Open Geospatial Consortium, Inc.

OpenGIS® Web Map Server Interface Implementation Specification

Revision 1.0.0

OpenGIS Project Document 00-028

Release Date: 19 April 2000

WARNING: The Open Geospatial Consortium (OGC) releases this specification to the public without warranty. It is subject to change without notice. This specification is currently under active revision by the OGC Technical Committee.

Requests for clarification and/or revision can be made by contacting the OGC at revisions@opengeospatial.org.

This page intentionally left blank.

Copyright 1999, 2000 BBN Technologies Copyright 1999, 2000 Cadcorp Ltd. Copyright 1999, 2000 CubeWerx Inc. Copyright 1999, 2000 IONIC Software s.a. Copyright 1999, 2000 Laser-Scan Limited Copyright 1999, 2000 SICAD Geomatics GmbH & Co. oHG Copyright 1999, 2000 Social Change Online Pty Ltd Copyright 1999, 2000 US Army Engineer Research and Development Center

The companies listed above have granted the Open Geospatial Consortium, Inc. (OGC) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version.

This document does not represent a commitment to implement any portion of this specification in any company's products.

OGC's Legal, IPR and Copyright Statements are found at http://www.opengeospatial.org/about/?page=ipr&view=ipr

NOTICE

Permission to use, copy, and distribute this document in any medium for any purpose and without fee or royalty is hereby granted, provided that you include the above list of copyright holders and the entire text of this NOTICE.

We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

No right to create modifications or derivatives of OGC documents is granted pursuant to this license. However, if additional requirements (as documented in the Copyright FAQ at http://www.opengeospatial.org/about/?page=ipr&view=ipr_faq) are satisfied, the right to create modifications or derivatives is sometimes granted by the OGC to individuals complying with those requirements.

THIS DOCUMENT IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with copyright holders.

RESTRICTED RIGHTS LEGEND. Use, duplication, or disclosure by government is subject to restrictions as set forth in subdivision (c)(1)(ii) of the Right in Technical Data and Computer Software Clause at DFARS 252.227.7013

OpenGIS®, OGCTM, OpenGeospatialTM, OpenLSTM, and Open GIS Consortium, Inc.TM are trademarks or registered trademarks of Open Geospatial Consortium, Inc. in the United States and in other countries.

This page intentionally left blank.

1 PREFACE	6
2 SUBMITTING COMPANIES	6
2.1.1 SUBMISSION CONTACT POINTS	7
2.2 DOCUMENT CONVENTIONS	7
2.3 REVISION HISTORY	
2.4 CHANGES TO THE OpenGIS [®] ABSTRACT SPECIFICATION	ON8
3 OVERVIEW	9
4 BACKGROUND	
5 ARCHITECTURE	
5.1 Overview	
5.2 Components and their relationship to each other	
6 NORMATIVE DESCRIPTION	
6.1 Version Numbering and Negotiation	19
6.1.1 Version Number Form	19
6.1.2 Appearance in Requests and Capabilities	19
6.1.3 Version Changes	19
6.1.4 Negotiation Rules	19
6.2 The Interfaces of a Map Server	20
6.2.5 General Request Rules	20
6.2.6 General Response Rules (HTTP)	21
6.2./ Capabilities (required)	21
0.2.0 Wap (required)	22
6.2.9 FeatureInfo (optional)	29
7 REFERENCES	32
8 WEB MAPPING TESTBED PARTICIPANTS	33
9 GLOSSARY	2.5
A CAPABILITIES DTD (Normative)	38
A.1 Capabilities DTD	38

1 PREFACE

This document is a formal description of the accomplishments of the Open GIS Consortium Web Mapping Testbed (WMT). The authors are indebted to those organizations that sponsored and/or participated in the WMT. Section 8 lists the WMT participants.

2 SUBMITTING COMPANIES

The following companies submitted this implementation specification to the OGC as a Request For Comment:

BBN Technologies Cadcorp Ltd. CubeWerx Inc. IONIC Software s.a. Laser-Scan Limited SICAD Geomatics GmbH & Co. oHG Social Change Online Pty Ltd US Army Engineer Research and Development Center

2.1.1 SUBMISSION CONTACT POINTS

All questions regarding this submission should be directed to the editor or the submitters:

Editor

Allan Doyle <u>International Interfaces, Inc.</u> 946 Great Plain Ave. PMB-182 Needham, MA 02492 USA adoyle@intl-interfaces.com

Submitting Companies

Don Dietrick <u>BBN Technologies</u> 10 Moulton Street Cambridge, MA 02138 USA ddietrick@bbn.com

Adam Gawne-Cain <u>Cadcorp Ltd</u> Sterling Court Norton Road Stevanage, HERTS SG1 2JY United Kingdom adam@cadcorpdev.co.uk

Edric Keighan <u>CubeWerx, Inc.</u> 200 Montcalm, Suite R-13 Hull, Quebec J8Y 3B5 Canada ekeighan@cubewerx.com

Vincent Dessard <u>IONIC Software</u> 128 Avenue de l'Observatoire B-4000 LIEGE Belgium vincent.dessard@ionicsoft.com Adrian Cuthbert Laser-Scan Limited Cambridge Science Park Milton Road Cambridge CB4 0FY United Kingdom adrian@lsl.co.uk

Dr. Peter Ladstaetter <u>SICAD Geomatics GmbH & Co. oHG</u> Otto-Hahn-Ring 6 D-81739 Munich Germany peter.ladstaetter@sicad.de

Rob Atkinson Social Change Online Pty Ltd 6A Nelson St Annandale, NSW 2038 Australia rob@socialchange.net.au

Jeff Harrison <u>US Army Engineer Research and</u> <u>Development Center</u> 7701 Telegraph Road Alexandria, VA 22315 USA jharris@tec.army.mil

2.2 DOCUMENT CONVENTIONS

This document contains sections that are "informative", meaning they serve as explanation and background. Other sections are "normative", meaning they contain the formal specification against which conformance testing can be done. The normative sections are labeled as such.

In the sections labeled as normative, 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 [15].

2.3 REVISION HISTORY

The specification described in this document has already undergone several revisions during the WMT Phase I development cycle. As submitted to the OGC by the submitters, the revision number was 0.9.3. This version is 1.0.0.

List of revisions

0.0.1 through 0.0.6 – WMT development versions, March-September 1999

- 0.1 WMT demonstration on September 10, 1999
- 0.9 RFC submission on November 15, 1999
- 0.9.3 RFC resubmission on January 17, 2000
- **1.0.0** Submission for OGC TC Approval.

2.4 CHANGES TO THE OpenGIS[®] ABSTRACT SPECIFICATION

No changes to the Abstract Specification other than the additions noted in this document are required for this specification.

This specification should form the basis of a new Web Mapping Volume of the OGC Bookshelf.

3 OVERVIEW

This is a complex document, though the basic technology and concepts described are quite simple. The complexity of the document reflects the need to specify as completely as possible both the essential model of Web Map Server Interfaces as well as a specific implementation of that model using Hypertext Transfer Protocol (HTTP) [8] and eXtensible Markup Language (XML) [9].

Section 3 OVERVIEW (informative) Provides a simplified description and illustrative examples of Web Mapping.

Section 4 BACKGROUND (informative) Provides historical background and technical context for the specification.

Section 5 ARCHITECTURE (informative) Describes the Web Map Server architecture and relates the components to each other.

Section 6 NORMATIVE DESCRIPTION (normative) Specifies the Web Map Server Interfaces in detail and is the primary normative part of this document.

Sections 7, 8, and 9 (informative) Comprise the bibliography, list of participants, and glossary.

Annex A CAPABILITIES DTD (Normative) Contains the XML DTD, which specifies the machine-readable formatting of Map Server capability information.

It is expected that future implementation models for other distributed computing platforms will be contained in additional Annex attachments to this specification.

Before the reader becomes inundated with detail, this overview provides a brief, simplified description.

This document describes a Web Map Server (or just Map Server). A Map Server can do three things. It can:

- 1. Produce a map (as a picture, as a series of graphical elements, or as a packaged set of geographic feature data),
- 2. Answer basic queries about the content of the map, and
- 3. Tell other programs what maps it can produce and which of those can be queried further.

To first order, a standard web browser can ask a Map Server to do these things just by submitting requests in the form of Uniform Resource Locators (URLs) [12]. The content of such URLs depends on which of the three tasks is requested. All URLs include a Web Mapping Technology specification version number and a request type parameter. In addition,

- 1. To produce a map, the URL parameters indicate which portion of the Earth is to be mapped, the coordinate system to be used, the type(s) of information to be shown, the desired output format, and perhaps the output size, rendering style, or other parameters.
- 2. To query the content of the map, the URL parameters indicate what map is being queried and which location on the map is of interest.
- 3. To ask a Map Server about its holdings, the URL parameters includes the "capabilities" request type.

