## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/4/2008</td>
<td>0.1</td>
<td>New</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>6/27/2008</td>
<td>1.0</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>12/12/2008</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>7/13/2009</td>
<td>1.02</td>
<td>Major</td>
<td>Changes made for template compliance</td>
</tr>
<tr>
<td>8/28/2009</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/6/2009</td>
<td>1.04</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>2/19/2010</td>
<td>2.0</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>3/31/2010</td>
<td>2.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>4/30/2010</td>
<td>2.02</td>
<td>Minor</td>
<td>Updated the technical content</td>
</tr>
<tr>
<td>6/7/2010</td>
<td>2.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/29/2010</td>
<td>2.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>7/23/2010</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/27/2010</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/18/2011</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>6/10/2011</td>
<td>2.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>1/20/2012</td>
<td>3.0</td>
<td>Major</td>
<td>Significantly changed the technical content</td>
</tr>
<tr>
<td>4/11/2012</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/12/2012</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>3.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>3.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/30/2013</td>
<td>3.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>3.3</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------</td>
<td>----------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>3.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>3.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>3.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>3.4</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/26/2016</td>
<td>4.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>7/15/2016</td>
<td>4.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>4.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/24/2018</td>
<td>5.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>10/1/2018</td>
<td>6.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>12/11/2018</td>
<td>6.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>3/19/2019</td>
<td>6.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction ........................................................................................................ 6  
  1.1 Glossary ........................................................................................................... 6  
  1.2 References ....................................................................................................... 7  
  1.2.1 Normative References ................................................................................... 7  
  1.2.2 Informative References .................................................................................. 8  
  1.3 Overview .......................................................................................................... 8  
  1.4 Relationship to Other Protocols ....................................................................... 8  
  1.5 Prerequisites/Preconditions .............................................................................. 9  
  1.6 Applicability Statement .................................................................................... 9  
  1.7 Versioning and Capability Negotiation ............................................................. 9  
  1.8 Vendor-Extensible Fields .................................................................................. 9  
  1.9 Standards Assignments ..................................................................................... 9  

2 Messages ............................................................................................................. 10  
  2.1 Transport .......................................................................................................... 10  
  2.2 Common Message Syntax .................................................................................. 10  
    2.2.1 Namespaces .................................................................................................. 10  
    2.2.2 Messages ...................................................................................................... 10  
    2.2.3 Elements ....................................................................................................... 10  
    2.2.4 Complex Types ............................................................................................... 11  
    2.2.5 Simple Types .................................................................................................. 11  
    2.2.6 Attributes ...................................................................................................... 11  
    2.2.7 Groups .......................................................................................................... 11  
    2.2.8 Attribute Groups ............................................................................................ 11  

3 Protocol Details .................................................................................................. 12  
  3.1 Server Details ................................................................................................... 12  
    3.1.1 Abstract Data Model ...................................................................................... 12  
    3.1.2 Timers .......................................................................................................... 12  
    3.1.3 Initialization .................................................................................................. 12  
    3.1.4 Message Processing Events and Sequencing Rules .................................... 12  
    3.1.4.1 Query ......................................................................................................... 13  
      3.1.4.1.1 Messages .............................................................................................. 13  
      3.1.4.1.1.1 queryRequestSoapIn ......................................................................... 13  
      3.1.4.1.1.2 queryRequestSoapOut .................................................................... 13  
      3.1.4.1.2 Elements ............................................................................................... 13  
      3.1.4.1.2.1 queryRequest .................................................................................... 14  
      3.1.4.1.2.2 queryResponse .................................................................................. 14  
      3.1.4.1.2.3 authentication .................................................................................... 14  
      3.1.4.1.2.4 dataRoot ............................................................................................ 15  
      3.1.4.1.2.5 request ............................................................................................... 15  
      3.1.4.1.2.6 versions ............................................................................................. 15  
    3.1.4.1.3 Complex Types ....................................................................................... 16  
      3.1.4.1.3.1 DSQuery ........................................................................................... 16  
      3.1.4.1.3.1.1 System Metadata Response ........................................................... 18  
      3.1.4.1.3.1.2 Web Metadata Response ............................................................... 22  
      3.1.4.1.3.1.3 List Data Response ...................................................................... 24  
      3.1.4.1.3.2 DsQuery ............................................................................................. 25  
      3.1.4.1.3.3 Fields ................................................................................................. 26  
      3.1.4.1.3.4 Field .................................................................................................... 26  
      3.1.4.1.3.5 AllFields ............................................................................................ 27  
      3.1.4.1.3.6 ArrayOfOrderField ......................................................................... 27  
      3.1.4.1.3.7 OrderField ........................................................................................ 27  
    3.1.4.1.4 Simple Types .......................................................................................... 27  
      3.1.4.1.4.1 OrderDirection ................................................................................ 28
1 Introduction

