[MS-ASWS]:
Access Services 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
07/13/2009	0.1	Major	Initial Availability
08/28/2009	0.2	Editorial	Revised and edited the technical content
11/06/2009	0.3	Editorial	Revised and edited the technical content
02/19/2010	1.0	Major	Updated and revised the technical content
03/31/2010	1.01	Editorial	Revised and edited the technical content
04/30/2010	1.02	Editorial	Revised and edited the technical content
06/07/2010	1.03	Minor	Updated the technical content
06/29/2010	1.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.04	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.0	Major	Significantly changed the technical content.
06/10/2011	2.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	2.1	Minor	Clarified the meaning of the technical content.

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 Protocol Overview (Synopsis) 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 10

2 Messages 11
2.1 Transport 11
2.2 Common Message Syntax 11
2.2.1 Namespaces 11
2.2.2 Messages 12
2.2.2.1 Faults 12
2.2.3 Elements 12
2.2.4 Complex Types 12
2.2.4.1 ArrayOfKeyValuePair 12
2.2.4.2 KeyValuePair 12
2.2.4.3 VersionType 13
2.2.5 Simple Types 13
2.2.5.1 UpdateCommand 13
2.2.6 Attributes 14
2.2.7 Groups 14
2.2.8 Attribute Groups 14

3 Protocol Details 15
3.1 Server Details 15
3.1.1 Abstract Data Model 15
3.1.1.1 Access Services Site Template 15
3.1.1.1.1 MSysASO 16
3.1.1.1.1.1 ID 16
3.1.1.1.1.2 Title 16
3.1.1.1.1.3 owshiddenversion 16
3.1.1.1.1.4 Type 17
3.1.1.1.1.5 Revision 18
3.1.1.1.1.6 ClientObject 18
3.1.1.1.1.7 ServerObject 20
3.1.1.1.1.8 ClientObjectProperties 20
3.1.1.1.1.9 Flags 21
3.1.1.1.1.10 Attachments 21
3.1.1.1.2 USysApplicationLog 22
3.1.1.1.2.1 ID 22
3.1.1.1.2.2 Created 22
3.1.1.1.2.3 owshiddenversion 23
3.1.1.1.2.4 Category 23

3.1.1.2 Access Services Site Version 23
3.1.2 Timers 24
3.1.3 Initialization 24
3.1.4 Message Processing Events and Sequencing Rules 24
3.1.4.1 GetAccessServicesVersion 24
3.1.4.1.1 Messages 25
3.1.4.1.1.1 GetAccessServicesVersionSoapIn 25
3.1.4.1.1.2 GetAccessServicesVersionSoapOut 25
3.1.4.1.2 Elements 25
3.1.4.1.2.1 GetAccessServicesVersion 25
3.1.4.1.2.2 GetAccessServicesVersionResponse 25
3.1.4.2 GetCurrentUserInfo 26
3.1.4.2.1 Messages 26
3.1.4.2.1.1 GetCurrentUserInfoSoapIn 26
3.1.4.2.1.2 GetCurrentUserInfoSoapOut 26
3.1.4.2.2 Elements 26
3.1.4.2.2.1 GetCurrentUserInfo 26
3.1.4.2.2.2 GetCurrentUserInfoResponse 27
3.1.4.3 GetDataMacroState 27
3.1.4.3.1 Messages 27
3.1.4.3.1.1 GetDataMacroStateSoapIn 27
3.1.4.3.1.2 GetDataMacroStateSoapOut 28
3.1.4.3.2 Elements 28
3.1.4.3.2.1 GetDataMacroState 28
3.1.4.3.2.2 GetDataMacroStateResponse 28
3.1.4.3.3 Complex Types 29
3.1.4.3.3.1 DataMacroInstanceState 29
3.1.4.3.4 Simple Types 29
3.1.4.3.4.1 DataMacroState 29
3.1.4.4 GetServerInformation 30
3.1.4.4.1 Messages 30
3.1.4.4.1.1 GetServerInformationSoapIn 30
3.1.4.4.1.2 GetServerInformationSoapOut 30
3.1.4.4.2 Elements 31
3.1.4.4.2.1 GetServerInformation 31
3.1.4.4.2.2 GetServerInformationResponse 31
3.1.4.4.3 Complex Types 31
3.1.4.4.3.1 AccessServerInformationType 31
3.1.4.5 RunDataMacro 32
3.1.4.5.1 Messages 32
3.1.4.5.1.1 RunDataMacroSoapIn 32
3.1.4.5.1.2 RunDataMacroSoapOut 32
3.1.4.5.2 Elements 32
3.1.4.5.2.1 RunDataMacro 32
3.1.4.5.2.2 RunDataMacroResponse 33
3.1.4.6 SetAccessServicesVersion 33
3.1.4.6.1 Messages 34
3.1.4.6.1.1 SetAccessServicesVersionSoapIn 34
3.1.4.6.1.2 SetAccessServicesVersionSoapOut 34
3.1.4.6.2 Elements 34
3.1.4.6.2.1 SetAccessServicesVersion 34
3.1.4.6.2.2 SetAccessServicesVersionResponse 35
3.1.4.7 StartCompilation 35

3.1.4.7.1 Messages 35
3.1.4.7.1.1 StartCompilationSoapIn 35
3.1.4.7.1.2 StartCompilationSoapOut 35
3.1.4.7.2 Elements 36
3.1.4.7.2.1 StartCompilation 36
3.1.4.7.2.2 StartCompilationResponse 36
3.1.4.8 UpdateLists 36
3.1.4.8.1 Messages 37
3.1.4.8.1.1 UpdateListsSoapIn 37
3.1.4.8.1.2 UpdateListsSoapOut 37
3.1.4.8.2 Elements 37
3.1.4.8.2.1 UpdateLists 37
3.1.4.8.2.2 UpdateListsResponse 40
3.1.4.8.3 Complex Types 40
3.1.4.8.3.1 Update 40
3.1.4.8.3.2 FieldValue 42
3.1.4.8.3.3 UpdateListsResultInfo 42
3.1.5 Timer Events 42
3.1.6 Other Local Events 42

4 Protocol Examples 43
4.1 GetCurrentUserInfo 43
4.2 Use UpdateLists to Insert Items into a List 43
4.3 Use UpdateLists to Insert Items into Two Lists with Lookup Relationships 45
4.4 RunDataMacro and GetDataMacroState 46

5 Security 49
5.1 Security Considerations for Implementers 49
5.2 Index of Security Parameters 49

6 Appendix A: Full WSDL 50

7 Appendix B: Product Behavior 58

8 Change Tracking 59

9 Index 61

1 Introduction

This document specifies the Access Services Protocol. This protocol enables a protocol client to run and monitor tasks on a server application.

Sections 1.8, 2, and 3 of this specification are normative and contain RFC 2119 language. Sections 1.5 and 1.9 are also normative but cannot contain RFC 2119 language. All other sections and examples in this specification are informative.

1.1 Glossary

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

Augmented Backus-Naur Form (ABNF)
Coordinated Universal Time (UTC)
GUID
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)

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

attachment
available site template
current user
data culture
data macro
database application
endpoint
field
file extension
list
list identifier
list item
list item identifier
lookup field
major version
minor version
session
Simple Object Access Protocol (SOAP)
site
site collection
SOAP action
SOAP body
SOAP fault
subsite
top-level site
Uniform Resource Identifier (URI)
user profile
Web Services Description Language (WSDL)
WSDL operation
XML namespace

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 Specification documents do not include a publishing year because links are to the latest version of the 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-ACCDT] Microsoft Corporation, "Access Template File Format Specification".

[MS-AXL] Microsoft Corporation, "Access Application Transfer Protocol Structure Specification".

[MS-LISTSWS] Microsoft Corporation, "Lists Web Service Protocol Specification".

[MS-SITESS] Microsoft Corporation, "Sites Web Service Protocol Specification".

[MS-SPSTWS] Microsoft Corporation, "SharePoint Security Token Service Web Service Protocol Specification".

[MS-UGS] Microsoft Corporation, "UserGroup Web Service Protocol Specification".

[MS-WSSFO2] Microsoft Corporation, "Windows SharePoint Services (WSS): File Operations Database Communications Version 2 Protocol Specification".

[MS-WSSTS] Microsoft Corporation, "Windows SharePoint Services Technical 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

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.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

[XML] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Fourth Edition)", W3C Recommendation, August 2006, http://www.w3.org/TR/2006/REC-xml-20060816/

[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".

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

1.3 Protocol Overview (Synopsis)

This protocol enables protocol clients to manage data exposed as **lists (1)** by a protocol server. The protocol client issues requests to a protocol server. The protocol server receives, processes, and responds to the requests of protocol clients.

The protocol provides the following sets of functionality:

- Retrieve **user profile** information: The protocol client can get information about the current user from the protocol server through the Web service described in this protocol.
- Manage data stored within lists (1): The protocol client can add, modify, delete, and retrieve data exposed as lists (1) by a protocol server through the Web service described in this protocol.
- Trigger a **data macro**: The protocol client can trigger a data macro through the Web service described in this protocol. The protocol server will respond with a token for this data macro instance to the protocol client. To know whether the data macro has finished, or is still processing, or encountered any error, the protocol client can get the status of the data macro instance from the protocol server by using the token for the data macro instance.
- Managing the **Access Services Site Version** (section 3.1.1.2): The protocol client can retrieve or attempt to set the value of the Access services site version for a **site (2)** or **subsite**.
- Compiling the **database application** defined by a site (2): The protocol client can notify a protocol server to start compiling the database application defined in the **MSysASO** list (section 3.1.1.1.1) of the current site (2) or subsite from the object definitions stored within that list.

The Web services are documented in detail in section 3.

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].

The following diagram shows the underlying messaging and transport stack used by the protocol:

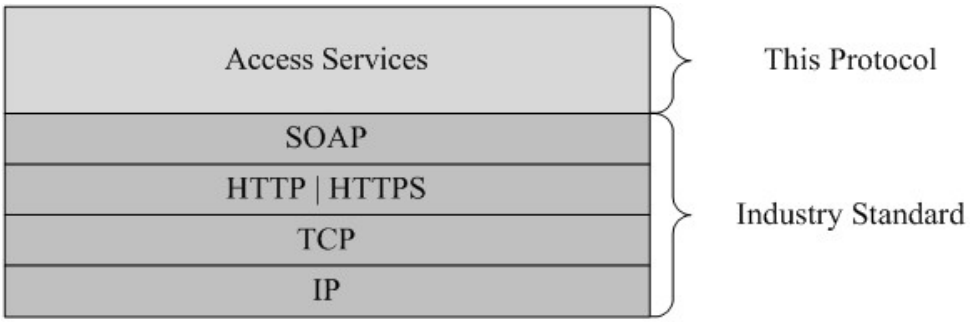

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a site (2) that is identified by a URI that is known by protocol clients. The protocol server **endpoint (4)** is formed by appending "/_vti_bin/ ACCSRV/AccessServer.asmx" to the **URI** of the site (2), for example:

http://www.example.com/Repository/_vti_bin/ ACCSRV/AccessServer.asmx

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

1.6 Applicability Statement

The Access Services Web Services Protocol is applicable in the following scenarios:

- Retrieving user profile information about the **current user**.
- Running a data macro on the protocol server.
- Inserting, updating, or deleting data on the protocol server.
- Retrieving and setting the Access services site version (section 3.1.1.2).
- Triggering compilation of a database application.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported transports:** This protocol can be implemented by using transports that support sending Simple Object Access Protocol (SOAP) messages, as specified in section 2.1.
- **Protocol versions:** This protocol is not versioned.