Each of these will be described in further detail later. We first provide some sample URLs and their resulting maps:

This requests a US National Oceanographic and Atmospheric Administration AVHRR image:

http://a-map-co.com/mapserver.cgi?WMTVER=1.0.0&REQUEST=map& SRS=EPSG%3A4326&BBOX=-97.105,24.913,78.794,36.358& WIDTH=560&HEIGHT=350&LAYERS=AVHRR-09-27%3AMIT-mbay&STYLES=default& FORMAT=PNG&BGCOLOR=0xFFFFF&TRANSPARENT=TRUE& EXCEPTIONS=INIMAGE&QUALITY=MEDIUM



Figure 3-1 NOAA AVHRR Image of the Gulf of Mexico

This requests three layers, "built up areas", political boundaries, and coastlines:

http://b-maps.com/map.cgi?WMTVER=1.0.0&REQUEST=map& SRS=EPSG%3A4326&BBOX=-97.105,24.913,78.794,36.358& WIDTH=560&HEIGHT=350&LAYERS=BUILTUPA_1M%3ACubeWerx, COASTL_1M%3ACubeWerx,POLBNDL_1M%3ACubeWerx &STYLES=0XFF8080,0X101040,BLACK&FORMAT=PNG&BGCOLOR=0xFFFFFF& TRANSPARENT=FALSE&EXCEPTIONS=INIMAGE&QUALITY=MEDIUM



Figure 3-2 Political, Coastline, and Populated Areas, Southeastern United States

Notice that in both of these URLs the spatial information is identical: "SRS=EPSG%3A4326& BBOX=-97.105,24.913,78.794,36.358& WIDTH=560&HEIGHT=350". Because both maps were produced with the same bounding box, spatial reference system, and output size, the results can actually be overlaid by placing the latter map on top of the former. By enabling the use of image formats that provide for transparency information, maps that are meant to be overlaid over other maps can be produced by Map Servers. In this example, background areas of the second map are transparent (because the URL parameter "TRANSPARENT=TRUE" was supplied). Figure 3-3 shows the result of overlaying Figure 3-2 on top of Figure 3-1 to produce a map from the result of two separate Map requests. Finally, note that in this example the two maps were requested from different Map Servers. By standardizing the way in which maps are requested, clients of Map Servers can tailor which layers to request from which servers, thus building up maps that would not have been practical to assemble without the Web Mapping Interface Specification.



Figure 3-3 Combined AVHRR Image and Political/Cultural Map

If either of these maps were queryable, a client could request information about a feature on the map by adding to the map URL two additional parameters specifying a location (as an X, Y offset from the upper left corner).

Because each Map Server is likely to have different kinds of information for which it can produce maps, each Map Server must be able to provide a machine-parseable list of its capabilities. That enables the construction of searchable catalogs that can direct clients to particular Map Servers.

4 BACKGROUND

Web Mapping within the OGC was first described in "WWW Mapping Framework" [1]. The first OGC consensus position of the WWW Mapping SIG, a core task force of the OGC, is described in "User Interaction with Geospatial Data" [2]. From these documents, as well as from "A Web Mapping Scenario" [3], an OGC-sponsored initiative was begun. Known as the Web Mapping Testbed (WMT), this initiative was first described in the Request For Technology (RFT) [4] and then in the Request for Quotation (RFQ) [5].

Figure 4-1 shows the original decomposition of the portrayal process from [2]. There are four distinct processing stages that occur as data is moved from the data source to the display where it becomes visible to a user.



Figure 4-1 Portrayal

In Figure 4-2 these four components are pictured as services, each with interfaces which can be invoked by clients of that service. Originally it was thought that the location of the services with respect to a network was sufficient to define the terms "thin", "medium", and "thick" client. In other words, if a user is using a computer connected to a network, and that computer contains just the display service, then that user would be said to be using a thin client. If the user's computer additionally contained a render service, then that user would be said to be using a medium client. And finally, if the user's computer also contained the display element generator service, that would indicate the user is using a thick client. After some consideration, it was decided that while this distinction may be somewhat helpful in describing web mapping, the terms "thick client" and "thin client" were already encumbered by very imprecise definitions used in marketing literature and were therefore not suitable for continued use in the WMT.



Figure 4-2 Services View of Portrayal

Upon examination of the responses to the RFT [4], it became apparent that web mapping could be thought of in terms of what kind of information crosses the boundary between a client computer and a web server (and additionally, how that information is packaged). This led to the three "cases" – namely the "Picture Case" (Figure 4-3), the "Graphic Element Case" (Figure 4-4), and the "Data Case" (Figure 4-5).

In the Picture Case, what travels across the Internet in response to a client's request is essentially a picture of a map, such as a GIF, JPEG or PNG image constructed by a Map Server.





Figure 4-3 The Picture Case

Figure 4-4 The Graphic Element Case



Figure 4-5 The Data Case

In the Graphic Element Case, what travels between the web server and the client is a packaged set of individual elements, typically already in a projected reference system, and with already defined symbolization for geographic features. Thus, a road might be a two-pixel thick polyline, colored red, a lake could be a blue polygon, and so on. Of course, some of the graphic elements could themselves be pictures in the sense of a bitmap or pre-drawn fragment of a map, so the graphic element case may or may not also include the picture case as a subset. In particular, in WMT Phase I, teams used SVG and WebCGM as graphic element formats.

Finally, the data case provides the ability to send geographic feature data from the server to the client. In WMT Phase I, experiments with the use of XML as the data encoding language resulted in a draft OGC Recommendation for "Geographic Markup Language," an encoding of Open GIS Simple Features [6] in XML. Current opinion on the data case within the Web Mapping Testbed community is that it really should be called the "Feature Case" (the rest of this document uses the latter term).

The WMT Phase I process culminated in a demonstration of web mapping centered around interfaces developed during the process and incorporating the Open GIS Catalog Services [7].

5 ARCHITECTURE

5.1 Overview

This specification covers primarily the Picture Case. The Graphic Element and Feature Cases have not yet been as extensively tested by working implementations. Figure 5-1 shows the essential components of the architecture in the context of the diagrams described above. The figure shows two gray boxes, the "Viewer Client" and the "Map Server". The interfaces covered by this specification are those that a Map Server exposes to a Viewer Client (or to any client equipped to invoked the same interfaces, such as the Service Registry described below). There are three operations that a Map Server can perform. In the interest of focusing on these three interfaces, the details of the internal services of a Map Server or Viewer Client will no longer be discussed in terms of the Display, Render, Display Element Generator, or Filter Services. While these services are still helpful in understanding the overall model, they are not needed to describe this specification. (It is expected that the services we are leaving behind here will reappear in the course of WMT Phase II or other testbed activities).



Figure 5-1 WMT Implementation of the Picture Case

The three interfaces on a Map Server are introduced in Figure 5-2. These are the Map, FeatureInfo, and Capabilities interfaces. Individually, they are sometimes informally referred to as GetMap, GetFeatureInfo, and GetCapabilities. The Get- form should be considered synonymous with the non-Get- form. Figure 5-2 introduces another element, the Service Registry. In WMT Phase I, a Service Registry was constructed using the OpenGIS Catalog Services specification [7]. It was populated by using the results of invoking the Capabilities request on each functioning Map Server, and it provided a registry of Map Servers to Viewer Clients capable of issuing queries and interpreting the results of catalog searches. The Service Registry interfaces, though not part of this specification, are mentioned to illustrate the point that this specification meshes well with existing OpenGIS specifications.



Figure 5-2 Map Server Interfaces

5.2 Components and their relationship to each other

Figure 5-2 provided a glimpse at the component names that embody the interfaces discussed in this document. Section 6.2.5.1 declares that a Map Server may have its Capabilities interface provided by a different server. The three interfaces--Map, FeatureInfo, and Capabilities--are interfaces on a Web Map Server, but are so in only a very loose sense. There is no requirement that a single piece of software implement all three interfaces (and, of course, the Feature Info interface is optional, cf. Section 6.2.9). When implemented in an HTTP environment, typically the interfaces are reached via a standard HTTP server which has the ability to dispatch requests to specific applications (for instance using the Common Gateway Interface [CGI] mechanism). These applications can be identified from portions of the URL that carry the Web Map Server requests described in this document. There is no requirement in this specification dictating the format or content of those identifying portions. Rather, this document describes only the packaging of the parameters that are ultimately received by the Map Server software.

During the course of the Web Mapping Testbed, a component type arose known as a "Cascading Map Server". The Cascading Map Server is a component that behaves like a client of other Map Servers and behaves like a Map Server to other clients. This provides a convenient mechanism for aggregating the capabilities of the individual Map Servers into one logical "place." Additionally, a Cascading Map Server can perform additional services. Consider a Cascading Map Server that can convert many different graphics formats (e.g., GIF, PNG, JPEG, etc. into GIF format). Then Viewer Clients that can only display GIF could still benefit from the output of map servers that produce only JPEG or PNG. Similarly, a Cascading Map Server might perform coordinate transformations on behalf of other servers.

6 NORMATIVE DESCRIPTION

6.1 Version Numbering and Negotiation

6.1.1 Version Number Form

The specification version number contains up to three positive integers, separated by decimal points. The following forms are possible: "x", "x.y", or "x.y.z".

6.1.2 Appearance in Requests and Capabilities

The version number appears in at least two places: it is one of the arguments of a Capabilities, Map or FeatureInfo request (e.g., the WMTVER parameter in an HTTP URL), and it is an attribute of the <WMT_MS_Capabilities> element in the Capabilities XML described in Annex A of this document.

