Internet Draft
Document: draft-ietf-liang-irpdl-03.txt
Expires: April 2004

Wang Liang,et al
hust

November 2003

Information Retrieval Protocol for Digital Resources
draft-liang-irpdl-03.txt

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of [RFC2026].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April 10, 2004.

Copyright Notice

Abstract

This document specifies an information retrieval protocol for heterogeneous information resources. This protocol has two parts: standard search Webservice, which defines the format of query words and the search results; and a method to find and select such search Webservice. By using this protocol, all the databases including web page database, digital issue database, and video database, can release the uniform search Webservice, though these databases may have different metadata standards and architectures. And these Webservice can be easily found and visited by search systems. This very protocol makes it possible that users can obtain all kinds of

information on Internet in single search engine, but not visit lots of different search engines one by one.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119].

Table of Contents

1.Introduction 2
2.Standard Search Webservice.....3
2.1 Data encoding.....3
2.2 The format of query words.....3
2.3 class and function.....5
2.3.1 Class Search.....5
2.3.2 Class Search Response.....7
2.3.3 Class Result Element.....7
2.4 WSDL of standard search Webservice.....8
3.The description of the search Webservice.....10
3.1 The XML schema of the SDDI.....11
3.2 The API of SDDI.....14
3.2.1 Publish.....14
3.2.2 Inquiry API.....14
4.Security consideration.....15
5.Reference.....15
5.1 Normative references.....15
5.2 Informative references.....15
6.Author's Address.....15
7. Copyright Statement.....16

1. Introduction

Nowadays there are many kinds of information on Internet, such as web pages, FTP, and hundreds of databases in many libraries. It has become a kind of acrobatics for us to find the complete and precise results about our query form so many data resources. Everyone hopes to obtain all kinds of information in one search engine, such as web pages, Videos, but does not care where the information lies in. Webservice [4] give us a good method to realize this desire. As long as these databases can provide Webservice, it will be an easy mission to integrate all kinds of information resources in one search engine. Now Google [5] and some other databases have provided search Webservice. But standard protocol for these searches Webservice does not exist. Even different web search enginesÆ Webservice have distinct formats of queries and search results, needless to mention the Webservice of many other kinds of databases. Thus, a uniform

Webservice applicable for all the information resources and an efficient method to find such Webservice should be established. This memo just achieves these two goals. The protocol comprises of two interacting parts, Standard Search Webservice (SSW) which can be applied to all databases and Search Webservice Description, Discovery and Integration (SDDI) which provides an efficient way to find the appropriate search Webservice.

2 Standard Searches Webservice

Standard Searches Webservice defines a standard search Webservice with its classes and functions. Most of databases can distribute the uniform search Webservice by using this definition.

2.1 data encoding

In order to support searching documents in multiple languages, all requests and responses should be in accordance with the UTF-8 encoding.

2.2. The format of query words

The query words specify attribute based boolean queries. Different communities will require their own sets of attributes, so these query words is flexible enough to allow attributes from different communities.

There are three kinds of query words : basicQuery, advancedQuery, fullQuery. The XML Schema of query word defined as follow.

Element "logops" contains three logical operators "AND","NOT","OR".The XML Schema of "logops" is defined as follow.

```
<xsd:simpleType name="logops">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="AND">
<xsd:enumeration value="NOT">
<xsd:enumeration value="OR">
</xsd:restriction>
</xsd:simpleType>
```

Element "operator" defines the string operation, whose value can be "contain", "equal" or "like".

equal: operator that provide simple equality matching on property values.
contain operator that search the documents that contain the special words.

like: operator that give simple wildcard-based pattern matching ability. Wildcard can be "?" or "%".The "?" wildcard matches exactly one character. The "%" wildcard matches zero or more characters.

The XML schema of element "operator" is defined as follows.

```
<xsd:simpleType name="operator">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="contain">
<xsd:enumeration value="equal">
<xsd:enumeration value="like">
</xsd:restriction>
</xsd:simpleType>
```

All the query words will be defined as a series of combination of logical operator, operator, and search field. For example:

```
<logops>AND</logops>
<oprator>contain<operator>
<title>java</title>
```

It represents that the title of result must contain the "java".

