

U.S. Department of the Interior
U.S. Geological Survey

The National Map Catalog Technical Discussion Paper

The National Map Catalog Service: A Guide for Application Developers

Document version 0.3.3

For catalog service versions 1.1.0 and 2.0.0

Jeff Wendel
Larry Moore

Mid-Continent Mapping Center
March 2005

This document is for internal use by USGS Geography Discipline personnel. It has not been reviewed for conformance with USGS editorial standards and has not been approved for formal publication. Any use of trade names is for descriptive purposes only and does not constitute endorsement by the US Government.

Table of Contents

1	Introduction.....	2
1.1	Notes on Version Numbers.....	2
1.2	Notes on Formal Standards Processes.....	3
1.3	Catalog Service Version 2.0.0.....	3
2	Hosts.....	4
3	General Form of Catalog Service Requests.....	4
4	Query Strings.....	5
4.1	Query String Formal Definition.....	5
4.2	versionParameter.....	7
4.3	requestParameter.....	7
4.3.1	getCapabilities.....	7
4.3.2	describeRecord.....	7
4.3.3	getRecords.....	7
4.3.4	getClassifications (version 2.x only).....	8
4.3.5	getElementSets (version 2.x only).....	8
4.3.6	getElementStatuses (version 2.x only).....	8
4.3.7	getServiceStatuses (version 2.x only).....	9
4.4	querySpecParameter.....	9
4.4.1	bboxParameter.....	10
4.4.2	classificationsParameter.....	10
4.4.3	scaleParameter.....	10
4.4.4	allClassificationsParameter.....	10
4.4.5	rankingParameter.....	11
4.4.6	elementParameter.....	11
4.4.7	elementSetParameter.....	11
4.4.8	serviceStatusesParameter.....	12
4.4.9	elementStatusesParameter.....	12
4.4.10	serviceAvailableParameter.....	12
5	Interactive Query Form for the Catalog Service.....	12
6	References.....	14
	Attachment A. Reading XML Responses with Software.....	15
	Attachment B. Example response.....	16
	Attachment C. Summary of Differences Between Catalog Service Versions 1.1.0 and 2.0.0..	19

1 Introduction

The National Map manages a database that tracks information about Web Map Services (WMS). This database can be queried through a web application running in a servlet container on a USGS computer. This web application is a **catalog service** that is consistent with the Open Geospatial Consortium (OGC)¹ Draft Candidate Specification “OGC Web Services Stateless Catalog Profile².”

The National Map catalog is therefore composed of two parts: a database and a service for accessing the database. The database is implemented in Oracle. Direct connections to this database from outside USGS firewalls are not allowed. The catalog service makes information from the database available to any application anywhere on the Internet.

Requests to the catalog service are in http syntax. Requests are typically sent by software applications, but can also be interactively entered in a browser address window. Responses are in Extensible Markup Language (XML) document type definition (DTD) form. The XML responses must be interpreted by the client program. Because XML is human-readable, requests typed into a browser address window will return documents that can be read in the browser. This is useful for training and some types of development and debugging, but is of limited practical value for most work.

The catalog service is a stateless application, meaning there is no concept of a session.

1.1 Notes on Version Numbers

The first documented release of the catalog service (2003) was numbered 1.1.0. By unfortunate coincidence, this is the same version number as one of the important versions of the OGC WMS Specification. The confusion this would cause was not appreciated until it was too late to do anything about it. There is actually no relationship between the version of the WMS specification and the version of the catalog service.

The catalog service underwent a significant upgrade in the summer of 2004. The version number of the catalog service changed from 1.1.0 to 2.0.0.

This paper documents catalog service 2.0.0, but also contains information needed to write or maintain applications that depend on catalog service version 1.1.0.

This documentation changes more frequently than the software it describes. The version numbers of these documents do not match the version numbers of the software; they are related more closely to the authors' subjective ideas about the maturity of the documentation. Document versions 0.1.0 through 0.1.3 describe catalog service version 1.1.0. Document versions 0.2.0 and later describe catalog service version 2.x.x.

¹ Formerly "Open GIS Consortium." The name was changed to "Open Geospatial Consortium" in the summer of 2004. The acronym OGC is unaffected by the name change. The older name is still used in many documents, including documents published by the OGC.

² Though used as a guide for the original design of the catalog service, this document is now deprecated. The effects are discussed in section 1.2.

1.2 Notes on Formal Standards Processes

The catalog service was originally designed to be consistent with the Open Geospatial Consortium (OGC) discussion paper "Web Registry Services," later renamed Draft Candidate Specification "OGC Web Services Stateless Catalog Profile." Version 0.06 of this draft specification is posted on *The National Map* catalog Web site. In the introduction, this document states an intention to "...integrate this document with the OGC Catalog Services v 1.0..."

