

[MS-DSPSTSS]: Data-Source Adapter SharePoint Team Services Web Service Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: <http://www.microsoft.com/interop/osp>) or the Community Promise (available here: <http://www.microsoft.com/interop/cp/default.aspx>). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Changes made for template compliance
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Editorial	Revised and edited the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Minor	Updated the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction	5
1.1 Glossary	5
1.2 References	5
1.2.1 Normative References	5
1.2.2 Informative References	6
1.3 Protocol Overview (Synopsis)	6
1.4 Relationship to Other Protocols	6
1.5 Prerequisites/Preconditions	7
1.6 Applicability Statement	7
1.7 Versioning and Capability Negotiation	7
1.8 Vendor-Extensible Fields	7
1.9 Standards Assignments	7
2 Messages	8
2.1 Transport	8
2.2 Common Message Syntax	8
2.2.1 Namespaces	8
2.2.2 Messages	8
2.2.3 Elements	8
2.2.4 Complex Types	9
2.2.4.1 SOAPFaultDetails	9
2.2.5 Simple Types	9
2.2.6 Attributes	9
2.2.7 Groups	9
2.2.8 Attribute Groups	9
2.3 Directory Service Schema Elements	9
3 Protocol Details	10
3.1 Server Details	10
3.1.1 Abstract Data Model	10
3.1.2 Timers	10
3.1.3 Initialization	10
3.1.4 Message Processing Events and Sequencing Rules	10
3.1.4.1 Query	11
3.1.4.1.1 Messages	11
3.1.4.1.1.1 queryRequestSoapIn	11
3.1.4.1.1.2 queryRequestSoapOut	11
3.1.4.1.2 Simple Types	11
3.1.4.1.2.1 OrderDirection	11
3.1.4.1.2.2 ResultContentType	12
3.1.4.1.2.3 ColumnMappingType	12
3.1.4.1.2.4 DocumentType	13
3.1.4.1.2.5 MethodType	13
3.1.4.1.3 Complex Types	13
3.1.4.1.3.1 DSQuery	13
3.1.4.1.3.1.1 System Metadata Response	16
3.1.4.1.3.1.2 Web Metadata Response	20
3.1.4.1.3.1.3 List Data Response	21
3.1.4.1.3.2 DspQuery	23
3.1.4.1.3.3 Fields	23

3.1.4.1.3.4	Field	24
3.1.4.1.3.5	AllFields	24
3.1.4.1.3.6	ArrayOfOrderField.....	24
3.1.4.1.3.7	OrderField	24
3.1.4.1.4	Elements.....	25
3.1.4.1.4.1	queryRequest.....	25
3.1.4.1.4.2	queryResponse.....	25
3.1.4.1.4.3	authentication	25
3.1.4.1.4.4	dataRoot	25
3.1.4.1.4.5	request	26
3.1.4.1.4.6	versions	26
3.1.5	Timer Events	26
3.1.6	Other Local Events	27
4	Protocol Examples.....	28
4.1	Obtaining list data and schema.....	28
4.2	Obtaining the list schema	28
4.3	Obtaining Filtered List Data.....	29
5	Security.....	31
5.1	Security Considerations for Implementers.....	31
5.2	Index of Security Parameters	31
6	Appendix A: Full WSDL.....	32
7	Appendix B: Product Behavior.....	36
8	Change Tracking.....	37
9	Index	38

1 Introduction

This document specifies the Data-Source Adapter SharePoint Team Services Service Protocol. This protocol enables a client to obtain structured tabular data from a server. This protocol also provides access to metadata about the server and how the tabular data is organized.

1.1 Glossary

The following terms are defined in [\[MS-GLOS\]](#):

language code identifier (LCID)

The following terms are defined in [\[MS-OFCGLOS\]](#):

list
site
site collection
SOAP action
SOAP body
SOAP fault
URL (Uniform Resource Locator)
Web service
WSDL (Web Services Description Language)
XML namespace
XML namespace prefix
XML Schema

The following terms are specific to this document:

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [\[RFC2119\]](#). All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, <http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624>, as an additional source.

[MS-WPPS] Microsoft Corporation, "[Web Part Pages Web Service Protocol Specification](#)", June 2008.

[MS-WSSCAML] Microsoft Corporation, "[Collaborative Application Markup Language \(CAML\) Structure Specification](#)", June 2008.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <http://www.ietf.org/rfc/rfc2119.txt>

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, <http://www.ietf.org/rfc/rfc2616.txt>

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, <http://www.ietf.org/rfc/rfc2818.txt>

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006, <http://www.ietf.org/rfc/rfc4648.txt>

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624>

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part2-20030624>

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

[XMLNS] World Wide Web Consortium, "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation 8 December 2009, <http://www.w3.org/TR/REC-xml-names/>

[XMLSCHEMA1] Thompson, H.S., Ed., Beech, D., Ed., Maloney, M., Ed., and Mendelsohn, N., Ed., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "[Windows Protocols Master Glossary](#)", March 2007.

[MS-OFCGLOS] Microsoft Corporation, "[Microsoft Office Master Glossary](#)", June 2008.

1.3 Protocol Overview (Synopsis)

This protocol provides access to **list** data via a **Web service**. The Web service accepts a query which describes the location of the data to be retrieved and any filtering or sorting used to format the requested data.

This protocol provides the following specific functionality:

- The ability to retrieve data about the server, such as its supported query type and version.
- The ability to retrieve data about the lists and Web sites accessible via the server.
- The ability to retrieve list data.

1.4 Relationship to Other Protocols

This protocol uses the SOAP messaging protocol for formatting requests and responses as specified either in [\[SOAP1.1\]](#) or in [\[SOAP1.2/1\]](#) and [\[SOAP1.2/2\]](#). It transmits these messages using the HTTP protocol as specified in [\[RFC2616\]](#) or the HTTPS protocol as specified in [\[RFC2818\]](#).

This protocol uses SOAP over HTTP(S) as shown in the following layering diagram:

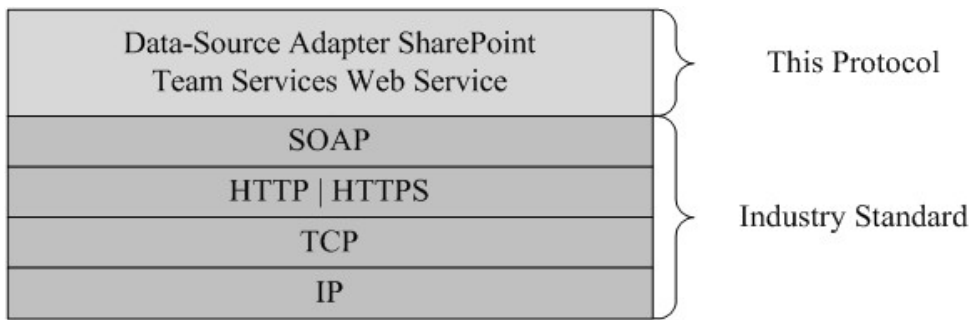


Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a **site** that is identified by a **URL** that is known by protocol clients. The protocol server endpoint is formed by appending `"/_vti_bin/DspSts.asmx"` to the URL of the site; for example, `http://www.contoso.com/Repository/_vti_bin/DspSts.asmx`.

