[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 Open Specification Promise or the Community Promise. 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.

Preliminary Documentation. This Open Specification provides documentation for past and current releases and/or for the pre-release (beta) version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is preliminary documentation for the pre-release (beta) versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

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.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
01/20/2012	3.0	Major	Significantly changed the technical content.
04/11/2012	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	3.0	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

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.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
language code identifier (LCID)

The following terms are defined in [MS-OFCGLOS]:

list
Simple Object Access Protocol (SOAP)
site
site collection
SOAP action
SOAP body
SOAP fault
Uniform Resource Locator (URL)
Web service
Web Services Description Language (WSDL)
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

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.

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-WSSCAML] Microsoft Corporation, "Collaborative Application Markup Language (CAML) Structure Specification".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.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

[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] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[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".

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

[MS-WPPS] Microsoft Corporation, "Web Part Pages Web Service Protocol Specification".

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

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** message protocol for formatting request and response messages, as described in [SOAP1.1], [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages by using **HTTP**, as described in [RFC2616], or **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [RFC2818].

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

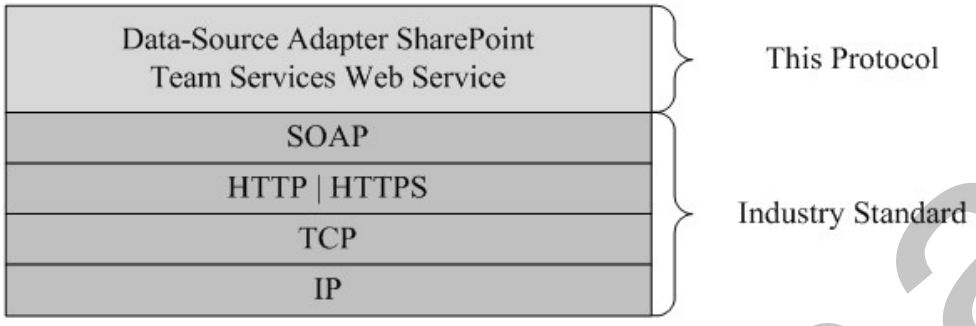

Figure 1: This protocol in relation to other protocols

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 describes methods for obtaining data from data sources and includes the following 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 described 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 differ from 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 might specify differences 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, or in [SOAP1.2/1] section 5. Protocol server faults MUST be returned either using HTTP Status Codes, as specified in [RFC2616] section 10, or using **SOAP faults**, as specified either in [SOAP1.1] section 4.4, or in [SOAP1.2/1] section 5.4.

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses **XML schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and **WSDL**, 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

This specification does not define any common WSDL message definitions.

2.2.3 Elements

This specification does not define any common XML schema element definitions.

2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions defined by this specification. XML schema complex type definitions that are specific to a particular operation are described with the operation.

Complex type	Description
SOAPFaultDetails	

2.2.4.1 SOAPFaultDetails

The details of a SOAP fault. This complex type is defined as follows.

```
<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

This specification does not define any common XML schema simple type definitions.

2.2.6 Attributes

This specification does not define any common XML schema attribute definitions.

2.2.7 Groups

This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

3 Protocol Details

In the following sections, the schema definition might differ from 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 might specify differences 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 [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, or a **Detail** element, as specified in [SOAP1.2/1] section 5.4.5, which conforms to the XML schema of the **SOAPFaultDetails** complex type specified in section 2.2.4.1. 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 specification does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this specification.

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.

The following table summarizes the list of WSDL operations as defined by this specification.

Operation	Description
Query	Accepts a request consisting of two parts.

3.1.4.1 Query

The **Query** method accepts a request that 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. It is defined as follows.

```
<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<1> respond with a **queryRequestSoapOut** response message that consists of either an XML data payload or a schema, or both.

3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
queryRequestSoapIn	A request to initiate a **Query** operation on the protocol server.
queryRequestSoapOut	A response from the protocol server at completion of the **Query** operation.

3.1.4.1.1.1 queryRequestSoapIn

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

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

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

3.1.4.1.1.2 queryRequestSoapOut

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

3.1.4.1.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
queryRequest	Body of the **queryRequestSoapIn** message.
queryResponse	Body of the **queryRequestSoapOut** message
authentication	Specifies what authentication headers are expected in the request message.

Element	Description
dataRoot	Specifies the site that the protocol uses for processing queries.
Request	Describes the type of data being requested and the type of method being called.
versions	Describes the version of the protocol that is being used for the query.