The "Web Services Stateless Catalog Profile" has now been replaced by the "Catalog Services Specification," but the two were in parallel development for several years. In August 2004 version 2.0 of the Catalog Services Specification (OGC 04-021r2) was released.

The USGS intends to implement *The National Map* catalog in conformance with the OGC Catalog Services Specification. This is not a simple ambition, because of the evolving nature of the various standards and systems involved:

- The catalog service is a working software system that existing applications depend on to function properly. It is absolutely critical that changes and enhancements to the catalog service interface do not break features of the service in previous software versions. All new versions **must** be backwardly compatible with earlier versions.
- The OGC Catalog Services Specification has been evolving and expanding at a relatively rapid rate in recent years. It is not clear whether this period of standard definition is over or not.
- Functional requirements for the catalog service continue to evolve in response to political and technical demands within the USGS and from *The National Map* business partners. These requirements are more or less independent of formal standards processes.

At this time (fall 2004) we believe the catalog service implementation is consistent **in spirit** with the OGC Catalog Services Specification, and that in time – assuming this project lives and prospers – true conformance can be achieved. The current implementation of the catalog service is probably technically inconsistent with the OGC specification, and is certainly incomplete.

As a separate but related issue, the USGS has its own product standards, and is currently working on a family of standards that will describe *The National Map*. This paper that describes the catalog service is proposed to be one such standard, and is being rewritten to conform to the appropriate USGS editorial guidelines. When that process is complete, this document will be replaced by a more formal document. The new standard will have essentially the same content, but will have a different appearance and probably a different version numbering convention.

1.3 Catalog Service Version 2.0.0

The first documented version of the catalog service was 1.1.0, released in March 2004. *The National Map* is still in a relatively early stage of development, and requirements for all parts of the system continue to evolve. In August 2004 a significantly different version of the catalog service, version 2.0.0 was released.

Version 2.0.0 is backwardly compatible with version 1.1.0. This means that existing applications based on 1.1.0 will continue to work. Upgrading applications to 2.0.0, while relatively easy, is not completely painless. An application that uses the additional features available in 2.0.0 must explicitly request that version in its requests to the catalog service. Applications written in Java or

other procedural languages may require syntax changes to some function calls. See the software release notes for more details.

The differences between the two versions are summarized in Attachment C of this document, noted throughout the body of the document, and are also summarized in the software release notes.

2 Hosts

At this writing, the catalog service runs concurrently on two hosts. The primary host, from the perspective of USGS-owned applications of *The National Map*, is

```
nmcatalog.usgs.gov:8080
```

which is an alias to a host computer physically located at Mid-Continent Mapping Center in Rolla, Missouri. When this instance is down, USGS applications switch to

```
t204dwimdn.er.usgs.gov
```

This machine is physically located at a USGS Water Resources Discipline (WRD) office in Wisconsin.

The two instances of the service are synchronized each night, so an application can use either at any time. The database instance on nmcatalog is the USGS working copy, and data changes made on nmcatalog are not transmitted to t204dwimdn until midnight of the same day.

3 General Form of Catalog Service Requests

A request to the catalog service has the general form

```
"http://" hostName ":[port]/" servletPath "?" (query string)?
```

The syntax of this string is explained in the grammar in section 4.

For example:

```
http://nmcatalog.usgs.gov:8080/catprod/start?request=getRecords&
querySpec='bbox=-96,35,-95,36|classifications=ROADS|ranking=NONE|
allClassifications=FALSE|scale=.0001|'
```

In this example:

- host name = nmcatalog.usgs.gov
- port = 8080
- servlet path = catprod/start
- query string is

```
request=getRecords&
querySpec=
  'bbox=-96,35,-95,36|
  classifications=ROADS|
  ranking=NONE|
  allClassifications=FALSE|
  scale=.0001|'
```

(line breaks and indents are inserted in this example to improve readability).

The domain name, port, and servlet path parts of the request are straight-forward, but the query string part has a complex structure of its own. Query strings are explained in section 4.

4 Query Strings

This section defines the structure of the 'query string' part of a catalog service request. A query string has three subcomponents:

- An optional version parameter to specify the catalog service version.
- A required request parameter. Seven request parameter values are defined. A query string must contain exactly one of the seven.
- An optional query spec parameter. Two of the seven request parameters may be constrained with a query spec.

Request parameters and query specs are discussed in sections 4.2 and 4.4 respectively.

4.1 Query String Formal Definition

The formal grammar of a query string is given in this specification using a simple Extended Backus-Naur Form (EBNF) notation. Each rule in the grammar defines one symbol, in the form

`symbol ::= expression`

Literals are quoted. Character classes are enclosed with square brackets []. Symbols and literals are represented in lowerCamelCase for readability; an actual instance of a queryString is case insensitive.