Capability negotiation: This protocol does not support version negotiation.

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 proprietary 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 or HTTPS.

All protocol messages MUST be transported by using HTTP bindings at the transport level.

Protocol messages MUST be formatted as specified in either [SOAP1.1] section 4 or [SOAP1.2/1] section 5. Protocol server faults MUST be returned by using either HTTP status codes, as specified in [RFC2616] section 10, or **SOAP faults**, as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4.

If the HTTPS transport is used, a server certificate MUST be deployed.

This protocol MAY transmit an additional SOAP header, the **ServiceContext** header, as specified in [MS-SPSTWS].

This protocol does not define any means for activating a protocol server or protocol client. The protocol server MUST be configured and begin listening in an implementation-specific way. In addition, the protocol client MUST know the format and transport that is used by the protocol server, for example, the SOAP format over an HTTP transport.

2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses XML Schema as defined in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL as defined in [WSDL].

2.2.1 Namespaces

This specification defines and references various XML namespaces using the mechanisms specified in [XMLNS]. Although this specification associates a specific 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
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
tns	http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/	
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]

2.2.2 Messages

This section contains common definitions used by this protocol. The syntax of the definitions uses XML Schema as defined in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL as defined in [WSDL].

2.2.2.1 Faults

In the event of an application error, the protocol server returns a SOAP fault as a response to the operation, as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4.

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
ArrayOfKeyValuePair	An array of **KeyValuePair** elements.
KeyValuePair	Specifies the name and value of a parameter ([MS-AXL] section 2.2.3.18) or a return variable ([MS-AXL] section 2.1.3.3.3) for a data macro ([MS-AXL] section 2.1.3).
VersionType	Specifies the **Access Services Site Version** (section 3.1.1.2) of a site (2) or subsite formatted as XML.

2.2.4.1 ArrayOfKeyValuePair

An array of **KeyValuePair** elements.

```
<xs:complexType name="ArrayOfKeyValuePair">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="KeyValuePair"
type="tns:KeyValuePair"/>
  </xs:sequence>
</xs:complexType>
```

KeyValuePair: Specified in section 2.2.4.2.

2.2.4.2 KeyValuePair

Specifies the name and value of a parameter, as specified in [MS-AXL] section 2.2.3.18, or a return variable, as specified in [MS-AXL] section 2.1.3.3.3, for a data macro, as specified in [MS-AXL] section 2.1.3.

```
<xs:complexType name="KeyValuePair">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="Key"/>
    <xs:element minOccurs="1" maxOccurs="1" name="Value" nillable="true"/>
  </xs:sequence>
```

```
</xs:complexType>
```

Key: A value that uniquely identifies the **KeyValuePair** within a collection. MUST be present. MUST be of type **xs:string** or **xs:int**.

Value: Data associated with a given **Key**, which can have any value as long as the document remains well-formed, as specified in [XML] section 2. MUST be present. MUST be formatted in the **data culture** of the **session (2)**.

2.2.4.3 VersionType

Specifies the **Access Services Site Version** (section 3.1.1.2) of a site (2) or subsite formatted as XML.

```
<xs:complexType name="VersionType">
  <xs:attribute name="Major" type="xs:int" use="required"/>
  <xs:attribute name="Minor" type="xs:int" use="required"/>
</xs:complexType>
```

Major: The **major version** of the **Access Services Site Version** (section 3.1.1.2).

Minor: The **minor version** of the **Access Services Site Version** (section 3.1.1.2).

2.2.5 Simple Types

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

Simple type	Description
UpdateCommand	Used to specify whether an **Update** element is specifying an Update, Insert, or Delete command.

2.2.5.1 UpdateCommand

The **UpdateCommand** simple type is used to specify whether an **Update** element is specifying an Update, Insert, or Delete command.

```
<xs:simpleType name="UpdateCommand">
  <xs:restriction base="xs:string">
    <xs:enumeration value="u"/>
    <xs:enumeration value="i"/>
    <xs:enumeration value="d"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for UpdateCommand:

Value	Meaning
u	Update. The data in the **Update** element MUST replace the data in the **list item** specified by the

Value	Meaning
	id attribute of the **Update** element.
i	Insert. The data in the **Update** element MUST be used to insert a new item into a list (1).
d	Delete. The list item that has a **list item identifier** matching the **id** attribute of the **Update** element MUST be deleted from the list (1).

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, present, and not null.

The protocol client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the protocol 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] section 10.

This protocol allows protocol servers to notify protocol clients of application-level faults using SOAP faults. 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.

Access Services Site: A site (2) based on the Access services site template (section 3.1.1.1) that supports Access services site versions (section 3.1.1.2).

Lists: Lists (1), as specified in [MS-WSSTS] section 2.1.2.7, and the list items, as specified in [MS-WSSTS] section 2.1.2.10, contained within them.

Data Macro: A set of built-in activities that act upon list items, as specified in the **CT_DataMacro** element specified in [MS-AXL] section 2.2.3.49. A protocol client can trigger a data macro indirectly through **UpdateLists** (section 3.1.4.8) or explicitly by calling **RunDataMacro** (section 3.1.4.5). The status of a data macro instance is obtained by calling **GetMacroState** (section 3.1.4.3).

3.1.1.1 Access Services Site Template

The protocol server MUST have an **available site template** named "ACCSRV#0". This means that the protocol server MUST list "ACCSRV#0" as a template name in a response to **GetSiteTemplate**, as specified in [MS-SITESS] section 3.1.4.5, and accept "ACCSRV#0" in the **templateName** parameter to **CreateWeb**, as specified in [MS-SITESS] section 3.1.4.9.

A site (2) created with the **ACCSRV#0** template is used for representing a database application.

All sites (2) created with **the ACCSRV#0** template MUST contain at least two lists (1), one with the title "MSysASO" and one with the title "USysApplicationLog", each with schema specified as follows.

3.1.1.1.1 MSysASO

The **MSysASO** list (1) is used to specify objects and settings of a database application. Each list item in the list (1) specifies either a single object or a group of settings, depending on the **Type field (2)**.

Each list item in **MSysASO** MUST have a unique combination of values in their **Title** and **Type** fields (2).

The **MSysASO** list MUST have at least the following fields (2), with the following **FieldDefinitions**, as specified in [MS-WSSFO2] section 2.2.8.3.3.

3.1.1.1.1.1 ID

Specifies a unique identifying value for each list item.

FieldDefinition property	Value
Name	ID
DisplayName	ID
Type	Counter
PrimaryKey	TRUE

3.1.1.1.1.2 Title

Specifies the name of the object in the database application that is represented by this list item.

FieldDefinition property	Value
Name	Title
DisplayName	Title
Type	Text

3.1.1.1.1.3 owshiddenversion

Specifies a value used for conflict detection, as specified in [MS-LISTSWS] section 3.1.4.31.2.1, and by the **UpdateLists** command specified in section 3.1.4.8.

FieldDefinition property	Value
Name	owshiddenversion
DisplayName	owshiddenversion
Type	Integer

3.1.1.1.1.4 Type

Specifies the type of object in the database application represented by this list item.

FieldDefinition property	Value
Name	Type
DisplayName	Type
Type	Integer

The following table specifies the valid values of the **Type** field (2) and their meanings.

Value	Meaning
0	A list (1) and its associated data macros, as specified in [MS-AXL] section 2.1.3.2.
1	Query, as specified in [MS-AXL] section 2.1.4.
2	Form, as specified in [MS-AXL] section 2.1.2.
3	Report, as specified in [MS-AXL] section 2.1.5.
4	User interface macro, as specified in [MS-AXL] section 2.1.3.1.
5	Client data that MUST be preserved by the protocol server.
6	Client data that MUST be preserved by the protocol server.
7	Client data that MUST be preserved by the protocol server.
8	Client data that MUST be preserved by the protocol server.
9	Client data that MUST be preserved by the protocol server.
10	Client data that MUST be preserved by the protocol server.
11	Client data that MUST be preserved by the protocol server.
12	Shared image, as specified in [MS-AXL] section 2.1.6.
13	Client data that MUST be preserved by the protocol server.
14	Client data that MUST be preserved by the protocol server.
15	This record contains server private data. Protocol clients MUST NOT change the values in this record.
16	Client data that MUST be preserved by the protocol server.
17	Client data that MUST be preserved by the protocol server.
18	Client data that MUST be preserved by the protocol server.

When the **Type** field (2) of this list item is zero, there MUST be a list (1) in this site (2) where the title of the list (1) matches the value of this list item's **Title** field (2). That list (1) specifies the data for the list (1) object represented by this list item.

When the **Type** field (2) is one of the values in the following table, the **Title** field (2) MUST be the value specified in the table.

Type	Title
9	Navigation Pane
10	VBA References
11	DBProps

3.1.1.1.1.5 Revision

Specifies the number of updates that have been made to this list item that have changed any of the **ClientObject**, **ServerObject**, **ClientObjectProperties**, or **Attachments** fields (2).

FieldDefinition property	Value
Name	Revision
DisplayName	Revision
Type	Integer

3.1.1.1.1.6 ClientObject

Specifies a client-specific object of the database application. The contents of this field (2) are created by the client and MUST be ignored by the server.

FieldDefinition property	Value
Name	ClientObject
DisplayName	ClientObject
Type	Note
RichText	FALSE
Required	FALSE

The contents of the **ClientObject** field (2) depend on the value in the **Type** field (2) of this list item, as specified in the following table.

Type field value	ClientObject field contents
1	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Query".
2	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part ,as specified in [MS-ACCDT] section 2.1.4.14, where

Type field value	ClientObject field contents
	the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Form".
3	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Report".
4	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Macro".
5	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Module".
6	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "Link".
7	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from an **Object** part, as specified in [MS-ACCDT] section 2.1.4.14, where the value of **AccessObject/Type** in the **Object Metadata** part, as specified in [MS-ACCDT] section 2.1.4.15, was "SQLLink".
8	The contents MUST have been specified by the client by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8).
9	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from a **Navigation Pane** part, as specified in [MS-ACCDT] section 2.1.4.13.
10	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8), or obtained from a **Visual Basic References** part, as specified in [MS-ACCDT] section 2.1.4.26.
16	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8).
17	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8).
18	The contents MUST have either been specified by the client directly by a call to **UpdateListItems**, as specified in [MS-LISTSWS] section 3.1.4.31, or **UpdateLists** (section 3.1.4.8).

The **ClientObject** field (2) MUST be empty for all other values of the Type field (2).

3.1.1.1.1.7 ServerObject

Specifies an object of the database application using [MS-AXL]. The target **XML namespaces** used in all the Xml elements referenced in this section are dependent on the Access Services Site Version (section 3.1.1.2) for the current site (2).

FieldDefinition property	Value
Name	ServerObject
DisplayName	ServerObject
Type	Note
RichText	FALSE
Required	FALSE

The contents of the **ServerObject** field (2) depend on the value in the **Type** field (2) of this list item, as specified in the following table.

Type field value	ServerObject field contents
0	When the list (1) represented by this list item has data macros, as specified in [MS-AXL] section 2.1.3.2, the **ServerObject** field (2) contains a **DataMacros** element, as specified in [MS-AXL] section 2.2.1.5, specifying those **data macros** belonging to this list (1). When the list (1) has no **data macros**, the **ServerObject** field (2) MUST be empty.
1	MUST be XML beginning with a **Query** element, as specified in [MS-AXL] section 2.2.1.2, or nothing.
2	MUST be XML beginning with a **View** element, as specified in [MS-AXL] section 2.2.1.7, or nothing.
3	MUST be XML beginning with a **Report** element, as specified in [MS-AXL] section 2.4.1.1, or nothing.
4	MUST be XML beginning with a **UserInterfaceMacro** element, as specified in [MS-AXL] section 2.2.1.6, or nothing.
11	MUST be XML beginning with an **Application** element, as specified in [MS-AXL] section 2.2.1.1, or nothing.
15	The protocol server can use this field to hold any type of string data. Protocol clients MUST NOT change the values in this record.

The **ServerObject** field (2) MUST be empty for all other values of the **Type** field (2).

3.1.1.1.1.8 ClientObjectProperties

Specifies client-specific properties of an object in the database application. The contents of this field (2) are created by the client and MUST be ignored by the server.

FieldDefinition property	Value
Name	ClientObjectProperties
DisplayName	ClientObjectProperties
Type	Note
RichText	FALSE
Required	FALSE

The **ClientObjectProperties** field (2) MUST be empty when the value of the **Type** field (2) in this list item is not an integer between zero and seven inclusive.

When the field (2) is not empty, its contents MUST have either been specified by the protocol client directly by a call to **UpdateListItems** ([MS-LISTSWS] section 3.1.4.31) or **UpdateLists** (section 3.1.4.8), or obtained from an **Object Properties** part, as specified in [MS-ACCDT] section 2.1.4.16.

3.1.1.1.1.9 Flags

Specifies whether the object in the database application represented by this list item is a reserved server-specific object. A value of "1" specifies that this object is a reserved server-specific object, which a protocol client MUST not make changes to. A value of zero ("0") or an empty field (2) specifies that this is not a reserved server-specific object.

FieldDefinition property	Value
Name	Flags
DisplayName	Flags
Type	Integer
Required	FALSE

3.1.1.1.1.10 Attachments

Specifies data for images and themes used by the database application.

FieldDefinition property	Value
Name	Attachments
Type	Attachments

The contents of the **Attachments** field (2) depend on the value in the **Type** field (2) of this list item. If the **Type** field (2)'s value is specified in the following table, there MUST be one **attachment** in the **Attachments** field (2). The contents of the attachment are specified in the following table.

Type field value	Attachment
12	A file of type PNG, GIF, or JPG. (The file's name without the **file extension** SHOULD match the **Title** field (2) of this list item, but MUST be ignored by the server.) It MUST have been

Type field value	Attachment
	uploaded by the client directly by a call to **AddAttachment**, as specified in [MS-LISTSWS] section 3.1.4.1, or had its contents obtained from an **Image** part as specified in [MS-ACCDT] section 2.1.4.6.
13	A file named "Office Theme.thmx". The contents of this file are created by the client and MUST be ignored by the server. It MUST have been uploaded by the client directly by a call to **AddAttachment**, as specified in [MS-LISTSWS] section 3.1.4.1, or had its contents obtained from a **Theme** part ,as specified in [MS-ACCDT] section 2.1.4.24.
14	A file of type PNG. (The file's name without the file extension SHOULD match the **Title** field (2) of this list item, but MUST be ignored by the server.) It MUST have been uploaded by the client directly by a call to **AddAttachment**, as specified in [MS-LISTSWS] section 3.1.4.1, or had its contents obtained from an **ImageCluster** part, as specified in [MS-ACCDT] section 2.1.4.7.

The **Attachments** field (2) MUST contain zero attachments for all other values of the **Type** field (2).

3.1.1.1.2 USysApplicationLog

The **USysApplicationLog** list (1) is used to specify a log of events generated by a database application on the protocol server. Each list item in USysApplicationLog contains represents a single event.

The **USysApplicationLog** list (1) MUST have at least the following fields (2), with the following **FieldDefinitions** (as specified in [MS-WSSFO2] section 2.2.8.3.3).

3.1.1.1.2.1 ID

Specifies a unique identifying value for each list item.

FieldDefinition property	Value
Name	ID
DisplayName	ID
Type	Counter
PrimaryKey	TRUE

3.1.1.1.2.2 Created

Specifies the **UTC** time in which this list item was added to the list (1).

FieldDefinition property	Value
Name	Created
DisplayName	Created
Type	DateTime
StorageTZ	TRUE

3.1.1.1.2.3 owshiddenversion

Specifies a value used for conflict detection, as specified in [MS-LISTSWS] section 3.1.4.31.2.1.

FieldDefinition property	Value
Name	owshiddenversion
DisplayName	owshiddenversion
Type	Integer

3.1.1.1.2.4 Category

Specifies the type of event represented by this list item.

FieldDefinition property	Value
Name	Category
DisplayName	Category
Type	Choice

3.1.1.2 Access Services Site Version

The **Access Services Site Version** is used to determine which target XML namespaces are used when referencing application object schemas from [MS-AXL]. This version number MUST consist of a major version and a minor version that can be expressed either as XML as specified in section 2.2.4.3, or as a string as specified in the following **ABNF**, as specified in [RFC5234]:

```
access-services-site-version = major-version dot minor-version
major-version = 1*3DIGIT

dot = "."

minor-version = 1*4DIGIT
```

If the Access Services Site Version is 1.2 then the Xml referenced in [MS-AXL] MUST use the following target XML namespaces:

MS-AXL schema	Required target namespace
AXL ([MS-AXL] section 2.2)	http://schemas.microsoft.com/office/accessservices/2009/11/application
RDL ([MS-AXL] section 2.4)	http://schemas.microsoft.com/office/accessservices/2009/11/reports
Form XAML ([MS-AXL] section 2.3.3)	http://schemas.microsoft.com/office/accessservices/2009/11/forms
WPF ([MS-AXL] section 2.3.2)	http://schemas.microsoft.com/client/2009/11
XAML 2006 ([MS-AXL] section 2.3.1)	http://schemas.microsoft.com/winfx/2009/04/xaml

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

This specification includes the following **WSDL operations**:

WSDL Operation	Description
GetAccessServicesVersion	The **GetAccessServicesVersion** command is used to get the **Access Services Site Version** (section 3.1.1.2) of the current site (2) or subsite.
GetCurrentUserInfo	This operation returns XML that describes the user profile being used when sending requests to the protocol server.
GetDataMacroState	The **GetDataMacroState** command is used to get the current state of a data macro ([MS-AXL] section 2.1.3.2).
GetServerInformation	The **GetServerInformation** command is used to get information about which **Access Services Site Versions** (section 3.1.1.2) the protocol server supports and the **Access Services Site Version** of the current site (2) or subsite.
RunDataMacro	This operation triggers the running of a data macro ([MS-AXL] section 2.2.1.5).
SetAccessServicesVersion	The **SetAccessServicesVersion** command is used to set the **Access Services Site Version** (section 3.1.1.2) of the current site (2) or subsite.
StartCompilation	The **StartCompilation** command is used to notify a protocol server to start compiling the database application defined in the MSysASO list (section 3.1.1.1.1) of the current site (2) or subsite from the object definitions stored within that list (1).
UpdateLists	This operation is used to insert, update, and delete list items in one or more lists (1).

3.1.4.1 GetAccessServicesVersion

The **GetAccessServicesVersion** command is used to get the **Access Services Site Version** (section 3.1.1.2) of the current site (2) or subsite.

```
<wsdl:operation name="GetAccessServicesVersion">
  <wsdl:input message="tns:GetAccessServicesVersionSoapIn"/>
  <wsdl:output message="tns:GetAccessServicesVersionSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **GetAccessServicesVersionSoapIn** request message and the protocol server sends a **GetAccessServicesVersionSoapOut** response message as follows:

- In the event of an application error on the protocol server during this operation, a SOAP fault is returned, as described in section 2.2.2.1.

- If the **GetAccessServicesVersionSoapIn** request message is sent to an **Access Services** site (2), the protocol server MUST return the **Access Services Site Version** of that site (2). If the request was sent to a site (2) that is not an Access Services site (2), the protocol server MUST return a major version of "-1" and a minor version of zero ("0").

3.1.4.1.1 Messages

3.1.4.1.1.1 GetAccessServicesVersionSoapIn

The requested WSDL message for the **GetAccessServicesVersion** WSDL operation.

The **SOAP action** value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetAccessServicesV
ersion
```

The **SOAP body** contains the **GetAccessServicesVersion** element.

3.1.4.1.1.2 GetAccessServicesVersionSoapOut

The response WSDL message for the **GetAccessServicesVersion** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetAccessServicesV
ersion
```

The SOAP body contains the **GetAccessServicesVersionResponse** element.

3.1.4.1.2 Elements

3.1.4.1.2.1 GetAccessServicesVersion

The input data for the **GetAccessServicesVersion** WSDL operation.

```
<xs:element name="GetAccessServicesVersion">
  <xs:complexType/>
</xs:element>
```

3.1.4.1.2.2 GetAccessServicesVersionResponse

The result data for the **GetAccessServicesVersion** WSDL operation.

```
<xs:element name="GetAccessServicesVersionResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="Version" nillable="true"
type="tns:VersionType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Version: The **Access Services Site Version** of the site (2) or subsite as specified in section 2.2.4.3.

3.1.4.2 GetCurrentUserInfo

This operation returns XML that describes the user profile being used when sending requests to the protocol server.

```
<wsdl:operation name="GetCurrentUserInfo">
  <wsdl:input message="tns:GetCurrentUserInfoSoapIn"/>
  <wsdl:output message="tns:GetCurrentUserInfoSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **GetCurrentUserInfoSoapIn** request message and the protocol server responds with a **GetCurrentUserInfoSoapOut** response message as follows:

- In the event of an application error on the protocol server during this operation, a SOAP fault is returned as described in section 2.2.2.1.
- Otherwise, the protocol server MUST respond with a **GetCurrentUserInfoSoapOut** response message that contains information about the user profile.

3.1.4.2.1 Messages

3.1.4.2.1.1 GetCurrentUserInfoSoapIn

The requested WSDL message for the **GetCurrentUserInfo** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetCurrentUserInfo
```

The SOAP body contains the **GetCurrentUserInfo** element.

3.1.4.2.1.2 GetCurrentUserInfoSoapOut

The response WSDL message for the **GetCurrentUserInfo** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetCurrentUserInfo
```

The SOAP body contains the **GetCurrentUserInfoResponse** element.

3.1.4.2.2 Elements

3.1.4.2.2.1 GetCurrentUserInfo

The input data for the **GetCurrentUserInfo** WSDL operation.

```
<xs:element name="GetCurrentUserInfo">
  <xs:complexType/>
```

```
</xs:element>
```

3.1.4.2.2.2 GetCurrentUserInfoResponse

The result data for the **GetCurrentUserInfo** WSDL operation.

```
<xs:element name="GetCurrentUserInfoResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="GetCurrentUserInfoResult">
        <xs:complexType mixed="true">
          <xs:sequence>
            <xs:any/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

GetCurrentUserInfoResult: This element MUST be present and MUST have one child element named **GetCurrentUserInfo**. The **GetCurrentUserInfo** element MUST be present and MUST have two child elements. The first MUST be an element named **User** of type **User**, as specified in [MS-UGS] section 2.2.4.3. The second element MUST be named **Groups** and be of type **Groups**, as specified in [MS-UGS] section 2.2.4.2.

3.1.4.3 GetDataMacroState

The **GetDataMacroState** command is used to get the current state of a data macro ([MS-AXL] section 2.1.3.2).