The Data-Source Adapter SharePoint Team Services Service 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 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

**authentication**: The act of proving an identity to a server while providing key material that binds the identity to subsequent communications.

**Hypertext Transfer Protocol (HTTP)**: An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

**Hypertext Transfer Protocol Secure (HTTPS)**: An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [SSL3] and [RFC5246].

**language code identifier (LCID)**: A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.

**list**: A container within a SharePoint site that stores list items. A list has a customizable schema that is composed of one or more fields.

**site**: A group of related pages and data within a SharePoint site collection. The structure and content of a site is based on a site definition. Also referred to as SharePoint site and web site.

**site collection**: A set of websites that are in the same content database, have the same owner, and share administration settings. A site collection can be identified by a GUID or the URL of the top-level site for the site collection. Each site collection contains a top-level site, can contain one or more subsites, and can have a shared navigational structure.

**SOAP**: A lightweight protocol for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics. SOAP 1.2 supersedes SOAP 1.1. See [SOAP1.2-1/2007].

**SOAP action**: The HTTP request header field used to indicate the intent of the SOAP request, using a URI value. See [SOAP1.1] section 6.1.1 for more information.

**SOAP body**: A container for the payload data being delivered by a SOAP message to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

**SOAP fault**: A container for error and status information within a SOAP message. See [SOAP1.2-1/2007] section 5.4 for more information.

**Uniform Resource Locator (URL)**: A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].
Web service: A unit of application logic that provides data and services to other applications and can be called by using standard Internet transport protocols such as HTTP, Simple Mail Transfer Protocol (SMTP), or File Transfer Protocol (FTP). Web services can perform functions that range from simple requests to complicated business processes.

Web Services Description Language (WSDL): An XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are described abstractly and are bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints, which describe a network service. WSDL is extensible, which allows the description of endpoints and their messages regardless of the message formats or network protocols that are used.

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML namespace prefix: An abbreviated form of an XML namespace, as described in [XML].

XML schema: A description of a type of XML document that is typically expressed in terms of constraints on the structure and content of documents of that type, in addition to the basic syntax constraints that are imposed by XML itself. An XML schema provides a view of a document type at a relatively high level of abstraction.

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

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

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.

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


1.2.2 Informative References

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


1.3 Overview

This protocol provides access to list data via a web service. The web service accepts a query that 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:
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 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, not null, and present.

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 [XMLSCHEMA1/2] and [XMLSCHEMA2/2], 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.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace URI</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>soap</td>
<td><a href="http://schemas.xmlsoap.org/wsd1/soap/">http://schemas.xmlsoap.org/wsd1/soap/</a></td>
<td>[SOAP1.1]</td>
</tr>
<tr>
<td>tns</td>
<td><a href="http://schemas.microsoft.com/sharepoint/dsp">http://schemas.microsoft.com/sharepoint/dsp</a></td>
<td></td>
</tr>
<tr>
<td>s</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
<td>[XMLSCHEMA1], [XMLSCHEMA2]</td>
</tr>
<tr>
<td>soap12</td>
<td><a href="http://schemas.xmlsoap.org/wsd1/soap12/">http://schemas.xmlsoap.org/wsd1/soap12/</a></td>
<td>[SOAP1.2/1], [SOAP1.2/2]</td>
</tr>
<tr>
<td>(none)</td>
<td><a href="http://schemas.microsoft.com/sharepoint/dsp">http://schemas.microsoft.com/sharepoint/dsp</a></td>
<td></td>
</tr>
<tr>
<td>wsd1</td>
<td><a href="http://schemas.xmlsoap.org/wsd1/">http://schemas.xmlsoap.org/wsd1/</a></td>
<td>[WSDL]</td>
</tr>
</tbody>
</table>

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

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.
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, not null, and present.

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. 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.