Comments are enclosed by /* and */.

The following operators are defined and listed in order of precedence:

1. Grouping is indicated by enclosing an expression with the (and) characters.
2. Optionality and repetition are indicated by trailing ?, +, and * characters.
 - a. ? indicates an expression appears 0 or 1 time.
 - b. + indicates an expression appears one or more times.
 - c. * indicates an expression occurs zero or more times.
3. Concatenation is indicated by juxtaposition; spaces are for readability only.
4. Alternation is indicated by the | character.

The notation used here is identical to that used by the W3C to describe the formal grammar of XML. More information can be found at:

<http://www.w3.org/TR/2004/REC-xml-20040204/#sec-notation>

A query string is defined by:

```
queryString ::= requestParameter ("&" versionParameter)? ("&" querySpecParameter)?  
/* versionParameter is discussed in section 4.2*/  
versionParameter ::= "version=" ("1.1.0" | "2.0.0")  
/* requestParameter is discussed in section 4.3 */  
requestParameter ::= "request=" requestValue
```

```

requestValue ::= "describeRecord" | "getCapabilities" | "getRecords" |
  "getClassifications" | "getElementSets" | "getElementStatuses" |
  "getServiceStatuses"

/* querySpecParameter is discussed in section 4.4 */
querySpecParameter ::= "querySpec=" "'" querySpecValue "'"

querySpecValue ::= (querySpecValueParameter "|" )+

querySpecValueParameter ::= bboxParameter |
  classificationsParameter | scaleParameter |
  allClassificationsParameter | rankingParameter | elementParameter |
  elementSetParameter | serviceStatusesParameter |
  elementStatusesParameter | serviceAvailableParameter

  bboxParameter ::= "bbox=" minX "," minY "," maxX "," maxY

    minX ::= number
    minY ::= number
    maxX ::= number
    maxY ::= number

  classificationsParameter ::= "classifications="
    (classificationsList | "all")

    classificationsList ::= stringList

  scaleParameter ::= "scale=" nonNegativeNumber | ("-" 1)

  allClassificationsParameter ::= "allClassifications=" ("true" |
    "false")

  rankingParameter ::= "ranking=" ("none" | "scaleClass" |
    "application" | "scaleClass_application")

  elementParameter ::= "element=" elementId

    elementId ::= positiveInteger

  elementSetParameter ::= "elementSet=" (elementSetId | "0")

    elementSetId ::= positiveInteger

  serviceStatusesParameter ::= "serviceStatuses=" (serviceStatusesList
    | "all")

    serviceStatusesList ::= stringList

  elementStatusesParameter ::= "elementStatuses=" (elementStatusesList
    | "all")

    elementStatusesList := stringList

  serviceAvailableParameter ::= "serviceAvailable=" ("t" | "f" |
    "all")

positiveInteger ::= [1-9] [0-9]*
number ::= "-"? nonNegativeNumber
nonNegativeNumber ::= [0-9]+ "."? [0-9]*
stringList := string ("," string)*
string ::= [ a-zA-Z0-9]+

```

The following sections provide narrative explanations and examples for the three parameter subcomponents of a query string: versionParameter, requestParameter and querySpecParameter.

4.2 *versionParameter*

versionParameter specifies the catalog service version. This parameter is optional. If it is not present, version 1.1.0 is assumed. Version 1.1.0 may also be explicitly requested. The only other permissible value at this time is 2.0.0.

4.3 *requestParameter*

requestParameter is defined in the grammar as:

```
requestParameter ::= "request=" requestValue
    requestValue ::= "describeRecord" | "getCapabilities" | "getRecords" |
        "getClassifications" | "getElementSets" | "getElementStatuses" |
        "getServiceStatuses"
```

Version 1.1.0 of the catalog service supports only three of these seven request values:

1. *getCapabilities*
2. *describeRecord*
3. *getRecords*

Version 2.0.0 of the catalog service supports four additional request values:

4. *getClassifications*
5. *getElementSets*
6. *getElementStatuses*
7. *getServiceStatuses*

The following sections are narrative discussions of the request values.

4.3.1 *getCapabilities*

A *getCapabilities* request is used to gather metadata about the catalog service itself:

```
http://nmcatalog.usgs.gov:8080/catprod/start?request=getCapabilities
```

The response to this request is an OGC-compliant capabilities XML document that describes the capabilities of the catalog service (as opposed to the capabilities of a WMS, a more common example of a capabilities document). The catalog service capabilities document is currently a static file that contains mainly contact and access URL information.