```
<wsdl:operation name="GetDataMacroState">
  <wsdl:input message="tns:GetDataMacroStateSoapIn"/>
  <wsdl:output message="tns:GetDataMacroStateSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **GetDataMacroStateSoapIn** request message and the protocol server sends a **GetDataMacroStateSoapOut** response message, as follows:

- If the specified **macroToken** element is a valid string that represents a data macro that was triggered on the protocol server, the protocol server MUST respond with the current state of the data macro.
- If the specified **macroToken** is not a valid string, or does not represent a data macro on the protocol server, a SOAP fault is returned as described in section 2.2.2.1.

3.1.4.3.1 Messages

3.1.4.3.1.1 GetDataMacroStateSoapIn

The requested WSDL message for the **GetDataMacroState** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetDataMacroState
```

The SOAP body contains the **GetDataMacroState** element.

3.1.4.3.1.2 GetDataMacroStateSoapOut

The response WSDL message for the **GetDataMacroState** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetDataMacroState
```

The SOAP body contains the **GetDataMacroStateResponse** element.

3.1.4.3.2 Elements

3.1.4.3.2.1 GetDataMacroState

The input data for the **GetDataMacroState** WSDL operation.

```
<xs:element name="GetDataMacroState">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="macroToken" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

macroToken: The identifier of a data macro instance. This element MUST be present, and MUST have a value obtained from one of the following:

- The **RunDataMacroResult** element of a **RunDataMacroResponse** element returned by the **RunDataMacro** operation as specified in section 3.1.4.5.2.2.
- The **mit** attribute of an **Update** element in an **UpdateListsReponse** element returned by the **UpdateLists** operation as specified in section 3.1.4.8.2.2.

3.1.4.3.2.2 GetDataMacroStateResponse

The result data for the **GetDataMacroState** WSDL operation.

```
<xs:element name="GetDataMacroStateResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="GetDataMacroStateResult"
type="tns:DataMacroInstanceState"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

GetDataMacroStateResult: The current state of the data macro instance as specified in section 3.1.4.3.3.1. MUST be specified.

3.1.4.3.3 Complex Types

3.1.4.3.3.1 DataMacroInstanceState

The state of a data macro instance on the server, including any errors and return variables, as specified in [MS-AXL] section 2.1.3.3.3.

```
<xs:complexType name="DataMacroInstanceState">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="State" type="tns:DataMacroState"/>
    <xs:element minOccurs="1" maxOccurs="1" name="ErrorNumber" type="xs:int"/>
    <xs:element minOccurs="0" maxOccurs="1" name="ErrorDescription" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="1" name="ReturnVars"
type="tns:ArrayOfKeyValuePair"/>
  </xs:sequence>
</xs:complexType>
```

State: The current state of the data macro instance as specified in section 3.1.4.3.4.1.

ErrorNumber: The error number returned by the data macro. See [MS-AXL] section 2.2.5.1.14.

ErrorDescription: The error description returned by the data macro. See [MS-AXL] section 2.2.5.1.14.

ReturnVars: Specifies the return variables, as specified in [MS-AXL] section 2.1.3.3.3, of the data macro. This MUST be an **ArrayOfKeyValuePair**, as specified in section 2.2.4.1, that MUST contain a **KeyValuePair** for each return variable returned from the data macro, and the **Key** of the **KeyValuePair** MUST be the same as the name of the return variable it is representing.

3.1.4.3.4 Simple Types

3.1.4.3.4.1 DataMacroState

The current state of a data macro instance.

```
<xs:simpleType name="DataMacroState">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Error"/>
    <xs:enumeration value="Running"/>
    <xs:enumeration value="Complete"/>
  </xs:restriction>
</xs:simpleType>
```

The following table specifies the allowable values for DataMacroState:

Value	Meaning
Error	The current data macro has stopped running because of an error.
Running	The current data macro is still running.
Complete	The current data macro has successfully finished running with no errors.

3.1.4.4 GetServerInformation

The **GetServerInformation** command is used to get information about which **Access Services Site Versions** (section 3.1.1.2) the protocol server supports and the **Access Services Site Version** of the current site (2) or subsite.

```
<wsdl:operation name="GetServerInformation">
  <wsdl:input message="tns:GetServerInformationSoapIn"/>
  <wsdl:output message="tns:GetServerInformationSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **GetServerInformationSoapIn** request message, and the protocol server sends a **GetServerInformationSoapOut** response message, as follows:

- In the event of an application error on the protocol server during this operation, a SOAP fault is returned, as described in section 2.2.2.1.
- Otherwise, the protocol server MUST respond with a **GetServerInformationSoapOut** response message that contains information about the **Access Services Site Version** (section 3.1.1.2) supported on the protocol server. If the **GetServerInformationSoapIn** request message is sent to an **Access Services** site (2) or subsite, the protocol server MUST also return the **Access Services Site Version** of that site (2) or subsite. If the request was sent to a **top-level site** of a **site collection**, or a site (2) that is not an **Access Services** site (2), the protocol server MUST return a major version of "-1" and a minor version of zero ("0").

3.1.4.4.1 Messages

3.1.4.4.1.1 GetServerInformationSoapIn

The requested WSDL message for the **GetServerInformation** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetServerInformation
```

The SOAP body contains the **GetServerInformation** element.

3.1.4.4.1.2 GetServerInformationSoapOut

The response WSDL message for the **GetServerInformation** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetServerInformation
```

The SOAP body contains the **GetServerInformationResponse** element.

3.1.4.4.2 Elements

3.1.4.4.2.1 GetServerInformation

The input data for the **GetServerInformation** WSDL operation.

```
<xs:element name="GetServerInformation">
  <xs:complexType/>
</xs:element>
```

3.1.4.4.2.2 GetServerInformationResponse

The result data for the **GetServerInformation** WSDL operation.

```
<xs:element name="GetServerInformationResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="AccessServerInformation" nillable="true"
type="tns:AccessServerInformationType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

AccessServerInformation: Information about the minimum and maximum **Access Services Site Version**, as specified in section 3.1.1.2, supported by the protocol server, and the current **Access Services Site Version**, as specified in section 3.1.4.4.3.1.

3.1.4.4.3 Complex Types

3.1.4.4.3.1 AccessServerInformationType

The **AccessServerInformationType** complex type contains information about the minimum and maximum **Access Services Site Version** (section 3.1.1.2) the protocol server supports, and the **Access Services Site Version** of the current site (2) or subsite.

```
<xs:complexType name="AccessServerInformationType">
  <xs:sequence>
    <xs:element minOccurs="0" maxOccurs="1" name="MinimumAccessServicesVersion"
type="tns:VersionType"/>
    <xs:element minOccurs="0" maxOccurs="1" name="MaximumAccessServicesVersion"
type="tns:VersionType"/>
    <xs:element minOccurs="0" maxOccurs="1" name="SiteVersion" type="tns:VersionType"/>
  </xs:sequence>
</xs:complexType>
```

MinimumAccessServicesVersion: The minimum **Access Services Site Version** that is supported on the protocol server. The format of this element MUST be as specified in section 2.2.4.3.

MaximumAccessServicesVersion: The maximum **Access Services Site Version** that is supported on the protocol server. The format of this element MUST be as specified in section 2.2.4.3.

SiteVersion: The current **Access Services Site Version** of the site (2) or subsite, as specified in section 2.2.4.3. If the request was sent to a top-level site (2) of a site collection, this MUST have a major version of "-1" and a minor version of zero ("0").

3.1.4.5 RunDataMacro

This operation triggers the running of a data macro ([MS-AXL] section 2.2.1.5).

```
<wsdl:operation name="RunDataMacro">
  <wsdl:input message="tns:RunDataMacroSoapIn"/>
  <wsdl:output message="tns:RunDataMacroSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **RunDataMacroSoapIn** request message and the protocol server responds with a **RunDataMacroSoapOut** response message as follows:

- In the event of an application error on the **protocol server** during this operation, a SOAP fault is returned as described in section 2.2.2.1.
- If the macro name specified by the protocol client does not match the name of a data macro on the protocol server, a SOAP fault is returned as described in section 2.2.2.1.
- Otherwise, the protocol server MUST respond with a **RunDataMacroSoapOut** response message that contains an identifier for the data macro instance the protocol server started.

3.1.4.5.1 Messages

3.1.4.5.1.1 RunDataMacroSoapIn

The requested WSDL message for the **RunDataMacro** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/RunDataMacro
```

The SOAP body contains the **RunDataMacro** element.

3.1.4.5.1.2 RunDataMacroSoapOut

The response WSDL message for the **RunDataMacro** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/RunDataMacro
```

The SOAP body contains the **RunDataMacroResponse** element.

3.1.4.5.2 Elements

3.1.4.5.2.1 RunDataMacro

The input data for the **RunDataMacro** WSDL operation.

```
<xs:element name="RunDataMacro">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="macroName" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="1" name="parameters"
type="tns:ArrayOfKeyValuePair"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

macroName: The name of the data macro to be run. MUST be specified. The value of **macroName** MUST be the title of a list (1) within the current site (2), followed by a period, followed by the name of the data macro to run. The title of the list (1) and the name of the data macro MUST be of type **ST_ObjectName**, as specified in [MS-AXL] section 2.2.4.1. If the title of the list (1) contains a space, it MUST be surrounded by square brackets. If the name of the data macro contains a space, it MUST be surrounded by square brackets.

parameters: Specifies the parameters for the data macro. MUST be present if the data macro requires parameters, as specified in the **Parameters** element of a **CT_DataMacro**, as specified in ([MS-AXL] section 2.2.3.49. Comparisons between the **Key** in a **KeyValuePair** and the parameters in a data macro MUST be case insensitive. If a parameter is specified in more than one **KeyValuePair**, the protocol server MUST use the last **Value**. If any **Key** in **KeyValuePair** does not match a parameter in the data macro, it MUST be ignored.

3.1.4.5.2.2 RunDataMacroResponse

The result data for the **RunDataMacro** WSDL operation.

```
<xs:element name="RunDataMacroResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="RunDataMacroResult" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

RunDataMacroResult: A new identifier for the data macro instance that was triggered. MUST be specified. This identifier can be supplied to **GetDataMacroState** (section 3.1.4.3) to obtain state information about the data macro instance.

3.1.4.6 SetAccessServicesVersion

The **SetAccessServicesVersion** command is used to set the **Access Services Site Version** (section 3.1.1.2) of the current site (2) or subsite.

```
<wsdl:operation name="SetAccessServicesVersion">
  <wsdl:input message="tns:SetAccessServicesVersionSoapIn"/>
  <wsdl:output message="tns:SetAccessServicesVersionSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **SetAccessServicesVersionSoapIn** request message and the protocol server sends a **SetAccessServicesVersionSoapOut** response message as follows:

- In the event of an application error on the protocol server during this operation, a SOAP fault is returned as described in section 2.2.2.1.
- If the version specified by **Version** is not supported by the protocol server, or is an invalid **VersionType**, a SOAP fault is returned as described in section 2.2.2.1.
- Otherwise, if the protocol server is a site, set the **Access Services Site Version** (section 3.1.1.2) to the values specified by the **Version** element of the request. The protocol server MUST respond with a **SetAccessServicesVersionSoapOut** response message.

