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

IC/DoD Keyword Query Language Specification

Version 2.0

Approval Date: 3-OCT-2012

Distribution Notice

This document has been approved for Public Release and is available for use without restriction

REVISION/HISTORY

Doc Revision	Revised By	Revision Date	Revisions
1.1	CDR IPT	01 March 2012	Draft Release for Internal Review.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Specification Overview

This document defines a keyword query language for use with Content Discovery & Retrieval (CDR) Search Component implementations. A *keyword*, in the context of a basic search, is one of the strings used to find matching content resources. It was popularized during the early days of search engine development, as it was not possible to send natural language queries to those search engines and find the desired sites. Searches typically gave the best results if only a few keywords were chosen and searched for. These keywords attempted to capture the essence of the topic in question on the basis that the keywords were likely to be present on all sites listed by the search engine.

This specification defines a common syntax, providing enough information for Search Service providers and consumers to create and use CDR-conformant Keyword Query Language Search Services.

1.2 Scope

The Keyword Query Language Specification defines the "basic" syntax and behavior of handling that syntax in a keyword search. This specification does not, however, provide the ability for a consumer or provider to provide a keyword query using more advanced functionality such as wildcards and proximity. This specification also does not provide detailed guidance for single keyword matching algorithms or techniques such as stemming or n-grams.

1.3 Artifact Overview

This specification is a part of the set of specifications that define the concrete, implementation-specific guidance for the services defined under the auspices of the Content Discovery & Retrieval (CDR) Integrated Project Team (IPT). The CDR Reference Architecture [CDR-RA] prescribes an abstract-to-concrete model for the development of architecture elements and guidance for content discovery and retrieval. Each layer or tier of the model is intended to provide key aspects of the overall guidance to achieve the goals and objectives for joint DoD/IC content discovery and retrieval. The following graphic, discussed in detail within the CDR Reference Architecture, illustrates this model.

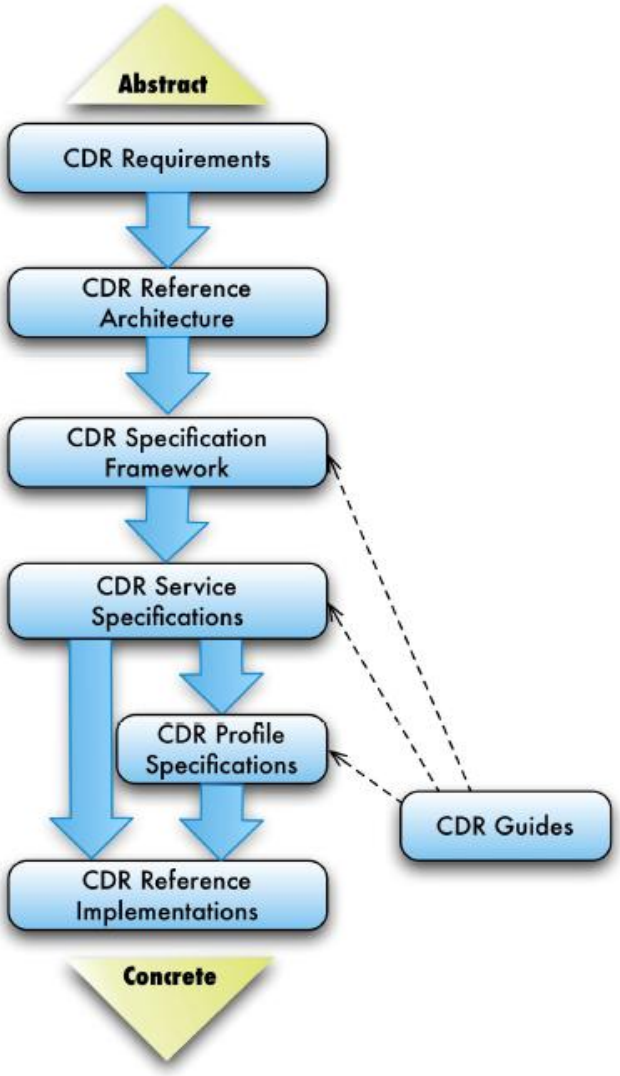

Figure 1: CDR Architecture Model

As illustrated in Figure 1, the CDR Specification Framework [CDR-SF] derives from the CDR Reference Architecture [CDR-RA] and describes behavior in terms of the capabilities, components, and usage patterns defined in the RA. Multiple CDR Service Specifications are derived from the CDR-SF, with separate specifications associated with the components of the architecture (e.g., Retrieve) and, for each service, separate specifications to address Representational State Transfer (REST) and SOAP implementations.