4.3.2 *describeRecord*

A *describeRecord* request will return a DTD describing the format of a *getRecords* response:

```
http://nmcatalog.usgs.gov:8080/catprod/start?request=describeRecord
```

```
http://nmcatalog.usgs.gov:8080/catprod/start?request=describeRecord&version=2.0.0
```

The DTD returned by this request contains an embedded data dictionary (in comment fields) that describe the data elements of the catalog service.

4.3.3 *getRecords*

The most important and complex request value is *getRecords*. In a sense, the *getRecords* request is the purpose of the entire service, and everything else exists to support it. This request can be issued

with or without a `querySpecParameter`, though most requests will include a query specification to constrain the request. If the `querySpecParameter` is omitted, the request is essentially an unconstrained query for everything the service can report about the database:

<http://nmcatalog.usgs.gov:8080/catprod/start?version=2.0.0&request=getRecords>

This request (with no `querySpecParameter`) will return a large XML document containing information for all public services and all public layers. This is nominally an unconstrained query, but because some `querySpecParameters` have defaults, it returns information only for services and layer that are (1) available, (2) public, and (3) part of *The National Map* element set. To return information about all services and layers, these constraints must be explicitly overridden with a `querySpec` parameter.

An example of a more normal `getRecords` request, with a `querySpecParameter`, is given in section 3.

4.3.4 `getClassifications` (version 2.x only)

Returns a classification list – that is, a list of subthemes. `getClassifications` may be constrained with an optional `querySpecParameter`.

4.3.5 `getElementSets` (version 2.x only)

Returns a list of all element sets that contain at least one element:

<http://nmcatalog.usgs.gov:8080/catprod/start?request=getElementSets&version=2.0.0>

This request cannot be constrained with a `querySpecParameter`.

Element sets support customized applications. The catalog database contains information about a large and growing number of WMSs. Though the database's primary purpose is to support applications of the USGS's *The National Map*, it is easy to see how this inventory of data sources could be useful for other applications. But not all applications are interested in exactly the same datasets. For example, an application designed by and for State X will not be interested in datasets that only cover State Y.

More importantly, some applications may need data that should not be visible to other applications. A general public viewer and a Department of Homeland Security (DHS) disaster-relief application will both need certain base data layers, but the DHS application may also depend on data layers that should not be available to the general public.

Element sets are an elegant solution to this problem. Elements (WMS layers) can be grouped together in the catalog database in sets. Applications that know what element sets they are interested in can access only those sets, and ignore all other sets. This does not, by itself, provide security for sensitive data layers. Actual data security must be provided by the services themselves through password protection or encryption. But grouping layers into sets allows cooperating applications to show only what they are interested in and avoid confusing login screens and "access denied" messages from other services.

4.3.6 `getElementStatuses` (version 2.x only)

Returns a status list containing the domain of element statuses.

<http://nmcatalog.usgs.gov:8080/catprod/start?request=getElementStatuses&version=2.0.0>

This request cannot be constrained with a querySpec.

Status is an attribute of both services and elements. The field is used as a hint to applications about using the service or element. Examples of status:

- A value of PUBLIC³ for both an element and the service it comes from means this GIS layer is within the domain of *The National Map*, has been evaluated by the USGS, is believed to be accurate and important, and is not highly duplicative of other PUBLIC layers.
- REVIEW means the service or element is being considered for inclusion in *The National Map* element set, but is still under evaluation. In some cases the element may be lacking some characteristic (such as associated geospatial metadata) required by *The National Map*, but is expected to acquire that characteristic soon.
- NOT USED means the service or element has been evaluated and is considered inappropriate for or outside the scope of *The National Map*. There are many reasons for an element to be marked this way, some of them having nothing to do with quality or completeness. NOT USED does not necessarily mean there is anything wrong with the data.

The status attribute is a simple text string. The domain of possible values is currently not limited.

4.3.7 getServiceStatuses (version 2.x only)

getServiceStatuses – Returns a status list containing the domain of service statuses. This request cannot be constrained with a querySpec.

Though not required to have the same domain of values, service status and element status are comparable. Status is an attribute of both element and service primarily to allow all the layers in a service to be flagged together with a particular status.

4.4 querySpecParameter

Two of the seven request values, getRecords and getClassifications, can be constrained with an optional query specification. Constraining these requests is a critical feature of the catalog service, particularly for the getRecords request. Without this ability to impose constraints, the service would be of very limited value to GIS applications, because all requests would return a huge set of responses.