6.1.3 Version Changes

The version number **shall** be changed with each revision of this specification. The number **shall** increase monotonically and **shall** comprise no more than three integers separated by decimal points, with the first integer being the most significant. There may be gaps in the numerical sequence.

6.1.4 Negotiation Rules

In response to a Capabilities, Map or FeatureInfo request containing a version number, a map server **must** either respond with output that conforms to that version of the specification, **or** negotiate a mutually agreeable version if the requested version is not implemented on the server. If no version number is specified in the request, the server **should** respond with the highest version it understands and label the response accordingly (for those output formats which include a protocol version number field, such as Capabilities XML).

Version number negotiation occurs as follows:

- 1) If the server implements the requested version number, the server **shall** send that version.
- 2) If a version unknown to the server is requested, the server **shall** send the highest version less than the requested version and label it accordingly.

2b) If the client request is for a version lower than any of those known to the server, then the server **shall** send the lowest version it knows and label it accordingly.

3) If the client does not understand the new version number sent by the server, it may either cease communicating with the server or send a new request with a new version number that the client does understand but which is less than that sent by the server (if the server had responded with a lower version).

3b) If the server had responded with a higher version (because the request was for a version lower than any known to the server), and the client does not understand the proposed higher version, then the client **may** send a new request with a version number higher than that sent by the server.

4) The process is repeated until a mutually understood version is reached, or until the client determines that it will not or cannot communicate with that particular server.

Example A: server understands versions 1, 2, 4, 5 and 8. Client understands versions 1, 3, 4, 6, and 7. Client requests version 7. Server responds with version 5. Client requests version 4. Server responds with version 4, which the client understands, and the negotiation ends successfully.

Example B: server understands versions 4, 5 and 8. Client understands version 3. Client requests version 3. Server responds with version 4. Client does not understand that version or any higher version, so negotiation fails and client ceases communication with that server.

6.2 The Interfaces of a Map Server

As previously stated, the three interfaces specified by this document are Capabilities, Map, and FeatureInfo. Each of these is described in this section. The descriptions tell how to package the requests as URLs. The specification also allows the packaging of requests as HTTP POST operations. In that case, the parameter names and their values are send as part of the POST operation to a server with a given URL.

The tables below present the requests in standard CGI style. This means the parameters consist of name-value pairs in the form "name=value" and the pairs are separated by the "&" character.

6.2.5 General Request Rules

A Map Server request, in an HTTP GET or POST request **shall** be packaged per RFC 2616 [8] as follows:

A prefix followed by a set of valid name/value pairs

Additional parameters **may** be included in the request only as described in Section 6.2.5.1.4 and 6.2.5.1.5.

Parameters consisting of lists, such as LAYERS and STYLES **shall** use the comma (",") as the separator between items in the list. Additional white space **shall not** be used to delimit list items.

URL Component	Description
http://server_address/path/script?	URL prefix of server
	(section 6.2.5.1.1)
WMTVER=1.0.0	Request version
	Required
	(section 6.2.5.1.2)
REQUEST= <name></name>	Request name
	Required
	(section 6.2.5.1.3)
Additional parameters	Use for Map (section 6.2.8) and FeatureInfo
•	(section 6.2.9)
Vendor-specific parameters	Vendor-specific parameters
	Optional
	(section 6.2.5.1.5)

List items may be empty.

Table 6.1- A general Map Server Request

6.2.5.1 Parameters Shared by All Requests

6.2.5.1.1 Prefix

The prefix shall conform to the description in RFC 2616 Section 3.2.2 "http URL" [8]

6.2.5.1.2 WMTVER

WMTVER specifies the protocol version number, and allows negotiation as described in Section 6.1.

The format of WMTVER values **shall** be as described in section 6.1.1.

6.2.5.1.3 REQUEST

REQUEST indicates which of the three interfaces is being invoked, and **shall** take on the values "capabilities", "map", or "feature_info".

Only the parameters in the query string after the prefix are mandated by this specification; the prefix portion is implementation-dependent.

6.2.5.1.4 Additional Parameters

Parameters are expressed in terms of name/value pairs. They are designed to be easily composed into a CGI query packaged in a URL. (i.e., to be issued via an HTTP GET request). In principle, they could also be packaged as an HTTP POST request; future revisions of this specification are expected to include an XML encoding of the parameters.

Parameter names shall not be case sensitive.

Parameter values **shall** be case sensitive.

Parameters in a request may be specified in any order.

A Map Server **must** be prepared to encounter parameters that are not part of this specification. In terms of producing results per this specification, a Map Server **shall** ignore such parameters.

6.2.5.1.5 Vendor-Specific Parameters

Finally, the requests allow vendors to specify additional optional parameters which will enhance the results of a request. A Map Server **must** produce a valid map even if vendor-specific parameters are missing or malformed (i.e., the Map Server **shall** supply a default value), or if vendor-specific parameters are supplied that are not known to the Map Server (i.e., the Map Server **shall** ignore unknown parameters in the URL).

A Map Server **may** declare vendor-specific parameters (VSPs) within the internal DTD of its Capabilities XML. A Map Server **may** choose not to advertise some or all of its VSPs. Clients **may** read the internal DTD and formulate requests using any VSPs advertised therein.

Vendors **should** choose parameter names with care when defining vendor-specific parameters. This is because the Map request (section 6.2.8) is duplicated inside the FeatureInfo request (section 6.2.9) which could, for poorly chosen parameter names, result in parameters being included twice in one request.

6.2.6 General Response Rules (HTTP)

As a practical matter, in the WWW environment, a client should be prepared to receive either a valid result, or nothing, or any other result. This is because the client may itself have formed a non-conforming request that inadvertently triggered a reply by something other than a Map Server, because a client may have encountered a non-conforming Map Server, etc..

6.2.7 Capabilities (required)

6.2.7.1 General

The Capabilities interface is designed to provide clients of Map Servers with a machine-parseable listing of what interfaces a Map Server supports, what map layers it can serve, what formats it can serve them in, and so on.

A Map Server **shall** either implement the Capabilities interface **or** ensure that there exists another web server that provides the Capabilities response on the Map Server's behalf. If there is no means of a client accessing the capabilities XML, then the Map Server is not a conforming Map Server.

Internally, the Map Server **may** choose either to generate the Capabilities response dynamically **or** to ship return a static XML file in response to the Capabilities request.

6.2.7.2 Request

The interface has two input parameters – the request name and the request version. Table 6.2 shows a the Capabilities request.

URL Component	Description
http://server_address/path/script?	URL prefix of server
	(section 6.2.5.1.1)
WMTVER=1.0.0	Request version
	Required
	(section 6.2.5.1.2)
REQUEST=capabilities	Request name
	Required.
Vendor-specific parameters	(section 6.2.5.1.5)

Table 6.2 - The Capabilities Request.

6.2.7.3 Result

Output **shall** be in the form of XML, which **must** be valid against the Capabilities DTD (discussed in Annex A) defined for that request version (or for a different version using the negotiation scheme described in Section 6). In the case of HTTP, the MIME type [11] of the returned XML **shall** be "text/xml". Note that text/xml is favored by RFC 2376 [16] when the content of the message is mainly plain text (e.g., no 16-bit chars, but with HTTP even UTF-16 is permitted).

6.2.7.4 Exceptions

If a server does not return XML which can be validated against the DTD in Annex A, then the Map Server **is not** a conforming Map Server.

If there is no means of a client accessing the capabilities XML, then the Map Server **is not** a conforming Map Server.

In the event a Map Server receives a request which is not described by that Map Server's capabilities XML, the Map Server **must** throw an exception.

Exception responses in the case of an exception in the Capabilities request are not specified. In this case, it is encouraged that the Map Server return an XML fragment as defined in section 6.2.8.2.8.

6.2.8 Map (required)

6.2.8.1 General

The Map interface is designed to provide clients of a Map Server with pictures of maps, possibly from multiple Map Servers.

Note that in this specification, the Map request is capable of returning Graphic Elements and Features. The intent of the WMT sponsors is to fully flesh this capability out in a future version of the specification.

Upon receiving a Map request, a Map Server **must** either satisfy the request **or** throw an exception in accordance with the exception instructions contained in the Map request (as described more fully below).

6.2.8.2 Request

Table 6.3 describes the Map Request.

URL Component	Description
http://server_address/path/script?	URL prefix of server
	(section 6.2.5.1.1)
WMTVER=1.0.0	Request version
	Required
REQUERT	(section 6.2.5.1.2)
REQUEST=map	Request name
LAVEDS-lover list	Required.
LAYERS=layer_list	Comma-separated list of one or more map
	layers. Required.
STYLES=style_list	Comma-separated list of one rendering
	style per requested layer.
	Required.
SRS=srs_identifier	Spatial Reference System
	Required.
BBOX=xmin,ymin,xmax,ymax	Bounding box corners (lower left, upper
- ,,,,,,	right) in SRS units.
	Required.
WIDTH=output_width	Width in pixels of map picture.
	Required.
HEIGHT=output_height	Height in pixels of map picture.
	Required.
FORMAT=output_format	Output format of map.
	Required.
TRANSPARENT=true_or_false	"TRUE" "FALSE"
	If TRUE, then the background color of the
	picture is to made transparent if the image
	format supports transparency. Optional. Default=FALSE
BGCOLOR=color_value	A hexadecimal red-green-blue color value
	(0xrrggbb) for the background color.
	Optional; default=0xFFFFF
EXCEPTIONS=exception_format	The format in which exceptions are to be
	reported by the Map Server.
	Optional; default=INIMAGE
Vendor-specific parameters	(section 6.2.5.1.5)

Table 6.3 – The Map Request.