This protocol assumes that authentication has been performed by the underlying protocols.

1.6 Applicability Statement

This protocol is intended for use by clients to access list data and Web metadata via a Web service. Another protocol, [\[MS-WPPS\]](#), also specifies methods for obtaining data from data sources and includes the preferred methods:

- **GetDataFromDataSource** is the preferred choice for obtaining list data and schema information for list structures. **GetDataFromDataSource** uses a different query semantic in comparison to this protocol.
- **GetXmlDataFromDataSource** is the preferred choice for obtaining Web metadata, and can accept the same DSQuery as input to access List data, in addition to other possible types of data sources.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- Supported Transports: This protocol uses multiple transports with SOAP as specified in section [2.1](#).

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

In the following sections, the schema definition might be less restrictive than the processing rules imposed by the protocol. The **WSDL** in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL specifies additional restrictions that reflect actual Microsoft product behavior. For example, the schema definition might allow for an element to be empty, null, or not present but the behavior of the protocol as specified restricts the same elements to being non-empty, present, and not null.

2.1 Transport

Protocol servers **MUST** support SOAP over HTTP. Protocol servers **SHOULD** additionally support SOAP over HTTPS for securing communication with clients.

Protocol messages **MUST** be formatted as specified either in [\[SOAP1.1\]](#), section 4 (SOAP Envelope), or in [\[SOAP1.2/1\]](#), section 5 (SOAP Message Construct). Protocol server faults **MUST** be returned either using HTTP Status Codes as specified in [\[RFC2616\]](#), section 10 (Status Code Definitions), or using **SOAP faults** as specified either in [\[SOAP1.1\]](#), section 4.4 (SOAP Fault), or in [\[SOAP1.2/1\]](#), section 5.4 (SOAP Fault).

2.2 Common Message Syntax

This section contains common structures used by this protocol. The syntax of the structures uses **XML Schema**, as specified in [\[XMLSCHEMA1\]](#) and [\[XMLSCHEMA2\]](#), and Web Services Description Language, as specified in [\[WSDL\]](#).

2.2.1 Namespaces

This protocol specifies and references **XML namespaces** using the mechanisms specified in [\[XMLNS\]](#). Although this document associates an **XML namespace prefix** for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
tns	http://schemas.microsoft.com/sharepoint/dsp	
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] , [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] , [SOAP1.2/2]
(none)	http://schemas.microsoft.com/sharepoint/dsp	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

2.2.2 Messages

None.

2.2.3 Elements

None.

2.2.4 Complex Types

One complex type definition is used by this specification: SOAPFaultDetails. XML schema complex type definitions that are specific to a particular operation are described with the operation.

2.2.4.1 SOAPFaultDetails

The details of a SOAP fault.

```
<s:schema xmlns:s="http://www.w3.org/2001/XMLSchema"
  targetNamespace=" http://schemas.microsoft.com/sharepoint/soap">
  <s:complexType name="SOAPFaultDetails">
    <s:sequence>
      <s:element name="faultstring" type="s:string"/>
      <s:element name="faultcode" type="s:string" minOccurs="0"/>
      <s:element name="detail" type="s:SOAPDetail" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
  <s:complexType name="SOAPDetail">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:any minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </s:complexType>
</s:schema>
```

errorstring: A human readable text explaining the application-level fault.

faultcode: The hexadecimal representation of a 4-byte result code.

Detail: The details of the SOAP fault in XML markup.

2.2.5 Simple Types

None.

2.2.6 Attributes

None.

2.2.7 Groups

None.

2.2.8 Attribute Groups

None

2.3 Directory Service Schema Elements

None

3 Protocol Details

In the following sections, the schema definition might be less restrictive than the processing rules imposed by the protocol. The WSDL in this specification matches the WSDL that shipped with the product and provides a base description of the schema. The text that introduces the WSDL specifies additional restrictions that reflect actual Microsoft product behavior. For example, the schema definition might allow for an element to be empty, null, or not present but the behavior of the protocol as specified restricts the same elements to being non-empty, present, and not null.

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

Except where specified, protocol clients SHOULD interpret HTTP Status Codes returned by the protocol server as specified in **Status Code Definitions** in [\[RFC2616\]](#).

This protocol allows protocol servers to notify protocol clients of application-level faults using SOAP faults. This protocol allows protocol servers to provide additional details for SOAP faults by including either a **detail** element as specified in [\[SOAP1.1\]](#), section 4.4 (SOAP Fault), or a **Detail** element as specified in [\[SOAP1.2/1\]](#), section 5.4.5 (SOAP Detail Element), which conforms to the XML schema of the **SOAPFaultDetails** complex type specified in [SOAPFaultDetails](#). Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

This protocol allows protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults either using HTTP Status Codes or using SOAP faults as specified previously in this section.

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

This protocol has a single operation: **Query**. The **Query** method provides access to list data as well as server and Web metadata.

3.1.4.1 Query

The **Query** method accepts a request which consists of two parts: an expression to specify the source of the data and a description of how to manipulate the data before it is returned.

```
<wsdl:operation name="Query">
  <wsdl:input name="queryRequest" message="queryRequestSoapIn"/>
  <wsdl:output name="queryRequest" message="queryRequestSoapOut"/>
</wsdl:operation>
```

When the client sends a **queryRequestSoapIn** request message, the server **MUST** respond with a **queryRequestSoapOut** response message that consists of either XML data payload or a schema, or both.

3.1.4.1.1 Messages

The following message definitions are specific to this operation.

3.1.4.1.1.1 queryRequestSoapIn

The **SOAP action** value of the message is defined as:

http://schemas.microsoft.com/sharepoint/dsp/query

The **SOAP body** contains a **queryRequest** element.

3.1.4.1.1.2 queryRequestSoapOut

The SOAP action value of the message is defined as:

http://schemas.microsoft.com/sharepoint/dsp/query

The SOAP body contains a **queryResponse** element.

3.1.4.1.2 Simple Types

3.1.4.1.2.1 OrderDirection

Specifies whether the sort order of a given list field is ascending (specified with the value of "ASC") or descending (specified with the value of "DESC").

```
<s:simpleType name="OrderDirection">
  <s:restriction base="s:string">
    <s:enumeration value="ASC"/>
    <s:enumeration value="DESC"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **OrderDirection**:

Value	Meaning
ASC	Sort order is ascending

Value	Meaning
DESC	Sort order is descending

3.1.4.1.2.2 ResultContentType

ResultContentType specifies what content to include in the response.