3.1.4.6.1 Messages

3.1.4.6.1.1 SetAccessServicesVersionSoapIn

The requested WSDL message for the **SetAccessServicesVersion** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/SetAccessServicesV
ersion
```

The SOAP body contains the **SetAccessServicesVersion** element.

3.1.4.6.1.2 SetAccessServicesVersionSoapOut

The response WSDL message for the **SetAccessServicesVersion** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/SetAccessServicesV
ersion
```

The SOAP body contains the **SetAccessServicesVersionResponse** element.

3.1.4.6.2 Elements

3.1.4.6.2.1 SetAccessServicesVersion

The input data for the **SetAccessServicesVersion** WSDL operation.

```
<xs:element name="SetAccessServicesVersion">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="1" name="Version" nillable="true"
type="tns:VersionType"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Version: The version of the site (2) or subsite as specified in section 2.2.4.3.

3.1.4.6.2.2 SetAccessServicesVersionResponse

The result data for the **SetAccessServicesVersion** WSDL operation.

```
<xs:element name="SetAccessServicesVersionResponse">
  <xs:complexType/>
</xs:element>
```

3.1.4.7 StartCompilation

The **StartCompilation** command is used to notify a protocol server to start compiling the database application defined in the MSysASO list (section 3.1.1.1.1) of the current site (2) or subsite from the object definitions stored within that list (1).

```
<wsdl:operation name="StartCompilation">
  <wsdl:input message="tns:StartCompilationSoapIn"/>
  <wsdl:output message="tns:StartCompilationSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **StartCompilationSoapIn** request message and the protocol server responds with a **StartCompilationSoapOut** response message as follows:

- In the event of an application error on the protocol server during this operation, a SOAP fault is returned as described in section 2.2.2.1.
- Otherwise, the protocol server MUST respond with a **StartCompilationSoapOut** response message.

3.1.4.7.1 Messages

3.1.4.7.1.1 StartCompilationSoapIn

The requested WSDL message for the **StartCompilation** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/StartCompilation
```

The SOAP body contains the **StartCompilation** element.

3.1.4.7.1.2 StartCompilationSoapOut

The response WSDL message for the **StartCompilation** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/StartCompilation
```

The SOAP body contains the **StartCompilationResponse** element.

3.1.4.7.2 Elements

3.1.4.7.2.1 StartCompilation

The input data for the **StartCompilation** WSDL operation.

```
<xs:element name="StartCompilation">
  <xs:complexType/>
</xs:element>
```

3.1.4.7.2.2 StartCompilationResponse

The result data for the **StartCompilation** WSDL operation.

```
<xs:element name="StartCompilationResponse">
  <xs:complexType/>
</xs:element>
```

3.1.4.8 UpdateLists

This operation is used to insert, update, and delete list items in one or more lists (1).

```
<wsdl:operation name="UpdateLists">
  <wsdl:input message="tns:UpdateListsSoapIn"/>
  <wsdl:output message="tns:UpdateListsSoapOut"/>
</wsdl:operation>
```

The protocol client sends an **UpdateListsSoapIn** request message and the protocol server sends an **UpdateListsSoapOut** response message that consists of zero or more **u** elements. The protocol server MUST process each **u** element as follows:

1. If the list (1) name specified in the **ln** attribute of the **u** element is a valid **GUID** and corresponds to the identifier of a list (1) on the site (2), use that list (1) for the operation.

2. If the specified list (1) name is not a valid GUID or does not correspond to the identifier of a list (1) on the site (2), check if the list name corresponds to the title of a list (1) on the site (2). If it does, use that list (1) if performing either an **Insert** or **Update** command. If the list (1) name corresponds to the title of a list (1) on the site (2) and performing a **Delete** command, return an error code of "-2130575322" in the **ec** attribute of the **u** element in the response that corresponds to this part of the request.

3. If the specified list (1) name does not match a list (1) based on either of the two previous conditions, the protocol server MUST return a SOAP fault, as described in section 2.2.2.1.

4. If there is a macro token specified in the **mit** element of the request, the protocol server MUST wait for the data macro that corresponds to that macro token to finish running before processing the **u** elements, or return an error code of "-2147467259" in the **ec** attribute of the **u** elements that failed to be updated.

5. Otherwise, the protocol server MUST process the operations on the list (1) and return success or failure in the **ec** attribute of the **u** element in the response that corresponds to this part of the request.

3.1.4.8.1 Messages

3.1.4.8.1.1 UpdateListsSoapIn

The requested WSDL message for the **UpdateLists** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/UpdateLists
```

The SOAP body contains the **UpdateLists** element.

3.1.4.8.1.2 UpdateListsSoapOut

The response WSDL message for the **UpdateLists** method.

The SOAP action value is:

```
http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/UpdateLists
```

The SOAP body contains the **UpdateListsResponse** element.

3.1.4.8.2 Elements

3.1.4.8.2.1 UpdateLists

The input data for the **UpdateLists** WSDL operation.

```
<xs:element name="UpdateLists">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" name="u" type="tns:Update"/>
      <xs:element minOccurs="1" maxOccurs="1" name="par" type="xs:boolean"/>
      <xs:element minOccurs="1" maxOccurs="1" name="mit" nillable="true" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

u: Specifies an operation to perform on a list item. MUST be specified.

The following table lists the three different operations that can be performed on a list item. For more information about constructing a **u** element, see section 3.1.4.8.3.1

Value	Description
Insert (i)	Add a new list item to the specified list (1). For each **f** element within the **u** element, the protocol server MUST set the value of the field (2) specified by the **n** attribute of that **f** element to the value stored in the **v** attribute of that element or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update. The protocol server MUST ignore the value in the **id** attribute of the **u** element in the client's request, and MUST return the assigned list item identifier of this inserted list item in the **id** attribute of the **update** element in the protocol server's response to this update.

Value	Description
Update (u)	Update fields (2) for a specific list item. The **id** attribute of the **u** element MUST be specified with a value that equals the list item identifier of a list item that already exists on the protocol server. For each **f** element within the **u** element, the protocol server MUST set the value of the field (2) specified by the **n** attribute of that **f** element to the value stored in the **v** attribute of that element, or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update.
Delete (d)	Delete a specified list item. The **id** attribute of the **u** element MUST be specified with a value that equals the list item identifier of a list item that MUST be deleted by the protocol server, or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update.

There are two types of fields (2) specified by the **n** attribute of an **f** element of the **u** element that require special treatment:

- owshiddenversion field (2)
- **lookup fields**

The **owshiddenversion** field (2), as specified in [MS-WSSTS] section 2.4.2, is used for conflict detection as follows:

- Insert operations MUST NOT have a value specified for the **owshiddenversion** field (2) by the protocol client.
- Update operations use the **owshiddenversion** field (2) as follows:
  - When the **owshiddenversion** field (2) is not specified by the protocol client, the protocol server MUST overwrite any changes in the list item or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update indicating why the update failed.
  - When the **owshiddenversion** value specified by the protocol client is equal to the **owshiddenversion** field's (2) value for the list item on the protocol server, the protocol server MUST update the list item or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update indicating why the update failed. The protocol server MUST NOT update this field (2) with the value that the protocol client includes in the request.
  - When the **owshiddenversion** specified by the protocol client is different from the current value of the **owshiddenversion** field (2) for the list item on the protocol server, the protocol server MUST return error code "-2130575305" and return all of the values of the fields (2) currently stored on the protocol server using the **f** elements of the **u** element in the response that corresponds to the current update.
- Delete operations MUST have a value specified for the **owshiddenversion** field (2) by the protocol client. Delete operations use the **owshiddenversion** field (2) as follows:
  - When the **owshiddenversion** value specified by the protocol client is equal to the **owshiddenversion** field's (2) value for the list item on the protocol server, the protocol server MUST delete the list item or return an error code in the **ec** attribute of the **u** element in the protocol server's response to this update indicating why the delete failed.
  - When the **owshiddenversion** specified by the protocol client is different from the current value of the **owshiddenversion** field (2) for the list item on the protocol server, the protocol server MUST return error code "-2130575339" and return all of the values of the fields (2) currently

stored on the protocol server using the **f** elements of the **u** element in the response that corresponds to the current update.

If the field (2) is a lookup field, as specified in [MS-WSSTS] section 2.3.1, the value MUST match a list item identifier in the target list (1) of the lookup, or the protocol server MUST perform identifier fixup on the list item as specified in the following paragraph. The value of a lookup field might refer to a list item that is being inserted in the same **UpdateListsSoapIn** request.

If a lookup field is referenced in an **f** element of a **u** element with **cmd** attribute value "i" in a request from the protocol client, and the value of that lookup field, as specified by the **v** attribute of the **f** element, is set to a value that does not correspond to a list item identifier in the target list (1) of the lookup, the list item that is the target of the lookup field has not yet been added to the protocol server. In this case, the value of the lookup field MUST reference the **id** of a **u** element in the current **UpdateLists** request, or the current **UpdateLists** request MUST have the **par** element set to "true". If the value of the lookup field references the **id** of a **u** element with **cmd** attribute value "i" in the current **UpdateLists** request, the protocol server MUST process the other updates in the **UpdateLists** request before attempting to process this update. After processing the other updates and assigning list item identifiers to any list items inserted by the **UpdateLists** request, the protocol server MUST replace the value of the lookup field with the actual list item identifier that the protocol server assigned to the inserted list item in the current request that had the **id** that matched the value of the lookup field, and expressed as specified in [MS-WSSTS] section 2.4.1. If the target of the lookup is not included in a previous **u** element from the current **u** element with **cmd** attribute value "i" in the same protocol client request, the protocol server MUST return an error in the **ec** attribute of the **u** element in the protocol server's response to this update, and MUST not commit any data for the current **u** element unless the **par** element of this **UpdateLists** request is set to "true", as specified in the following paragraph.

par: The value of this parameter determines whether insert operations in this update are treated as regular inserts or partial inserts. If **par** is set to "true", all **u** elements with a **cmd** attribute of value "i" are treated as partial inserts. Partial inserts can be used when the target of a lookup in a new list item will be inserted in a future request. In a partial insert, the server MUST commit any valid data in the **u** element, and MUST attempt to fix values in lookup fields, as specified in the preceding paragraph. If the protocol server cannot set a lookup field's value, the protocol server behavior is as follows:

- If the lookup field is not required to have a value, the protocol server MUST return –"2130575159" in the **ec** attribute of the **u** element in the protocol server's response to this update and MUST commit any valid data in the **u** element. A value MUST NOT be committed for the lookup field of this list item.
- If the lookup field is required to have a value, the protocol server MUST return "-2130575163" in the **ec** attribute of the **u** element in the protocol server's response to this update and MUST NOT commit any data from the **u** element.

If **par** is set to "false", all **u** elements with a **cmd** attribute value "i" will be treated as regular inserts. If all data in the **u** element is valid after attempting to fix values in the lookup fields, as previously specified, the protocol server MUST commit those values in a new list item. If the protocol server cannot set a lookup field's value for a **u** element, it must return "-2130575159" in the **ec** attribute of the **u** element in the protocol server's response to this update and MUST NOT commit any data from the **u** element.

mit: A macro token representing a data macro instance. If a macro token is specified, the protocol server MUST wait for the data macro instance to complete before attempting the update, and the macro token MUST have come from an **UpdateListsResultInfo** or a **RunDataMacroResponse**.

