[MS-PRSTFR]:

ADO XML Persistence Format

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **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 can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **License Programs**. To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.
- **Trademarks**. The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are 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 as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications documentation does 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 documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Support. For questions and support, please contact dochelp@microsoft.com.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.01	Major	Initial Availability
6/27/2008	1.0	Editorial	Revised and edited technical content
8/6/2008	1.01	Editorial	Revised and edited technical content
12/12/2008	1.02	Editorial	Revised and edited technical content
7/13/2009	1.03	Major	Revised and edited the technical content
8/28/2009	1.04	Editorial	Revised and edited the technical content
11/6/2009	1.05	Editorial	Revised and edited the technical content
2/19/2010	2.0	Editorial	Revised and edited the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Editorial	Revised and edited the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.5	Minor	Clarified the meaning of the technical content.
4/11/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2013	2.5	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.6	Minor	Clarified the meaning of the technical content.

Date	Revision History	Revision Class	Comments
11/18/2013	2.6	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.6	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.7	Minor	Clarified the meaning of the technical content.
7/31/2014	2.7	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.8	Minor	Clarified the meaning of the technical content.
2/26/2016	3.0	Major	Significantly changed the technical content.
7/15/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.
7/24/2018	4.0	Major	Significantly changed the technical content.
10/1/2018	5.0	Major	Significantly changed the technical content.
6/18/2019	5.0	None	No changes to the meaning, language, or formatting of the technical content.
7/20/2021	6.0	Major	Significantly changed the technical content.
10/5/2021	7.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The ADO XML Persistence Format Protocol is a subset of the ADO XML Persistence Format, as described in [MSFT-PRXF], that is used by some communication protocols to represent the tabular data retrieved from relational databases. This document does not specify the complete ADO XML Persistence Format. The full ADO XML Persistence Format has capabilities not specified in this document as they are not used by these communication protocols.

Sections 1.7 and 2 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

attribute group: A collection of attributes that can be used to decorate an **XML element**, as described in [XMLSCHEMA1].

Coordinated Universal Time (UTC): A high-precision atomic time standard that approximately tracks Universal Time (UT). It is the basis for legal, civil time all over the Earth. Time zones around the world are expressed as positive and negative offsets from UTC. In this role, it is also referred to as Zulu time (Z) and Greenwich Mean Time (GMT). In these specifications, all references to UTC refer to the time at UTC-0 (or GMT).

data type: A property of a field that defines the kind of data that is stored in the field, or defines the kind of data returned by an expression when the expression is evaluated.

XML: The Extensible Markup Language, as described in [XML1.0].

XML element: An **XML** structure that typically consists of a start tag, an end tag, and the information between those tags. Elements can have attributes and can contain other elements.

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

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

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

1.2 References

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

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[IEEE754] IEEE, "IEEE Standard for Binary Floating-Point Arithmetic", IEEE 754-1985, October 1985, http://ieeexplore.ieee.org/servlet/opac?punumber=2355

[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

[RFC4122] Leach, P., Mealling, M., and Salz, R., "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005, http://www.rfc-editor.org/rfc/rfc4122.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

[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/

[XMLSCHEMA1/2] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures Second Edition", W3C Recommendation, October 2004, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/

[XMLSCHEMA2/2] Biron, P., and Malhotra, A., Eds., "XML Schema Part 2: Datatypes Second Edition", W3C Recommendation, October 2004, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

1.2.2 Informative References

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

[MS-SITEDATS] Microsoft Corporation, "Site Data Web Service Protocol".

[MSFT-PRXF] Microsoft Corporation, "Persisting Records in XML Format", http://msdn.microsoft.com/en-us/library/ms681538%28VS.85%29.aspx

[W3C-XMLNote] Layman, A., Jung, E., Maler, E., et al., "XML-Data", W3C Note, January 1998, http://www.w3.org/TR/1998/NOTE-XML-data-0105

1.3 Overview

The ADO XML Persistence Format, as described in [MSFT-PRXF], provides an open, vendor-neutral way of accessing data stored in a variety of relational databases. This document describes a subset of ADO XML Persistence Format.

The ADO XML Persistence Format is used to define an **XML** representation of the tabular data retrieved from relational database sources so that it can be unambiguously interpreted by any client. It describes the schema of fields; including name, title, and data type, and the way the rows containing the data are formatted.

ADO XML Persistence Format is based on XML-Data Reduced (XDR). XDR provides a way to define **XML schema** by using XML itself. XDR (see [W3C-XMLNote]) was proposed in the early period of XML evolution. While this proposal has not reached the status of standard, it has nevertheless influenced the ultimately adopted XSD schema definition standard [XMLSCHEMA1/2]. This document defines the XDR notation using standard XSD schema conventions, and defines how XDR is used to define the schema of tabular data and how the tabular data itself is represented.

Elements and attributes of XDR schema both define elements and attributes of a valid document. To avoid confusion, this document calls them *schema elements* and *schema attributes*, as opposed to *elements* and *attributes* of the valid document.

1.4 Relationship to Protocols and Other Structures

The ADO XML Persistence Format is used in various protocols that deal with tabular or quasi-tabular data, such as Site Data ([MS-SITEDATS]) and Lists ([MS-LISTSWS]).