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

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

When the client sends a **queryRequestSoapIn** request message, the server SHOULD 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.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryRequestSoapIn</td>
<td>A request to initiate a <strong>Query</strong> operation on the protocol server.</td>
</tr>
<tr>
<td>queryRequestSoapOut</td>
<td>A response from the protocol server at the completion of the <strong>Query</strong> operation.</td>
</tr>
</tbody>
</table>

#### 3.1.4.1.1.1 queryRequestSoapIn

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

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

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.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryRequest</td>
<td>The body of the <strong>queryRequestSoapIn</strong> message.</td>
</tr>
</tbody>
</table>
### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queryResponse</td>
<td>The body of the queryRequestSoapOut message</td>
</tr>
<tr>
<td>authentication</td>
<td>Specifies which authentication headers are expected in the request message.</td>
</tr>
<tr>
<td>dataRoot</td>
<td>Specifies the site that the protocol uses for processing queries.</td>
</tr>
<tr>
<td>Request</td>
<td>Describes the type of data that is being requested and the type of method that is being called.</td>
</tr>
<tr>
<td>versions</td>
<td>Describes the version of the protocol that is being used for the query.</td>
</tr>
</tbody>
</table>

#### 3.1.4.1.2.1 queryRequest

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

```xml
<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.

```xml
<s:element name="queryResponse">
    <s:complexType mixed="true">
        <s:sequence>
            <s:element name="any" minOccurs="0" maxOccurs="unbounded"/>
        </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.

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

```xml
<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 will be "http://server/site". If set, the value will 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 will 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 is returned by the server. It is defined as follows.

```xml
<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 is returned by the server. It is defined as follows.

```xml
<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.

<table>
<thead>
<tr>
<th>Complex type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSQuery</td>
<td>The core structure of a request.</td>
</tr>
<tr>
<td>DspQuery</td>
<td>Describes how the data is manipulated.</td>
</tr>
<tr>
<td>Fields</td>
<td>Contains either a list of Field elements or the AllFields element.</td>
</tr>
<tr>
<td>Field</td>
<td>Specifies the name of the list column to be returned.</td>
</tr>
<tr>
<td>AllFields</td>
<td>All the fields in the list.</td>
</tr>
<tr>
<td>ArrayOfOrderField</td>
<td>Contains a list of OrderField elements.</td>
</tr>
<tr>
<td>OrderField</td>
<td>Specifies the field to be sorted in the request.</td>
</tr>
</tbody>
</table>

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.

```xml
<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="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.
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.

<table>
<thead>
<tr>
<th>Document type</th>
<th>select attribute</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>&quot;/&quot;, &quot;/dataRoot&quot;, &quot;/authentication&quot;, &quot;/versions&quot;, &quot;/querySupport&quot;</td>
<td>System Metadata Response. See 3.1.4.1.3.1.1</td>
</tr>
<tr>
<td>Document type</td>
<td>select attribute</td>
<td>Response</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Content</td>
<td>&quot;/&quot;,&quot;/web[@id='GUID']&quot;</td>
<td>Web Metadata Response. See 3.1.4.1.3.1.2.</td>
</tr>
<tr>
<td>Content</td>
<td>&quot;/list[@id='GUID']&quot;</td>
<td>List Data Response. See 3.1.4.1.3.1.3.</td>
</tr>
</tbody>
</table>

**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 the query, the value of startPosition SHOULD be empty. The response result contains the value to do next page request, in pagingInfo 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 a language code identifier (LCID).