```
<s:simpleType name="ResultContentType">
  <s:restriction base="s:string">
    <s:enumeration value="both"/>
    <s:enumeration value="schemaOnly"/>
    <s:enumeration value="dataOnly"/>
  </s:restriction>
</s:simpleType>
```

The following table defines the allowable values for **ResultContentType**:

Value	Meaning
both	Return both schema and list data.
schemaOnly	Return only schema info.
dataOnly	Return only list data.

3.1.4.1.2.3 ColumnMappingType

ColumnMappingType is used to determine the format of the data returned to the client. If **ColumnMappingType** is set to the value of "element", the data MUST be formatted so that each column of data is returned as a child element to the row element. For example:

```
<Widgets_Row>
  <Title>Widget C</Title>
  <Count>23</Count>
  <Stock>1</Stock>
  <ID>3</ID>
</Widgets_Row>
```

When the **ColumnMappingType** is set to the value of "attribute", the data MUST be formatted so that each column of data is returned as an attribute of the row element. For example:

```
<Widgets_Row Title="Widget A" Count="50" Stock="0" ID="1" />
```

When **ColumnMappingType** is set to "attribute" the result data MUST NOT contain any data annotations that are used to comment return data. The "attribute" for **ColumnMappingType** setting is designed to increase performance when a client is requesting only row data from a data source.

```
<s:simpleType name="ColumnMappingType">
  <s:restriction base="s:string">
    <s:enumeration value="element"/>
  </s:restriction>
</s:simpleType>
```

```

    <s:enumeration value="attribute"/>
  </s:restriction>
</s:simpleType>

```

The following table defines the allowable values for **ColumnMappingType**:

Value	Meaning
element	The server MUST send result XML using elements to store column data.
attribute	The server MUST send result XML using attributes on to store column data.

3.1.4.1.2.4 DocumentType

The **DocumentType** is used to specify the source of the data to be returned to the client. **DocumentType** determines how the **select** attribute of the **DSQuery** element is interpreted.

```

<s:simpleType name="DocumentType">
  <s:restriction base="s:string">
    <s:enumeration value="content"/>
    <s:enumeration value="system"/>
  </s:restriction>
</s:simpleType>

```

The following table defines the allowable values for **DocumentType**:

Value	Meaning
content	Return information about a list or Web site.
system	Return information about the DSPSTSS Web service.

3.1.4.1.2.5 MethodType

MethodType is the type of operation that the server can perform. **MethodType** MUST be set to the value of "query".

```

<s:simpleType name="MethodType">
  <s:restriction base="s:string">
    <s:enumeration value="query"/>
  </s:restriction>
</s:simpleType>

```

3.1.4.1.3 Complex Types

3.1.4.1.3.1 DSQuery

DSQuery is the core structure of a request for data using this protocol. **DSQuery** contains all the necessary information to fully describe the source of the data, what to include in the response, any sorting or filtering information about the data, and the formatting for the response.