3.1.4.8.2.2 UpdateListsResponse

The result data for the **UpdateLists** WSDL operation.

```
<xs:element name="UpdateListsResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="UpdateListsResult"
type="tns:UpdateListsResultInfo"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

UpdateListsResult: Specifies the results for the UpdateLists request sent by the protocol client.

The number of elements in **UpdateListsResult** MUST be equal to the number of **Update** elements in the **UpdateLists** element supplied to the operation. If any of the **Update** elements use the **ut** attribute, the matching **Update** element in the response MUST have the same value for the **ut** attribute. This allows the protocol client to match responses and requests.

If a field (2) has its hidden attribute set to "true", as specified in [MS-WSSFO2] section 2.2.8.3.3.2, and the field (2) is not the **ID** field or the **owshiddenversion** field, as specified in [MS-WSSTS] section 2.4.2, the protocol server MUST NOT return these fields (2) to the client in any response. Fields (2) of this type will be referenced later in insert and update responses.

For an insert request, if the operation is successful, the protocol server MUST return all of the fields (2) that were created on the protocol server for the new list item, except for the fields (2) that are hidden as specified earlier. The protocol server MUST return a new unique list item identifier as the **id** attribute of the response, which the protocol client MUST use to reference the list item in subsequent update or delete requests. If the operation is successful, the protocol server MUST set **ec** to zero ("0") and **em** to an empty string. Otherwise, the protocol server MUST set **ec** to an error code and MUST set **em** to an error message. See section 3.1.4.8.3.1 for a list of error codes.

For an update request, if the operation is successful, the protocol server MUST return all of the fields (2) that have changed on the server, except for the fields (2) that are hidden as specified earlier. The protocol server MUST return the columns **owshiddenversion**, **Modified**, and **Editor**, as defined in [MS-WSSTS] section 2.4.2, even if these fields (2) have not changed. If the operation is successful, the protocol server MUST set **ec** to zero ("0") and **em** to an empty string. Otherwise, the protocol MUST set **ec** to an error code and MUST set **em** to an error message. See section 3.1.4.8.3.1 for a list of error codes.

For a delete request, if the operation is successful, the protocol server MUST return a **u** element with no **f** elements, and MUST set **ec** to zero ("0") and **em** to an empty string. If the operation is not successful, the protocol MUST set **ec** to an appropriate error code and MUST set **em** to an error message. See section 3.1.4.8.3.1 for a list of error codes.

3.1.4.8.3 Complex Types

3.1.4.8.3.1 Update

The **Update** complex type contains information about an insert, update, or delete to a particular list item.

```
<xs:complexType name="Update">
  <xs:sequence>
```

```
    <xs:element minOccurs="0" maxOccurs="unbounded" name="f" nillable="true"
type="tns:FieldValue"/>
  </xs:sequence>
  <xs:attribute name="ec" type="xs:int"/>
  <xs:attribute name="em" type="xs:string"/>
  <xs:attribute name="cmd" type="tns:UpdateCommand" use="required"/>
  <xs:attribute name="ut" type="xs:string"/>
  <xs:attribute name="ln" type="xs:string"/>
  <xs:attribute name="id" type="xs:int" use="required"/>
</xs:complexType>
```

f: The **FieldValue** elements to be updated or that have been updated. See section 3.1.4.8.3.2 for details.

ec: The error code returned for this **Update**. This MUST only be set in a protocol server response. If the operation described by the current **Update** succeeded, this MUST be set to zero ("0"). The following table specifies error codes that MUST be returned for particular scenarios.

Error code	Error meaning
-2130575166	Delete cannot occur because of a restrict delete relationship.
-2130575339	There is a data conflict for a **Delete** operation because the information that is on the protocol server and the protocol client has different **owshiddenversion** field values.
-2130575322	A **Delete** operation was attempted by specifying a list (1) by name in the **ln** attribute.
-2130575305	There is a data conflict for an update operation because the information that is on the protocol server and the protocol client have different **owshiddenversion** field values.
-2130575169	An attempt was made to set a field (2) that is marked to not allow duplicate values as a duplicate of another list item.
-2130575163	A value for a required field (2) was not specified in the update request.
-2130575159	Trying to set the value of a lookup field to an **ID** that does not exist in the parent table.
-2147467259	General failure.

em: The error message returned for this **Update**. This MUST only be set in a protocol server response.

cmd: Specifies which **UpdateCommand** is being executed in this **Update**. MUST be present. See section 2.2.5.1 for details.

ut: An update tag used by the protocol client to associate protocol client requests with protocol server responses. If the protocol client is sending multiple **Update** elements in an **UpdateLists** request, the protocol client MUST specify a monotonically incrementing integer for **ut**, and the protocol server MUST send a reply with the same **ut**. If the protocol client is sending a single **Update** in the **UpdateLists** request, the protocol client MUST set **ut** to zero ("0"), and the protocol server MUST send a response with **ut** set to zero ("0").

ln: The title or **list identifier** of the list (1) that is being updated. MUST be present.

id: The list item identifier for the list item being updated. MUST be present.

3.1.4.8.3.2 FieldValue

The **FieldValue** complex type contains the name and value of a field (2) for a specific list item.

```
<xs:complexType name="FieldValue">
  <xs:attribute name="n" type="xs:string"/>
  <xs:attribute name="v" type="xs:string"/>
</xs:complexType>
```

n: Specifies the field internal name of a field (2) in the list (1).

v: Specifies the value of the field (2). If present, MUST be a string representation of a value that matches the data type of the field (2) specified by the **n** attribute, and MUST be formatted in the data culture of the session (2).

3.1.4.8.3.3 UpdateListsResultInfo

The results of an **UpdateLists** request.

```
<xs:complexType name="UpdateListsResultInfo">
  <xs:sequence>
    <xs:element minOccurs="1" maxOccurs="1" name="mit" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="Update" type="tns:Update"/>
  </xs:sequence>
</xs:complexType>
```

mit: If the protocol server has triggered an **AfterInsert**, **AfterUpdate**, or **AfterDelete** data macro, as specified in [MS-AXL] section 2.1.3.2, in response to this update, this attribute specifies a macro token that can be used as an input to **GetDataMacroState** or **UpdateLists** operations. If the protocol server does not trigger an **AfterInsert**, **AfterUpdate**, or **AfterDelete** data macro, **mit** MUST be NULL.

Update: An array of **Update**. Specifies the results of all of the updates that were executed as part of the **UpdateLists** request. See section 3.1.4.8.3.1.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

4.1 GetCurrentUserInfo

This example describes how **GetCurrentUserInfo** method works. To get the current user information, the protocol client sends the following message to the protocol server:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetCurrentUserInfo
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
    </GetCurrentUserInfo>
  </soap:Body>
</soap:Envelope>
```

The protocol server then responds with the following:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetCurrentUserInfoResponse
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <GetCurrentUserInfoResult>
        <GetCurrentUserInfo
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
          <User ID="25" Sid="S-1-5-21-2127521184-1604012920-1887927527-4150143"
           Name="Andrew Cencini" LoginName="Northwind\andrew"
           Email="andrew@northwindtraders.com"
           Notes=""
           IsSiteAdmin="True"
           IsDomainGroup="False" Flags="0" />
          <Groups>
            <Group ID="3" Name="Team Site Owners" Description="Use this group to give people
full control permissions to the SharePoint site: Team Site"
             OwnerID="3" OwnerIsUser="False" />
          </Groups>
        </GetCurrentUserInfo>
      </GetCurrentUserInfoResult>
    </GetCurrentUserInfoResponse>
  </soap:Body>
</soap:Envelope>
```

4.2 Use UpdateLists to Insert Items into a List

This example describes how to use **UpdateLists** method to insert list items into a list (1).

The protocol client sends the following message to the protocol server to insert a list item into a list (1):

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateLists
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
```

```
      <u cmd="i" ln="{3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}" ut="0" id="0">
        <f n="JobTitle" v="Sales Representitive" />
      </u>
      <par>false</par>
    </UpdateLists>
  </soap:Body>
</soap:Envelope>
```

In the **u** element, as defined in section 3.1.4.8.3.1, the **cmd** attribute has the value "i", indicating that this is an insertion.

ln equals "{3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}". In this example, this is the GUID of an existing list (1). The insertion happens to this list (1).

ut equals zero ("0"). This value is used to match the **UpdateLists** response with **UpdateLists** request. The **ut** value in the corresponding response for this particular **UpdateLists** request is also zero ("0").

In an **insert** command, the **id** attribute in the request is ignored on the protocol server. The protocol client in this example sends a zero ("0").

In this example, the protocol client requests to insert a list item that has one field (2), named "JobTitle" Therefore, there is an **f** element, as defined in section 3.1.4.8.3.2.

The protocol server responds with the following message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateListsResponse
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <UpdateListsResult>
        <mit xsi:nil="true" />
        <Update ec="0" em="" cmd="i" ut="0" ln="{3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}"
id="1">
          <f n="JobTitle" v=" Sales Representitive " />
          <f n="LinkTitleNoMenu" v="" />
          <f n="Editor" v="25;#Andrew Cencini" />
          <f n="Author" v="25;#Andrew Cencini " />
          <f n="Modified" v="05/11/2009 08:02:36" />
          <f n="Created" v="05/11/2009 08:02:36" />
          <f n="ID" v="5" />
          <f n="owshiddenversion" v="1" />
          <f n="Attachments" v="False" />
        </Update>
      </UpdateListsResult>
    </UpdateListsResponse>
  </soap:Body>
</soap:Envelope>
```

In this example, the **id** attribute equals "1", meaning that the list item inserted has an identifier equal to "1" in the list (1). In subsequent **UpdateLists** calls, both the protocol client and the protocol server use an identifier equal to "1" to refer to this list item.

4.3 Use UpdateLists to Insert Items into Two Lists with Lookup Relationships

This example shows how to use **UpdateLists** to insert list items into two lists (1) that have lookup relationships. In this example, List 1 is referred to by GUID {E5BDB272-1DFB-4752-903E-BF7BFF2052FE}. List 2 is referred to by GUID {3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}. List 1 has a field (2) named "Occupation" that is of lookup type. It references the **id** attribute of the list items in List 2. The list item to be inserted into List 1 references a list item that is to be inserted into List 2 in the same **UpdateLists** call.

The protocol client sends the following message to the protocol server to insert list items into two lists (1) in one call:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateLists
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <u cmd="i" ln="{3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}" ut="0" id="-1">
        <f n="JobTitle" v="Sales Representative" />
        <f n="_OldID" v="2" />
      </u>
      <u cmd="i" ln="{E5BDB272-1DFB-4752-903E-BF7BFF2052FE}" ut="4" id="-1">
        <f n="FullName" v="Nancy Freehafer" />
        <f n="Account" v="nancy@northwindtraders.com" />
        <f n="_OldID" v="4" />
        <f n="Occupation" v="-1" />
      </u>
      <par>false</par>
    </UpdateLists>
  </soap:Body>
</soap:Envelope>
```

Because there are two lists (1) to be updated, there are two **u** elements in the message that is shown later. In the first **u** element, **ln** equals {3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}, meaning the insertion happens to List 2. The second **u** element in the message that is shown later has the **ln** attribute equals {E5BDB272-1DFB-4752-903E-BF7BFF2052FE}, referring to List 1.

If the list item to be inserted to List 1 references a list item that already exists in List 2 on the protocol server, the value of the field (2) named "Occupation" is filled with the identifier of the list item being referenced. However, in this example, the list item to be inserted to List 1 references a list item that is to be inserted in List 2 in the same **UpdateLists** call. Therefore, at the time the protocol client sends the message, it does not know the identifier value for the list item being referenced. However, it knows that in the first **u** element, the **id** attribute equals "-1". That is the list item to be inserted into List 2. Therefore, the value of the fourth **f** element in the second **u** element is "-1" to refer to the list item to be inserted in List 2 in the first **u** element of this **UpdateLists** call.