**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.

**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.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>Schema</td>
</tr>
</tbody>
</table>

```
<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>
```
<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>Data</td>
</tr>
</tbody>
</table>
|            |     <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:dspSts> |
| /dataRoot  | Schema |
|            |     <x:element name="dspSts">  
|            |     <x:complexType>  
|            |         <x:all>  
|            |         <x:element name="dataRoot">  
|            |         </x:complexType>  
|            |     </x:element>  
|            |     </x:complexType>  
|            |     </x:schema> |
| /dataRoot  | Data   |
|            |     <d:dataRoot>  
|            |     <d:rootFormat>URL</d:rootFormat>  
|            |     </d:dataRoot>  
<p>|            |     &lt;/d:dspSts&gt; |</p>
<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>/querySupport</td>
<td>Schema</td>
</tr>
<tr>
<td></td>
<td>&lt;xs:schema xmlns:d=&quot;<a href="http://schemas.microsoft.com/sharepoint/dsp">http://schemas.microsoft.com/sharepoint/dsp</a>&quot;</td>
</tr>
<tr>
<td></td>
<td>targetNamespace=&quot;<a href="http://schemas.microsoft.com/sharepoint/dsp">http://schemas.microsoft.com/sharepoint/dsp</a>&quot;</td>
</tr>
<tr>
<td></td>
<td>elementFormDefault=&quot;qualified&quot;</td>
</tr>
<tr>
<td></td>
<td>attributeFormDefault=&quot;unqualified&quot;</td>
</tr>
<tr>
<td></td>
<td>xmlns:x=&quot;<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>&quot;</td>
</tr>
</tbody>
</table>
|                    |     <xs:element name="dspSts">
|                    |         <xs:complexType>
|                    |             <xs:all>
|                    |                 <xs:element name="querySupport">
|                    |                   <xs:complexType>
|                    |                     <xs:sequence>
|                    |                         <xs:element name="queryType"                     |
|                    |         maxOccurs="unbounded" type="xs:string" />                    |
|                    |                     </xs:sequence>
|                    |                   </xs:complexType>
|                    |                 </xs:element>
|                    |             </xs:all>
|                    |         </xs:complexType>
|                    |     </xs:element>
|                    | </xs:schema>                                                          |
| /querySupport      | Data                                                                    |
|                    |     <d:querySupport>                                                  |
|                    |         <d:queryType>DSPQ</d:queryType>                               |
|                    |     </d:querySupport>                                                |
|                    | </d:dspSts>                                                           |
| /versions          | Schema                                                                 |
|                    | <xs: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"                        |
|                    |     <xs:element name="dspSts">
|                    |         <xs:complexType>
|                    |             <xs:all>
|                    |                 <xs:element name="versions">
|                    |                   <xs:complexType>
|                    |                     <xs:sequence>
|                    |                         <xs:element name="version"                     |
|                    |         maxOccurs="unbounded" type="xs:string" />                    |
|                    |                     </xs:sequence>
|                    |                   </xs:complexType>
|                    |                 </xs:element>
|                    |             </xs:all>
|                    |         </xs:complexType>
|                    |     </xs:element>
<p>|                    | &lt;/xs:schema&gt;                                                          |</p>
<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>/versions</td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td><code>&lt;d: versions&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;d: version&gt;1.0&lt;/d: version&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/d: versions&gt;</code></td>
</tr>
<tr>
<td>/authentication</td>
<td>Schema</td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:element name=&quot;dspSts&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:complexType&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:all&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:element name=&quot;authentication&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:complexType&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:sequence&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;x:element name=&quot;authMethod&quot; minOccurs=&quot;0&quot; maxOccurs=&quot;unbounded&quot; type=&quot;x:string&quot; /&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:sequence&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:complexType&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:element&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>/all&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:complexType&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:element&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/x:schema&gt;</code></td>
</tr>
<tr>
<td>/authentication</td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td><code>&lt;d: authentication /&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;/d: dspSts&gt;</code></td>
</tr>
</tbody>
</table>

### 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.
comparisonLocale: 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:

```xml
<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>
```

web.web: Specifies a subsite of the requested web object.
web.list: Specifies a list of the requested web object.
id: A string that identifies the requested web object.

The following table describes properties on the list or web object that is returned.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Required. A string that identifies the list or web object.</td>
</tr>
<tr>
<td>displayName</td>
<td>Optional. Specifies a string containing the friendly name for the list or web object. This property is used as required.</td>
</tr>
<tr>
<td>contentType</td>
<td>Required. Specifies the content type of the object returned, which is set to RowReturning for list objects and to TableReturning for web objects.</td>
</tr>
<tr>
<td>serverParameters</td>
<td>Required. <strong>None</strong> for list and web objects.</td>
</tr>
</tbody>
</table>
### 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.

**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, 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 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.

<table>
<thead>
<tr>
<th>columnMapping attribute</th>
<th>Schema</th>
</tr>
</thead>
</table>
  <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:attribute />
          </x:complexType>
        </x:element>
      </x:sequence>
    </x:complexType>
  </x:element>
</x:schema>``` |
  <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. It is defined as follows.
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.

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.

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.

```xml
<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.