6.2.8.2.1 LAYERS/STYLES

The model of a Map Server in this specification is one which publishes its ability to return drawn maps rather than one which publishes its ability to access specific data holdings. The way this is accomplished is via pairs of parameters, namely "Layers" and "Styles". A Layer is a particular type of georeferenced information. A Style indicates the appearance of that layer. A map server advertises in its Capabilities XML what Layers it offers and, optionally, what Styles are available for each layer.

A Layer is advertised by a <Layer> element in the Capabilities XML. The Layer Name indicates what value **must** be used in a Map request LAYERS parameter to request that Layer. The Title is a human-readable string for presentation in a menu and **shall** be present. Optionally, an Abstract and Keywords **may** be specified. See the Capabilities DTD in Annex A for further details.

Zero or more Styles may be advertised for a Layer or collection of Layers using <Style> elements, which also have Name, Title and other components. The Name is used in the Map request STYLES parameter. If only a single style is available, that style is known as the "default" style and

need not be advertised by the server. A client requests the default style using a null value as in "STYLES=".

In a Map request, the layers and styles parameters are expressed in the form of a commaseparated list, where there is a left-to-right correspondence of the layer values and the style values. A Map Server **shall** render the requested layers by drawing the **leftmost** in the list **bottommost**, the next one over that, and so on.

No controlled vocabulary has been defined, so at present Layer and Style names and titles are arbitrary. However, a Map Server **shall** advertise the names it uses/offers/recognizes, and a client **shall not** request a layer or style that has not been advertised. A server **shall** throw an exception if an unadvertised layer or style is requested.

Figure 6-1shows three examples of named Layers and Styles. In part A, there is a layer named "GRATICULE" which provides pictures of graticules (i.e., grids at evenly-spaced coordinate values). This layer does not have any selectable styles. A Map Server does not need a data source at all to provide this kind of a layer. Instead, all it needs to do is generate a picture with properly spaced graticule lines, say every 10 degrees of latitude and every 10 degrees of longitude. In part B, there is a layer named "CONTEXT" which provides pictures of maps having only simple political and ocean boundaries. This layer could be generated from a data collection such as Digital Chart of the World or some other suitable small-scale data source. This layer also has only a default style. In part C, there is a layer named "ATMAX" showing maximum atmospheric temperature. This layer has three associated styles, named "POINTS", "CONTOURS", and "REFERENCE" in this example. The Map Server draws maps using the "GLOBE" data source when a request is made for LAYERS=ATMAX and STYLES=POINTS or LAYERS=ATMAX and STYLES=CONTOURS. However, if a request is made for LAYERS=ATMAX and STYLES=REFERENCE, the Map Server uses numerical weather prediction models to produce the map being returned.

The use of named layers and styles is somewhat analogous to stored functions in a database management system. In the case of a Map Server, it **shall** advertise these layer and style names in the Capabilities response. The Capabilities response also provides a means to publish layer and style information in the form of titles and abstracts, meant to provide users with more information when searching for and choosing MapServers and layer/style combinations.



Figure 6-1 Layers and Styles

6.2.8.2.2 SRS

In order to receive a picture of a map in which every pixel of the returned map has a known relationship to the original geographic area being shown, several parameters are used to set what is known as the spatial context. The parameters are SRS, BBOX, WIDTH, and HEIGHT.

<u>SRS</u> - The SRS (Spatial Reference System) is a text parameter that names a horizontal coordinate reference system code. The name includes a namespace prefix, a colon, a numeric identifier, and possibly a comma followed by additional parameters. This specification defines two namespaces: EPSG and AUTO. Map Servers are *not* required to support all possible projections, but **shall** advertise in their Capabilities XML those projections which they do offer and **shall** accept Map requests for all advertised projections.

In the Capabilities XML, there is a top-level Layer element which encapsulates all the other Layers. this parent element **shall** include a list of SRS values that every layer available from the Map Server supports (in other words, the set of values in the parent SRS list is the intersection of the set of values in the individual Layer SRS lists). This list **may** be empty if no SRS is common to all layers. Any SRS in the parent list is inherited by all child Layers. Subsidiary Layer elements **may** contain additional SRS supported by that Layer or by its children. Lower-level layers **may** repeat an SRS mentioned from a parent; such duplication **must** be ignored by clients.

If a request contains an SRS and one or more layers which do not support that SRS, a server **must** throw an exception.

The form of the EPSG and AUTO SRS codes is as follows

EPSG: The EPSG namespace makes use of the European Petroleum Survey Group tables
[13], which define numeric identifiers (the EPSG "horiz_cs" code) for many common projections
and which associate projection or coordinate metadata (such as measurement units or central
meridian) for each identifier. For example, EPSG code 4326 refers to WGS84
latitude/longitude coordinates in degrees with Greenwich as the central meridian. An SRS
name in the EPSG namespace includes only the prefix and the identifier (e.g., "EPSG:4326").
This format is used both as the value of the SRS parameter in a Map request and as the value
of an <SRS> element in the Capabilities XML (see Annex A).

AUTO: The AUTO namespace is used for "automatic" projections; that is, a class of projections that includes an arbitrary center point [14]. An SRS name in the AUTO namespace includes an identifier, a code indicating what units are to be used for bounding boxes in that SRS, and values for the central latitude and longitude:

 "AUTO:wm_id,epsg_units,lon,lat". wm_id is one of the projection IDs defined by this specification in [14]; epsg_units is one of the EPSG-defined [13] numbers for units, and lon,lat are longitude,latitude in decimal degrees. For example,
 "AUTO:42003,9001,-100,45" is auto orthographic projection, bounding box in meters, center at 100 West, 45 North. The bounding box for Auto Orthographic projections is specified in a plane perpendicular to the line of sight and tangent to the Earth.

In addition to the Auto Orthographic projection, Auto UTM (Universal Transverse Mercator), Auto Transverse Mercator (with the given central meridian), and Auto Equirectangular (which uses the given central latitude to scale the X axis) are defined. See reference [14] for additional details.

In a Map request, the complete AUTO SRS is specified including latitude, longitude, and units. In Capabilities XML, the latitude, longitude, and units specifier being arbitrary, are omitted.

6.2.8.2.3 BBOX

BBOX – The BBOX (Bounding Box) is a set of four comma-separated decimal, scientific notation, or integer values (if integers are provided where floating point is needed, the decimal point is assumed at the end of the number). These values specify the minimum X, minimum Y, maximum X, and maximum Y ranges, in that order, expressed in units of the SRS of the request such that a rectangular area in the units of the SRS is defined. Where the SRS units are not units of linear measure but are expressed in latitude and longitude, X is taken to mean Longitude and Y is taken to mean Latitude.



Figure 6-2 - BBOX representation

The relation of the BBOX to the image pixel matrix is shown in Figure 6-2. The bounding box goes around the "outside" of the pixels of the image rather than through the centers of the border pixels. In this context, individual pixels have an area.

A BBOX **should not** have zero area.

If a request contains an invalid BBOX (e.g., one whose minimum X >= maximum X or minimum Y >= maximum Y) the server **may** throw an exception.

If a request contains a BBOX whose area is outside of the BBOX (as advertised in the Capabilities XML) for any layer in the LAYERS list, the server **should** return an empty map.

If a request contains a valid BBOX for the given SRS but for which the server can't draw a map (e.g., for which it has no data), then a server **shall** return an empty map.

6.2.8.2.4 WIDTH/HEIGHT -

The WIDTH and HEIGHT parameters are expressed as positive integers. Taken together, they specify a rectangle of pixels into which the resulting map should be drawn and returned. The returned picture, regardless of its return format, **shall** have exactly that width and height in pixels. In the case where the aspect ratio of the BBOX and that of the WIDTH/HEIGHT are different, the Map Server **must** assume that it is to stretch the returned map so that the resulting pixels could themselves be rendered in the aspect ratio of the BBOX. In other words, it should be possible using this definition to request a map for a device whose output pixels are themselves non-square, or to stretch a map into an image area of a different aspect ratio.

If a request is for a vector or feature return format, the WIDTH and HEIGHT values **must** be present and **may** be used by the server as helpful information but this specification currently considers this use to be experimental.

6.2.8.2.5 FORMAT

The value of this parameter is to be taken from the KnownFormats list in the Capabilities DTD. This list is meant to be easily overridden to allow for addition of new formats without having to revise this specification. Annex A.1 contains a list of the formats included at the time this specification was published.

The FORMATS can be divided into four basic groups.

