.

6.4.3 QM-Update Function

The QM-Update function allows a consumer component to change a Saved Search. The Consumer Component sends an updated Search Request and/or Target Search Capability to the QM Component. This differs from the QM-Create Activity in that the Consumer Component MUST also send a valid Saved Search ID corresponding to a previously created Saved Search. The QM-Update function will replace Search Request and/or the Target Capability in the existing Saved Search to correspond to the update input. The Saved Search ID will remain the same.

6.4.3.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity") that they are authorized to use QM-Update and a Policy Enforcement Point allows QM-Update to be invoked for the provider.
- A Policy Decision Point has determined, based on subject and the Saved Search Bundle security attributes that the consumer is authorized to update the Saved Search, and a Policy Enforcement Point allows the consumer to update the Saved Search Bundle.
- Saved Search is under management of QM and may be accessed through reference to its Saved Search ID for purposes of update.

6.4.3.2 Input

Table 23 - QM-Update Function Inputs

Input Name	Required/Optional
Saved Search ID	Required
Saved Search	Required
QM Properties	Optional

Saved Search ID – As defined in Section 3.4.1.3.

Saved Search – As defined in Section 3.4.1.3.

QM Properties – As defined in Section 3.4.1.2.

6.4.3.3 Output

Table 24 - QM-Update Function Outputs

Output Name	Required/Optional
Saved Search ID	Required
Saved Search	Optional
Saved Search Description	Optional

Saved Search ID – As defined in Section 3.4.1.3.

Saved Search – As defined in Section 3.4.1.3.

Saved Search Description – As defined in Section 3.4.1.3.

6.4.3.4 Post-conditions

- Saved Search Bundle reflects specified updates.
- Saved Search Bundle is referenced by the Saved Search ID.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the QM function or activities within the function.
- If the update to the Saved Search changes its security attributes, the attributes are updated in the Policy Decision Point database.

6.4.3.5 Other Considerations

This specification does not define the process for updating the Saved Search. e.g. whether the Saved Search collection completely overwrites the existing Saved Search or internally maintains a history of past versions. While such a versioning strategy SHOULD be developed and made known to potential Consumer Component, the specifics of such a strategy are beyond the scope of this document.

6.4.4 QM-Delete Function

The QM-Delete function removes a Saved Search resource from the Saved Search collection managed by the QM Component.

6.4.4.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity") that they are authorized to use QM-Delete and a Policy Enforcement Point allows QM-Delete to be invoked for the provider.
- A Policy Decision Point has determined, based on subject and the Saved Search Bundle attributes that the consumer is authorized to delete the Saved Search Bundle, and a Policy Enforcement Point allows the consumer to delete the Saved Search Bundle.
- Saved Search is under management of QM and may be accessed through reference to its Saved Search ID for purposes of delete.

6.4.4.2 Input

Table 25 - QM-Delete Function Inputs

Input Name	Required/Optional
Saved Search ID	Required
QM Properties	Optional

Saved Search ID – As defined in Section 3.4.1.3.

QM Properties – As defined in Section 3.4.1.2.

6.4.4.3 Output

Table 26 - QM-Delete Function Outputs

Output Name	Required/Optional
Confirmation	Required

Confirmation – If the Delete Activity is successful, it MUST return a notification of a successful deletion.

6.4.4.4 Post-conditions

- Saved Search Bundle is no longer accessible by QM functions.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of QM function or activities within the function.
- The security attributes of the deleted Saved Search are removed from the Policy Decision Point database.

6.4.4.5 Other Considerations

There are a number of issues that relate to Delete that will be left to the Saved Search collection implementation and are considered out of scope for the current version.

- A Delete may be a “soft” Delete and only mark an item as deleted without removing it from the collection or a “hard” which is permanent and removes the deletion from the collection. A Delete may also be reversible in that for some time after a Saved Search has been designated for deletion, the designation can be removed. These are all left as option for the implementation.
- A Saved Search which has been soft deleted SHOULD be omitted by default from the response to a Search Saved Searches function request, but SHOULD be optionally accessible.
- Any verification to prevent accidental deletion, such as a GUI prompt asking the user to verify their intentions, is left to the Consumer Component.
- Management of external subscribers is out of scope for the QM Component and is left to the Consumer Component of the QM-Execute activity, so if the Delete function deletes a resource that is referenced by an external subscriber, the external subscriber must be able to recognize the fault condition and respond appropriately.