Query specifications are defined in the grammar by:

```
querySpecParameter ::= "querySpec=" "'" querySpecValue "'"  
querySpecValue ::= (querySpecValueParameter "|" )+
```

³ Examples in the next few pages use UPPER CASE to indicate string-literal parameters and database field values. This is a typographical convention to improve the readability of this document. Requests to the catalog service are actually case-insensitive.

```
querySpecValueParameter ::= bboxParameter |
    classificationsParameter | scaleParameter |
    allClassificationsParameter | rankingParameter |
    elementParameter | elementSetParameter |
    serviceStatusesParameter | elementStatusesParameter |
    serviceAvailableParameter
```

Syntactical definitions of `querySpecValueParameter` are given in the grammar at the beginning of this section. The following sections are narrative discussion of what these parameters mean.

4.4.1 `bboxParameter`

The `bbox` parameter is used to spatially constrain the query. The catalog database explicitly associates WGS84 geographic polygonal footprints with each layer. The catalog service uses these stored footprints to determine intersection with the `bbox` and return information about layers where such an intersection occurs.

The `bbox` is defined with coordinates in this order: `minX,minY,maxX,maxY`, where all values are WGS84 geographic decimal degrees, west negative. In the northwest quadrant of the globe, the first X,Y pair correspond to the southwest corner of the box and the second X,Y pair correspond to the northeast corner.

4.4.2 `classificationsParameter`

The `classifications` parameter is used to constrain the query to selected data subthemes. *The National Map* categorizes all data by theme and subtheme – “classification” is the term used in the database for subtheme. An example of a theme is hydrography. Classifications in the hydrography theme include steams, lakes, and wetlands.

Version 1.1.0 contains no efficient method to dynamically obtain the `classifications` list. This shortcoming is removed in version 2.0.0 with the addition of the `getClassifications` `requestType` (see section 4.3.4).

4.4.3 `scaleParameter`

Every element (GIS layer) in the catalog has assigned min and max viewscales. These values are hints to applications about the appropriate range of display scales. The `scale` parameter means "return a layer only if the `scale` parameter is between `minviewscale` and `maxviewscale` inclusive."

Viewscales and the `scale` parameter are expressed in decimal degrees per screen pixel. For a detailed discussion of viewscales, see the paper "Viewscales and their Effect on Data Display" at http://mcmweb.er.usgs.gov/catalog/tnm_catalog_page2.html#techinfo.

4.4.4 `allClassificationsParameter`

The `allClassifications` parameter partially overrides constraints on `classifications`. If `allClassifications=TRUE` then all are included, but layers not classified with a member of the `classifications` parameter will have a 'requested=false' attribute in the response. Layers that are classified with a member of the `classifications` parameter will have an attribute of 'requested=true' in the response. For example, the `querySpec` fragment

```
classifications=ROADS,STREAM NETWORK|allClassifications=TRUE|
```

will return records for layers with any classification, but attribute those with a classification of ROADS or STREAM NETWORK with requested=true and all others with requested=false. If allClassifications is FALSE or absent, then only those layers with a classification in the classifications parameter will be returned in the response and every layer will be attributed with requested=true.

4.4.5 rankingParameter

The ranking parameter determines how the response layer attribute 'recommended' is populated.

If this parameter is absent or ranking=NONE, then all layers will have the attribute 'recommended=true'.

If ranking=SCALECLASS then only the largest-scale layers in each classification necessary to completely fill the bbox will have the attribute 'recommended=true', the rest will have the attribute 'recommended=false'.

If ranking=APPLICATION, the behavior is determined by the value of the field *element_set_members.recommended*. This is a non-null field with a domain of NEUTRAL, NEVER, and ALWAYS with NEUTRAL being the default. The application ranking method returns true for the layer's recommended attribute when the *element_set_members.recommended* value is ALWAYS, and false otherwise. Note that because the field is in the *element_set_members* table, the element can have different recommended values when it is a member of more than one element set.

SCALECLASS_APPLICATION ranking is a hybrid approach. When the *element_set_members.recommended* field is ALWAYS, the layer will have a recommended=true attribute. When it is NEVER it will have a recommended=false attribute, and when it is NEUTRAL it will have a recommended=true attribute if it is needed to fill the querySpec bbox with content for a requested classification, and false otherwise.

Note that, like scaleclass ranking, only the classifications that are explicitly passed in the classifications parameter are affected by these new methods. For example, if classifications=roads,orthoimagery is passed and allclassifications=true is also passed, then only layers classified as roads and orthoimagery will be subjected to ranking. Layers from other classifications will all have their recommended attributes set to false. To rank all classifications, classifications=all can be passed or they can all be listed in the classifications parameter.

It is recommended ranking only be used by advanced clients with a good understanding of its behavior.

4.4.6 elementParameter

Specifies by element ID number which element should be returned.