1.5 Applicability Statement

ADO XML Persistence Format can be used to represent tabular data, originating from a database or similar system, in XML notation.

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

The ADO XML Persistence Format (similar to the XDR schema, which is used as its notation mechanism) allows arbitrary vendor extensions, as long as the core namespaces and interpretations of their components are not redefined. If both the sender and receiver of the data are aware of those vendor extensions, they can take advantage of them; however, if the receiver party is unaware of vendor extensions, it still should be able to process the data within the limits of semantics defined in this document.

2 Structures

2.1 Namespaces

This specification defines and references various **XML namespaces** by 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.

The following table shows the standard XML namespaces used by the ADO XML Persistence Format and the alias, or prefix, used in the remaining sections of this specification.

Prefix	Namespace URI	Reference
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1/2] and [XMLSCHEMA2/2]
rs	urn:schemas-microsoft-com:rowset	
xdr	uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882	
dt	uuid:C2F41010-65B3-11d1-A29F-00AA00C14882	

2.2 ADO XML Persistence Format

The following XML schema defines the ADO XML Persistence Format:

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           attributeFormDefault="unqualified"
           elementFormDefault="qualified"
           xmlns:xdr="uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882"
           xmlns:dt="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882"
           xmlns:rs="urn:schemas-microsoft-com:rowset">

  <xs:import namespace="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882" />
  <xs:import namespace="uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882" />
  <xs:import namespace="urn:schemas-microsoft-com:rowset" />

  <xs:complexType name="xml">
    <xs:sequence>
      <xs:element ref="xdr:Schema"/>
      <xs:element ref="rs:data"/>
    </xs:sequence>
  </xs:complexType>

</xs:schema>
```

xml: The root **XML element**.

Schema: The schema of the data as specified in the XML-Data Schema element (see section 2.3.1.2).

data: The data conforming to the XML schema element as specified in the rowset (see section 2.4). Within the data element, zero or more 'z:row' sub-elements can be present, denoting rows of the tabular data. Each row has at most as many attributes as there are columns, with values formatted according to their data types as defined by the schema element. Missing attributes in z:row denote null values, see section 3.2 for more information.

2.3 XML-Data Reduced Schema

XML-Data Reduced (XDR) schema is well-formed XML that specifies the structure of another XML document, or documents, called valid documents. XDR schema consists of the following schema elements:

- **ElementType:** Specifies the element tags allowed in the valid document, as well as their contents, which can contain text and other elements.
- **AttributeType:** Specifies the valid attributes, whether they are required, data types, and default values elements in the valid document.
- **element:** Specifies the context where elements of **ElementType** can occur.
- **attribute:** Specifies the attributes of **AttributeType** that can decorate elements of **ElementType**.
- **group:** Elaborates on the structure of the valid document, referring to **ElementType** definitions.

XDR schemas permit attributes and elements from other namespaces to be used.

In the context of ADO XML Persistence Format, there are several limitations of the generic XDR model:

- **AttributeType** schema element MUST NOT be global; it is always a child of **ElementType** schema element. **ElementType** MUST have at least one attribute, because tabular data always has one or more columns.
- **ElementType** schema element defining row of the tabular data MUST appear once and only once in the **Schema** element.

2.3.1 Elements

2.3.1.1 description

The **description** schema element can appear anywhere in the XDR schema. It is used for comments and can contain any mix of text and XML. It does not affect the formal meaning of the XDR schema. The following XML schema defines the **description** schema element:

```
<xs:element  name='description'>
  <xs:complexType  mixed='true'>
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:any  processContents='skip'
               namespace='##any'/>

    </xs:choice>
  </xs:complexType>
</xs:element>
```

2.3.1.2 Schema

The **Schema** element is the root XML element, or document element, of an XDR schema. It can contain definitions of elements (see section 2.3.1.4) and attributes (see section 2.3.1.5) that are permitted to appear in a valid document. The following XML schema defines the **Schema** element:

```
<xs:element  name='Schema'>
  <xs:complexType >
    <xs:choice  minOccurs='0' maxOccurs='unbounded'>
      <xs:element  ref='xdr:AttributeType'/>
      <xs:element  ref='xdr:ElementType'/>
      <xs:element  ref='xdr:description'/>
      <xs:any  namespace='##other' processContents='skip'/>
    </xs:choice>
    <xs:attribute  name='name' type='xs:string'/>
    <xs:attribute  name='id' type='xs:ID'/>
    <xs:anyAttribute  namespace='##other'
                      processContents='skip'/>
  </xs:complexType>
</xs:element>
```

AttributeType: One or more **attribute** elements (see section 2.3.1.5) that can be used to decorate various elements in a valid document. In ADO XML Persistence schema global **AttributeType** schema elements are not supported. That is, **AttributeType** cannot be child of **Schema** element.

ElementType: One or more definitions of elements that can appear in valid document. In ADO XML Persistence schema, **ElementType** MUST appear once and only once inside a **Schema** element.

According to XDR schema definitions, element tags and element classes are tightly coupled. That is, the tag defines the class. This is unlike XSD, where one can define complex type, and then prescribe that type to one or more elements, perhaps with different tags.

description: A description for the **Schema** element.

name: The name of the schema.

id: The identifier of the schema. If it is prefixed with the number sign (#), it can be used as a namespace if the XML document contains both the schema and data sections.

2.3.1.3 datatype

The **datatype** schema element is used to specify the **data type** of the attribute or of the element.

The data type provides a set of possible values and a notation to represent specific data. For example, the data type for calendar dates can be defined as 10-character notation, for example: **02/12/2008**, which is interpreted as "the 12th of February, year 2008 A.D." using the Gregorian calendar. The correct interpretation of the date is based on the assumption that 02 means February, which is valid for US-English tradition, but possibly misinterpreted by a French or Russian reader, and even more so by a reader accustomed to the Hijri calendar. There is a limitation on the set of values: dates B.C., such as the death of Julius Caesar, on March 15 in year 44 B.C., cannot be shown in that notation unless the data type is duly extended.

The following XML schema defines the **datatype** schema element:

```
<xs:element  name='datatype'>
  <xs:complexType  mixed='true'>
    <xs:attributeGroup  ref='dt:typeAttributes'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>
```

For more information, see section 2.5. Note that the **datatype** schema attributes MAY be used instead of the **datatype** schema element.

2.3.1.4 ElementType

The **ElementType** element specifies the type (for example, tags, possible contents, and child elements) of the elements that appear in the valid document. The following XML schema defines the **ElementType** element:

```
<xs:element  name='ElementType'>
    <xs:complexType >
      <xs:choice  minOccurs='0' maxOccurs='unbounded'>
        <xs:element  ref='xdr:AttributeType'/>
        <xs:element  ref='xdr:attribute'/>
        <xs:element  ref='xdr:element'/>
        <xs:element  ref='xdr:group'/>
        <xs:element  ref='xdr:datatype'/>
        <xs:element  ref='xdr:description'/>
        <xs:element  ref='xdr:extends'/>
        <xs:any  namespace='##other'
                 processContents='skip'/>
      </xs:choice>
      <xs:attribute  name='name'
                     type='xs:ID'/>
      <xs:attribute  name='content'>
        <xs:simpleType >
          <xs:restriction  base='xs:string'>
            <xs:enumeration  value='empty'/>
            <xs:enumeration  value='textOnly'/>
            <xs:enumeration  value='eltOnly'/>
            <xs:enumeration  value='mixed'/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute  ref='xdr:model'/>
      <xs:attribute  ref='xdr:order'/>
      <xs:attributeGroup  ref='dt:typeAttributes'/>
      <xs:anyAttribute  namespace='##other'/>
    </xs:complexType>
  </xs:element>
```

AttributeType: Defines the attribute that is local to this element, that is, not applicable to other **ElementType** elements (see section 2.3.1.5).

attribute: A reference, with partial redefinition, to an **AttributeType**, (see section 2.3.1.7), which can be used to decorate this XML element.

element: A reference to other **ElementType** elements that can appear inside of this **element** (see section 2.3.1.6).

group: A structural construct to control the order and number of the elements listed inside of this **group** (see section 2.3.1.8).

datatype: The data type of the element value, if its content is text, or mixed, otherwise not applicable (see section 2.3.1.3). This element MUST NOT appear more than once inside of the defined element type. The data type can be defined with **dt:typeAttributes**.

description: A description for the **ElementType** element.

extends: A reference to other **ElementType** in this XDR schema or in a foreign namespace: a "base type". Extended element type inherits all the attributes of the base **ElementType**.

name: The tag of the element, as it appears in the valid document. This is also the name used by schema element references in the XDR schema. This attribute is mandatory and has to be unique in the scope of the XDR schema.

content: Prescribes which **content** this element can have. Values are specified in the **content** attribute (see section 2.3.2.4). The **content** attribute value is by default "mixed". The **content** attribute value affects the default value for the **order** attribute: "mixed" content has default order "many", while "eltOnly" content has default order "seq".

model: Indicates whether this element is "open", that is, whether it can contain elements and attributes having a namespace not defined by this schema, or "closed". The default value is "open", which allows child elements and attributes from foreign namespaces. Whether to interpret or to ignore those foreign child elements and attributes is left to the discretion of the data receiver.

order: Controls how the child elements appear. See section 2.3.2.1 for the definition. See section 2.3.1.8, for the examples of **order** usage.

dt:typeAttributes: A simple way of defining value data type for the element. It is applicable only if content is "textOnly" or "mixed", otherwise this attribute MUST NOT be used.

XDR Schema does not prescribe which element, among defined **ElementTypes**, can appear as the root tag in the valid document. In fact, any **ElementType** can be the valid root.

2.3.1.5 AttributeType

AttributeType schema element introduces the attribute that can be used to decorate elements in the valid document. The following XML schema defines the **AttibuteType** schema element:

```
<xs:element  name='AttributeType'>
    <xs:complexType >
      <xs:choice  minOccurs='0'
                  maxOccurs='unbounded'>
        <xs:element  ref='xdr:datatype'/>
        <xs:element  ref='xdr:description'/>
        <xs:any  namespace='##other'
                 processContents='skip'/>
      </xs:choice>
      <xs:attribute  name='default'
                     type='xs:string'/>
      <xs:attribute  name='name'
                     type='xs:ID'/>
      <xs:attribute  ref='xdr:required'/>
      <xs:attributeGroup  ref='dt:typeAttributes'/>
      <xs:anyAttribute  namespace='##other'/>
    </xs:complexType>
  </xs:element>
```

datatype: The data type that the values of this attribute MUST have.

description: A description for the **AttributeType** element.

default: The value that the defined attribute is presumed to have if it is omitted (see section 2.3.2.2). This value MUST be valid for the **datatype** of the **AttributeType**.

name: The name of the attribute when used in valid documents and referred to by attribute element of XDR schema. Attribute names MUST be unique in the scope of a given XDR schema.

required: Indicates whether the defined attribute is mandatory (`required="yes"`) or optional (`required="no"`). By default, any attribute is optional.

dt:typeAttributes: An alternative form to supply data type information.

The **AttributeType** schema element SHOULD appear as a child of the **Schema** element, or SHOULD appear as a child of the **ElementType** schema element.

In the first case (**global** attribute: child of **Schema** element) **AttributeType** just defines the type of attribute, not yet associated with any **ElementType** (see **attribute** schema element, section 2.3.1.7, which provides such an association). If **AttributeType** appears inside of **ElementType** (**local** attribute), it both defines attribute type and associates it with its parent **ElementType**. In ADO XML Persistence Format schema, **AttributeType** MUST be local to **ElementType**.

2.3.1.6 element

The **element** element refers to an **ElementType** (as defined in this XDR Schema in section 2.3.1.4, or imported from a foreign namespace) and indicates that the referred **ElementType** can appear in the context of another ElementType, either directly or indirectly via the group construct (see section 2.3.1.8). The following XML schema defines the **element** element:

```
<xs:element  name='element'>
  <xs:complexType >
    <xs:attribute  name='type'
                   type='xs:NMTOKEN'/>
    <xs:attributeGroup  ref='xdr:minmax'/>
  </xs:complexType>
</xs:element>
```

type: An **ElementType** defined in this schema or in a foreign namespace.

minmax: The number of occurrences in the given context (see section 2.3.3.1).

2.3.1.7 attribute

The **attribute** element refers to **AttributeType** elements, defined in this XDR Schema in section 2.3.1.5 or imported from a foreign namespace. The attribute schema element appears in **ElementType** context (see section 2.3.1.4) and indicates that the attribute of the referred **AttributeType** can be used to decorate elements of this **ElementType**, which is called "parent" **ElementType** in the following example. The **attribute** schema element can override the **required** value and the **default** value of the referred **AttributeType** in a valid document, but cannot override the data type. The following XML schema defines the **attribute** schema element:

```
<xs:element  name='attribute'>
  <xs:complexType >
    <xs:attribute  name='type'
                   type='xs:NMTOKEN'/>
    <xs:attribute  name='default'
                   type='xs:string'/>
    <xs:attribute  ref='xdr:required'/>
  </xs:complexType>
</xs:element>
```

type: Name of the **AttributeType** element (see section 2.3.1.5), defined in this XDR schema or imported from the foreign namespace.

default: The default value. If this attribute is defined, the value overrides the default value that the referenced **AttributeType** defines. Overriding affects only the parent (see section 2.3.1.4). The default value MUST be valid for the data type as specified by the referred **AttributeType**.

required: Indicates whether this attribute, in the context of parent **ElementType**, is optional or mandatory. By default, attributes are optional.

If **required** is set to "yes" and the default value is provided, all elements of the parent **ElementType** in the valid document MUST have this attribute present with the value equal to default.

2.3.1.8 group

The **group** element is an anonymous construct, used in the context of **ElementType** elements or another **group**, that helps to organize child elements; their valid sequences and number of occurrences. For more information about **ElementType** elements, see section 2.3.1.4. The following XML schema defines the **group** schema element:

```
<xs:element  name='group'>
  <xs:complexType >
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:element  ref='xdr:description'/>
      <xs:element  ref='xdr:element'/>
      <xs:element  ref='xdr:group'/>
      <xs:any  namespace='##other'
               processContents='skip'/>
    </xs:choice>
    <xs:attribute  ref='xdr:order'/>
    <xs:attributeGroup  ref='xdr:minmax'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>
```

description: A description for the **group** element.

element: One or more **element** elements that might appear in that **group**. For more information about **element** elements, see section 2.3.1.6.

group: A structural construct to control the order and number of the elements listed inside of this **group** schema element. In other words, **group** can contain another **group** (or **groups**).

order: A schema attribute indicating in which order the elements can appear. Possible values are defined in section 2.3.2.1.

minmax: The number of occurrences in the given context, see section 2.3.3.1.

2.3.2 Attributes

2.3.2.1 order

The **order** attribute type is used to define the order in which elements from the given list can appear in the context of a parent element or **group**. The **order** schema attribute applies to the **ElementType** element (see section 2.3.1.4) and the **group** element (see section 2.3.1.8). The following XML schema defines the **order** attribute:

```
<xs:attribute name="order">
 <xs:simpleType>
    <xs:restriction base="xs:string">
    <xs:enumeration value="one"/>
    <xs:enumeration value="seq"/>
    <xs:enumeration value="many"/>
   </xs:restriction>
 </xs:simpleType>
</xs:attribute>
```

The allowable values in the **order** attribute are specified in the following table.

Value	Meaning
one	One and only one of the list.
seq	All listed elements in exactly that order.
many	Each element from the list can or cannot appear, in any order and any number.

2.3.2.2 required

The **required** attribute type denotes whether an attribute is optional or mandatory in the valid document. It applies to **AttributeType**, and **attribute** definitions. The following XML schema defines the **required** attribute:

```
<xs:attribute name="required">
  <xs:simpleType>
     <xs:restriction base="xs:string">
       <xs:enumeration value="yes"/>
       <xs:enumeration value="no"/>
     </xs:restriction>
  </xs:simpleType>
</xs:attribute>
```

For more usage information, see AttributeType (section 2.3.1.5), and attribute (section 2.3.1.7). By default, **required** has the value "no" that means not required or optional.

2.3.2.3 model

The **model** attribute applies to the **ElementType** element (see section 2.3.1.4) and denotes "open" or "closed" content of the element. The default value of the **model** attribute is "open". If the **model** attribute is "open", the elements and attributes prefixed by other, or foreign, namespaces can appear inside of the schema attribute in the valid document. If the **model** attribute is "closed", the elements and attributes prefixed by other, or foreign, namespaces cannot appear inside of the schema attribute in the valid document. The following schema defines the **model** attribute:

```
<xs:attribute  name='model'>
  <xs:simpleType >
    <xs:restriction  base='xs:string'>
      <xs:enumeration  value='open'/>
      <xs:enumeration  value='closed'/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
```

2.3.2.4 content

The **content** attribute applies to the **ElementType** element (see section 2.3.1.4), and restricts the content of this element. The following schema defines the **content** attribute:

```
<xs:attribute  name='content'>
  <xs:simpleType >
    <xs:restriction  base='xs:string'>
      <xs:enumeration  value='empty'/>
      <xs:enumeration  value='textOnly'/>
      <xs:enumeration  value='eltOnly'/>
```

```
        <xs:enumeration  value='mixed'/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
```

The allowable values in the **content** schema attribute are specified in the following table.

Value	Meaning
empty	Element can have attributes, but no text or child elements.
textOnly	Element can have attributes and text, but no child elements.
eltOnly	Element can have attributes and child elements, but no text.
mixed	Element can contain both text and elements.

The **content** attribute value is by default "mixed". The value of the **content** attribute affects the default value for the **order** schema attribute. **content** that is "mixed" has default **order** "many", while "eltOnly" **content** has default **order** "seq". If the value of **model** is "open", the value of **content** MUST NOT be "empty".

2.3.3 Attribute Groups

2.3.3.1 minmax

The **minmax** attribute group in XDR schema indicates how many times, and the minimum and maximum times, the element can appear in the context of parent element or group in the valid XML document. The following schema defines the **minmax** attribute:

```
<xs:attributeGroup name="minmax">
 <xs:attribute name="minOccurs">
  <xs:simpleType name="minOccurs">
    <xs:restriction base="xs:string">
      <xs:enumeration value="0"/>
      <xs:enumeration value="1"/>
    </xs:restriction>
  </xs:simpleType>
 </xs:attribute>
 <xs:attribute name="maxOccurs">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="1"/>
      <xs:enumeration value="*"/>
    </xs:restriction>
  </xs:simpleType>
 </xs:attribute>
</xs:attributeGroup>
```

minOccurs: Specifies minimum number of times an element can occur.

maxOccurs: Specifies maximum number of times an element can occur.

For usage information, see sections 2.3.1.6 and 2.3.1.8. By default, **minOccurs** has the value of 1, and **maxOccurs** has the value of 1; `maxOccurs="*"` means "unlimited number of times".

2.4 rowset

The **rowset** namespace contains a set of attributes that are used to specify the data and its underlying attributes. The following XML schema defines a subset of **rowset** namespace that is relevant in this context:

```
<xs:schema xmlns="urn:schemas-microsoft-com:rowset"
           targetNamespace="urn:schemas-microsoft-com:rowset"
           attributeFormDefault="unqualified"
           elementFormDefault="qualified"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="number" type="xs:int" />
  <xs:attribute name="precision" type="xs:unsignedByte" />
  <xs:attribute name="scale" type="xs:unsignedByte" />
  <xs:attribute name="CommandTimeout" type="xs:int"/>
  <xs:element name="data">
    <xs:complexType>
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:any minOccurs="0" maxOccurs="unbounded" namespace="##any"
                processContents="lax" />
      </xs:sequence>


       <xs:anyAttribute namespace="##any" processContents="skip" />
     </xs:complexType>
   </xs:element>
 </xs:schema>
```

name: The name of the field in the underlying data model.

number: The ordinal number of the field in the list of fields. It starts from 1.

precision: The precision of the data, wherever applicable. Precision is the number of digits in a floating point number.

scale: The scale of the data, wherever applicable. Scale is the number of digits to the right of the decimal point**.**

CommandTimeout: The timeout value for this operation.

data: The actual data that can contain any XML attribute and any XML element conforming to the schema defined by XML-Data Reduced Schema inside of the same document.

2.5 datatype

The **datatype** namespace contains a set of attributes that are used to define the data type of the underlying data. The following XML schema defines the namespace:

```
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882"
   xmlns:dt="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882"
   attributeFormDefault="qualified"
   elementFormDefault="qualified">

  <xs:attributeGroup name="typeAttributes">
      <xs:attribute ref="dt:type" />
```

```
      <xs:attribute ref="dt:values"/>
      <xs:attribute ref="dt:minLength"/>
      <xs:attribute ref="dt:maxLength"/>
  </xs:attributeGroup>
  <xs:attribute name="type">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="bin.hex"/>
        <xs:enumeration value="boolean"/>
        <xs:enumeration value="date"/>
        <xs:enumeration value="datetime"/>
        <xs:enumeration value="enumeration"/>
        <xs:enumeration value="float"/>
        <xs:enumeration value="i1"/>
        <xs:enumeration value="i2"/>
        <xs:enumeration value="i4"/>
        <xs:enumeration value="i8"/>
        <xs:enumeration value="int"/>
        <xs:enumeration value="number"/>
        <xs:enumeration value="r4"/>
        <xs:enumeration value="string"/>
        <xs:enumeration value="time"/>
        <xs:enumeration value="Ui1"/>
        <xs:enumeration value="ui1"/>
        <xs:enumeration value="ui4"/>
        <xs:enumeration value="ui8"/>
        <xs:enumeration value="uuid"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="values" type="xs:string"/>
  <xs:attribute name="minLength" type="xs:int"/>
  <xs:attribute name="maxLength" type="xs:int"/>
  </xs:schema>
```

typeAttributes: The **attribute group** that can be used to specify the attributes that can be used to define the data type.

type: The type of the element. The following table specifies the various data types permitted in the type attribute and their specifications.

Type	Description	Data type specification
bin.hex	Hex-encoded binary data	**xs:hexBinary**
boolean	Either a XSD Boolean value or a bit value with 0 specifying false and 1 specifying **true**.	0, 1, **true**, or **false**. The syntax in Augmented BNF notation ([RFC5234]) is: BOOLEANVALUE = "0" / "1" / xs:boolean
date	String specifying date in **UTC**	**xs:date** having no **timezone**.
datetime	String specifying date and time in UTC	**xs:dateTime** having no **timezone**.
enumeration	One of the string values specified in the values attribute.	**xs:string** which is one of the values specified in the values attribute.
float	IEEE 64-bit double precision floating point [IEEE754]	**xs:double**
i1	8-bit signed integer	**xs:byte**
i2	16-bit signed integer	**xs:short**

Type	Description	Data type specification
i4	32-bit signed integer	**xs:int**
i8	64-bit signed integer	**xs:long**
int	32-bit signed integer	**xs:int**
number	IEEE 64-bit double precision floating point [IEEE754]	**xs:double**
r4	IEEE 32-bit single precision floating point [IEEE754]	**xs:float**
string	String	**xs:string**
time	String specifying time in UTC	**xs:time** having no **timezone**.
Ui1	8-bit unsigned integer	**xs:unsignedByte**
ui1	16-bit unsigned integer	**xs:unsignedShort**
ui4	32-bit unsigned integer	**xs:unsignedInt**
ui8	64-bit unsigned integer	**xs:unsignedLong**
uuid	Universally Unique Identifier	A UUID within curly braces. The syntax in Augmented BNF notation ([RFC5234]) is: UUIDVALUE = '{' UUID '}' Where UUID is defined in [RFC4122].

values: Specifies the valid set of values when the type attribute is "enumeration". This is a string in which each valid value is separated by a space character.

minLength: Specifies the minimum allowed length of the data. Length in this context means the binary value length in bytes and not the textual representation length.

maxLength: Specifies the maximum allowed length of the data. Length in this context means the binary value length in bytes and not the textual representation length.

3 Structure Examples

3.1 Document Defined by the Schema

This section shows an example of the XML schema defining a class document, which represents a class from an imaginary secondary school that consists of zero or more students. Each student has a name, an integer identification (for example, a badge number), a grade-point average (GPA) which is a float, and, optionally, a gender, which defaults to "unknown" but can be "male" or "female". The **student** element can contain arbitrary text (for example, remarks from the teacher), which is indicated by `content='mixed'`. In the next section, the schema and data in the ADO XML Persistence Format is represented.

```
<Schema xmlns= 'uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882'
        xmlns:dt='uuid:C2F41010-65B3-11d1-A29F-00AA00C14882'>
  <AttributeType name='gender'
                 dt:type='enumeration'
                 dt:values='unknown male female'
                 default='unknown'
                 required='no'/>
  <AttributeType name='studentID'
                 dt:type='int'
                 required='yes'/>
  <ElementType name='name'
               content='textOnly'/>
  <ElementType name='GPA'
               content='textOnly'
               dt:type='float'/>
  <ElementType name='student'
               content='mixed'>
    <attribute type='gender'/>
    <attribute type='studentID'/>
    <element type='name'/>
    <element type='GPA'/>
  </ElementType>
  <ElementType name='class'
               content='eltOnly'>
    <element type='student'
             maxOccurs='*'/>
  </ElementType>
</Schema>
```

The following is a valid class document conforming to the preceding schema:

```
<class >
  <student studentID='123'
           gender='male'>
    <name> Steve Masters</name>
    <GPA>3.8</GPA> interested in mathematics
    </student>
  <student studentID='678'>
    <name> Lori Penor</name>
    <GPA>4.5</GPA>
  </student>
</class>
```

The reader of the document will know that Steve Masters is "male", while the gender of Lori Penor is "unknown". There is a remark in the student 123 record.

3.2 Data Represented by the Persistence Format

Consider a back-end data store that contains the fields and field properties described in the following table.

Name	Number	Data Type	Max Length
name	1	string	10
bin	2	bin.hex	8
GUID	3	uuid	16
date	4	dateTime	16
float	6	float	8
flag	7	boolean	2

There are two rows, and the data contained in their fields is shown in the following table.

name	bin	GUID	date	float	flag
sample1	00000000499602d2	{8AC68D3D-8A09-4403-8860-D0E494BBE894}	2008-01-25T13:04:00Z	3.1415926535897932	0
sample2	null	null	2008-02-13T18:49:00Z	null	1

The preceding schema and data can be represented by using the ADO XML Persistence Format as follows:

```
<xml xmlns:s='uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882'
  xmlns:dt='uuid:C2F41010-65B3-11d1-A29F-00AA00C14882'
    xmlns:rs='urn:schemas-microsoft-com:rowset'
    xmlns:z='#RowsetSchema'>
<s:Schema id='RowsetSchema'>
  <s:ElementType name='row' content='eltOnly' rs:CommandTimeout='30'>
        <s:AttributeType name='name' rs:number='1' >
            <s:datatype dt:type='string' dt:maxLength='10'  />
        </s:AttributeType>
        <s:AttributeType name='bin' rs:number='2' >
            <s:datatype dt:type='bin.hex' dt:maxLength='8' />
        </s:AttributeType>
        <s:AttributeType name='GUID' rs:number='3'>
            <s:datatype dt:type='uuid' dt:maxLength='16' />
        </s:AttributeType>
        <s:AttributeType name='date' rs:number='4' >
            <s:datatype dt:type='dateTime'  dt:maxLength='16'
                        rs:scale='0' rs:precision='16' />
        </s:AttributeType>
        <s:AttributeType name='float' rs:number='6' >
            <s:datatype dt:type='float' dt:maxLength='8'
                        rs:precision='17' />
        </s:AttributeType>
        <s:AttributeType name='flag' rs:number='7' >
            <s:datatype dt:type='boolean' dt:maxLength='2' />
        </s:AttributeType>
  </s:ElementType>
</s:Schema>
<rs:data>
    <z:row name='sample1' bin='00000000499602d2'
           GUID='{8AC68D3D-8A09-4403-8860-D0E494BBE894}'
           date='2008-01-25T13:04:00Z'
```

```
            float='3.1415926535800001' flag='0'/>
        <z:row name='sample2' date='2008-02-13T18:49:00Z' flag='1'/>
    </rs:data>
    </xml>
```

The absence of an attribute, defined in **RowsetSchema**, in **z:row** element means that the field has a null value. For example, the data row with `name="sample2"` has null values of fields "bin", "GUID", and "float".

4 Security

4.1 Security Considerations for Implementers

None.

4.2 Index of Security Fields

None.

5 Appendix A: XML-Data Reduced Schema

```
<?xml version="1.0"?>
<xs:schema  version='1.0'
            xmlns:xs='http://www.w3.org/2001/XMLSchema'
            xmlns:xdr='uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882'
            xmlns:dt='uuid:C2F41010-65B3-11d1-A29F-00AA00C14882'
            targetNamespace='uuid:BDC6E3F0-6DA3-11d1-A2A3-00AA00C14882'
            attributeFormDefault='unqualified'
            elementFormDefault='qualified'>
  <xs:import  namespace='uuid:C2F41010-65B3-11d1-A29F-00AA00C14882'/>

  <!--  optional or mandatory, applies to attribute  -->
  <xs:attribute  name='required'>
    <xs:simpleType >
      <xs:restriction  base='xs:string'>
        <xs:enumeration  value='yes'/>
        <xs:enumeration  value='no'/>
        <!--  default  -->
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

  <!--  open or closed model, applies to ElementType  -->
  <xs:attribute  name='model'>
    <xs:simpleType >
      <xs:restriction  base='xs:string'>
        <xs:enumeration  value='open'/>
        <!--  default  -->
        <xs:enumeration  value='closed'/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

  <!--  order is one of, or sequential, or random  -->
  <xs:attribute  name='order'>
    <xs:simpleType >
      <xs:restriction  base='xs:string'>
        <xs:enumeration  value='one'/>
        <xs:enumeration  value='seq'/>
        <!--  default for eltOnly  -->
        <xs:enumeration  value='many'/>
        <!--  default for mixed  -->
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>