This specification supports the implementation of both the IC/DoD Content Discovery & Retrieval SOAP [CDR-SS] and REST [CDR-RS] Interface Specifications for CDR Search 3.0. It is intended to parallel corresponding commercial query language specifications such as the XQuery and OpenGIS Filter specifications. Additional CDR Guides, Profile Specifications, or Reference Implementations may provide additional guidance on implementing this specification in a particular context.

1.4 Notational Convention

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

When describing concrete XML schemas and example XML documents, this specification uses XPath as the notational convention. Each member of an XML schema is described using an XPath notation (e.g., `/x:RootElement/x:ChildElement/@Attribute`). The use of {any} indicates the presence of an element wildcard (<xs:any/>). The use of @{any} indicates the presence of an attribute wildcard (<xs:anyAttribute/>).

Items contained in curly braces ({item}) are meant to indicate template or notional values to be replaced by actual values (without the use of curly braces) when in actual use.

Examples in this text are distinguished by a black border. These are meant to be illustrative and only one way that the described syntax can be used.

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

1.5 Conformance

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

1.6 Namespaces

Namespaces referenced in this document and the prefixes used to represent them are listed in the following table. The namespace prefix of any XML Qualified Name (QName) used in any example in this document should be interpreted using the information below.

Table 1: Referenced XML Namespaces

Prefix	URI	Description
soap	http://www.w3.org/2003/05/soap-envelope	SOAP 1.2 Envelope
cdrs	urn:cdr:search:2.0	The CDR IPT Search binding for SOAP implementations

2 Query Language Definition

2.1 *Identifier*

The Keyword Query Language defines its unique URI as follows:

```
urn:cdr:query:keyword:2.0
```

2.2 *Syntax*

The syntax of the keyword query is as follows as specified in EBNF[1]:

```
<keyword-query-expression> ::= <term> (<boolean-operator> <term>)*;

<boolean-operator> ::= " "|" AND "|" OR "|" NOT "|" AND NOT "|" OR NOT ";

<term> ::= <keyword> | <phrase> | <group>;

<phrase> ::= '"' <keyword> (' '<keyword>)* '"';

<group> ::= '('<keyword-query-expression>')';

<keyword> ::= (<uppercase-alpha> | <lowercase-alpha> | <digit>)+

<uppercase-alpha> ::= <any US-ASCII uppercase letter "A".."Z">

<lowercase-alpha> ::= <any US-ASCII lowercase letter "a"..."z">

<digit> ::= <any US-ASCII digit "0"..."9">
```

Figure 2: Keyword Query Expression in EBNF

2.3 *Keyword Query Expression*

A keyword query expression in its entirety is a string of Unicode [Unicode] characters. The keyword query expression when parsed can broken into Boolean operators and three types of terms: keyword, phrase and groups.

2.3.1 Order of Precedence

The default order of precedence in evaluating a keyword query expression is to evaluate terms moving from left to right. This ordering can be modified through the use of parentheses ( ) and double quotes " ".

2.4 *Keyword*

A keyword is a single string (containing uppercase or lowercase letters and/or numeric digits with no whitespaces) such as "test" or "hello".

[1] *Extended Backus-Naur Form (EBNF) is a notation for expressing the syntac of a language [EBNF].*

2.4.1 Common Words

An implementation SHOULD ignore any commonplace words in the keyword expression, such as “a” and “the,” unless encapsulated by quotes. The words to ignore are left as an implementation detail.