4.4.7 elementSetParameter

The elementSet parameter determines which element sets to consider. The parameter is the numerical database key of an element set. *The National Map* element set id is 1 and is the default if this parameter is absent. A parameter value of 0 may be passed to consider all elements without regard to which element sets they belong.

4.4.8 serviceStatusesParameter

The serviceStatuses parameter constrains the services to be considered to those with a status attribute in the comma delimited value list. A default value of PUBLIC is used if this parameter is absent. A value of ALL includes all services without regard to the status attribute.

The most common status values are PUBLIC (service is available to *The National Map* public viewer), REVIEW or NEW (data under review, may soon be available to *The National Map* public viewer), and NOT USED (data not available to *The National Map* public viewer). The complete domain of service statuses can be obtained with the getServiceStatuses requestType (see section 4.3.7).

4.4.9 elementStatusesParameter

The elementStatuses parameter constrains the elements to be considered to those with a status attribute in the comma delimited parameter value list. A default value of PUBLIC is used if this parameter is absent. A value of ALL includes all elements without regard to the status attribute.

The most common status values are PUBLIC (element is available to *The National Map* public viewer, provided the element's service is also available), REVIEW or NEW (data under review, may soon be available to *The National Map* public viewer), and NOT USED (data not available to *The National Map* public viewer). The complete domain of element statuses can be obtained with the getElementStatuses requestType (see section 4.3.6).

4.4.10 serviceAvailableParameter

USGS-hosted software continually checks the availability of services registered in the catalog. When a service does not respond to a well-formed query, the service is marked as unavailable. Applications may wish to avoid querying services that are known by the catalog to be off-line. The serviceAvailable parameter constrains the services to be considered to those with an available attribute equal to the parameter value. A default value of T, for true, is used if this parameter is absent. A value of F returns only services that are not available. A value of ALL returns all services without regard to the availability attribute.

5 Interactive Query Form for the Catalog Service

Programmers at Mid-Continent Mapping Center have written a simple forms interface to the catalog service. This form is not intended to hide the service details, but it does make queries easier to construct, and also permits stylesheets to be applied to some results (as opposed to displaying only raw XML responses).

The catalog service query form is at <http://helios.er.usgs.gov/catalog/CatalogQuery.html>. Figure 1 is a screen shot of the query form.

The form data entry fields accept information to constrain the query. The constructed query is displayed in the bottom box, and is updated dynamically as each field is filled in.

The screenshot shows a web form for querying a catalog. At the top, a green bar contains the text "Choose Server:" followed by a dropdown menu showing "t204dwimdn.er.usgs.gov". Below this, there are several input fields: "Scale:" with a text box containing ".0001", "Ranking:" with a dropdown menu, "Element:" with a text box, and "Element Set:" with a text box containing "3". A central map of the United States shows a "Geographic Bounding Box" with coordinates: West = -91, East = -90, North = 40, and South = 39. Below the map, there are three text boxes for "Classifications" (containing "roads"), "Element Statuses" (containing "public,new"), and "Service Statuses" (containing "public,new"). A note below these boxes says "(separate multiple entries with a comma)". There are two dropdown menus: "List only available services" and "Formatting Options:" (containing "None (basic XML)"). A "Submit Query" button is located below these. At the bottom, there are two links: "Application Capabilities" and "Interpreting the Results". A large text box at the bottom displays the constructed query:

```
http://t204dwimdn.er.usgs.gov/catprod/start?
request=getRecords&querySpec='bbox=-91,39,-
90,40|scale=0.0001|classifications=roads|elementset=3|servic
estatuses=public,new|elementStatuses=public,new|serviceAvail
able=T' &xsl=
```

Figure 1. Screen shot of the catalog query form, with example parameters entered.

The example parameters in Figure 1 search for roads layers, with status of either NEW or PUBLIC, within a 1-degree box in the central U.S., where $\text{minviewscale} < .0001 < \text{maxviewscale}$, and restricted to THE NATIONAL MAP element set. The parameters are:

- Geographic bounding box:
 - West = -91
 - East = -90
 - North = 40
 - South = 39

- Classifications = roads
- Service statuses = public,new
- Element statuses = public,new
- Scale = .0001
- Element set = 3
- Show only services that are currently available

The resulting constructed query, displayed in the bottom box of the form, is:

```
http://t204dwimdn.er.usgs.gov/catprod/start?request=getRecords&
  querySpec='bbox=-91,39,-90,40|
  scale=0.0001|
  classifications=roads|
  elementset=3|
  serviceStatuses=public,new|
  elementStatuses=public,new|
  serviceAvailable=T|'
&xsl=
```

(line breaks and indents inserted to improve readability).

Entries in the form fields must obey the grammar rules defined elsewhere in the body of this document.