The basicQuery is defined as follow. "title" represents the basic description of a recorder. Field "title" is available in all the metadata and databases. Book, video, webpage database all can provide the search in "title".

```
<xsd:complexType name="BasicQuery">
<xsd:complexType name="field" minOccurs="0" maxOccurs="unbounded">
<xsd:sequence>
<xsd:element ref="logops">
<xsd:element ref="operator">
<xsd:element name="title" type="xsd:string">
</xsd:sequence>
</xsd:complexType>
</xsd:complexType>
```

The advancedQuery is defined as follow. Selecting element "title", "keywords", "author", "abstract" as the search fields is because these fields are normally available in most databases and metadata and represent the same meaning.

```
<xsd:simpleType name="AdvancedField">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="title">
<xsd:enumeration value="keywords">
<xsd:enumeration value="author">
<xsd:enumeration value="abstract">
</xsd:restriction>
```

```
</xsd:simpleType>

<xsd:complexType name="advancedQuery">
<xsd:complexType name="field" minOccurs="0" maxOccurs="unbounded">
<xsd:sequence>
<xsd:element ref="logops">
<xsd:element ref="operator">
<xsd:element ref="AdvancedField">
</xsd:sequence>
</xsd:complexType>
</xsd:complexType>
```

The fullQuery is defined as follow. FullQuery will provide all the search field of one database. They will be decided by the database owners.

```
<xsd:simpleType name="FullField">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="title">
<xsd:enumeration value="keywords">
<!--and so on-->
</xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="FullQuery">
<xsd:complexType name="field" minOccurs="0" maxOccurs="unbounded">
<xsd:sequence>
<xsd:element ref="logops">
<xsd:element ref="operator">
<xsd:element ref="FullField">
</xsd:sequence>
</xsd:complexType>
</xsd:complexType>
```

2.3 class and function

The standard Webservice with its class structures and functions are detailed here and also presented in the form of WSDL [1].

There are three function components in a search Webservice.1 receive the query words and return results.2 analyze and explain the results. 3 depict every recorder of the results. All these functions are implemented with three classes of Webservice.

2.3.1 Class Search

Main function of this class is to submit a query string and a set of parameters to the search service and receive in return a set of search results.

There are three levels of search function according to the three kinds of query words in this class: basic search, advanced search, full search.

a. Basic search

```
basicSearch(
query as basicQuery,
start as integer,
maxResults as Integer)
```

query: XML format parameter, it accords with the different definitions of query words.

start: Zero-based index of the first desired result.

maxResults: Number of results desired per query. The maximum value per query set to 100, and the minimum is defined as 1. If you make a query that doesn't have many matching items, the actual number of results you get may be smaller than that of you request.

b. Advanced search

```
advancedSearch(
query as advancedQuery,
dateStart as date,
dateEnd as date,
start as Integer,
maxResults as Integer,
orderby as String,
order as string)
```

dateStart,dateEnd: present date range. If you want to limit your results to document that are published within a specific date range, you can use this query term to accomplish this.

orderby: the sort order of the results. It can be "date" which means sorting by date or "Relevance" which means sorting by the relation between results recorders and query, or "title", sorting by field "title".

Order: can be "descending" or "ascending", recorders is sorted in descending or ascending.

c. Full search

```
fullSearch(
query as fullQuery,
 Start as Integer,
 maxResults as Integer,
other parameters)
```

Full search will provides all the query formats of one database. They will be guaranteed by the database owners.

2.3.2 Class Search Response

Each time you issue a search request to the search service, a response is returned back to you. This class describes the meanings of the values returned to you. The characters of this class are described as follows.

TotalResultsCount: The estimated total number of results that exist for the query.

resultElements: An array of "resultElement" items. This corresponds to the actual list of search results.

startIndex:Indicates the index (1-based) of the first search result in "resultElements".

endIndex: Indicates the index (1-based) of the last search result in "resultElements".

searchTime :Text, floating-point number indicating the total server time to return the search results, which measured in seconds.

2.3.3 Class Result Element

This class describes every record in return results. This Class has three characters as follows.

Sourcename: name of the information source.

Title: title of the search recorder.

URL: The URL of the recorder, returned as text, with an absolute URL path.

Otherinformation: some information such as a snippet of a webpage, author of the recorder. This character will be defined according to different search Webservice.