- 1. Picture Formats GIF, JPEG, PNG, TIFF, GeoTIFF, PPM, WBMP
- 2. Graphic Element Formats WebCGM, SVG
- 3. Feature Formats GML.1, GML.2, GML.3
- 4. Other Formats MIME, INIMAGE, BLANK, WMS_XML

The 4th type of format is not used as the output of a Map request, rather, these formats are listed here for use as Map request parameters and FeatureInfo request parameters. They are described in the appropriate sections of this document.

A Map Server is *not* required to offer all of these formats, but the server shall advertise those formats it does support and shall accept requests for any format it advertised. A Map Server may optionally add a new format to the list by redefining the KnownFormats entity as described in the DTD.

6.2.8.2.6 TRANSPARENT

The ability to return pictures drawn with transparent pixels allows results of different Map requests to be layered. The TRANSPARENT parameter can take on two values, **TRUE** or **FALSE**.

When TRANSPARENT is set to **TRUE** and the FORMAT parameter contains a Picture format (Section 6.2.8.2.5), then a Map Server **shall** return, *where permitted by the requested FORMAT*, a result where all of the pixels not drawn into by the Map Server are set to a transparent value.

When TRANSPARENT is set to FALSE, those pixels shall be set to the value of BGCOLOR.

When the FORMAT parameter contains a non-Picture format, the TRANSPARENT parameter **shall** be included in the request but its value **shall** be ignored by the Map Server.

6.2.8.2.7 BGCOLOR

The general format of BGCOLOR is a hexadecimal encoding of an RGB value where two hexadecimal characters are used for each of Red, Green, and Blue color values. The values can range between 00 and FF for each (0 and 255, base 10). The format is 0xRRGGBB and either upper or lower case characters are allowed for RR, GG, and BB values. The "0x" prefix **shall** have a lower case 'x'.

When not set, BGCOLOR defaults to "0xFFFFFF", which is equivalent to "white".

When FORMAT is a Picture format (Section 6.2.8.2.5), a Map Server **shall** render its output on a Picture whose pixels were initially uniformly of the color encoded in BGCOLOR (or 0xFFFFFF if the BGCOLOR parameter is not present in the request).

When FORMAT is a Graphic Element format (Section 6.2.8.2.5), a Map Server should avoid use of that color since elements of that color would not be visible on a background of that color.

6.2.8.2.8 EXCEPTIONS

The EXCEPTIONS parameter dictates how a Map Server shall behave if it encounters an error. The error can either be due to a malformed request, or due to other circumstances. In all cases of error, a Map Server may simply return nothing. However, Map Servers should attempt to produce the exception behavior described in the request.

The behavior is as follows for each possible value EXCEPTIONS can take:

<u>INIMAGE</u>

In the case of a Picture format (Section 6.2.8.2.5), if the EXCEPTIONS parameter is set to INIMAGE, an error message to the user **shall** be graphically returned as part of the content. This would usually take the form of text containing the message being painted into the returned map.

In non-picture output formats, the response to this value is experimental and no specific use is required or suggested by this specification.

<u>BLANK</u>

In the case of a Picture format (Section 6.2.8.2.5), if the EXCEPTIONS parameter is set to BLANK, the Map Server **shall**, upon detecting an error, return an object of the type specified in FORMAT whose content is uniformly "off". In the case of an image format such as GIF or JPEG, that would be an object containing only pixels of one color (the background color if BACKGROUND is specified). In the case of a picture format supporting transparency, if TRANSPARENT=TRUE is specified the pixels **shall** all be transparent.

In non-picture output formats, such as vector-based formats, this specification **suggests** that no visible graphic elements be returned.

WMS XML

In an HTTP environment, the MIME type of the returned XML **shall** be "text/xml" rather than the type requested in the FORMAT parameter. The XML **shall** have the following structure (there is no DTD, so the XML **shall** be well-formed but cannot be validated):

<WMTException version="1.0.0" > error information </WMTException>

Note that the placeholder above ("error information") is meant to contain a plain text description of the problem. The description should not include additional XML.

Additional attributes to the WMTException element **may** be used. The version attribute **shall** contain the WMTVER the Map Server was using (following version negotiation). The "error information" is a placeholder for anything that conforms to well-formed XML.

Note that a client should also be prepared for other returned values and types since there is a possibility that a Map Server is poorly behaved or that a request was directed at a non-Map Server.

6.2.8.3 Result

The return value of a Map request **shall** be the same as the type requested in the FORMAT parameter. In an HTTP environment, the MIME type of the returned value **shall** match the return value **unless** there is no MIME type for that value. There are two exceptions to this which are covered in Section 6.2.8.4

6.2.8.4 Exceptions

A Map Server throwing an exception **shall** either return nothing **or shall** adhere to the value of the EXCEPTIONS parameter. The latter behavior is strongly encouraged.

A Map request **may**, due to circumstances beyond its control, return nothing (in the case of HTTP this condition could result from the HTTP server's behavior and could be caused by a malformed Map request, by an invalid HTTP request, by access violations, or several other conditions) **or**

A Map request **may** return an XML structure containing error information **if** the Map Server detected an error **and** the EXCEPTIONS parameter was set to "WMS_XML". See Section 6.2.8.2.8 for the description of this case.

6.2.9 FeatureInfo (optional)

6.2.9.1 General

The FeatureInfo interface is designed to provide clients of a Map Server with more information about points in the pictures of maps that were returned by previous Map requests. The basic interface provides the ability for a client to specify which pixel is being asked about, which layer(s) should be asked, and what format the information should be returned in. In order to provide a stateless protocol, a Map request is itself one of the parts of a FeatureInfo request. The general use of FeatureInfo is that a user sees the result of a Map request and chooses a point on that map to get more information. The only mechanism, in this specification, to indicate to a Map Server which map the user is looking at is by repeating the Map request to the Map Server (minus the prefix, WMTVER, and REQUEST parts). A Map Server can use this repeated Map request to either look up the previously returned, but saved map, or can rebuild a map. From that map, along with the X,Y position the user chose, the Map Server can (possibly) return additional information about that point.

The actual semantics of how a Map Server decides what to return more information about, or what exactly to return is left up to the Map Server developer. This mechanism, when used in a commonsense (or maybe even in a very clever) way, can provide more benefit to an end-user than simply being able to view maps.

The behavior described above is geared toward the picture case. In the graphic element case, the semantics of FeatureInfo is less defined, and in the feature case, even less so. The intent is to gain experience with this version of the request and perhaps provide additional rigor in future versions of the specification.

6.2.9.2 Request

The parameters of a FeatureInfo request are listed in the table below.

URL Component	Description
http://server_address/path/script?	URL prefix of server (section 6.2.5.1.1)
WMTVER=1.0.0	Request version Required (section 6.2.5.1.2)
REQUEST=feature_info <map copy="" request=""></map>	Request name Copy of the Map request parameters that generated the map for which information is desired (Section 6.2.8.2)
QUERY_LAYERS=layer_list	Comma-separated list of one or more layers to be queried
INFO_FORMAT=output_format	Return format of feature information Optional; default=MIME
FEATURE_COUNT=number	How many features to return information about Optional; default=1
X=pixel_column	X coordinate in pixels of feature (measured from upper left corner=0)
Y=pixel_row	Y coordinate in pixels of feature (measured from upper left corner=0)
Vendor-specific parameters	(section 6.2.5.1.5)

Table 6.4 - The components of a FeatureInfo request.

6.2.9.2.1 <map request copy>

This is a placeholder. This is not a name/value pair as in all other parameters. Instead, the entire "map part" of a Map request is repeated here. The first three portions of a Map request (section 6.2.8.2) are omitted, namely the prefix, the WMTVER, and the REQUEST. The rest shall be embedded contiguously.

6.2.9.2.2 QUERY_LAYERS

This is a list of layers and follows the rules for the LAYERS parameter of the Map request set forth in section 6.2.8.2.1.

This parameter **shall** contain a list of one or more map layers from which feature information is desired to be retrieved.

This parameter shall contain at least one layer name.

If any layer in this sequence is not contained in the Capabilities XML of the Map Server, the results are undefined and the Map Server **shall** produce an exception response.

6.2.9.2.3 INFO_FORMAT

This parameter takes its values from the KnownFormats list of the Capabilities DTD. Only the following values from that list are allowed:

MIME

The Map Server **shall** choose the return format. The return format **shall** be any format for which there is a MIME value.

In an HTTP environment, the MIME type shall be set on the returned object.

GML.1, GML.2, or GML.3

The Map Server **shall** return information that conforms to the GML variant requested. GML is described in an OpenGIS document due to be published in early 2000. See <u>http://www.opengis.org</u> to find the document.

6.2.9.2.4 FEATURE_COUNT

Maximum number of features for which feature information should be returned. Inclusion of a value for this attribute is optional. When included, the value **shall** be greater than zero.

6.2.9.2.5 X/Y

The X and Y parameters **shall** identify a single point within the borders of the WIDTH and HEIGHT parameters of the embedded Map request. The origin is set to (0,0) centered in the pixel at the upper left corner and X increases to the right, Y increases downward. The query

6.2.9.3 Result

The Map Server **shall** return a result according to the description in section 6.2.9.2.3.

6.2.9.4 Exceptions

Upon encountering an exception, a Map Server **shall** either return nothing **or** return a MIME-typed error message. If the INFO_FORMAT was set to a GML variant, the Map Server **should** return an error as defined in GML.