6.4.5 QM-Execute Function

The Execute function is accomplished through the use of CDR Search. Search request are sent to the Target Search Service which in turn executes the search request. The target search service may be a Search Component or Brokered Search Component.

6.4.5.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity") that they are authorized to use QM-Execute and a Policy Enforcement Point allows QM-Execute to be invoked for the provider.
- A Policy Decision Point has determined, based on subject and the Saved Search attributes that the consumer is authorized to execute the Saved Search Bundle, and a Policy Enforcement Point allows the consumer to execute the Saved Search Bundle.
- Saved Search is under management of QM and may be accessed through reference to its Saved Search ID for purposes of execute.

6.4.5.2 Input

Table 27 - QM-Execute Function Inputs

Input Name	Required/Optional
Modified Search Function Inputs	Required/Optional per Table 3, as modified below
Saved Search ID	Required (as replacement for Search Function query)
Search QM Properties	Optional

Modified Search Function Inputs – As defined in Section 3.4.1.2 except the REQUIRED Query is replaced by the Saved Search ID. Note, other than query, all Search Function inputs are OPTIONAL.

Saved Search ID – As defined in Section 3.4.1.3.

QM Properties – As defined in Section 3.4.1.2.

6.4.5.3 Output

Table 28 - QM-Execute Function Outputs

Output Name	Required/Optional
Search Results	Required

Search Results – Results of a search as defined in Section 2.3 and further discussed in Section 3.4.1.3.

6.4.5.4 Post-conditions

- Saved Search Bundle is not affected by its use as part of QM-Execute.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the QM function or activities within the function.

6.4.6 QM-Search Saved Search Function

The QM-Search function enables a Consumer Component to search the QM Collection. The use of CDR Search for this function provides a consistent, well-defined approach to discovering the data in a QM Collection. QM-Search provides flexible means to access Saved Searches. One use of QM-Search would be to 'list' all Saved Searches that correspond to a particular user.

6.4.6.1 Preconditions

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

- A Policy Decision Point has determined, based on the provider's identity and authentication of that identity ("authenticated identity") that they are authorized to use QM-Search Saved Searches and a Policy Enforcement Point allows QM-Search Saved Searches to be invoked for the provider.
- A Policy Decision Point has determined, based on subject and the Saved Search attributes that the consumer is authorized to search the QM collection, and a Policy Enforcement Point allows the consumer to search Saved Searches in the QM Collection.
- Collection must support search if search function is to be used.

6.4.6.2 Input

Table 29 - QM-Search Function Inputs

Input Name	Required/Optional
Search Function Inputs	Required/Optional per Table 3
QM Properties	Optional

Search Function Inputs – Inputs conforming to the Search Function inputs as defined in Section 3.4.1.1. These inputs may be used directly or processed as necessary for use in the QM-Search function.

QM Properties – As defined in Section 3.4.1.2.

Note that query is the only REQUIRED Search Function input and its content and format is not defined in Section 3.4.1.2. For use with QM-Search, the query could be a simple "list" command as defined by the QM Component implementation.

6.4.6.3 Output

Table 30 - QM-Search Function Outputs

Output Name	Required/Optional
Search Function Outputs	Required

Search Function Outputs – Outputs conforming to the Search Function outputs as defined in Section 3.4.1.3. The results set comprise results and associated metadata for identified Saved Searches in the QM Collection.

6.4.6.4 Post-conditions

- Saved Search Bundle is not affected by search Saved Search.
- Only the Saved Search Bundles that the consumer is authorized to access will be passed back to the consumer. Authorization will be determined based on the consumer's attributes and the access policy for the Saved Search Bundle.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the QM function or activities within the function.

6.4.7 Fault Conditions

Fault conditions as described below provide a brief description of the faults that may occur during QM Component processing. Individual specifications SHOULD support these faults and MAY expand their description of fault conditions to address additional situations or provide additional detail on the fault. The faults explicitly listed here are in addition to the faults related to the use of CDR Search/Brokered Search by the QM-Execute and QM-Search functions, and those faults are described in Sections 3.4.3 and 4.4.3.

Table 31- Query Management Faults

Fault Name	Fault Description
Validation Failure	The search request to be saved did not meet the validation criteria.
Resource Not Found	The Saved Search requested at that Saved Search ID was not found or has been deleted.
Resource Moved	The Saved Search requested at that Saved Search ID is now located elsewhere. The Query Management Component SHOULD return the new location.