2.5 *Boolean Operators*

The ***Search*** Service MUST respect Boolean operators found in a keyword expression (for example, the search `treaty OR verification` would return metadata for items that match *either* “treaty” or “verification,” while the search `treaty AND verification` would only return metadata for items that match both). In the following sections, the behaviors of operators are described, where Apple, Orange, and Banana should be interpreted as arbitrary keywords, phrases, or groups.

2.5.1 AND Operator

The AND operator searches for a result that contains both the term before and after it. The AND operator is case-sensitive.

Apple AND Orange

Figure 3: AND Operator Example

Multiple search terms separated by a space but no explicit operator are processed as an AND operation, i.e. the search should find all the terms in the keyword-query-expression.

Apple Orange

Figure 4: Alternative AND Operator Example

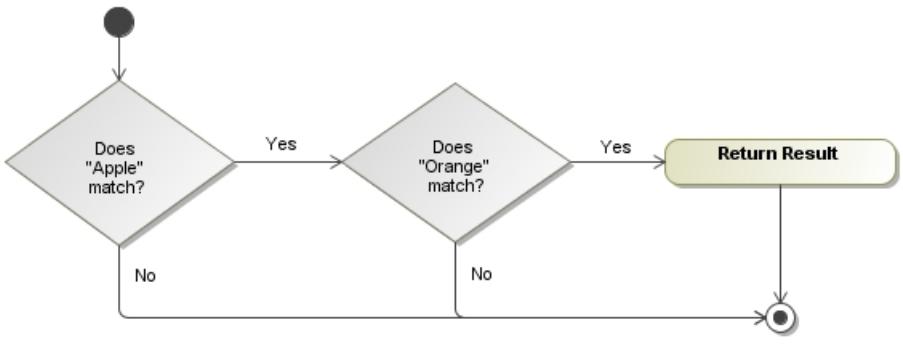

Figure 5: AND Operator Logic

2.5.2 OR Operator

Gives a choice. The OR operator links two terms and finds a matching result if either of the terms exist, e.g. Apples OR Oranges. The OR operator is case-sensitive.

Apple OR Orange

Figure 6: OR Operator Example

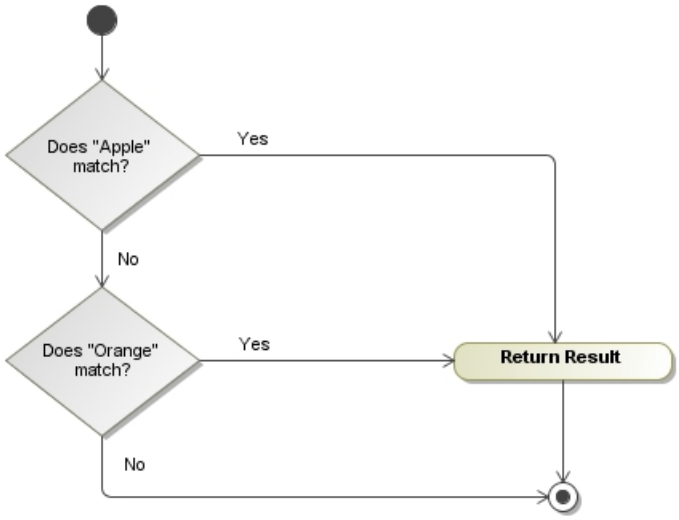

Figure 7: OR Operator Logic

2.5.3 NOT Operator

Exclude the term after this operator. The NOT operator is case-sensitive.