7 REFERENCES

- [1] OpenGIS Project Document 97-009: "WWW Mapping Framework", Allan Doyle, BBN Corporation
- [2] OpenGIS Project Document 98-060: "User Interaction with Geospatial data", Adrian Cuthbert, Laser-Scan
- [3] OpenGIS Project Document 98-068: "A Web Mapping Scenario", Kenn Gardels, University of California, Berkeley
- [4] OpenGIS "A Request for Technology In Support of a Web Mapping Technology Testbed", October 28, 1998
- [5] OpenGIS "Request For Quotation and Call For Participation in the OGC Web Mapping Testbed Initial Operating Capability and Demonstration", March 8, 1999
- [6] OpenGIS Specification for Simple Features for OLE/COM
- [7] OpenGIS Catalog Interface Implementation Specification (Version 1.0)
- [8] Fielding et. al., "RFC 2616: Hypertext Transfer Protocol HTTP 1/1," June 1999, Internet Society Network Working Group, <u>ftp://ftp.isi.edu/in-notes/rfc2616.txt</u>
- [9] Bray, Paoli, Sperberg-McQueen, eds., "Extensible Markup Language (XML) 1.0", 10 February, 1998, World Wide Web Consortium
- [10] OpenGIS Project Document 99-086: "Standardize Spatial Reference System (SRS) Identifiers", Arliss Whiteside, BAE SYSTEMS Mission Solutions
- [11] Borenstein N., and N. Freed, "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1521, Bellcore, Innosoft, September 1993.
- [12] Berners-Lee, T., Fielding, N., and Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", IETF RFC 2396, http://www.ietf.org/rfc/rfc2396.txt
- [13] European Petroleum Survey Group, "EPSG Geodesy Parameters", http://www.petroconsultants.com/products/geodetic.html
- [14] Web Mapping Testbed, "WM Automatic Projections", <u>http://www.digitalearth.gov/wmt/auto.html</u>
- [15] Bradner, Scott, "RFC 2119 Key words for use in RFCs to Indicate Requirement Levels," March 1997, <u>ftp://ftp.isi.edu/in-notes/rfc2119.txt</u>
- [16] Whitehead, E. et. Al., "XML Media Types," July 1998, <u>ftp://ftp.isi.edu/in-notes/rfc2376.txt</u>

8 WEB MAPPING TESTBED PARTICIPANTS

Name	Organization	E-mail address
Rob Atkinson	Social Change Online	rob@socialchange.net
Sam Bacharach	Intergraph	sabachar@ingr.com
David Beddoe	ESRI	DBeddoe@esri.com
Steve Blake	AUSLIG	SteveBlake@auslig.gov.au
Lee Bock	ESRI	LBock@esri.com
Craig Bruce	CubeWerx	csbruce@cubewerx.com
Kurt Buehler	OGC	kurt@opengis.org
Adrian Cuthbert	Laser Scan Ltd.	adrian@lsl.co.uk
John Davidson	Georeferenced Systems	jvdson@erols.com
Vincent Dessard	Ionic Software s.a.	vincent.dessard@ionicsoft.com
Jeff de La Beaujardiere	CESDIS/NASA GSFC	delabeau@iniki.gsfc.nasa.gov
Allan Doyle	International Interfaces	adoyle@intl-interfaces.com
William Euerle	BBN Technologies	wjeuerle@bbn.com
John Evans	MIT	jdevans@mit.edu
Kristin Fishburn	Lockheed Martin M&DS	kristin.fishburn@lmco.com
Adam Gawne-Cain	Cadcorp Ltd.	adam@cadcorpdev.co.uk
Barend Gehrels	Geodan IT b.v.	barend@geodan.nl
Jeff Harrison	USACE-TEC	Jharris@tec.army.mil
Chuck Heazel	MITRE	cmheazel@ingr.com
Steve Hirsch	MITRE	hirsch@mitre.org
Peter Ladstaetter	SICAD Geomatics	peter.ladstaetter@sicad.de
Ron Lake	Northwood	ron_lake@focalpoint.org
Serge Margoulies	Ionic Software s.a.	<pre>serge.margoulies@ionicsoft.com</pre>
Brian May	CubeWerx	bmay@cubewerx.com
Dimitri Monie	Ionic Software s.a.	dimitri.monie@ionicsoft.com
Doug Nebert	FGDC	ddnebert@usgs.gov
Barry O'Rourke	Compusult	barry@compusult.nf.ca
George Percivall	SGT	george@sgt-inc.com
Davin Peterson	Environment Australia	davin@erin.gov.au
Paul Pilkington	Laser-Scan Inc.	paulpi@lsiva.com
Keith Pomakis	CubeWerx	pomakis@cubewerx.com

Greg Roy	Autodesk	greg.roy@autodesk.com
Shigeru Shimada	Hitachi CRL	shimada@crl.hitachi.co.jp
Hyam Singer	Object/FX	HyamS@ObjectFX.com
Dan Specht	USACE-TEC	Daniel.L.Specht@tec.army.mil
Jim Stephens	Lockheed Martin M&DS	james.t.stephens@lmco.com
Glenn Stowe	CubeWerx	gstowe@cubewerx.com
Didier Vidal	ILOG	vidal@ilog.fr
Frank Warmerdam	Chesapeake Analytics	warmerda@home.com
Frank Xia	ESRI	FXia@esri.com

Arliss Whiteside of BAE SYSTEMS Mission Solutions produced the UML model and tirelessly and willingly worked with the editor to revise it several times. Arliss is to be thanked for his invaluable contribution to the document and thus to the eventual Abstract Specification.

9 GLOSSARY

Bounding Box	A floating point or integer parameter that specifies the minimum X, minimum Y, maximum X, and maximum Y ranges of the geographic area to be portrayed by a Map Server. Expressed in units of the Spatial Reference System such that a rectangular area in the units of the Spatial Reference System is defined. Where Spatial Reference System units are not units of linear measure but are expressed in latitude and longitude, X is taken to mean Longitude and Y is taken to mean Latitude.
Capabilities	An XML document which conforms to the Capabilities Interface DTD and lists what interfaces a Map Server supports and what Map Layers it can serve in response to invocations of the Map interface. The listing also contains information about whether a Map Server supports the optional FeatureInfo interface.
Capabilities Interface	An interface designed to provide clients of MapServers with a machine- parseable listing of what interfaces a Map Server supports, what map layers it can serve, what formats it can serve them in among other parameters. The interface has one input parameter – the request version and its output is in the form of XML.
Cascading Map Server	A Map Server can report the Capabilities of other MapServers as its own and transform layers from those MapServers into different projections and formats, even if those MapServers cannot serve those projections and formats themselves.
Case	A model of client-server distribution of web-based portrayal services. Cases are distinguished by how much actual geographic data, graphic elements (e.g., WebCGM), or symbolized map graphics (e.g., GIF or jpeg images) travel across the web and the client-server distribution of services necessary to support that transfer. In this RFC three cases are defined: Feature Case, Graphic Element Case, and the Picture Case. These Cases can also be combined.
Catalog Service	A component that delivers dataset identifiers (among other possible information) across a common interface, also provides an interface for publishing datasets (and associating default symbology) to any client.
Data Case	The "old" term for Feature Case (q.v.)
Data Server	A component that delivers data across a common interface (typically as the result of a query) to any client.
Display	A portrayal service that makes the rendered map visible to the user by displaying it on a suitable display device.
Display Element Generation	A portrayal service which converts geographic data into a sequence of display elements, essentially a plotting command. These display elements are used to build up a representation of features or application-generated annotation and are passed on to the rendering service. These are sufficiently atomic that they can be individually rendered into a rendered map and the process does not require access to the geographic data source.

- Feature Case A model of client-server distribution of web-based portrayal services in which the client performs all portrayal services except Filter. Essentially what travels across the Internet in response to a client request is geographic feature data. In WMT Phase 1, experiments with XML as the transport format resulted in a possible RFC covering encoding of Open GIS Simple Features in XML.
- FeatureInfo (Interface) An interface designed to provide clients of a Map Server with more information about points in the pictures of maps which were returned by previous Map requests. The basic interface provides the ability for a client to specify which pixel is being asked about, which layer(s) should be asked, and what format the information should be returned in.
- Filter A portrayal service which selects geographic data to be displayed from a geographic data source.
- Format A parameter that specifies the format of a returned map (e.g., GIF, JPEG, PNG, WebCGM, SVG, GML.1, etc.).
- Graphic Element Case A model of client-server distribution of web-based portrayal in which the client performs Display and Render services but not Display Element Generation and Filter. Essentially what travels across the Internet in response to a client request is a packaged set of individual display elements, typically already in a projected reference system, and with already defined symbolization for geographic features.
- Layer/Style In Capabilities, a list of published XML elements that identify a Map Server's ability to return a single rendered map or series of rendered maps from a geographic data source. In a Map request, the layers and styles parameters are expressed in the form of a comma-separated list, where there is a left-to-right correspondence of the layers value and the styles values. A Map Server will render the requested layers by drawing the leftmost in the list first, the next second, and so on.
- Map (Interface)An interface designed to provide viewer clients with pictures of maps from
multiple MapServers.
- Map Server A component that delivers symbolized graphics across a common interface to a viewer client.
- MIME Type A naming convention, originally for multi-media email, which allows content types (e.g. JPEG, text, etc.) to be identified in such a way that a application can know what the internal format of the content is.
- Picture Case A model of client-server distribution of web-based portrayal services where the client only performs Display. Essentially what travels across the Internet to a clients request is an image in GIF, JPEG, PNG or other format that was constructed by a Map Server.
- Portrayal Services The processing stages which occur as data is moved from the data source to a display where it becomes visible to a user. These stages include Filter, Display Element Generation, Render, and Display.
- Render A portrayal service which constructs a rendered map from a sequence of display elements. In general, the rendering service may provide named shortcuts for a variety of plotting parameters, including fonts, colors, line-

styles, fill-styles, and symbols.