6.5 Future Considerations

6.5.1 Persistent Search

Persistent Search allows a consumer to identify a Saved Search and associate a trigger that will cause the search request to be submitted to the target search capability. An example use of Persistent Search is to receive automatic updates whenever new content is available at the target search capability. Support for a Persistent Search capability will require the QM mechanism to include or work in collaboration with several additional,

non-trivial capabilities. Using the simple working definition of Persistent Search as defined in Section 2.3, it is possible to identify sets of functionality that may need to be included in future versions of this specification:

- **Subscription** - Enables consumers to express interest (subscribe) in information resources to which a consumer's Saved Search can be submitted.
- **Filtering** - Processes large volumes of information which can be suitable for the satisfaction of a consumer's needs.
- **Event Detection** - Defines the conditions associated with a trigger. Triggers serve to initiate work (processes) within the Persistent Search environment.
- **Alerts** - notifies subscribers when information of interest becomes available.

Identifying and coordinating the collaboration of the functional sets identified above will be an essential part of making full use of the QM features specified to this point.

7 Retrieve Component

7.1 Component Overview

The Retrieve Component serves as the primary content access mechanism. It encompasses the capability to retrieve an identified content resource from the Content Collection in which it is stored. The content resource may be identified in the result set returned from the Search Component or the retrieval information could be obtained elsewhere by the requester.

7.2 Component Scope

The Retrieve Component is general enough to be applied to an identified resource.

7.3 Component Behavior

The Retrieve Component provides a common interface and behavioral model for Intelligence Community (IC) and Department of Defense (DoD) content collections, enabling content consumers to retrieve content resources from disparate collections across the IC/DoD enterprise. Specifically, the Retrieve 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. Thus, it unambiguously conveys a request for the content without knowing or setting requirements on the implementation of the underlying content collection.

7.4 Interface Model

The Retrieve function is the single function of the Retrieve component. All Retrieve Component implementations MUST implement the Retrieve function.

Table 32 - Retrieve Component Functions

Function Name	Required/Optional
Retrieve	Required

7.4.1 Retrieve Function (required)

7.4.1.1 Preconditions

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

- Retrieve must include the invocation of security services responsible for authentication, authorization, IA metadata binding, access enforcement, and cross domain flow control to ensure authorized use of the retrieve function and protect retrieved content (See Section 9.1.).
- The resource to be retrieved must be accessible using the pattern described for the retrieve component.
- A Policy Decision Point has determined, based on the requestor's identity and authentication of that identity ("authenticated identity"), that the consumer is authorized to issue the retrieve request, and a Policy Enforcement Point allows Retrieve to be invoked for the consumer.

- A Policy Decision Point has determined, based on subject and object attributes that the consumer is authorized to retrieve the specified content resource, and a Policy Enforcement Point allows the retrieve request to be processed.

7.4.1.2 Input

Table 33 - Retrieve Function Inputs

Input Name	Required/Optional
Resource Identifier	Required
Retrieve Source	Required
Retrieve Properties	Optional

Resource Identifier - Identifies the content resource to be retrieved. The identifier MUST be unique within the data set. CDR Retrieve specifications MUST include an identifier.

Retrieve Source – Indicator of the Resource Collection or Search Component to which the retrieve request is directed. This may be part of the Resource Identifier.

Retrieve Properties - Information needed by the Retrieve Component to respond to the retrieve request. The specifics of the Retrieve Properties, including whether specific Retrieve Properties are mandatory or optional, are specified by the Service Specification of the Retrieve Component implementation. An example of Retrieve Properties could be content types the Consumer Component is prepared to receive.

7.4.1.3 Output

Table 34 - Retrieve Function Outputs

Output Name	Required/Optional
Resource	Required
Resource Metadata	Optional

Resource - Resource Content that has been retrieved and needs to be returned per the Consumer Component request.

Resource Metadata - Metadata about the Resource that is relevant to retrieving (and delivering) the Resource and may not be included in the Result Metadata.

7.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the retrieve function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- Results correspond to defined outputs.
- Only the Content Resource that the consumer is authorized to access will be passed back to the consumer. Authorization will be determined based on the consumer's attributes and the retrieved Content Resource's IA metadata

- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Retrieve function or activities within the function.

7.4.2 Fault Conditions

Fault conditions as described below provide a brief description of faults that may occur during Deliver Component processing. Individual specifications SHOULD support these faults and MAY expand their description of fault conditions to address additional situations or provide additional detail on the fault.