```

<s:complexType name="DSQuery">

```

```
<s:sequence>
  <s:element name="Query" type="tns:DspQuery" minOccurs="0"/>
</s:sequence>
<s:attribute name="select" type="s:string"/>
<s:attribute name="resultContent" type="tns:ResultContentType"
  default="both"/>
<s:attribute name="columnMapping" type="tns:ColumnMappingType"
  default="element"/>
<s:attribute name="resultNamespace" type="s:string"/>
<s:attribute name="resultPrefix" type="s:string"/>
<s:attribute name="resultRoot" type="s:string"/>
<s:attribute name="resultRow" type="s:string"/>
<s:attribute name="startPosition" type="s:string"/>
<s:attribute name="comparisonLocale" type="s:string"/>
</s:complexType>
```

The following figure shows how the various settings of **DocumentType**, **ColumnMappingType**, and the **select** attribute of **DSQuery** determine the type of response that is generated by this protocol.

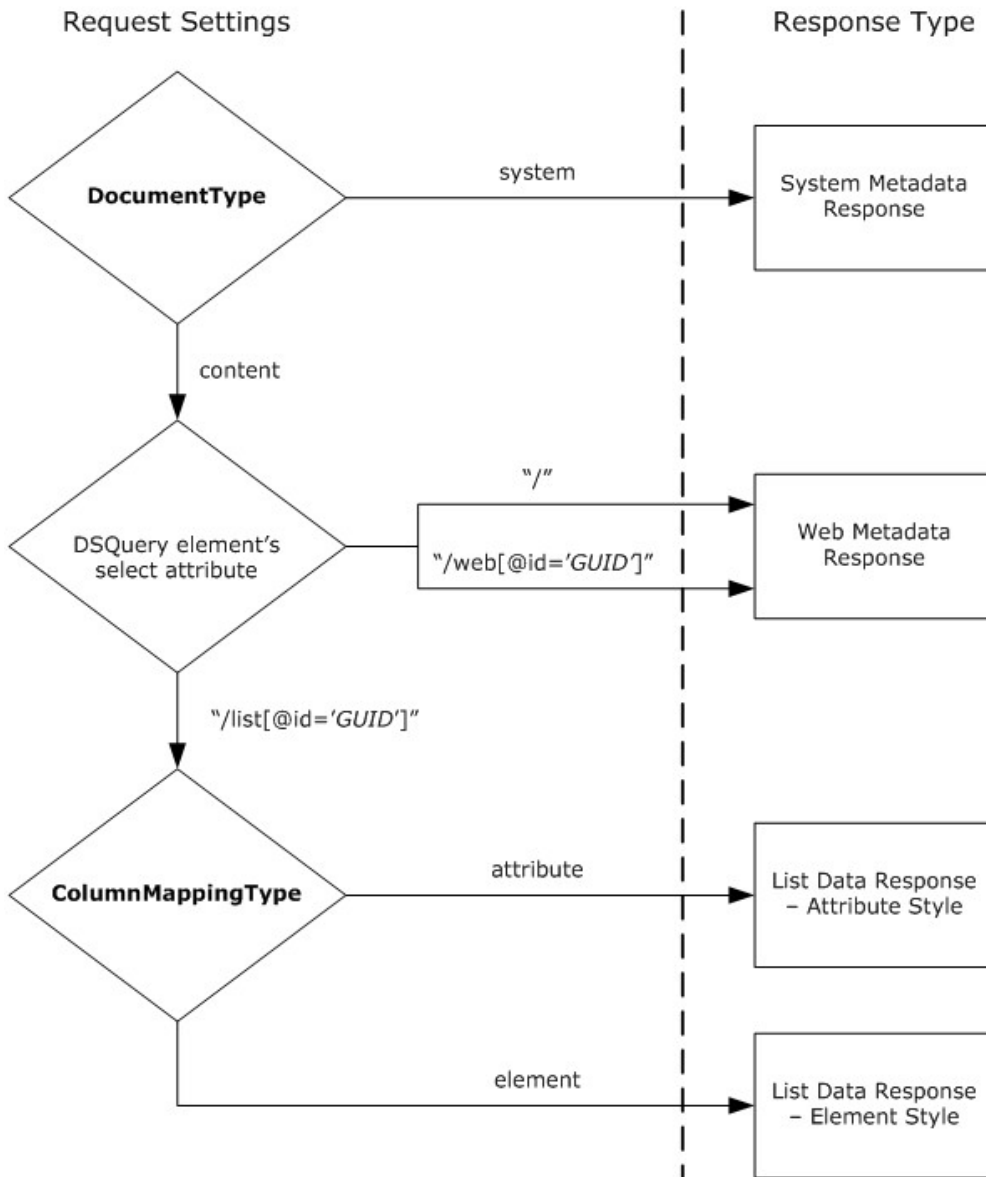


Figure 2: How various request settings determine the response type

The following table shows how the various combinations of the **DocumentType** attribute of the request element and **select** attributes of the **DSQuery** determine what type of response is generated. The specific behavior for the elements and attributes of the **DSQuery** change based on the type of response, as detailed in [3.1.4.1.3.1.1](#), [3.1.4.1.3.1.2](#), and [3.1.4.1.3.1.3](#).

Document type	select attribute	Response
System	"/", "/dataRoot", "/authentication", "/versions", "/querySupport"	System Metadata Response. See 3.1.4.1.3.1.1

Document type	select attribute	Response
Content	"/","/web[@id='GUID']"	Web Metadata Response. See 3.1.4.1.3.1.2.
Content	"/list[@id='GUID']"	List Data Response. See 3.1.4.1.3.1.3.

select: Specifies an expression that selects the data upon which the query is applied. The possible valid expressions depend on the setting specified in **DocumentType**.

StartPosition: Used to provide paging support. If set, the value MUST be a Base64 encoded **string** as defined in [\[RFC4648\]](#), Section 4, where the **string** is of the format "t_ID=ROWID" and ROWID is the identifier value for the first row in the requested set of rows. When the first page is requested in query, the value of **StartPosition** should be empty. The request result contains the value to do next page request, in **pagingInfo** element inside the **resultRoot** element. The value can be used to issue queries for subsequent paging.

resultNamespace: Used to set the XML namespace for the XML data payload.

resultPrefix: Used to set the prefix for the XML namespace for the XML data payload.

resultRoot: Used to specify the name of the root element for the XML data payload.

resultRow: Used to specify the name of the row elements for the XML data payload.

comparisonLocale: Used to specify the locale used for **string** comparisons. If set, the value MUST be an **LCID**.

pagingInfo: : Used to specify the value of **StartPosition** for query to the next page.

resultContent: Element as specified in [3.1.4.1.2.2.](#)

columnMapping: Element as specified in [3.1.4.1.2.3.](#)

Query: Element as specified in [3.1.4.1.3.2.](#)

3.1.4.1.3.1.1 System Metadata Response

If **DocumentType** is set to "system", the attributes and elements of the **DSQuery** element MUST follow the behavior specified in the following list. If not explicitly noted, the behavior is as specified in section [3.1.4.1.3.1.](#)

StartPosition: This attribute MUST be ignored.

resultNamespace: If this attribute specifies an XML namespace, the XML data payload in the response MUST use the specified namespace. If not set, an empty **string** MUST be used as the namespace. If set to an invalid XML namespace **string**, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, the XML namespace for the XML data payload MUST be set as the **resultNamespace** with the XML namespace prefix specified as "resultPrefix". If **resultNamespace** is not set and **resultPrefix** is set, the response MUST be an exception. If **resultPrefix** is not set, the response MUST use a blank result XML namespace prefix for the XML namespace.

resultRoot: This attribute MUST be ignored.

resultRow: This attribute MUST be ignored.

comparisonLocale: This attribute MUST be ignored.

columnMapping: This attribute MUST be ignored.

Query: This element MUST NOT exist. If set, the response MUST be an exception.

The response MUST be one of the values in the following table, such that when the **select** attribute of the **DSQuery** is set to an expression from the Expression column, the response contains either the schema or the data, or both the schema and the data from the Result column.

Expression	Result
/	Schema <pre><x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="dspSts"> <x:complexType> <x:all> <x:element name="versions"> <x:complexType> <x:sequence> <x:element name="version" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> <x:element name="querySupport"> <x:complexType> <x:sequence> <x:element name="queryType" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> <x:element name="dataRoot"> <x:complexType> <x:sequence> <x:element name="rootFormat" minOccurs="0" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> <x:element name="authentication"> <x:complexType> <x:sequence> <x:element name="authMethod" minOccurs="0" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> </x:all> </x:complexType> </x:element> </pre></pre>

Expression	Result	
		</x:schema>
/	Data	<pre><d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" <d:versions> <d:version>1.0</d:version> </d:versions> <d:querySupport> <d:queryType>DSPQ</d:queryType> </d:querySupport> <d:dataRoot> <d:rootFormat>URL</d:rootFormat> </d:dataRoot> <d:authentication /> </d:dspSts></pre>
/dataRoot	Schema	<pre><x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="dspSts"> <x:complexType> <x:all> <x:element name="dataRoot"> <x:complexType> <x:sequence> <x:element name="rootFormat" minOccurs="0" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> </x:all> </x:complexType> </x:element> </x:schema></pre>
/dataRoot	Data	<pre><d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" <d:dataRoot> <d:rootFormat>URL</d:rootFormat> </d:dataRoot> </d:dspSts></pre>
/querySupport	Schema	<pre><x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="dspSts"> <x:complexType> <x:all></pre>

Expression	Result	
		<pre> <x:element name="querySupport"> <x:complexType> <x:sequence> <x:element name="queryType" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> </x:all> </x:complexType> </x:element> </x:schema> </pre>
/querySupport	Data	<pre> <d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:querySupport> <d:queryType>DSPQ</d:queryType> </d:querySupport> </d:dspSts> </pre>
/versions	Schema	<pre> <x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="dspSts"> <x:complexType> <x:all> <x:element name="versions"> <x:complexType> <x:sequence> <x:element name="version" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> </x:all> </x:complexType> </x:element> </x:schema> </pre>
/versions	Data	<pre> <d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:versions> <d:version>1.0</d:version> </d:versions> </d:dspSts> </pre>
/authentication	Schema	<pre> <x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema"> </pre>

Expression	Result
	<pre> <x:element name="dspSts"> <x:complexType> <x:all> <x:element name="authentication"> <x:complexType> <x:sequence> <x:element name="authMethod" minOccurs="0" maxOccurs="unbounded" type="x:string" /> </x:sequence> </x:complexType> </x:element> </x:all> </x:complexType> </x:element> </x:schema> </pre>
/authentication	<pre> <d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:authentication /> </d:dspSts> </pre>

3.1.4.1.3.1.2 Web Metadata Response

If the **DocumentType** is set to "content", the attributes and elements of the **DSQuery** element MUST follow the behavior specified in the following list. If not explicitly noted, the behavior is as specified in section [3.1.4.1.3.1](#).

StartPosition: This attribute MUST be ignored.

resultNamespace: If this attribute specifies an XML namespace, the XML data payload in the response MUST use the specified namespace. If not set, an empty **string** MUST be used as the namespace. If set to an invalid XML namespace **string**, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, the XML namespace for the XML data payload MUST be set as the **resultNamespace** with the XML namespace prefix specified as "resultPrefix". If **resultNamespace** is not set and **resultPrefix** is set, the response MUST be an exception. If **resultPrefix** is not set, the response MUST use a blank result XML namespace prefix for the XML namespace.

resultRoot: This attribute MUST be ignored.

resultRow: This attribute MUST be ignored.

comparisionLocale: This attribute MUST be ignored.

columnMapping: This attribute MUST be ignored.

select: If the value set for **select** is a backslash (/), the response MUST contain metadata conforming to the following schema for the top-level Web site in the **site collection**. If set to "/web[@id='Path']", where *Path* is the relative path of a site, the response MUST contain Web metadata for that site conforming to the following schema:

```

<x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"
targetNamespace="http://schemas.microsoft.com/sharepoint/dsp" elementFormDefault="qualified"
attributeFormDefault="unqualified" xmlns:x="http://www.w3.org/2001/XMLSchema">
  <x:complexType name="ObjectPropertiesType">
    <x:attribute name="id" type="x:string" use="required" />
    <x:attribute name="displayName" type="x:string" />
    <x:attribute name="contentType" type="x:string" use="required" />
    <x:attribute name="serverParameters" type="x:string" use="required" />
    <x:attribute name="supportFiltering" type="x:boolean" />
    <x:attribute name="supportOrdering" type="x:boolean" />
    <x:attribute name="supportPaging" type="x:string" />
    <x:attribute name="comparisonLocale" type="x:int" />
    <x:attribute name="unsafe" type="x:boolean" />
    <x:attribute name="querySupport" type="x:string" />
  </x:complexType>
  <x:element name="web">
    <x:complexType>
      <x:sequence>
        <x:element name="web" type="d:ObjectPropertiesType" minOccurs="0"
maxOccurs="unbounded" />
        <x:element name="list" type="d:ObjectPropertiesType" minOccurs="0"
maxOccurs="unbounded" />
      </x:sequence>
      <x:attribute name="id" type="x:string" use="required" />
    </x:complexType>
  </x:element>
</x:schema>

```

3.1.4.1.3.1.3 List Data Response

StartPosition: A string that specifies the beginning of the next page if paging is supported in the data payload. The string value can be used to retrieve next page data in the subsequent request for data using this protocol.

resultNamespace: If this attribute specifies an XML namespace, the XML data payload in the response MUST use the specified namespace. If not set, an empty **string** MUST be used as the namespace. If set to an invalid XML namespace **string**, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, then the XML namespace for the XML data payload MUST be set as the **resultNamespace** with the XML namespace prefix specified as "resultPrefix". If **resultNamespace** is not set and **resultPrefix** is set, then the response MUST be an exception. If **resultPrefix** is not set then the response MUST use a blank result XML namespace prefix for the XML namespace.

resultRoot: If set to a non-empty string, then the response MUST use the **resultRoot** value as the name of the root XSD element for the XML data payload. If not set, then the response MUST use the name of the list being queried as the name of the root element for the XML data payload.

resultRow: If set to a non-empty **string**, then the response MUST use the **resultRow** value as the name of the element for each row of data in the XML data payload. If not set or if set to an empty **string**, then the response MUST use the name of the list being queried with "_Row" appended as the name of the element for each row of data in the XML data payload.

comparisonLocale: If the locale is not present or not supported, the default locale of the server MUST be used. If set to a supported LCID value, then any string comparisons MUST use the comparisonLocale value.

Query: Element as specified in [3.1.4.1.3.2](#).

resultContent: Attribute as specified in [3.1.4.1.2.2](#).

columnMapping: Attribute as specified in [3.1.4.1.2.3](#).

Based on the **columnMapping** setting, the response SHOULD [<1>](#) contain data which conforms to the following table.

The value of the **columnMapping** attribute in the **columnMapping** column results in data which conforms to the schema in the "Schema" column of the following table for the corresponding row. The *resultRoot* and *resultRow* values shown in the table are placeholders for the actual **resultRow** and **resultRoot** values as described in the preceding list. The sequence attribute which is a child of the **resultRow** element MUST contain one XSD element or XSD attribute entry for each column of data which is returned in the response.

columnMapping attribute	Schema
element	<pre> <x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="resultRoot"> <x:complexType> <x:sequence maxOccurs="unbounded"> <x:element name="resultRow" minOccurs="0"> <x:complexType> <x:sequence> <x:element /> </x:sequence> </x:complexType> </x:element> </x:sequence> </x:complexType> </x:element> </x:schema> </pre>
attribute	<pre> <x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp" xmlns:x="http://www.w3.org/2001/XMLSchema"> <x:element name="resultRoot"> <x:complexType> <x:sequence maxOccurs="unbounded"> <x:element name="resultRow" minOccurs="0"> <x:complexType> <x:sequence> <x:attribute /> </x:sequence> </x:complexType> </x:element> </x:sequence> </x:complexType> </x:element> </x:schema> </pre>

3.1.4.1.3.2 DspQuery

The **DspQuery** element describes how the data is manipulated before it is formatted for return to the client. The **DspQuery** element describes the columns of data to be included in the results, how to sort or filter the data, and the maximum number of data rows to be returned.

```
<s:complexType name="DspQuery">
  <s:sequence>
    <s:element name="Fields" type="tns:Fields" minOccurs="0"/>
    <s:element name="Where" minOccurs="0">
      <s:complexType mixed="true">
        <s:sequence>
          <s:any/>
        </s:sequence>
      </s:complexType>
    </s:element>
    <s:element name="OrderBy" type="tns:ArrayOfOrderField" minOccurs="0"/>
  </s:sequence>
  <s:attribute name="RowLimit" type="s:long" default="-1"/>
</s:complexType>
```

Fields: Contains a list of Field elements as specified in [3.1.4.1.3.3](#). If not specified, the result MUST be handled as if only **AllFields** had been set.

Where: Contains filter information as specified in [\[MS-WSSCAML\]](#) section 2.2. If not set then the response MUST contain all the rows of data from the data source, limited only by the **RowLimit** value.

OrderBy: Element of type **ArrayOfOrderField** as specified in [3.1.4.1.3.6](#). If not set then the response MUST contain the rows of data in the order that they were retrieved from the data source.

RowLimit: Sets the paging limit of the request. If **RowLimit** is not specified, all rows of the list MUST be returned. If set, the response MUST contain a number of rows less than or equal to the row limit.

3.1.4.1.3.3 Fields

The **Fields** element MUST contain either a list of **Field** elements as specified in [3.1.4.1.3.4](#) or the **AllFields** element as specified in [3.1.4.1.3.5](#). The **Fields** element MUST NOT be empty. If the **Fields** element is empty, the response MUST be an exception.