2.4 WSDL of standard search Webservice

```
<definitions name="search"
targetNamespace="databaseSearch "
xmlns:typens=" databaseSearch "
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://schemas.xmlsoap.org/wsdl/">
<types>

<xsd:schema
xmlns=http://www.w3.org/2001/XMLSchema
targetNamespace=" databaseSearch ">
<xsd:complexType name="SearchResult">
<xsd:sequence>
<xsd:element name="ResultsCount" type="xsd:int" />
<xsd:element name="resultElements" type="typens:ResultElementArray"/>
<xsd:element name="startIndex" type="xsd:int" />
<xsd:element name="endIndex" type="xsd:int" />
<xsd:element name="searchTime" type="xsd:double" />
</xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ResultElement">
<xsd:sequence>
<xsd:element name="Sourcename" type="xsd:string" />
<xsd:element name="title" type="xsd:string" />
<xsd:element name="URL" type="xsd:string" />
<xsd:element name=" otherInfomation" type="xsd:string" />
</xsd:sequence>
</xsd:complexType>


<xsd:complexType name="ResultElementArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType"
wsdl:arrayType="typens:ResultElement[]" />
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>
</types>

<message name="basicSearch">
```

```
   <part ref="basicQuery"/>
```

```
   <part name="start" type="xsd:int" />
   <part name="maxResults" type="typens:intScope" />
   </message>

   <message name="advancedSearch">
   <part ref="advancedQuery"/>
   <part name=" dateStart " type="xsd:date" />
   <part name=" dateEnd" type="xsd:date" />
   <part name="start" type="xsd:int" />
   <part name="maxResults" type="typens:intScope" />
   <part name="order" type="xsd:string " />
   </message>

   <message name="fullSearch">
   <part ref="fullQuery"/>
   ......
   <part name="start" type="xsd:int" />
   <part name="maxResults" type="typens:intScope" />
   </message>

   <message name="searchResponse">
   <part name="return" type="typens:SearchResult" />
   </message>

   <portType name="SearchPort">
   <operation name="basicSearch">
   <input message="typens:basicSearch" />
   <output message="typens:searchResponse" />
   </operation>

   <operation name="advancedSearch">
   <input message="typens:advancedSearch" />
   <output message="typens:searchResponse" />
   </operation>

   <operation name="fullSearch">
   <input message="typens:fullSearch" />
   <output message="typens:searchResponse" />
   </operation>

   </portType>

   <binding name="SearchBinding" type="typens:SearchPort">
   <soap:binding style="rpc"
   transport="http://schemas.xmlsoap.org/soap/http" />
   <operation name="basicSearch">
   <soap:operation soapAction=" searchAction" />
   <input>
```

```
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>

<operation name="advancedSearch">
<soap:operation soapAction="searchAction" />
<input>
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</operation>

<operation name="fullSearch">
<soap:operation soapAction=" searchAction" />
<input>
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
<soap:body use="encoded" namespace="databaseSearch"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</binding>
<service name="SearchService">
<port name="SearchPort" binding="typens:SearchBinding">
<soap:address location=" " />
</port>
</service>
</definitions>
```

3 The description of the search Webservice

To describe the search Webservice, we refer to the UDDI [2] and UKOLN SCD [6] and define Search Webservice Description, Discovery and Integration (SDDI) in this part. SDDI will help the search system find and select the appropriate data sources.

3.1 The XML schema of the SDDI

We use the DC [3] standard to descript the character of the search Webservice. The other basic information for a web service is also added in the SDDI. All elements are divided into four groups.

The elements and attributes to describe a search Webservice are represented as follows.

1 content

Title A name given to the resource.

Description An account of the content of the resource.

Language A language of the intellectual content of the resource.

Source: Information about a second resource from which the present resource is derived. While it is generally recommended that elements contain information about the present resource only, this element may contain metadata for the second resource when it is considered important for discovery of the present resource.

2 copyright

Creator An entity primarily responsible for making the content of the resource.

Publisher An entity responsible for making the resource available

Rights Information about rights held in and over the resource.

Admin: The person or organization responsible for service operation and administration. Typically the value of Admin will be a name or name and email address.

3 Instantiation

Date A date of an event in the lifecycle of the resource

Identifier An unambiguous reference to the resource within a given context.

Format Typically, Format may include the media-type or dimensions of the resource. Format may be used to identify the software, hardware, or other equipment needed to display or operate the resource. Examples of dimensions include size and duration. Recommended best practice is to select a value from a controlled vocabulary.

Type: The category of the resource, such as home page, novel, poem, working paper, technical report, essay, dictionary. For the sake of interoperability, Type should be selected from an enumerated list that is currently under development in the workshop series.

Categorybag: This is an optional list of name-value pairs that are used to tag a resource with specific taxonomy information. Some classification methods according to subjects can be adopted, such as (CLC) Chinese Library Classification (LCC) Library of Congress Classification.

4 Interface: the definition of program interface.

AccessPolicy: A description of any constraints or legal prerequisites for accessing the collection or its component items.

UDDI the UDDI content of this Webservice if available.

accesspoint: URL of this Webservice

queryschema: all supporting search parameters.