Table 35 - Retrieve Component Fault

Fault Name	Fault Description
Content Not Found	A fault used when the provider does not have any content for the Content Identifier input.
Retrieve Property Unsupported	A fault used if a Retrieve Property is not supported.
Retrieve Property Error	A fault used if an error occurs while processing a Retrieve Property

8 Deliver Component

8.1 Component Overview

The Deliver Component allows a content resource to be sent to a specified destination, which may or may not be the requesting component.

The Deliver Component provides a behavioral model for IC and DoD content collections, enabling the delivery of content to content consumers of resources from disparate collections across the IC/DoD Enterprise. The Deliver Component provides a means to deliver a content resource.

8.2 Component Scope

The Deliver Component is general enough to be applied to the transfer of resource payloads.

8.3 Component Behavior

In its simplest form, Deliver will take a consumer-supplied payload and send it to another consumer as specified in the delivery property set. A second variation of Deliver may include additional processing, such as compression, encryption, or conversion that makes delivery of the payload suitable for its destination and the delivery path to be used. For example, if an image is to be delivered to a location that can only be reached using a low bandwidth path, Deliver processing may choose to apply interim processing of the information payload to transform it into a payload that is more appropriate for the destination. The third variation of Deliver occurs when a consumer requests the delivery of an information payload that must be retrieved prior to information delivery.

The outcome of using the Deliver Component is a payload resident at the specified destination. Successful delivery of a payload REQUIRES the Receiving Consumer to understand and be able to process the protocols used by the Deliver Component Implementation. Note that the mechanism for ensuring the attempted Deliver protocol and the protocols recognized by the Receiving Consumer are compatible is out of scope.

8.4 Interface Model

The Deliver function is the single function of the Deliver component. All Deliver Component implementations MUST implement the Deliver function.

Table 36 - Deliver Component Functions

Function Name	Required/Optional
Deliver	Required

8.4.1 Deliver Function (required)

8.4.1.1 Preconditions

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

- A Policy Decision Point has determined, based on the requestor's identity and authentication of that identity ("authenticated identity"), that the consumer is authorized to invoke Deliver, and a Policy Enforcement Point allows Deliver to be invoked for the consumer.
- A Policy Decision Point has determined, based on subject and object attributes of the object and of the Deliver Destination's attributes that the destination consumer is authorized to receive the content, and a Policy Enforcement Point allows the deliver request to be executed.
- A payload exists and is accessible for delivery.

8.4.1.2 Input

Table 37 - Deliver Function Inputs

Input Name	Required/Optional
Resource Payload	Required
Deliver Properties	Optional

Resource Payload – Content being delivered.

Deliver Properties – Information needed by the Deliver Component to process and deliver the content specified in the retrieve request. An example of one item in the Deliver Properties could be a delivery address other than the default address to which the retrieved resource would be returned.

8.4.1.3 Output

The outcome of using the Deliver Component is a content resource resident at a specified destination. There may not be specific Deliver Component output, although a status output MAY be defined.

8.4.1.4 Post-conditions

The following post-conditions MUST be the end result of the deliver function if it has successfully processed input and generated output as described. Fault conditions SHOULD result if post-conditions are not satisfied.

- The resource payload is resident at a specified destination.
- Only content that the Deliver destination is authorized to access will be delivered to it. Authorization will be determined based on the Deliver destination's attributes and the content's Information Assurance metadata.
- The audit security service collected information and fault reports to identify and analyze anomalous behavior or misuse during the execution of the Deliver function or activities within the function.

8.4.2 Fault Conditions

Fault conditions as described below provide a brief description of faults that may occur during deliver component processing. Individual specifications SHOULD support these faults and MAY expand their description of fault conditions to address additional situations or provide additional detail on the fault.

1519 **Table 38 - Deliver Component Faults**

Security Fault	A fault used if the consumer is not authenticated or is not authorized to use the Deliver function.
Deliver Component Unknown	A fault used if Deliver Properties are present in request but the deliver location is not specified in the Deliver Properties and there is no default defined.
Deliver Component Unavailable	A fault used if Deliver Properties are present in request but the deliver location cannot be reached.
Incompatible Deliver Protocol	A fault used if there is a protocol mismatch.