3.1.4.1.2.1 queryRequest

Element named "queryRequest" that contains a **DSQuery** element. It is defined as follows.

```
<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.

3.1.4.1.2.2 queryResponse

This element contains an XML document that contains the result data in response to a query. It is defined as follows.

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

3.1.4.1.2.3 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. It is defined as follows.

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

3.1.4.1.2.4 dataRoot

dataRoot specifies the site that the protocol uses for processing queries. It is defined as follows.

```
<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>
```

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".

3.1.4.1.2.5 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. It is defined as follows.

```
<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 3.1.4.1.4.4.

Method: Attribute as described in section 3.1.4.1.4.5.

3.1.4.1.2.6 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. It is defined as follows.

```
<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.

3.1.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
DSQuery	Core structure of a request.
DspQuery	Describes how the data is manipulated.
Fields	Contains either a list of **Field** elements or the **AllFields** element.
Field	Specifies the name of the list column to be returned.
AllFields	All the fields in the list.
ArrayOfOrderField	Contains a list of **OrderField** elements..
OrderField	Specifies the field to be sorted in the request.

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. It is defined as follows.

```
<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.

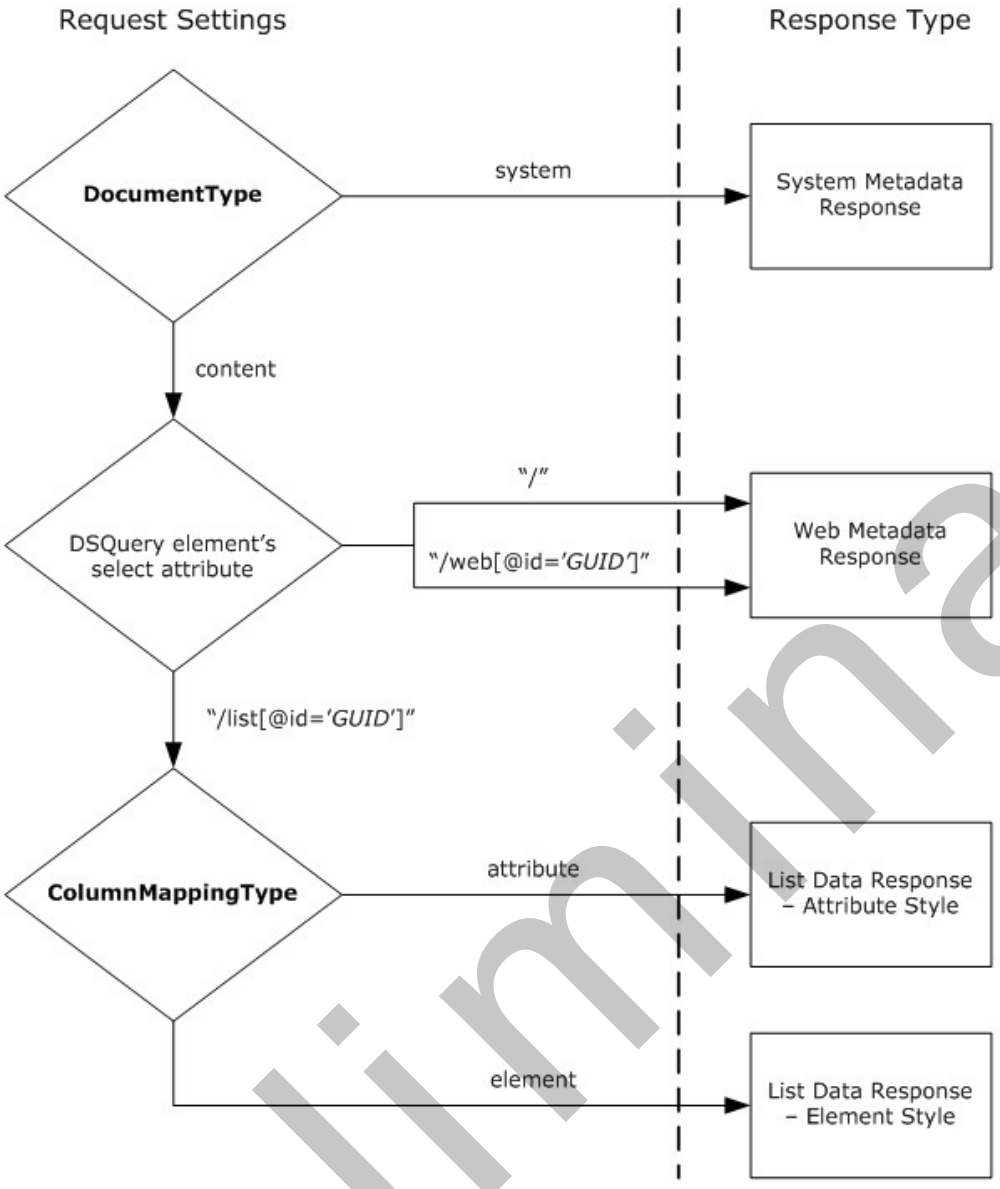

Figure 2: How various request settings determine the response type

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.

comparisionLocale: Used to specify the locale used for **string** comparisons. If set, the value MUST be a **language code identifier (LCID)**.