```
<s:complexType name="Fields">
  <s:choice>
    <s:element name="AllFields" type="tns:AllFields" />
    <s:sequence>
      <s:element name="Field" type="tns:Field" maxOccurs="unbounded" />
    </s:sequence>
  </s:choice>
</s:complexType>
```

Field: Element as defined in [3.1.4.1.3.4](#).

AllFields: Element as defined in [3.1.4.1.3.5](#).

3.1.4.1.3.4 Field

Specifies the name of the list column to be returned in the SOAP response.

```
<s:complexType name="Field">
  <s:attribute name="Name" type="s:string"/>
  <s:attribute name="Alias" type="s:string"/>
</s:complexType>
```

Name: The name of the list column.

Alias: The alternate name of the list column. If not set, the **displayName** for the column in the response will be set to the **name** of the column. If set, the **displayName** for the column in the response MUST be the **alias** specified. **displayName** is the name of the element in the response that represent the field.

3.1.4.1.3.5 AllFields

Returns all the fields in the list, except any hidden or computed fields

```
<s:complexType name="AllFields"> <s:attribute name="IncludeHiddenFields" type="s:boolean"
default="false"/></s:complexType>
```

IncludeHiddenFields: Determines whether hidden fields and computed fields are included in the result set. If false, the result set MUST NOT include hidden fields or computed fields. If true, the result set MUST include all hidden and computed fields.

3.1.4.1.3.6 ArrayOfOrderField

Contains a list of **OrderField** elements as described in [3.1.4.1.3.7](#). The result set MUST be sorted based on the **OrderField** elements specified, with the sorts applied iteratively.

```
<s:complexType name="ArrayOfOrderField">
  <s:sequence>
    <s:element name="OrderField" type="tns:OrderField" minOccurs="0"
      maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

OrderField: Element as defined in [3.1.4.1.3.7](#).

3.1.4.1.3.7 OrderField

Specifies the field to be sorted in the SOAP request. **OrderField** has two attributes: the **name** attribute specifies the internal name of the order by field, and the **direction** attribute specifies if the sort order is ascending (with value "ASC") or descending (with value "DESC"). The default value of **direction** is "ASC".

```
<s:complexType name="OrderField">
  <s:attribute name="Name" type="s:string"/>
  <s:attribute name="Direction" type="tns:OrderDirection" default="ASC"/>
</s:complexType>
```


3.1.4.1.4 Elements

3.1.4.1.4.1 queryRequest

Element named "queryRequest", which contains a **DSQuery** element.

```
<s:element name="queryRequest" nillable="true">
  <s:complexType>
    <s:sequence>
      <s:element name="dsQuery" type="tns:DSQuery" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

dsQuery: Element as specified in [3.1.4.1.3.1](#).

3.1.4.1.4.2 queryResponse

This element contains an XML document that contains the result data in response to a query.

```
<s:element name="queryResponse">
  <s:complexType mixed="true">
    <s:sequence>
      <s:any/>
    </s:sequence>
  </s:complexType>
</s:element>
```

3.1.4.1.4.3 authentication

Specifies what authentication headers are expected in the request message. This element **MUST NOT** exist in the query. If the element exists in the query, the response **MUST** be an exception.

```
<s:element name="authentication">
  <s:complexType>
    <s:sequence>
      <s:any minOccurs="0" maxOccurs="unbounded"/>
    </s:sequence>
    <s:anyAttribute/>
  </s:complexType>
</s:element>
```

3.1.4.1.4.4 dataRoot

dataRoot specifies the site that the protocol uses for processing queries.

```
<s:element name="dataRoot">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" name="root" type="s:string"/>
    </s:sequence>
    <s:attribute default="true" name="allowRemoteDataAccess" type="s:boolean"/>
    <s:anyAttribute/>
  </s:complexType>
</s:element>
```

```
</s:complexType>
</s:element>
```

AllowRemoteDataAccess: MUST be ignored by the server, as well as any sub-elements of the element.

root: If the value is null or empty, the site URL MUST be the URL by which the service is invoked. For example, if the service is invoked through the site path "http://server/site/<service>", the URL to the site must be "http://server/site". If set, the value must be the complete path to a site to which the query is posted. For example, if the site is "http://server/site", and the client needs to post a query to this site, the value of the **dataRoot** element must be "http://server/site".

3.1.4.1.4.5 request

Request describes the type of data being requested and the type of method being called. It MUST be present; otherwise, a SOAP fault as specified in section [3.1.4.1.3.1.3](#) is returned by the server.

```
<s:element name="request">
  <s:complexType>
    <s:attribute name="document" type="tns:DocumentType" use="required"/>
    <s:attribute name="method" type="tns:MethodType" use="required"/>
    <s:anyAttribute/>
  </s:complexType>
</s:element>
```

Document: Attribute as described in section [3.1.4.1.2.4](#).

Method: Attribute as described in section [3.1.4.1.2.5](#).

3.1.4.1.4.6 versions

The **versions** element describes the version of the protocol that is being used for the query. It MUST be present; otherwise, a SOAP fault as specified in section [3.1.4.1.3.1.3](#) is returned by the server.

```
<s:element name="versions">
  <s:complexType>
    <s:sequence>
      <s:element name="version" type="s:string" minOccurs="0"
        maxOccurs="unbounded"/>
    </s:sequence>
    <s:anyAttribute/>
  </s:complexType>
</s:element>
```

version: Is a child of the **versions** element. Its value MUST be set to 1.0 if it is present.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

The examples that follow use a list called "Widgets" that has a GUID based identifier of {C13E4B16-9982-4C30-B533-2B4068B0C623}. The list contains the following fields:

Field name	Field type
ID	Counter
Title	String
Count	Number
Stock	Boolean

The data for "Widgets" list is as follows:

ID	Title	Count	Stock
1	Widget A	50	No
2	Widget B	100	No
3	Widget C	23	Yes

4.1 Obtaining list data and schema

The minimal query simply sets the select attribute on the **dsQuery** node.

Request:

```
<queryRequest xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQuery select="/list[@id='{C13E4B16-9982-4C30-B533-2B4068B0C623}']" />
</queryRequest>
```

Response:

4.2 Obtaining the list schema

Setting the **resultContent** attribute to the value of "schemaOnly" will result in only schema data being returned.

Request:

```
<queryRequest xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQuery select="/list[@id='{C13E4B16-9982-4C30-B533-2B4068B0C623}']"
    resultContent="schemaOnly" />
</queryRequest>
```

Response:

```
<queryResponse xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQueryResponse status="success">
```