The protocol server responds with the following message:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <UpdateListsResponse
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <UpdateListsResult>
        <mit xsi:nil="true" />
```

```
        <Update ec="0" em="" cmd="i" ut="0" ln="{3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}"
id="1" >
          <f n="JobTitle" v="Sales Representative" />
          <f n="_OldID" v="2" />
          <f n="LinkTitleNoMenu" v=""/>
          <f n="Editor" v="25;#Andrew Cencini" />
          <f n="Author" v="25;#Andrew Cencini" />
          <f n="Modified" v="05/7/2009 06:32:01" />
          <f n="Created" v="05/7/2009 06:32:01"/>
          <f n="ID" v="1" />
          <f n="owshiddenversion" v="1" />
          <f n="Attachments" v="False" />
        </Update>
        <Update ec="0" em="" cmd="i" ut="4" ln="{E5BDB272-1DFB-4752-903E-BF7BFF2052FE}"
id="1" >
          <f n="FullName" v="Nancy Freehafer" />
          <f n="Account" v="nancy@northwindtraders.com" />
          <f n="_OldID" v="4" />
          <f n="LinkTitleNoMenu" v="" />
          <f n="Editor" v="25;#Andrew Cencini" />
          <f n="Author" v="25;#Andrew Cencini" />
          <f n="Modified" v="05/7/2009 06:32:01" />
          <f n="Created" v="05/7/2009 06:32:01" />
          <f n="Occupation" v="1;#Sales Representative" />
          <f n="ID" v="1" />
          <f n="owshiddenversion" v="1" />
          <f n="Attachments" v="False" />
        </Update>
      </UpdateListsResult>
    </UpdateListsResponse>
  </soap:Body>
</soap:Envelope>
```

In the first **Update** element, the list item is assigned an **id** attribute value "1" in List 2 ({3B6DEE82-D5AC-4ACE-A6E1-00774FA1E10F}). Once the new list item was created and assigned a permanent **id**, the protocol server performed identifier fixup on the remaining list items. Therefore, in the second **Update** element, the **f** element named "Occupation" in List 1 ({E5BDB272-1DFB-4752-903E-BF7BFF2052FE}) has the value "1; # Sales Representative" because it is referring to the list item with **id** = 1 in List 2.

4.4 RunDataMacro and GetDataMacroState

The following example describes how to use the **RunDataMacro** method to trigger a data macro called **AddAComment** and how to use the **GetDataMacroState** method to track the status of this data macro instance. In this example, the data macro **AddAComment** has two parameters, *ContactIDParam* and *CommentParam*. The protocol client sends the following message to the protocol server.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RunDataMacro
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <macroName>Comments.AddAComment</macroName>
      <parameters>
        <KeyValuePair>
          <Key xsi:type="xsd:string">contactIDParam</Key>
```

```
            <Value xsi:type="xsd:double">1</Value>
          </KeyValuePair>
          <KeyValuePair>
            <Key xsi:type="xsd:string">CommentParam</Key>
            <Value xsi:type="xsd:string"><![CDATA[abcded]]></Value>
          </KeyValuePair>
        </parameters>
      </RunDataMacro>
    </soap:Body>
  </soap:Envelope>
```

The protocol server responds with the following message, which contains the token for this data macro instance on the protocol server.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <RunDataMacroResponse
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <RunDataMacroResult>AAEAAAD/////AQAAAAAAAAAMAgAAAGJNaWNyb3NvZ
      nQuT2ZmaWNlLkFjY2Vzcy5TZXJ2ZXIsIFZlcnNpb249MTQuMC4LjAsIEN1bHR1cmU9bm
      V1dHJhbCwgUHVibGljS2V5VG9rZW49NzFlOWJjZTExMWU5NDI5YwUBAAAANU1pY3Jvc2
      9mdC5PZmZpY2UuQWNjZXNzLlNlcnZlci5EYXRhLkRhdGFNYWNyb0luc3RhbmNlBAAAAA
      xtX2luc3RhbmNlSWQNbV9zZXNzaW9uTmFtZRRtX3dvcmtmbG93SW5zdGFuY2VJZAdtX3
      N0YXRlAQEBBDJNaWNyb3NvZnQuT2ZmaWNlLkFjY2Vzcy5TZXJ2ZXIuRGF0YS5EYXRhTW
      Fjcm9TdGF0ZQIAAAACAAAABgMAAAAkNjQ4ODQ5NjYtZjI2My00ZTcxLWFhOGEtZTBjMz
      Q3M2IzODBhBgQAAADOATM2LjBhNTk2MWU4LThiZjgtNGFjNS04YmFiLWRjZDViNmJiM2
      U5MzE2My4xLlYyMi4xNXlMTjgrYjU4UlFrem8rY0JYUEJGOTAuNS5lbi1VUzUuZW4tVV
      M3My4rMDQ4MCMwMDAwLTExLTAwLTAxVDAyOjAwOjAwOjAwMDAjKzAwMDAjMDAwMC0wMy
      0wMC0wMlQwMjowMDowMDowMDAwIy0wMDYwMzYuMDAwMDAwMDAtMDAwMC0wMDAwLTAwMD
      AtMDAwMDAwMDAwMDAwMS5VCgX7////Mk1pY3Jvc29mdC5PZmZpY2UuQWNjZXNzLlNlcn
      Zlci5EYXRhLkRhdGFNYWNyb1N0YXRlAQAAAAd2YWx1ZV9fAAgCAAAAAQAAAAsAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==
      </RunDataMacroResult>
    </RunDataMacroResponse>
  </soap:Body>
</soap:Envelope>
```

The token returned in the **RunDataMacro** method can be used to track the status of this data macro instance. The protocol client calls **GetDataMacroState** by sending the following message to the protocol server.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetDataMacroState
xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
      <macroToken>AAEAAAD/////AQAAAAAAAAAMAgAAAGJNaWNyb3NvZ
      nQuT2ZmaWNlLkFjY2Vzcy5TZXJ2ZXIsIFZlcnNpb249MTQuMC4LjAsIEN1bHR1cmU9bm
```

```
        V1dHJhbCwgUHVibGljS2V5VG9rZW49NzFlOWJjZTExMWU5NDI5YwUBAAAANU1pY3Jvc2
        9mdC5PZmZpY2UuQWNjZXNzLlNlcnZlci5EYXRhLkRhdGFNYWNyb0luc3RhbmNlBAAAAA
        xtX2luc3RhbmNlSWQNbV9zZXNzaW9uTmFtZRRtX3dvcmtmbG93SW5zdGFuY2VJZAdtX3
        N0YXRlAQEBBDJNaWNyb3NvZnQuT2ZmaWNlLkFjY2Vzcy5TZXJ2ZXIuRGF0YS5EYXRhTW
        Fjcm9TdGF0ZQIAAAACAAAABgMAAAAkNjQ4ODQ5NjYtZjI2My00ZTcxLWFhOGEtZTBjMz
        Q3M2IzODBhBgQAAADOATM2LjBhNTk2MWU4LThiZjgtNGFjNS04YmFiLWRjZDViNmJiM2
        U5MzE2My4xLlYyMi4xNXlMTjgrYjU4UlFrem8rY0JYUEJGOTAuNS5lbi1VUzUuZW4tVV
        M3My4rMDQ4MCMwMDAwLTExLTAwLTAxVDAyOjAwOjAwOjAwMDAjKzAwMDAjMDAwMC0wMy
        0wMC0wMlQwMjowMDowMDowMDAwIy0wMDYwMzYuMDAwMDAwMDAtMDAwMC0wMDAwLTAwMD
        AtMDAwMDAwMDAwMDAwMS5VCgX7////Mk1pY3Jvc29mdC5PZmZpY2UuQWNjZXNzLlNlcn
        Zlci5EYXRhLkRhdGFNYWNyb1N0YXRlAQAAAAd2YWx1ZV9fAAgCAAAAAQAAAAsAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==
        </macroToken>
      </GetDataMacroState>
    </soap:Body>
  </soap:Envelope>
```

In this example, the data macro **AddAComment** is still processing. Therefore, the protocol server responds with the following message.

```
  <?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <GetDataMacroStateResponse 
  xmlns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/">
        <GetDataMacroStateResult>
          <State>Running</State>
          <ErrorNumber>0</ErrorNumber>
        </GetDataMacroStateResult>
      </GetDataMacroStateResponse>
    </soap:Body>
  </soap:Envelope>
```

In the response from the protocol server, the **GetDataMacroStateResult** element has the value "Running", meaning that the data macro is still processing.

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:tns="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
targetNamespace="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <xs:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="UpdateLists">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="unbounded" name="u" type="tns:Update" />
            <xs:element minOccurs="1" maxOccurs="1" name="par" type="xs:boolean" />
            <xs:element minOccurs="1" maxOccurs="1" name="mit" nillable="true"
type="xs:string" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="Update">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="f" nillable="true"
type="tns:FieldValue" />
        </xs:sequence>
        <xs:attribute name="ec" type="xs:int" />
        <xs:attribute name="em" type="xs:string" />
        <xs:attribute name="cmd" type="tns:UpdateCommand" use="required" />
        <xs:attribute name="ut" type="xs:string" />
        <xs:attribute name="ln" type="xs:string" />
        <xs:attribute name="id" type="xs:int" use="required" />
      </xs:complexType>
      <xs:complexType name="FieldValue">
        <xs:attribute name="n" type="xs:string" />
        <xs:attribute name="v" type="xs:string" />
      </xs:complexType>
      <xs:simpleType name="UpdateCommand">
        <xs:restriction base="xs:string">
          <xs:enumeration value="u" />
          <xs:enumeration value="i" />
          <xs:enumeration value="d" />
        </xs:restriction>
      </xs:simpleType>
      <xs:element name="UpdateListsResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="UpdateListsResult"
type="tns:UpdateListsResultInfo" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="UpdateListsResultInfo">
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="1" name="mit" nillable="true" type="xs:string"
/>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="Update" type="tns:Update" />
```

```
        </xs:sequence>
      </xs:complexType>
      <xs:element name="GetDataMacroState">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="macroToken" type="xs:string" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetDataMacroStateResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="GetDataMacroStateResult"
type="tns:DataMacroInstanceState" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:complexType name="DataMacroInstanceState">
        <xs:sequence>
          <xs:element minOccurs="1" maxOccurs="1" name="State" type="tns:DataMacroState" />
          <xs:element minOccurs="1" maxOccurs="1" name="ErrorNumber" type="xs:int" />
          <xs:element minOccurs="0" maxOccurs="1" name="ErrorDescription" type="xs:string" />
          <xs:element minOccurs="0" maxOccurs="1" name="ReturnVars"
type="tns:ArrayOfKeyValuePair" />
        </xs:sequence>
      </xs:complexType>
      <xs:simpleType name="DataMacroState">
        <xs:restriction base="xs:string">
          <xs:enumeration value="Error" />
          <xs:enumeration value="Running" />
          <xs:enumeration value="Complete" />
        </xs:restriction>
      </xs:simpleType>
      <xs:complexType name="ArrayOfKeyValuePair">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="unbounded" name="KeyValuePair"
type="tns:KeyValuePair" />
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="KeyValuePair">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="1" name="Key" />
          <xs:element minOccurs="1" maxOccurs="1" name="Value" nillable="true" />
        </xs:sequence>
      </xs:complexType>
      <xs:element name="RunDataMacro">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="macroName" type="xs:string" />
            <xs:element minOccurs="0" maxOccurs="1" name="parameters"
type="tns:ArrayOfKeyValuePair" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="RunDataMacroResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="RunDataMacroResult"
type="xs:string" />
```