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

resultContent: Element as specified in 3.1.4.1.4.2.

columnMapping: Element as specified in 3.1.4.1.4.3.

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 namespace, the 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 namespace **string**, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, the namespace for the data payload MUST be set as the **resultNamespace** with the 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 namespace prefix for the 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.

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	<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> </x:schema>

Expression	Result	
/	Data	<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>
/dataRoot	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: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>
/dataRoot	Data	<d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:dataRoot> <d:rootFormat>URL</d:rootFormat> </d:dataRoot> </d:dspSts>
/querySupport	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:element name="dspSts"> <x:complexType> <x:all> <x:element name="querySupport">

Expression	Result	
		<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>
/querySupport	Data	<d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:querySupport> <d:queryType>DSPQ</d:queryType> </d:querySupport> </d:dspSts>
/versions	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: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>
/versions	Data	<d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:versions> <d:version>1.0</d:version> </d:versions> </d:dspSts>
/authentication	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:element name="dspSts">

Expression	Result	
		<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>
/authentication	Data	<d:dspSts xmlns:d="http://schemas.microsoft.com/sharepoint/dsp"> <d:authentication /> </d:dspSts>

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 namespace, the 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 namespace string, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, the namespace for the data payload MUST be set as the **resultNamespace** with the 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 namespace prefix for the 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 slash (/), 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 a namespace, the 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 namespace **string**, the response MUST be an exception.

resultPrefix: If **resultPrefix** and **resultNamespace** are set, the namespace for the data payload MUST be the **resultNamespace** with the 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 namespace prefix for the namespace.

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

resultRow: If set to a non-empty **string**, the response MUST use the **resultRow** value as the name of the element for each row of data in the data payload. If not set or if set to an empty **string**, 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 data payload.

comparisionLocale: 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, any **string** comparisons MUST use the **comparisonLocale** value.

Query: Element as specified in section 3.1.4.1.3.2.

resultContent: Attribute as specified in section 3.1.4.1.4.2.

columnMapping: Attribute as specified in section 3.1.4.1.4.3.

Based on the **columnMapping** setting, the response SHOULD<2> contain data that conforms to the following table.

The value of the **columnMapping** attribute in the columnMapping column results in data that 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 that is a child of the **resultRow** element MUST contain one element or attribute entry for each column of data that is returned in the response.

columnMapping attribute	Schema
element	<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>
attribute	<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>

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. For example:

```
<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. It is defined as follows.

```
<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, as follows.

```
<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, as follows.

```
<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. It is defined as follows.

```
<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". It is defined as follows.

```
<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 Simple Types

The following table summarizes the XML schema simple type definitions that are specific to this operation.

Simple type	Description
OrderDirection	Specifies sort order.
ResultContentType	Specifies what content is returned.
ColumnMappingType	Used to determine the format of the data returned.
DocumentType	Specifies data source being returned.
MethodType	Type of operation to be performed.

3.1.4.1.4.1 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") as follows.

```
<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 possible values for **OrderDirection**.

Value	Meaning
ASC	Sort order is ascending
DESC	Sort order is descending

3.1.4.1.4.2 ResultContentType

ResultContentType specifies what content to include in the response. It is defined as follows.

```
<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 possible values for **ResultContentType**.

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

3.1.4.1.4.3 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. For example:

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

The following table defines possible 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.

3.1.4.1.4.4 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. It is defined as follows.

```
<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 possible values for **DocumentType**.

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

3.1.4.1.4.5 MethodType

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

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

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

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 fields described in the following table.

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

The data for the "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 Obtain 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 Obtain the List Schema

Setting the **resultContent** attribute to "schemaOnly" results 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 Obtain 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 and schema are provided in this appendix.

```
<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: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
- Microsoft® SharePoint® Designer 2013 Preview
- Windows® SharePoint® Services 2.0
- Windows® SharePoint® Services 3.0
- Microsoft® SharePoint® Foundation 2010
- Microsoft® SharePoint® Foundation 2013 Preview

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: SharePoint Foundation 2010 (SP1) and SharePoint Foundation 2010 return a SOAP fault with the error string "Request is empty."

<2> 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 data payload does respect this setting as described in section 3.1.4.1.4.3.

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

S

T

V

W