- Service Registry A component that delivers service identifiers (among other possible information) across a common interface (typically as the result of a metadata-based query) to any client. The Service Registry also provides interface for publishing service descriptions to a publisher client.
- Spatial Context Parameters that define the relationship between pixels in a displayed map and the original geographic area. Parameters include Spatial Reference System, Bounding Box, Width, and Height.
- Spatial Reference System A text parameter of Spatial Context that specifies a horizontal coordinate reference system code.
- True/False (Transparency) A boolean parameter that allows the background value of a layer to be considered transparent, allowing composite overlay of multiple layers.
- Viewer Client A component that renders (or possibly simply displays) graphics that come from Map Server components.
- Viewer Client Generator A component that provides server-side processing of viewer clientgenerated requests and returns responses as HTML pages.
- Web Mapping Testbed (WMT) An OGC Interoperability Initiative intended to advance interoperable web mapping technology by conducting collaborative development and testing of interoperable Web-based mapping standards and technology solutions.
- Width/Height A set of positive integer parameters of Spatial Context that specify a rectangle of pixels into which a resulting map should be portrayed. The map, regardless of format, should have exactly that width and height in pixels.
- WWW Mapping SIG That SIG (Special Interest Group) focuses on "achieving interoperability of map assembly and delivery middleware within a heterogeneous distributed computing environment".
- WMT Phase I The initial collaborative development period for the WMT covering Testbed kickoff in May 1999, Initial Operating Capability demonstrations in September 1999, and RFC Submission activities in November 1999.

A CAPABILITIES DTD (Normative)

This annex contains the Map Server Capabilities DTD corresponding to this version of the specification.

A.1 Capabilities DTD

<!-- NOTE: comments in this Document Type Definition impose additional constraints beyond those codified in the DTD syntax. A conformant Web Map Server must provide Capabilities XML that (1) validates against the DTD and (2) does not violate the constraints stated in comments herein. -->

<!-- The parent element of the Capabilities document includes as children a Service element with general information about the server and a Capability element with specific information about the kinds of functionality offered by the server. -->

<!ELEMENT WMT_MS_Capabilities (Service, Capability) >

<!-- The version attribute specifies the specification revision to which this DTD applies. Its format is one, two or three integers separated by periods: "x", or "x.y", or "x.y.z", with the most significant number appearing first. Future revisions are guaranteed to be numbered in monotonically increasing fashion, though gaps may appear in the sequence. All known versions may be found at http://www.digitalearth.gov/wmt/xml/ --> <!-- The updateSequence attribute is a sequence number for managing propagation of the contents of this document. For example, if a Map Server adds some data layers it can increment the update sequence to inform catalog servers that their previously cached versions are now stale. The format is a positive integer. -->

<!ATTLIST WMT_MS_Capabilities version CDATA #FIXED "1.0.0" updateSequence CDATA "0">

<!-- This WMT-wide list of possible output formats can be redefined by individual servers; see the sample XML. --> <!ENTITY % KnownFormats " GIF | JPEG | PNG | WebCGM | SVG | GML.1 | GML.2 | GML.3 | WBMP | WMS_XML | MIME | INIMAGE | TIFF | GeoTIFF | PPM | BLANK " >

<!-- The Service element provides metadata for the service as a whole. --> <!ELEMENT Service (Name, Title, Abstract?, Keywords?, OnlineResource, Fees?, AccessConstraints?) >

<!-- A service name defined within the Web Mapping Specification namespace. Currently only the name "GetMap" is defined. --> <!ELEMENT Name (#PCDATA) >

<!-- A human-readable title to briefly identify this server in menus. --> <!ELEMENT Title (#PCDATA) >

<!-- A descriptive narrative for more information about this server. --> <!ELEMENT Abstract (#PCDATA) >

<!-- Short words to help catalog searching. Currently, no controlled vocabulary has been defined. --> <!ELEMENT Keywords (#PCDATA) >

<!-- The top-level HTTP URL of this service. Typically the URL of a "home page" for the service. See also the onlineResource attributes of <DCPType> children, below. Currently, no non-HTTP platforms have been specified. --> <!ELEMENT OnlineResource (#PCDATA)>

<!-- Elements indicating what fees or access constraints are imposed.
The reserved keyword "none" indicates no constraint exists. -->
<!ELEMENT Fees (#PCDATA)>
<!ELEMENT AccessConstraints (#PCDATA)>

<!-- A Capability lists available request types, how exceptions may be reported, and whether any vendor-specific capabilities are defined. It also includes an optional list of map layers available from this server. -->

```
<!ELEMENT Capability
(Request, Exception?, VendorSpecificCapabilities?, Layer?) >
```

<!-- Available WMT-defined request types are listed here. At least one of the values is required, but more than one may be given. --> <!ELEMENT Request (Map | Capabilities | FeatureInfo)+ >

```
<!-- For each request method offered by the server, list the
available output formats and the supported distributed
computing platforms (DCPs). Example:
        <Map>
        <Format><PNG /><JPEG /><GML.1 /></Format>
        <DCPType><HTTP><Get onlineResource="URL" /></HTTP></DCPType>
        </Map>
```

```
-->
```

```
<!-- GetMap interface: Presence of the Map element means this server can
generate a map of a specified area, either as a picture or a feature
collection -->
<!ELEMENT Map (Format, DCPType+)>
```

<!-- GetCapabilities interface: Presence of the Capabilities element means this server can generate a description of its abilities and holdings formatted in XML that complies with this DTD. --> <!ELEMENT Capabilities (Format, DCPType+)>

<!-- GetFeatureInfo interface: Presence of the FeatureInfo element means this server can return information about a specific feature given a valid map

request and a location on that map. -->
<!ELEMENT FeatureInfo (Format, DCPType+)>

<!-- Available Distributed Computing Platforms (DCPs) are listed here. At present, only HTTP is defined. --> <!ELEMENT DCPType (HTTP) >

<!-- Available HTTP request methods. --> <!ELEMENT HTTP (Get | Post)+ >

<!-- HTTP request methods. The onlineResource attribute indicates the URL
prefix for HTTP GET requests (everything before the question mark and query
string: http://hostname[:port]/path/scriptname); for HTTP POST requests,
onlineResource is the complete URL. The HTTP GET syntax for Map, Capabilities
and FeatureInfo requests has been well defined and is described in the OGC
Web Map Server Interface Specification. The POST formalism, wherein GetMap
arguments are encoded in XML and POSTed to the server, has not yet been fully
developed. -->
<!ELEMENT Get EMPTY>

<!ATTLIST Get onlineResource CDATA #REQUIRED> <!ELEMENT Post EMPTY> <!ATTLIST Post onlineResource CDATA #REQUIRED>

<!-- Available formats. Not all formats are relevant to all requests. Individual servers MAY add new formats as shown in the sample XML accompanying this DTD. --> <!ELEMENT Format (%KnownFormats;) + > <!ELEMENT GIF EMPTY> <!-- Graphics Interchange Format --> <!ELEMENT JPEG EMPTY> <!-- Joint Photographics Expert Group --> <!ELEMENT PNG EMPTY> <!-- Portable Network Graphics --> <!ELEMENT PPM EMPTY> <!-- Portable PixMap --> <!ELEMENT TIFF EMPTY> <!-- Tagged Image File Format --> <!ELEMENT GeoTIFF EMPTY> <!-- Geographic TIFF --> <!ELEMENT WebCGM EMPTY> <!-- Web Computer Graphics Metafile --> <!ELEMENT SVG EMPTY> <!-- Scalable Vector Graphics --> <!ELEMENT WMS XML EMPTY> <!-- eXtensible Markup Language --> <!ELEMENT GML.1 EMPTY> <!-- Geography Markup Language, profile 1 --> <!ELEMENT GML.2 EMPTY> <!-- Geography Markup Language, profile 2 --> <!ELEMENT GML.3 EMPTY> <!-- Geography Markup Language, profile 3 --> <!ELEMENT WBMP EMPTY> <!-- Wireless Access Protocol (WAP) Bitmap --> <!ELEMENT MIME EMPTY> <!-- Multipurpose Internet Mail Extensions --> <!ELEMENT INIMAGE EMPTY> <!-- display text in the returned image --> <!ELEMENT BLANK EMPTY> <!-- return an image with all pixels transparent if supported by the image format, otherwise all pixels set to the BGCOLOR if present, otherwise all pixels set to the same (arbitrary) value -->