1520

1521

9 External Dependencies

9.1 Service Security

The Security focus area provides a set of security-focused services to the IC and DoD for protecting access to services, data, and their interactions within the IC/DoD Enterprise. Integration of Security capabilities is advocated, both from the service discovery and the service access standpoint, to protect content providers and consumers from attack from any unknown entities. Security capabilities are responsible for authenticating and authorizing of consumers and consumer agents, binding IA metadata to the information that it describes (query, search result, or retrieved content), controlling access to content resources, and enabling cross-domain search and retrieval. Furthermore, security capabilities provide integrity, confidentiality, and audit services that CDR providers can leverage. CDR providers together with their security engineers should reference the Joint IC/DoD Security Reference Architecture (SRA) for guidance on integrating and using the security services within and between CDR Components.[18] It is expected that the SRA and derived specifications will provide guidance for implementers of the CDR Components which identifies interface points for requesting security services. As appropriate, this guidance will be documented within the CDR Architecture Model (Para 1.6) to achieve secure CDR services. In some cases, e.g., cross domain search and retrieval, the CDR specifications may have to be augmented to address processing queries, query results, retrieve requests or retrieved content to ensure that they can be inspected by cross domain solutions in accordance with applicable cross domain policies.

9.1.1 Service Security Concerns

The following security relevant considerations are consolidated in this Section to more clearly define points of intersection and dependency upon the Joint IC/DoD SRA that may be of importance in realizing the CDR Specification Framework:

- Identification and Authentication: The operations defined here require the Consumer Component to provide an authenticated identity to the CDR Component it is calling. The authentication requirement extends to authenticating CDR Components acting on behalf of a consumer (chained authentication).
- Activity Authorization: The CDR Component must determine if the authenticated consumer is authorized to perform the requested activity. In addition, it must determine if the intended recipients of delivered metadata or resource content are authorized to receive it.
- Resource Authorization: In addition to authorizing activities in general, for operations that refer to one or more Saved Searches, the QM Component must determine if the Consumer Component is authorized to perform the requested activity on the Saved Search. For example, a Saved Search with public access may be readable and executable by all, but update and write permissions may be limited.
- Access Control: The CDR Component must abide by the access control policies for search results and retrieved content based on their IA Metadata, and on Consumer and CDR Component security attributes. Access control is applied to

[18] *This guidance could also cover the security aspects of CDR Components interacting with non-CDR Components.*

both the Content Collection and individual Content Resources within the Collection.

- Classification: General rules and specifications referring to the classification of saved resources also apply to CDR Components, but are not described in this framework.
- Auditing and Logging: General rules and specifications referring to the auditing and logging of data apply to CDR Components, but are not described in this framework.
- Protecting Confidentiality, Integrity, Availability and Non-Repudiation: General rules and specifications referring to these security concerns apply to CDR components, but are not described in this framework. This includes message level and transport level security.
- Cross Domain Search and Retrieval: While the general rules and specifications for cross domain solutions are not described in this framework, we anticipate providing guidance for augmenting CDR specifications to address processing CDR information before and after it is passed to a cross domain solution.

9.1.2 Security Fault Conditions

The following potential security fault conditions are common to most of the CDR capabilities:

- Action Not Authorized: The Consumer does not have permission to perform the requested function on the requested resource.
- Identity Not Authenticated: The Consumer could not be authenticated or the claimed identity could not be confirmed.

9.2 Messaging

The Messaging focus area provides the capabilities for handling the addressing and re-routing functions necessary for asynchronous messaging. Furthermore, Messaging capabilities provide reliable messaging capabilities for potentially large sets of content. CDR providers should reference the IC/DoD Messaging Reference Architecture[19] for guidance on utilizing the service discovery capabilities.

[19] *CDR-IPT Technical Artifacts @ Unclassified Intelink Web Site:* https://www.intelink.gov/site/odni/cio/i2e/focus/iads/cdript/default.aspx.

Appendix A – Reference Documents

The following references were used in the development of this specification:

1. CDR-RA, *IC/DoD Content Discovery and Retrieval Reference Architecture V1.1, February 2011*
2. CDR-IPT Requirements Specification, D2R-CDR Requirements FINAL 12-10-09-Mapping, (available via the CDR-IPT Section of the Unclassified Intelink Web Site [3]
3. CDR-IPT Technical Artifacts @ Unclassified Intelink Web Site: https://www.intelink.gov/site/odni/cio/i2e/focus/iads/cdript/default.aspx
4. Internet Engineering Task Force (IETF) RFC 2119, March 1997
5. Description Framework Artifacts: https://www.intelink.gov/wiki/Service_Description_Framework, https://www.intelink.gov/wiki/Content_Collection_Description_Framework, https://www.intelink.gov/wiki/General_Description_Framework