```xml
<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.

```xml
<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.

<table>
<thead>
<tr>
<th>Simple type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OrderDirection</td>
<td>Specifies the sort order.</td>
</tr>
<tr>
<td>ResultContentType</td>
<td>Specifies which content is returned.</td>
</tr>
</tbody>
</table>
### Simple type

<table>
<thead>
<tr>
<th>Simple type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColumnMappingType</td>
<td>Used to determine the format of the data that is returned.</td>
</tr>
<tr>
<td>DocumentType</td>
<td>Specifies the data source that is being returned.</td>
</tr>
<tr>
<td>MethodType</td>
<td>The type of operation to be performed.</td>
</tr>
</tbody>
</table>

### 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.

```xml
<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**.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASC</td>
<td>Sort order is ascending</td>
</tr>
<tr>
<td>DESC</td>
<td>Sort order is descending</td>
</tr>
</tbody>
</table>

### 3.1.4.1.4.2 ResultContentType

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

```xml
<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**.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>both</td>
<td>Return both schema and list data.</td>
</tr>
<tr>
<td>Value</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>schemaOnly</td>
<td>Return only schema info.</td>
</tr>
<tr>
<td>dataOnly</td>
<td>Return only list data.</td>
</tr>
</tbody>
</table>

### 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:

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

When 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:

```xml
<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:

```xml
<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.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>element</td>
<td>The server MUST send result XML using elements to store column data.</td>
</tr>
<tr>
<td>attribute</td>
<td>The server MUST send result XML using attributes on to store column data.</td>
</tr>
</tbody>
</table>

### 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.

```xml
<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**.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>Return information about a list or Web site.</td>
</tr>
<tr>
<td>system</td>
<td>Return information about the DSPSTSS web service.</td>
</tr>
</tbody>
</table>

### 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.

```xml
<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.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Field type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Counter</td>
</tr>
<tr>
<td>Title</td>
<td>String</td>
</tr>
<tr>
<td>Count</td>
<td>Number</td>
</tr>
<tr>
<td>Stock</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

The data for the "Widgets" list is as follows:

<table>
<thead>
<tr>
<th>ID</th>
<th>Title</th>
<th>Count</th>
<th>Stock</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Widget A</td>
<td>50</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>Widget B</td>
<td>100</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Widget C</td>
<td>23</td>
<td>Yes</td>
</tr>
</tbody>
</table>

4.1 Obtain List Data and Schema

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

Request

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

Response

```xml
<queryResponse xmlns="http://schemas.microsoft.com/sharepoint/dsp">
   <dsQueryResponse status="success">
         <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;Lt;Leq;Gt;Gt;Geq;Gteq;Contains;BeginsWith;" d:displayName="ID" type="x:int" />
                           <x:element name="Title" d:filterSupport="IsNull;IsNotNull;Eq;Neq;Lt;Lt;Leq;Gt;Gt;Geq;Gteq;Contains;BeginsWith;" d:displayName="Title" type="x:string" />
                           <x:element name="Count" minOccurs="0" d:filterSupport="IsNull;IsNotNull;Eq;Neq;Lt;Lt;Leq;Gt;Gt;Geq;Gteq;Contains;BeginsWith;" d:displayName="Count" type="x:float" />
                           <x:element name="Stock" minOccurs="0" d:filterSupport="IsNull;IsNotNull;Eq;Neq;Lt;Lt;Leq;Gt;Gt;Geq;Gteq;Contains;BeginsWith;" d:displayName="Stock" type="x:bool" />
                        </x:sequence>
                    </x:complexType>
                </x:element>
            </x:sequence>
        </x:element>
    </x:complexType>
</x:schema>
```

[MS-DSPSTSS] - v20190319
Data-Source Adapter SharePoint Team Services Web Service Protocol
Copyright © 2019 Microsoft Corporation
Release: March 19, 2019
4.2 Obtain the List Schema

Setting the resultContent attribute to "schemaOnly" results in only schema data being returned.

**Request**

```xml
<queryRequest xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQuery select="/list[@id='{C13B4B16-9982-4c30-B533-2B4068B0C623}']" resultContent="schemaOnly" />
</queryRequest>
```

**Response**

```xml
<queryResponse xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQueryResponse status="success">
      <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;" />
                  <x:element name="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

```xml
<queryRequest xmlns="http://schemas.microsoft.com/sharepoint/dsp">
  <dsQuery select="/list[@id='{(C13E4B16-9982-4C30-B533-2B406B0C623)'}']" resultContent="dataOnly">
    <Query>
      <Where>
        <Gt>
          <FieldRef Name="ID" />  
          <Value>1</Value>
        </Gt>
      </Where>
    </Query>
  </dsQuery>
</queryRequest>
```

Response

```xml
<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.