The XML schema of SDDI is defined as follows.

```
<xsd:element name = "SourceEntity">
<xsd:complexType>
<xsd:element ref= "Content" >
<xsd:element ref= "Instantiation" >
<xsd:element ref= "Copyright" >
<xsd:element ref= "Interface" >
</xsd:complexType>
</xsd:element>
```

The XML schema of element "Content":

```
<xsd:element name= "Content" >
<xsd:complexType>
<xsd:element name = "Title" type="xsd:string"/ >
<xsd:element name = "Description" type="xsd:string"/>
<xsd:element name = "Language" type="xsd:language"
maxOccurs="unbounded"/>
<xsd:element name = "Source" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
</xsd:complexType>
</xsd:element>
```

The XML schema of element "Copyright":

```
<xsd:element name = "Copyright" >
<xsd:complexType>
<xsd:element name = "Creator" type="xsd:string" minOccurs="1"
maxOccurs="unbounded"/>
<xsd:element name = "Publisher" type="xsd:string" />
<xsd:element name = "Rights" type="xsd:string"/>
<xsd:element name = "Admin" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
```

The XML schema of element " Instantiation ":

```
<xsd:element name = " Instantiation " >
<xsd:complexType>
<xsd:element name = "Date" type="xsd:date" />
<xsd:element ref = " Identifier " type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
<xsd:element ref = "Format" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
<xsd:element ref = "Type" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
<xsd:element ref = " Categorybag" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"/>
</xsd:complexType>
</xsd:element>

<xsd:element name = "Categorybag" />
<xsd:complexType>
<xsd:attribute name= "keyName"/>
<xsd:attribute name= "keyValue" use = "required"/>
</xsd:complexType>
```

The definition of "Identifier", "Format", "Type" is as same as that of "categorybag". They are all an optional list of name-value pairs.

The XML schema of element "Interface":

```
<xsd:element name = " Interface " >
<xsd:complexType>
<xsd:element name = " AccessPolicy" >
<xsd:element ref = "UDDI" minOccurs="0" maxOccurs="1"/ >
<xsd:element ref="Queryschema">
<xsd:element name= "Accesspoint" type="xsd:anyURL" use="required"/>
</xsd:complexType>

<xsd:element ref=" Queryschema ">
<xsd:complexType>
```

```
<xsd:sequence>
<xsd:element ref = "Searchfield" minOccurs="1" maxOccurs=
"unbounded"/>
<xsd:element ref = "otherParameter" minOccurs="0" maxOccurs=
"unbounded"/>
</xsd:sequence>
</xsd:complexType>
```

"Searchfield" is all supported search field. We can use it to construct the "Fullquery" in query words. "otherParameter" defines the other parameter supported by function "Fullsearch"." Queryschema" is defined by resource provider.

3.2 The API of SDDI

3.2.1 Publish

When a library purchase a database, a SDDI of this database will be authorized at the same time and saved at the local servers. The library can revise the SDDI itself according to its own needs.

3.2.2 Inquiry API

Inquiry API will provide two simple functions that help the search engine find the appreciate Webservice that matches the requirements of users. The definition of element should refer to the SDDI. Meanwhile, the element can be complex element.

1 find(element,value)

Element:the name of element.
Value: the value of element, which is xml format according to the different element of SDDI.

This function returns the "accesspoint" according to the element and its value. For example:

find(title, "ACM")

This function will return the "acesspoint" of SDDI whose element "title" is "ACM".

2 get(element1,value,element2)

Return the value of element2 according to the value of element1.the format of results is accordance with XML. For example:

Get((title, "ACM", character)

This function will return the element "character" of SDDI whose element "title" is "ACM".

4. Security Considerations

Since the databases are always purchased by some organization, the IP access control will be used to protect the copyright.

The organization will build their own search engine based on this protocol, so the Webservice and SDDI will not open to end user. The more additional security concerns should be decided by the corresponding organization.

5. Reference

5.1 Normative References

[1] WSDL, http://www.w3.org/TR/wsdl12
[2] UDDI, http://uddi.org/pubs/uddi_v3.htm
[3] S. Weibel, J. Kunze, "Dublin Core Metadata for Resource Discovery", rfc2413, September 1998.

5.2 Informative References

[4] Webservice , http://www.w3.org/2002/ws
[5] The web service of Google, http://www.google.com/apis/
[6] Simple Collection Description scheme, http://www.ukoln.ac.uk/metadata/cld/simple/

6. Author's Addresses

Wang liang
HUST
WUHAN 430074
P.R.China
Phone: 86-27-87553494
Email:wangliang_f@163.com

Guo YiPing
HUST
WUHAN 430074
P.R.China
Email:gyp@hust.edu.cn

Fang Ming
HUST

WUHAN 430074
P.R.China
Email:fangming_w@263.net

Full Copyright Statement