```

<x:schema xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"
  xmlns:x="http://www.w3.org/2001/XMLSchema">
  <x:element name="Widgets">
    <x:complexType>
      <x:sequence maxOccurs="unbounded">
        <x:element name="Widgets_Row" minOccurs="0">
          <x:complexType>
            <x:sequence>
              <x:element name="ID" minOccurs="0" d:readOnly="true"
                d:filterSupport="IsNull;IsNotNull;Eq;Neq;Lt;
                Gt;Leq;Geq;" d:displayName="ID" type="x:int" />
              <x:element name="Title"
                d:filterSupport="IsNull;IsNotNull;Eq;Neq;
                Lt;Gt;Leq;Geq;Contains;BeginsWith;"
                d:displayName="Title" type="x:string" />
              <x:element name="Count" minOccurs="0"
                d:filterSupport="IsNull;IsNotNull;Eq;Neq;Lt;Gt;
                Leq;Geq;" d:displayName="Count" type="x:float" />
              <x:element name="Stock" minOccurs="0"
                d:filterSupport="IsNull;IsNotNull;Eq;Neq;"
                d:displayName="Stock" type="x:boolean" />
            </x:sequence>
          </x:complexType>
        </x:element>
      </x:sequence>
    </x:complexType>
  </x:element>
</x:schema>
</dsQueryResponse>
</queryResponse>

```

4.3 Obtaining Filtered List Data

The following is an example of a query for obtaining list data for all list items with an identifier greater than 1. To obtain the list data, but not list schema, the **resultContent** attribute is set to the value of "dataOnly". Filtering is achieved by setting the **Where** clause of the **Query**.

Request:

```

<queryRequest xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQuery select="/list[@id='{C13E4B16-9982-4C30-B533-2B4068B0C623}']"
    resultContent="dataOnly">
    <Query>
      <Where>
        <Gt>
          <FieldRef Name="ID" />
          <Value>1</Value>
        </Gt>
      </Where>
    </Query>
  </dsQuery>
</queryRequest>

```

Response:

```

<queryResponse xmlns="http://schemas.microsoft.com/sharepoint/dsp">

```

```
<dsQueryResponse status="success">
  <Widgets xmlns="">
    <Widgets_Row>
      <ID>2</ID>
      <Title>Widget B</Title>
      <Count>100</Count>
      <Stock>0</Stock>
    </Widgets_Row>
    <Widgets_Row>
      <ID>3</ID>
      <Title>Widget C</Title>
      <Count>23</Count>
      <Stock>1</Stock>
    </Widgets_Row>
  </Widgets>
</dsQueryResponse>
</queryResponse>
```

5 Security

5.1 Security Considerations for Implementers

This protocol introduces no additional security considerations beyond those applicable to its underlying protocols.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL for Content **DocumentType** query is provided below:

```
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/dsp"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://schemas.microsoft.com/sharepoint/dsp">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/dsp">
      <s:import namespace="http://www.w3.org/2001/XMLSchema" />
      <s:element name="queryRequest" nillable="true">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" name="dsQuery" type="tns:DSQuery" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="DSQuery">
        <s:sequence>
          <s:element minOccurs="0" name="Query" type="tns:DspQuery" />
        </s:sequence>
        <s:attribute name="select" type="s:string" />
        <s:attribute default="both" name="resultContent"
type="tns:ResultContentType" />
        <s:attribute default="element" name="columnMapping"
type="tns:ColumnMappingType" />
        <s:attribute name="resultNamespace" type="s:string" />
        <s:attribute name="resultPrefix" type="s:string" />
        <s:attribute name="resultRoot" type="s:string" />
        <s:attribute name="resultRow" type="s:string" />
        <s:attribute name="startPosition" type="s:string" />
        <s:attribute name="comparisonLocale" type="s:string" />
      </s:complexType>
      <s:complexType name="DspQuery">
        <s:sequence>
          <s:element minOccurs="0" name="Fields" type="tns:Fields" />
          <s:element minOccurs="0" name="Where">
            <s:complexType mixed="true">
              <s:sequence>
                <s:any />
              </s:sequence>
            </s:complexType>
          </s:element>
          <s:element minOccurs="0" name="OrderBy"
type="tns:ArrayOfOrderField" />
        </s:sequence>
        <s:attribute default="-1" name="RowLimit" type="s:long" />
      </s:complexType>
      <s:complexType name="Fields">
        <s:choice>
          <s:element name="AllFields" type="tns:AllFields" />
          <s:sequence>
            <s:element name="Field" type="tns:Field" maxOccurs="unbounded" />
          </s:sequence>
        </s:choice>
      </s:complexType>
    </s:schema>
  </wsdl:types>

```



```

    </s:choice>
</s:complexType>
<s:complexType name="Field">
  <s:attribute name="Name" type="s:string" />
  <s:attribute name="Alias" type="s:string" />
</s:complexType>
<s:complexType name="AllFields">
  <s:attribute default="false" name="IncludeHiddenFields"
    type="s:boolean" />
</s:complexType>
<s:complexType name="ArrayOfOrderField">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="OrderField"
      type="tns:OrderField" />
  </s:sequence>
</s:complexType>
<s:complexType name="OrderField">
  <s:attribute name="Name" type="s:string" />
  <s:attribute default="ASC" name="Direction"
    type="tns:OrderDirection" />
</s:complexType>
<s:simpleType name="OrderDirection">
  <s:restriction base="s:string">
    <s:enumeration value="ASC" />
    <s:enumeration value="DESC" />
  </s:restriction>
</s:simpleType>
<s:simpleType name="ResultContentType">
  <s:restriction base="s:string">
    <s:enumeration value="both" />
    <s:enumeration value="schemaOnly" />
    <s:enumeration value="dataOnly" />
  </s:restriction>
</s:simpleType>
<s:simpleType name="ColumnMappingType">
  <s:restriction base="s:string">
    <s:enumeration value="element" />
    <s:enumeration value="attribute" />
  </s:restriction>
</s:simpleType>
<s:element name="queryResponse">
  <s:complexType mixed="true">
    <s:sequence>
      <s:element name="dsQueryResponse" type="tns:DSQueryResponse"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="DSQueryResponse">
  <s:complexType>
    <s:sequence>
      <s:any/>
    </s:sequence>
    <s:attribute name="status" type="s:string"/>
    <s:attribute name="comparisionLocale" type="s:string"/>
  </s:complexType>
</s:element>
<s:element name="authentication">
  <s:complexType>
    <s:sequence>

```