The default formatting option is "none." In this case, an XML document is returned. Selecting one of the style sheet options will filter and format the returned data.

6 References

Extensible Markup Language (XML) 1.0, 6-October-2000,
<http://www.w3.org/TR/2000/REC-xml-20001006.pdf>

OGC Web Services Stateless Catalog Profile, version 0.0.6, 29-August 2001.
<http://mcmcweb.er.usgs.gov/catalog/docs/2001-062StatelessCatalogProfile.pdf>

OpenGIS Catalog Services Specification, version 2.0, 11-May-2004
<http://www.opengeospatial.org/specs/>

The National Map catalog web site. <http://mcmcweb.er.usgs.gov/catalog>

Attachment A. Reading XML Responses with Software

The National Map viewer client is the first example of a web application that is driven by *The National Map* catalog service. The viewer software reads a `getRecords` response with a Java library generated with Java Architecture for XML Binding (JAXB), Early Access. Sample code from this application, and the JAXB EA jar file, can be retrieved from http://mcmcweb.er.usgs.gov/catalog/api/samples/catalog_web_app_examples.zip.

If the client is running in a servlet container, it is important that JAXB 1.0 (or later) not be loaded by the container when the container starts. If you are using the Tomcat container bundled with the JWSDP this can be accomplished by renaming the `jaxb-X.X` directory in your distribution to something like `jaxb-X.X-dist`. The `getRecords` reader and JAXB EA jar files should be placed in your web applications lib directory.

Attachment B. Example response

The service request

[http://nmcatalog.usgs.gov/catalog/CatalogQuery.html /catprod/start?request=getrecords](http://nmcatalog.usgs.gov/catalog/CatalogQuery.html/catprod/start?request=getrecords)

can be typed into a browser address window. It returns an XML document, which is a convenient format for machine parsing, but is also human-readable. This example is an unconstrained request for Catalog records. It essentially says “return all information about all services and all layers.” The resulting XML document would be some 350 pages long if printed. The body of this paper describes how to constrain such requests to return more limited and focused record sets.

Following are some excerpts from the response to this request to illustrate the types of information that can be retrieved through the catalog service.

Service Information

For each service registered in the catalog, a block similar to the following is returned:

```
- <Service>
  <Version>1.1.1</Version>
  <Title>Tahoe Pilot WMS</Title>
  <Abstract />
  <Status>PUBLIC</Status>
  <PartnerName>Tahoe Partners</PartnerName>
  <PartnerWebURL>http://mapsonline.wr.usgs.gov/nm_tahoe_feature/
    metadata/Lake_Tahoe_Partners.html</PartnerWebURL>
  <GetCapabilitiesURL>http://mapsonline.wr.usgs.gov/ogcwms/servlet/
    com.esri.ogc.wms.WMSServlet?servicename=WMS_tahoe_pilot
  </GetCapabilitiesURL>
  <GetMapURL>http://mapsonline.wr.usgs.gov/ogcwms/servlet/
    com.esri.ogc.wms.WMSServlet?servicename=WMS_tahoe_pilot
  </GetMapURL>
  <GetFeatureInfoURL>http://mapsonline.wr.usgs.gov/ogcwms/servlet/
    com.esri.ogc.wms.WMSServlet?servicename=WMS_tahoe_pilot
  </GetFeatureInfoURL>
  <BackingURL>http://mapsonline.wr.usgs.gov/servlet/
    com.esri.esrimap.Esrimap?ServiceName=tahoe_pilot</BackingURL>
  <BackingService>ESRI-ArcIMS</BackingService>
  <BackingVersion>1.1</BackingVersion>
```

The name of this service is “Tahoe Pilot WMS.” It is owned by the partner “Tahoe Partners.” The service has status PUBLIC, meaning its data are available for display in *The National Map* viewer.

The URLs to the provider home page and the service itself are the most important pieces of information. These allow applications to connect to the service and retrieve its GIS data for display or analysis.

Layer Information

One service can have any number of GIS layers. Within the `<Service>` tag block are sub-blocks with information about each layer. For example:

```
<Layer id="50" requested="true" recommended="true"
  transparency="false" queryable="false" downloadable="true">
  <Name>NLC_image</Name>
  <Title>Tahoe National Land Cover</Title>
  <PartnerName>Tahoe Partners</PartnerName>
  <PartnerWebURL>http://mapsonline.wr.usgs.gov/nm_tahoe_feature/
    metadata/Lake_Tahoe_Partners.html</PartnerWebURL>
  <Resolution>0</Resolution>
  <MinScale>0.0000214151486530036</MinScale>
  <MaxScale>0.0048</MaxScale>
  <TimePeriod>01-jan-1900</TimePeriod>
<Classification id="7" zdepth="208010.00005195">
  <Name>LAND COVER</Name>
  <Theme>LAND USE/LAND COVER</Theme>
</Classification>
<SRS>54008</SRS>
<SRS>4326</SRS>
<LatLon>Undefined</LatLon>
<BoundingBox>Undefined</BoundingBox>
<Status>PUBLIC</Status>
<Type>RASTER IMAGE</Type>
<MetadataURL>http://landcover.usgs.gov/
  nationallandcover.html</MetadataURL>
<LogoURL />
<SLDURL />
<LegendURL />
<BackingName>NLC_image</BackingName>
<Description>Image covering the project area showing the
  type of surface coverage at a resolution of 30 meters per
  pixel.</Description>
</Layer>
```