```
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetCurrentUserInfo">
        <xs:complexType />
      </xs:element>
      <xs:element name="GetCurrentUserInfoResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="GetCurrentUserInfoResult">
              <xs:complexType mixed="true">
                <xs:sequence>
                  <xs:any />
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetServerInformation">
        <xs:complexType />
      </xs:element>
      <xs:complexType name="AccessServerInformationType">
        <xs:sequence>
          <xs:element minOccurs="0" maxOccurs="1" name="MinimumAccessServicesVersion"
type="tns:VersionType" />
          <xs:element minOccurs="0" maxOccurs="1" name="MaximumAccessServicesVersion"
type="tns:VersionType" />
          <xs:element minOccurs="0" maxOccurs="1" name="SiteVersion" type="tns:VersionType"
/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="VersionType">
        <xs:attribute name="Major" type="xs:int" use="required" />
        <xs:attribute name="Minor" type="xs:int" use="required" />
      </xs:complexType>
      <xs:element name="GetServerInformationResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="AccessServerInformation"
nillable="true" type="tns:AccessServerInformationType" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="GetAccessServicesVersion">
        <xs:complexType />
      </xs:element>
      <xs:element name="GetAccessServicesVersionResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="Version" nillable="true"
type="tns:VersionType" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SetAccessServicesVersion">
        <xs:complexType>
          <xs:sequence>
```

```
            <xs:element minOccurs="1" maxOccurs="1" name="Version" nillable="true"
type="tns:VersionType" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SetAccessServicesVersionResponse">
        <xs:complexType />
      </xs:element>
      <xs:element name="StartCompilation">
        <xs:complexType />
      </xs:element>
      <xs:element name="StartCompilationResponse">
        <xs:complexType />
      </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:message name="UpdateListsSoapIn">
    <wsdl:part name="parameters" element="tns:UpdateLists" />
  </wsdl:message>
  <wsdl:message name="UpdateListsSoapOut">
    <wsdl:part name="parameters" element="tns:UpdateListsResponse" />
  </wsdl:message>
  <wsdl:message name="GetDataMacroStateSoapIn">
    <wsdl:part name="parameters" element="tns:GetDataMacroState" />
  </wsdl:message>
  <wsdl:message name="GetDataMacroStateSoapOut">
    <wsdl:part name="parameters" element="tns:GetDataMacroStateResponse" />
  </wsdl:message>
  <wsdl:message name="RunDataMacroSoapIn">
    <wsdl:part name="parameters" element="tns:RunDataMacro" />
  </wsdl:message>
  <wsdl:message name="RunDataMacroSoapOut">
    <wsdl:part name="parameters" element="tns:RunDataMacroResponse" />
  </wsdl:message>
  <wsdl:message name="GetCurrentUserInfoSoapIn">
    <wsdl:part name="parameters" element="tns:GetCurrentUserInfo" />
  </wsdl:message>
  <wsdl:message name="GetCurrentUserInfoSoapOut">
    <wsdl:part name="parameters" element="tns:GetCurrentUserInfoResponse" />
  </wsdl:message>
  <wsdl:message name="GetServerInformationSoapIn">
    <wsdl:part name="parameters" element="tns:GetServerInformation" />
  </wsdl:message>
  <wsdl:message name="GetServerInformationSoapOut">
    <wsdl:part name="parameters" element="tns:GetServerInformationResponse" />
  </wsdl:message>
  <wsdl:message name="GetAccessServicesVersionSoapIn">
    <wsdl:part name="parameters" element="tns:GetAccessServicesVersion" />
  </wsdl:message>
  <wsdl:message name="GetAccessServicesVersionSoapOut">
    <wsdl:part name="parameters" element="tns:GetAccessServicesVersionResponse" />
  </wsdl:message>
  <wsdl:message name="SetAccessServicesVersionSoapIn">
    <wsdl:part name="parameters" element="tns:SetAccessServicesVersion" />
  </wsdl:message>
  <wsdl:message name="SetAccessServicesVersionSoapOut">
    <wsdl:part name="parameters" element="tns:SetAccessServicesVersionResponse" />
  </wsdl:message>
  <wsdl:message name="StartCompilationSoapIn">
```

```
      <wsdl:part name="parameters" element="tns:StartCompilation" />
    </wsdl:message>
    <wsdl:message name="StartCompilationSoapOut">
      <wsdl:part name="parameters" element="tns:StartCompilationResponse" />
    </wsdl:message>
    <wsdl:portType name="AccessServerSoap">
      <wsdl:operation name="UpdateLists">
        <wsdl:input message="tns:UpdateListsSoapIn" />
        <wsdl:output message="tns:UpdateListsSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetDataMacroState">
        <wsdl:input message="tns:GetDataMacroStateSoapIn" />
        <wsdl:output message="tns:GetDataMacroStateSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="RunDataMacro">
        <wsdl:input message="tns:RunDataMacroSoapIn" />
        <wsdl:output message="tns:RunDataMacroSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetCurrentUserInfo">
        <wsdl:input message="tns:GetCurrentUserInfoSoapIn" />
        <wsdl:output message="tns:GetCurrentUserInfoSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetServerInformation">
        <wsdl:input message="tns:GetServerInformationSoapIn" />
        <wsdl:output message="tns:GetServerInformationSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="GetAccessServicesVersion">
        <wsdl:input message="tns:GetAccessServicesVersionSoapIn" />
        <wsdl:output message="tns:GetAccessServicesVersionSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="SetAccessServicesVersion">
        <wsdl:input message="tns:SetAccessServicesVersionSoapIn" />
        <wsdl:output message="tns:SetAccessServicesVersionSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="StartCompilation">
        <wsdl:input message="tns:StartCompilationSoapIn" />
        <wsdl:output message="tns:StartCompilationSoapOut" />
      </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="AccessServerSoap" type="tns:AccessServerSoap">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
      <wsdl:operation name="UpdateLists">
        <soap:operation
  soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/Update
  Lists" style="document" />
        <wsdl:input>
          <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
          <soap:body use="literal" />
        </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="GetDataMacroState">
        <soap:operation
  soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetDat
  aMacroState" style="document" />
        <wsdl:input>
          <soap:body use="literal" />
        </wsdl:input>
```

```
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RunDataMacro">
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/RunDat
aMacro" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCurrentUserInfo">
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetCur
rentUserInfo" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetServerInformation">
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetSer
verInformation" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAccessServicesVersion">
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetAcc
essServicesVersion" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetAccessServicesVersion">
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/SetAcc
essServicesVersion" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="StartCompilation">
```

```
      <soap:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/StartC
ompilation" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="AccessServerSoap12" type="tns:AccessServerSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="UpdateLists">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/Update
Lists" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetDataMacroState">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetDat
aMacroState" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RunDataMacro">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/RunDat
aMacro" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetCurrentUserInfo">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetCur
rentUserInfo" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetServerInformation">
```

```
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetSer
verInformation" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetAccessServicesVersion">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/GetAcc
essServicesVersion" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="SetAccessServicesVersion">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/SetAcc
essServicesVersion" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="StartCompilation">
      <soap12:operation
soapAction="http://schemas.microsoft.com/office/Access/Server/WebServices/AccessServer/StartC
ompilation" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body 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® Access® 2010
- Microsoft® SharePoint® Server 2010
- Microsoft® Access® 15 Technical Preview
- Microsoft® SharePoint® Server 15 Technical 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.

8 Change Tracking

This section identifies changes that were made to the [MS-ASWS] protocol document between the June 2011 and January 2012 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

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 or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.6 Applicability Statement	Clarified the prerequisites/preconditions information by adding URI examples and removing other details.	N	Content updated.
2.2 Common Message Syntax	Changed the term "common structures" to "common definitions".	N	Content updated.
2.2.1 Namespaces	Modified the description to include more details.	N	Content updated.
2.2.2 Messages	Added an introduction to this section.	N	Content updated.
2.2.3 Elements	Added an introduction to this section.	N	Content updated.
7 Appendix B: Product Behavior	Updated the product list.	N	Content updated.

9 Index

A

Abstract data model
  server 15
    Access services site template 15
    Access services site version 23
Access services site template 15
Access services site version 23
Applicability 9
ArrayOfKeyValuePair complex type 12
Attribute groups 14
Attributes 14

C

Capability negotiation 9
Change tracking 59
Client
  overview 15
Complex types 12
  ArrayOfKeyValuePair 12
  KeyValuePair 12
  VersionType 13

D

Data model - abstract
  server 15
    Access services site template 15
    Access services site version 23

E

Events
  local - server 42
  timer - server 42
Examples
  GetCurrentUserInfo 43
  RunDataMacro and GetDataMacroStatus 46
  use UpdateLists to insert items into a list 43
  use UpdateLists to insert items into two lists with lookup relationships 45

F

Fields - vendor-extensible 9
Full WSDL 50

G

GetCurrentUserInfo example 43
Glossary 6
Groups 14

I

Implementer - security considerations 49
Index of security parameters 49
Informative references 8
Initialization
  server 24
Introduction 6

K

KeyValuePair complex type 12

L

Local events
  server 42

M

Message processing
  server 24
Messages
  ArrayOfKeyValuePair complex type 12
  attribute groups 14
  attributes 14
  complex types 12
  elements 12
  enumerated 12
  Faults 12
  Faults message 12
  groups 14
  KeyValuePair complex type 12
  namespaces 11
  simple types 13
  syntax 11
  transport 11
  UpdateCommand simple type 13
  VersionType complex type 13

N

Namespaces 11
Normative references 7

O

Operations
  GetAccessServicesVersion 24
  GetCurrentUserInfo 26
  GetDataMacroState 27
  GetServerInformation 30
  RunDataMacro 32
  SetAccessServicesVersion 33
  StartCompilation 35
  UpdateLists 36
Overview (synopsis) 8

P

Parameters - security index 49
Preconditions 9

Prerequisites 9
Product behavior 58

R

References 7
  informative 8
  normative 7
Relationship to other protocols 8
RunDataMacro and GetDataMacroStatus example 46

S

Security
  implementer considerations 49
  parameter index 49
Sequencing rules
  server 24
Server
  abstract data model 15
    Access services site template 15
    Access services site version 23
  GetAccessServicesVersion operation 24
  GetCurrentUserInfo operation 26
  GetDataMacroState operation 27
  GetServerInformation operation 30
  initialization 24
  local events 42
  message processing 24
  overview 15
  RunDataMacro operation 32
  sequencing rules 24
  SetAccessServicesVersion operation 33
  StartCompilation operation 35
  timer events 42
  timers 24
  UpdateLists operation 36
Simple types 13
  UpdateCommand 13
Standards assignments 10
Syntax
  messages - overview 11

T

Timer events
  server 42
Timers
  server 24
Tracking changes 59
Transport 11
Types
  complex 12
  simple 13

U

UpdateCommand simple type 13
Use UpdateLists to insert items into a list example 43
Use UpdateLists to insert items into two lists with lookup relationships example 45

V

Vendor-extensible fields 9
Versioning 9
VersionType complex type 13

W

WSDL 50