```

        <s:any minOccurs="0" maxOccurs="unbounded" />
    </s:sequence>
    <s:anyAttribute />
</s:complexType>
</s:element>
<s:element name="dataRoot">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" name="root" type="s:string" />
        </s:sequence>
        <s:attribute default="true" name="allowRemoteDataAccess" type="s:boolean" />
        <s:anyAttribute />
    </s:complexType>
</s:element>
<s:element name="request">
    <s:complexType>
        <s:attribute name="document" type="tns:DocumentType"
            use="required" />
        <s:attribute name="method" type="tns:MethodType" use="required" />
        <s:anyAttribute />
    </s:complexType>
</s:element>
<s:simpleType name="DocumentType">
    <s:restriction base="s:string">
        <s:enumeration value="content" />
        <s:enumeration value="system" />
    </s:restriction>
</s:simpleType>
<s:simpleType name="MethodType">
    <s:restriction base="s:string">
        <s:enumeration value="query" />
    </s:restriction>
</s:simpleType>
<s:element name="versions">
    <s:complexType>
        <s:sequence>
            <s:element minOccurs="0" maxOccurs="unbounded" name="version"
                type="s:string" />
        </s:sequence>
        <s:anyAttribute />
    </s:complexType>
</s:element>
</s:schema>
</wsdl:types>
<wsdl:message name="queryRequestSoapIn">
    <wsdl:part name="Request" element="tns:queryRequest" />
</wsdl:message>
<wsdl:message name="queryRequestSoapOut">
    <wsdl:part name="queryRequestResult" element="tns:queryResponse" />
</wsdl:message>
<wsdl:message name="queryRequestauthentication">
    <wsdl:part name="authentication" element="tns:authentication" />
</wsdl:message>
<wsdl:message name="queryRequestdataRoot">
    <wsdl:part name="dataRoot" element="tns:dataRoot" />
</wsdl:message>
<wsdl:message name="queryRequestrequest">
    <wsdl:part name="request" element="tns:request" />
</wsdl:message>

```

```

<wsdl:message name="queryRequestversions">
  <wsdl:part name="versions" element="tns:versions" />
</wsdl:message>
<wsdl:portType name="StsAdapterSoap">
  <wsdl:operation name="Query">
    <wsdl:input name="queryRequest" message="tns:queryRequestSoapIn" />
    <wsdl:output name="queryRequest" message="tns:queryRequestSoapOut" />
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="StsAdapterSoap" type="tns:StsAdapterSoap">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="Query">
    <soap:operation
      soapAction="http://schemas.microsoft.com/sharepoint/dsp/queryRequest"
      style="document" />
    <wsdl:input name="queryRequest">
      <soap:body use="literal" />
      <soap:header message="tns:queryRequestauthentication"
        part="authentication" use="literal" />
      <soap:header message="tns:queryRequestdataRoot" part="dataRoot"
        use="literal" />
      <soap:header message="tns:queryRequestrequest" part="request"
        use="literal" />
      <soap:header message="tns:queryRequestversions" part="versions"
        use="literal" />
    </wsdl:input>
    <wsdl:output name="queryRequest">
      <soap:body use="literal" />
      <soap:header message="tns:queryRequestversions" part="versions"
        use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="StsAdapterSoap12" type="tns:StsAdapterSoap">
  <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <wsdl:operation name="Query">
    <soap12:operation
      soapAction="http://schemas.microsoft.com/sharepoint/dsp/queryRequest"
      style="document" />
    <wsdl:input name="queryRequest">
      <soap12:body use="literal" />
      <soap12:header message="tns:queryRequestauthentication"
        part="authentication" use="literal" />
      <soap12:header message="tns:queryRequestdataRoot" part="dataRoot"
        use="literal" />
      <soap12:header message="tns:queryRequestrequest" part="request"
        use="literal" />
      <soap12:header message="tns:queryRequestversions" part="versions"
        use="literal" />
    </wsdl:input>
    <wsdl:output name="queryRequest">
      <soap12:body use="literal" />
      <soap12:header message="tns:queryRequestversions" part="versions"
        use="literal" />
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
</wsdl:definitions>

```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® Office SharePoint® Designer 2007
- Microsoft® SharePoint® Designer 2010
- Windows® SharePoint® Services 2.0
- Windows® SharePoint® Services 3.0
- Microsoft® SharePoint® Foundation 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

[<1> Section 3.1.4.1.3.1.3:](#) In Windows SharePoint Services 3.0, the columnMapping setting has no effect on the schema returned in the response, always returning schema for the element setting of columnMapping. However, the XML data payload does respect this setting as described in 3.1.4.1.2.3.

8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

9 Index

A

Abstract data model
[server](#) 10
[Applicability](#) 7
[Attribute groups](#) 9
[Attributes](#) 9

C

[Capability negotiation](#) 7
[Change tracking](#) 37
Client
[overview](#) 10
[Complex types](#) 9
[SOAPFaultDetails](#) 9

D

Data model - abstract
[server](#) 10
[Directory service schema elements](#) 9

E

[Elements - directory service schema](#) 9
Events
[local - server](#) 27
[timer - server](#) 26
Examples
[obtaining Filtered List Data](#) 29
[obtaining list data and schema](#) 28
[obtaining the list schema](#) 28
[overview](#) 28

F

[Fields - vendor-extensible](#) 7
[Full WSDL](#) 32

G

[Glossary](#) 5
[Groups](#) 9

I

[Implementer - security considerations](#) 31
[Index of security parameters](#) 31
[Informative references](#) 6
Initialization
[server](#) 10
[Introduction](#) 5

L

Local events
[server](#) 27

M

Message processing
[server](#) 10
Messages
[attribute groups](#) 9
[attributes](#) 9
[complex types](#) 9
[elements](#) 8
[enumerated](#) 8
[groups](#) 9
[namespaces](#) 8
[simple types](#) 9
[SOAPFaultDetails complex type](#) 9
[syntax](#) 8
[transport](#) 8

N

[Namespaces](#) 8
[Normative references](#) 5

O

[Obtaining filtered list data example](#) 29
[Obtaining list data and schema example](#) 28
[Obtaining the list schema example](#) 28
Operations
[Query](#) 11
[Overview \(synopsis\)](#) 6

P

[Parameters - security index](#) 31
[Preconditions](#) 7
[Prerequisites](#) 7
[Product behavior](#) 36

R

References
[informative](#) 6
[normative](#) 5
[Relationship to other protocols](#) 6

S

[Schema elements - directory service](#) 9
Security
[implementer considerations](#) 31
[parameter index](#) 31
Sequencing rules
[server](#) 10
Server
[abstract data model](#) 10
[initialization](#) 10
[local events](#) 27
[message processing](#) 10

- [overview](#) 10
- [Query operation](#) 11
- [sequencing rules](#) 10
- [timer events](#) 26
- [timers](#) 10
- [Simple types](#) 9
- [SOAPFaultDetails complex type](#) 9
- [Standards assignments](#) 7
- Syntax
 - [messages - overview](#) 8

T

- Timer events
 - [server](#) 26
- Timers
 - [server](#) 10
- [Tracking changes](#) 37
- [Transport](#) 8
- Types
 - [complex](#) 9
 - [simple](#) 9

V

- [Vendor-extensible fields](#) 7
- [Versioning](#) 7

W

- [WSDL](#) 32