The “name” field is what the service calls this layer (NLC_image). The “title” field is populated by USGS personnel, and is used by USGS applications such as the viewer to describe this layer. In this case, the string “Tahoe National Land Cover” is used by the viewer to label this layer. Other information includes the min and max viewscales for the layer, the display status, metadata URL, and the spatial reference systems the service supports for this layer.

Much of this information is supplied by the WMS serving the data, and is stored in the USGS Catalog database through a semi-automated harvest process. However, some of the information, such as the viewscale values and metadata URL, are added and maintained by the USGS to assist *The National Map* applications.

Field definitions

The explanations of the two examples above are obviously not complete. A complete and formal data dictionary of the fields is needed. The documentation problem has two parts:

1. Most of the fields returned in catalog service responses correspond directly to a field in the catalog database. When the database data dictionary is fully populated, the definitions of these fields can be derived directly from the database. However, the names of the database fields are not exactly the same as the names of the service fields in all cases. Therefore, some cross-walk between field names will be needed.
2. In addition, the service returns some fields that are derived or calculated from information in the database. These field names therefore do not directly correspond to any database field, and their definitions will need to be maintained separately from the database.

Concurrent with the release of version 2.0.0 of the catalog service, data dictionary documentation was added to the catalog service DTD. Definitions of the catalog service fields can therefore be obtained with a describeRecord request:

<http://nmcatalog.usgs.gov/catalog/CatalogQuery.html/catprod/start?request=describeRecord>

This documentation method is not completely satisfactory. It is one more static document that must be manually kept consistent with the database and other documentation. On the other hand, the DTD is a technically critical component of the overall system; since the DTD fields **must** be kept current and consistent, it should be relatively easy to keep the field definition comments up to date as well.

Questions about specific questions about the meanings of fields, or suggestions for improving the DTD documentation, can be sent to the Catalog Support Team at USGScatalog@usgs.gov.

Attachment C. Summary of Differences Between Catalog Service Versions 1.1.0 and 2.0.0

In any software system, adding new features while retaining backward compatibility creates problems and requires compromises. Through 2004 it was obvious that the early versions of the catalog service needed additional requestTypes.

Version 2.0.0 of the catalog service adds four requestTypes. Implementing these in a way that does not break any existing applications is both critically important and fairly difficult.

Version 2.0.0 of the catalog service implements the idea of a capability specification to clearly specify which version of the service an application intends to use. The default is the original 1.1.0 version, so if requests to the service do not specify a version, 1.1.0 is assumed. If an application wants to use version 2.0.0, the application must specify this within each request string. A specification is selected by adding

version=<specification version>

parameter to the query string (see section 4 in the report body). If this parameter is missing, the service assumes 1.1.0. Version 1.1.0 may also be specified explicitly.

New specifications always support the request values of previous versions and never remove elements or attributes from previous version response formats. Rather, new versions add request values and extend responses with optional elements and attributes. This allows applications to use the features of a new specification with little change to existing source code.

The 1.1.0 specification was described in version 0.1.2 of this paper. As described in section 4.2 of this document, version 1.1.0 defined three request parameter values:

1. getCapabilities – returns a capabilities document, essentially a description of what the catalog service can do.
2. describeRecord – returns the DTD of the catalog service.
3. getRecords – returns information contained in the catalog database about services and GIS layers.

Version 2.0.0 adds four new request values:

4. getClassifications – Returns a classification list (that is, a list of subthemes) constrained with an optional querySpec.
5. getElementSets – Returns an element set list. The list contains all element sets that contain at least one element. This request cannot be constrained with a querySpec.
6. getElementStatuses – Returns a Status list containing the domain of element statuses. This request cannot be constrained with a querySpec.
7. getServiceStatuses – Returns a Status list containing the domain of service statuses. This request cannot be constrained with a querySpec.

The following additions were made to the 2.0.0 response DTD:

1. A searchResults element can now contain exactly one Service, Classification, ElementSet or Status list.
2. An optional id attribute was added to the Theme element.