  <!--  at least and at most number of occurrences  -->
  <xs:attributeGroup  name='minmax'>
    <xs:attribute  name='minOccurs'>
      <xs:simpleType >
        <xs:restriction  base='xs:string'>
          <xs:enumeration  value='0'/>
          <xs:enumeration  value='1'/>
          <!--  default  -->
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute  name='maxOccurs'>
      <xs:simpleType >
        <xs:restriction  base='xs:string'>
          <xs:enumeration  value='1'/>
          <!--  default  -->
          <xs:enumeration  value='*'/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:attributeGroup>
```

```
<!--  root element of XDR schema  -->
<xs:element  name='Schema'>
  <xs:complexType >
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:element  ref='xdr:AttributeType'/>
      <xs:element  ref='xdr:ElementType'/>
      <xs:element  ref='xdr:description'/>
      <xs:any  namespace='##other'
               processContents='skip'/>
    </xs:choice>
    <xs:attribute  name='name'
                   type='xs:string'/>
    <xs:attribute  name='id'
                   type='xs:ID'/>
    <xs:anyAttribute  namespace='##other'
                      processContents='skip'/>
  </xs:complexType>
</xs:element>

<!--   ElementType introduces elements, and appears in Schema only   -->
<xs:element  name='ElementType'>
  <xs:complexType >
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:element  ref='xdr:AttributeType'/>
      <xs:element  ref='xdr:attribute'/>
      <xs:element  ref='xdr:element'/>
      <xs:element  ref='xdr:group'/>
      <xs:element  ref='xdr:datatype'/>
      <xs:element  ref='xdr:description'/>
      <xs:element  ref='xdr:extends'/>
      <xs:any  namespace='##other'
               processContents='skip'/>
    </xs:choice>
    <xs:attribute  name='name'
                   type='xs:ID'/>
    <xs:attribute  name='content'>
      <xs:simpleType >
        <xs:restriction  base='xs:string'>
          <xs:enumeration  value='empty'/>
          <xs:enumeration  value='textOnly'/>
          <xs:enumeration  value='eltOnly'/>
          <xs:enumeration  value='mixed'/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute  ref='xdr:model'/>
    <xs:attribute  ref='xdr:order'/>
    <xs:attributeGroup  ref='dt:typeAttributes'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>

<!--   AttributeType introduces attribute,
       and appears in Schema or ElementType   -->
<xs:element  name='AttributeType'>
  <xs:complexType >
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:element  ref='xdr:datatype'/>
      <xs:element  ref='xdr:description'/>
      <xs:any  namespace='##other'
               processContents='skip'/>
    </xs:choice>
    <xs:attribute  name='default'
                   type='xs:string'/>
```

```
    <xs:attribute  name='name'
                   type='xs:ID'/>
    <xs:attribute  ref='xdr:required'/>
    <xs:attributeGroup  ref='dt:typeAttributes'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>

<!--  attribute applies to ElementType and refers to known AttributeType  -->
<xs:element  name='attribute'>
  <xs:complexType >
    <xs:attribute  name='type'
                   type='xs:NMTOKEN'/>
    <xs:attribute  name='default'
                   type='xs:string'/>
    <xs:attribute  ref='xdr:required'/>
  </xs:complexType>
</xs:element>

<!--  element appears inside of ElementType or group
      and refers to ElementType  -->
<xs:element  name='element'>
  <xs:complexType >
    <xs:attribute  name='type'
                   type='xs:NMTOKEN'/>
    <xs:attributeGroup  ref='xdr:minmax'/>
  </xs:complexType>
</xs:element>

<!--  group appears inside of ElementType
      (can be nested into another group)
      and organizes its elements  -->
<xs:element  name='group'>
  <xs:complexType >
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:element  ref='xdr:description'/>
      <xs:element  ref='xdr:element'/>
      <xs:element  ref='xdr:group'/>
      <xs:any  namespace='##other'
               processContents='skip'/>
    </xs:choice>
    <xs:attribute  ref='xdr:order'/>
    <xs:attributeGroup  ref='xdr:minmax'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>

<!--  description is arbitrary text with or without XML,
      and is intended for the human reader  -->
<xs:element  name='description'>
  <xs:complexType  mixed='true'>
    <xs:choice  minOccurs='0'
                maxOccurs='unbounded'>
      <xs:any  processContents='skip'
               namespace='##any'/>
    </xs:choice>
  </xs:complexType>
</xs:element>

<!--  refers to Data types, as defined in dt: namespace  -->
<xs:element  name='datatype'>
  <xs:complexType  mixed='true'>
    <xs:attributeGroup  ref='dt:typeAttributes'/>
    <xs:anyAttribute  namespace='##other'/>
  </xs:complexType>
</xs:element>

<xs:element  name='extends'>
```

```
      <xs:complexType >
        <xs:attribute  name='type'
                       type='xs:NMTOKEN'/>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

6 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Office SharePoint Portal Server 2003
- Microsoft Office SharePoint Server 2007
- Microsoft SharePoint Server 2010
- Microsoft SharePoint Server 2013
- Windows SharePoint Services 2.0
- Windows SharePoint Services 3.0
- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016
- Microsoft SharePoint Server 2019
- Microsoft SharePoint Server Subscription Edition

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

7 Change Tracking

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

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

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

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

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

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

Section	Description	Revision class
6 Appendix B: Product Behavior	Updated list of supported products.	Major

8 Index

X