Apple NOT Orange

Table 2: NOT Operator Syntax

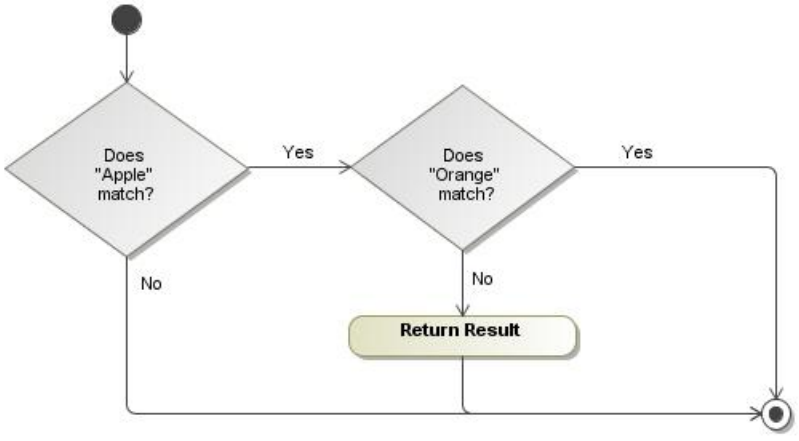

Figure 8: NOT Operator Logic

2.5.4 (Group)

A group uses parenthetical notation to control the order of precedence of a keyword query expression. By inserting a particular expression within parentheses, that subset of

the keyword query expression must be processed first before the rest of the keyword expression. Parenthetical Notation allows increasingly complex Boolean operations. This is illustrated in the following example:

```
(Apple AND(Orange OR Banana))
```

When evaluating an expression with multiple sets of parentheses, the innermost parentheses MUST be evaluated first, followed by the next innermost and so forth. In the previous example, the innermost set of parentheses is "(Orange OR Banana)".

2.5.5 "Phrase"

A phrase is any number of keywords separated by spaces and enclosed in double quotes such as "hello dolly". The results MUST contain the string (case-insensitive) within those quotation marks. For example, if the expression contained the phrase `"range acquisition technology"` in quotes as shown, the service would only return matches to the phrase and not data that matched one or two of the keywords.

2.6 *Processing Rules*

A Keyword Search is the most basic type of Search and has a minimal set of processing rules. The only mandatory requirement is the provided keywords MUST be applied against the available set of data resources to determine any potential matches. This query language definition does not place any restrictions on the exact keyword matching algorithm that should be used to determine matches between the provided keyword(s) and the data it manages. However, very useful additional functionality are added by supporting phrases, Boolean operators and parenthetical notation as in this specification and commonly found in many public search services on the Internet.

3 Implementation Guidance

The following example illustrates a Search Request using the Keyword Query Language in both the IC/DoD Content Discovery & Retrieval SOAP [CDR-SS] and REST [CDR-RS] Interface Specifications for CDR Search 3.0:

```
<?xml version='1.0' ?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Header>
    <wsa:Action>urn:cdr:search:3.0:request</wsa:Action>
  </soap:Header>
  <soap:Body>
    <cdrs:SearchRequest startIndex="1" count="10"
          responseFormat="urn:cdr:response:atom:1.0">
      <cdrs:Expression queryLanguage="urn:cdr:queryLanguage:keyword">
            watson ibm
      </cdrs:Expression>
    </cdrs:SearchRequest>
  </soap:Body>
</soap:Envelope>
```

Figure 9: Search Input (SOAP) Example

```
http://example.com/?q=watson+ibm&StartIndex=31&count=10
```

Figure 10: Search Input (REST) Example

References

[CDR-RA]
"CDR IPT Reference Architecture ", 1.1, 25 Feb 2011.

[CDR-RS]
"IC/DoD Content Discovery & Retrieval REST Interface Specification for CDR Search 3.0." 2012.

[CDR-SF]
"IC/DoD Content Discovery & Retrieval Specification Framework", 1.0 DRAFT, 9 May 2011.

[CDR-SS]
"IC/DoD Content Discovery & Retrieval SOAP Interface Specification for CDR Search 3.0." 2012.

[EBNF]
"Extended Backus-Naur Form." 1996. http://standards.iso.org/ittf/PubliclyAvailableStandards/s026153_ISO_IEC_14977_1996(E).zip"

[UNICODE]
"The Unicode Standard." 2003.

Appendix A. Mapping between other query syntaxes

This section provides some guidance between the items (operators, etc) in this specification and the equivalent in other similar query languages and tool sets.

CDR Keyword Specification	Lucene	Google	Sharepoint
AND, ‘ ‘	AND, &&	‘ ‘	_AND_
OR	OR, ||	OR	
NOT	NOT, !		_NOT_
( )	( )		
“ ”	“”	“”	“”