```xml
<?xml version="1.0" encoding="utf-8"?>
<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="comparisonLocales" 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" minOccurs="0" />
          </s:sequence>
        </s:choice>
      </s:complexType>
    </s:schema>
    <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="comparisonLocales" 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" minOccurs="0" />
          </s:sequence>
        </s:choice>
      </s:complexType>
    </s:schema>
  </wsdl:types>
</wsdl:definitions>
```
<s:complexType 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="comparisonLocale" type="s:string" />
  </s:complexType>
</s:element>

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

<s:element name="request">
  <s:complexType>
    <s:attribute name="document" type="tns:DocumentType"
<s:complexType 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" 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" />
        </wsdl:output>
    </wsdl:binding>
</wsdl:types>
<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 updates to those products.

- Microsoft Office SharePoint Designer 2007
- Microsoft SharePoint Designer 2010
- Microsoft SharePoint Designer 2013
- Windows SharePoint Services 2.0
- Windows SharePoint Services 3.0
- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016
- Microsoft SharePoint Server 2019

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates 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: Microsoft SharePoint Foundation 2010 Service Pack 1 and Microsoft 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; the schema for the element setting of columnMapping is returned. However, the data payload does respect this setting as described in section 3.1.4.1.4.3.
8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class Major means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class Minor means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class None means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Revision class</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.4.1</td>
<td>Query</td>
<td>Minor</td>
</tr>
<tr>
<td>3.1.4.1.3.1</td>
<td>DSQuery</td>
<td>Updated description for startPosition about pagingInfo element.</td>
</tr>
</tbody>
</table>
9 Index

A
Abstract data model

  server 12

Applicability 9

Attribute groups 11

Attributes 11

C

  Capability negotiation 9

  Change tracking 40

  Complex types 11

      server

          AllFields 27
         ArrayOfOrderField 27
          DspQuery 25
          DSQuery 16
          Field 26
          Fields 26
          OrderField 27

D

Data model - abstract

  server 12

E

Elements

      server

          authentication 14
          dataRoot 15
          queryRequest 14
          queryResponse 14
          request 15
          versions 15

Events

      local - server 30

      timer - server 30

Examples

      obtaining Filtered List Data 33

      obtaining list data and schema 31

      obtaining the list schema 32

      overview 31

F

Fields - vendor-extensible 9

      Full WSDL 35

G

Glossary 6

Groups 11

I

Implementer - security considerations 34

      Index of security parameters 34

      Informative references 8

 Initialization

      server 12

      Introduction 6

L

Local events

      server 30

M

Message processing

      server 12

Messages

      attribute groups 11

      attributes 11

      complex types 11

      elements 10

      enumerated 10

      groups 11

      namespaces 10

      server

          queryRequestSoapIn 13

          queryRequestSoapOut 13

          simple types 11

          syntax 10

          transport 10

N

Namespaces 10

Normative references 7

O

Obtaining filtered list data example 33

Obtaining list data and schema example 31

Obtaining the list schema example 32

Operations

      Query 13

      Overview (synopsis) 8

P

Parameters - security index 34

Preconditions 9

Prerequisites 9

Product behavior 39

Protocol Details

      overview 12

R

References 7

      informative 8

      normative 7

Relationship to other protocols 8

S

Security
implementer considerations 34
parameter index 34
Sequencing rules
server 12
Server
abstract data model 12
initialization 12
local events 30
message processing 12
Query operation 13
complex types 16
elements 13
messages 13
simple types 27
sequencing rules 12
timer events 30
timers 12
Simple types 11
server
ColumnMappingType 29
DocumentType 29
MethodType 30
OrderDirection 28
ResultContentType 28
Standards assignments 9
Syntax
messages - overview 10
T
Timer events
server 30
Timers
server 12
Tracking changes 40
Transport 10
Types
complex 11
simple 11
V
Vendor-extensible fields 9
Versioning 9
W
WSDL 35