<!-- An Exception element indicates which output formats are supported for reporting problems encountered when executing a request. Available Exception formats MUST include one or more of WMS_XML, INIMAGE, or BLANK. Example: <Exception><Format><INIMAGE /><WMS_XML /></Format></Exception>. -->
<!ELEMENT Exception (Format)>

<!-- The optional VendorSpecificCapabilities element lists any capabilities unique to a particular Map Server. Because the information is not known a priori, it cannot be constrained by this particular DTD. A vendor-specific DTD fragment must be supplied at the start of the XML Capabilities document, after the reference to the general WMT_MS_Capabilities DTD. See the sample XML for further information. -->

<!--DEFINE THIS ELEMENT AS NEEDED IN YOUR XML <!ELEMENT VendorSpecificCapabilities (your stuff here) > -->

<!-- Nested list of zero or more map Layers offered by this server. The Layer element can be omitted if the server has no layers (for example, if it offers only geoprocessing services but no actual data). -->

<!-- A Layer element has two functions: it either refers to a map layer which can be requested by Name in the LAYERS parameter of a GetMap request, or it is a category Title for all the layers nested within. In the latter case, the category itself MAY include a Name by which all of the nested layers can be requested at once. For example, a parent layer "Roads" may have children "Interstates" and "County Roads" and allow the user to request either child individually or both together. -->

<!-- A Map Server which advertises a Layer containing a Name element MUST be able to accept that Name as the value of LAYERS argument in a GetMap request and return the corresponding map. A Viewer Client MUST NOT attempt to request a Layer that has a Title but no Name. -->

<!-- A Map Server MUST include at least one <Layer> element for each map layer offered. If desired, data layers MAY be repeated in different categories when relevant. A Layer element MAY state the Name by which a map of the layer is requested, MUST give a Title to be used in human-readable menus, and MAY include: a human-readable Abstract containing further description, available Spatial Reference Systems (SRS), bounding boxes in Lat/Lon and SRS-specific coordinates indicating the available geographic coverage, styles in which the layer is available, a URL for more information about the data, and a hint concerning appropriate map scales for displaying this layer. Use of the nesting hierarchy is optional. -->

<!-- The following table specifies the number and source of the various elements (and one attribute) describing a Layer that has a Name. Without a Name, the Layer is merely a category title and all other elements are optional; if present, some of those elements may be inherited by children as described in the table.

inherit

| | | from
parent? | comments | |
|--|-----|-----------------|---|--|
| ====== | 1 | - | =======
this table only applies to Named Layers
(those which can be requested in a GetMap call)
no default | |
| Title | 1 | - | required in each Layer; no default | |
| Abstract | 0/1 | - | optional; no default | |
| Keywords | 0/1 | - | optional; no default | |
| SRS | 1 | add | one list is required; default from parent;
list may include multiple whitespace-separated
values | |
| LatLonBoundingBo | x 1 | replace | exactly one is required; default from parent | |
| BoundingBox | 0+ | replace | one is required per SRS other than EPSG:4326;
default from parent | |
| DataURL | 0/1 | - | optional; no default | |
| Style | 0+ | add | optional; default from parent | |
| ScaleHint | 0/1 | replace | optional; default from parent | |
| Layer | 0+ | - | optional; no default;
if present, each is a "child" of the enclosing
Layer and inherits some default values as
described in this table | |
| queryable
(attribute) | 1 | replace | optional; default is "0" (not queryable)
or parent value if present | |
| > | | | | |
| ELEMENT Layer (Name?, Title, Abstract?, Keywords?, SRS?,<br LatLonBoundingBox?, BoundingBox*, DataURL?,
Style*, ScaleHint?, Layer*) > | | | | |
| A data layer may be queryable, as specified by this optional binary<br attribute. 1 = queryable, 0 = not queryable. A server that declares a Layer
to by queryable MUST implement the GetFeatureInfo interface> | | | | |
| ATTLIST Layer queryable (0 1) "0" | | | | |
| Listing of available Spatial Reference Systems (SRS).</td | | | | |

- * The root Layer element must include a list of all SRSes which are common to *all* subsidiary layers. This allows clients to quickly determine that a server cannot possibly satisfy a request for a particular SRS. Use an empty element if there is no common SRS.
- * Layer-specific SRS: Optionally, layers may add to the global SRS list, or to the list inherited from a parent layer as described above. Any duplication should be ignored by clients. That is, a particular layer may list all of its SRSes, even if it repeats information in the top-level SRS.

The content of the SRS element is a free-text list of SRS names separated by whitespace. A name includes a namespace prefix, a colon, and one or more parameter values. Currently defined namespaces are:

| prefix | parameters | comment |
|--------|------------|--|
| | | ====== |
| EPSG | EPSG code | European Petroleum Survey Group geodesy parameters
http://www.petroconsultants.com/products/geodetic.html
Examples: 'EPSG:4326' is WGS84 lat/lon,
'EPSG:26986' is NAD83 / Massachusetts Mainland. |

| AUTO | WMT code, | WMT list of "automatic" projections. WMT code: |
|------|-------------|--|
| | EPSG units, | an integer identifier assigned by WMT (see |
| | Longitude, | http://www.digitalearth.gov/wmt/auto.html). EPSG units: |
| | Latitude | one of the EPSG codes for identifying units, |
| | | indicating what units are to be used in Bounding Boxes. |
| | | Longitude: central meridian of the projection (degrees). |
| | | Latitude: central latitude of the projection (degrees). |
| | | Example: 'AUTO:42003,9001,-100,45' is auto orthographic |
| | | projection, bbox units in meters, center at 100W 45N. |
| | | The bounding box is measured in a plane perpendicular to |
| | | the line of sight, *not* directly on the Earth. |

-->

<!ELEMENT SRS (#PCDATA) >

<!-- The LatLonBoundingBox attributes indicate the edges of the enclosing
rectangle in latitude/longitude decimal degrees (as in SRS EPSG:4326 [WGS1984
lat/lon]). LatLonBoundingBox MUST be supplied regardless of what SRS the map
server may support, but it MAY be approximate if EPSG:4326 is not supported.
Its purpose is to facilitate geographic searches without requiring coordinate
transformations by the search engine. -->
<!ELEMENT LatLonBoundingBox EMPTY>
<!ATTLIST LatLonBoundingBox
 minx CDATA #REQUIRED
 miny CDATA #REQUIRED
 maxx CDATA #REQUIRED</pre>

```
maxy CDATA #REQUIRED>
```

<!-- The BoundingBox attributes indicate the edges of the bounding box in units of the specified spatial reference system. --> <!ELEMENT BoundingBox EMPTY> <!ATTLIST BoundingBox SRS CDATA #REQUIRED minx CDATA #REQUIRED miny CDATA #REQUIRED maxx CDATA #REQUIRED maxy CDATA #REQUIRED

<!-- A Map Server MAY use DataURL to offer more information about the data underneath a particular layer. While the semantics are not well-defined, as long as the results of an HTTP GET request against the DataURL are properly MIME-typed, Viewer Clients and Cascading Map Servers can make use of this. --> <!ELEMENT DataURL (#PCDATA) >

<!-- A Style element lists the name by which a style is requested and a human-readable title for pick lists, optionally (and ideally) provides a human-readable description, and optionally gives a style URL. If a Layer is offered in only a single Style, the Map Server MAY choose not to give it a name. Nevertheless, when handling a GetMap request a server MUST accept '' (null) and 'default' in the STYLES parameter as synonyms for the default (or only) style. -->

<!ELEMENT Style (Name, Title, Abstract?, StyleURL?) >

<!-- A Map Server MAY use StyleURL to offer more information about the data or symbology underlying a particular Style. While the semantics are not well-defined, as long as the results of an HTTP GET request against the StyleURL are properly MIME-typed, Viewer Clients and Cascading Map Servers can make use of this. A possible use could be to allow a Map Server to provide legend information. --> <!ELEMENT StyleURL (#PCDATA) >

<!-- Minimum and maximum scale hints for which it is appropriate to display this layer. It is STRONGLY RECOMMENDED that for Picture Case return formats the min and max values be expressed as ground distance in meters of a southwest to northeast diagonal of the pixel whose X coordinate is floor(width/2) and whose Y coordinate is floor(height/2).

In the figure below, the ${\rm X}$ is in the pixel whose diagonal measurement is used.

| +===+===+ | +===+===+===+ | | | | | |
|-----------|---------------|-------|------|-------|----|--|
| | | | 1 | | I | |
| +===+===+ | +=== | =+=== | +=== | =+=== | =+ | |
| X | | X | 1 | | I | |
| +===+===+ | +=== | =+=== | +=== | =+=== | =+ | |
| | | | I | | I | |
| +===+===+ | +=== | =+=== | +=== | =+=== | =+ | |
| | | | I | | I | |

+===+===+===+

It is understood that this definition is not geodetically precise, but at the same time the hope is that by including it, conventions will develop around its use which can be later specified more clearly. --> <!ELEMENT ScaleHint EMPTY> <!ATTLIST ScaleHint min CDATA #REQUIRED max CDATA #REQUIRED>