[MS-XSSK-Diff]:

XML Serialization of Synchronization Knowledge

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~~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. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, ~~email~~e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
~~07~~7/13/2009	0.1	Major	Initial availability.
~~08/07~~8/7/2009	0.2	Minor	~~Updated~~Clarified the meaning of the technical content.
11/~~06~~6/2009	0.2.1	Editorial	~~Revised~~Changed language and ~~edited~~formatting in the technical content.
~~03/05~~3/5/2010	0.3	Minor	~~Updated~~Clarified the meaning of the technical content.
~~04~~4/21/2010	0.3.1	Editorial	~~Revised~~Changed language and ~~edited~~formatting in the technical content.
~~06/04~~6/4/2010	0.3.2	Editorial	~~Revised~~Changed language and ~~edited~~formatting in the technical content.
~~09/03~~9/3/2010	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~02/09~~2/9/2011	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~07/07~~7/7/2011	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
11/~~03~~3/2011	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~01~~1/19/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~02~~2/23/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~03~~3/27/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~05~~5/24/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~06~~6/29/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~07~~7/16/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
10/~~08~~8/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
10/23/2012	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~03~~3/26/2013	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
~~06~~6/11/2013	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
~~08/08~~8/8/2013	0.3.2	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
12/~~05~~5/2013	1.0	Major	~~Significantly changed~~Updated and revised the technical content.
~~02~~2/11/2014	2.0	Major	~~Significantly changed~~Updated and revised the technical content.
~~05~~5/20/2014	2.0	~~No change~~None	No changes to the meaning, language, or formatting of the technical content.
5/10/2016	2.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 ~~1~~ Introduction

The XML Serialization of Synchronization Knowledge protocol provides the structure of knowledge **XML serialization**. This type of serialization depends on the use of prescribed formatting for the data "knowledge", which represents a metadata structure that can be used to synchronize two or more sets of data as efficiently as possible. The knowledge can be represented as an XML file that conforms to its **XML** ~~Schema~~**schema**.

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

1.1 ~~1.1~~ Glossary

~~The~~ This document uses the following terms ~~are specific to this document~~:

attribute: A characteristic of some object or entity, typically encoded as a name/value pair.

change unit identifier: For tabular data, the identifier for a particular column.

change unit override: Part of the knowledge that associates a particular item identifier/change unit identifier pair with the clock vector.

clock vector: A list of synchronization versions that is ordered by the replica key.

data synchronization: The process of making two or more sets of data contain exactly the same items.

item identifier: For tabular data, the identifier for a particular row.

item override: Part of the knowledge that associates a pair comprising a particular item identifier and any change unit identifier with the clock vector, given that the knowledge does not contain a change unit override covering that pair.

range override: Part of the knowledge that associates all item identifiers in the closed range between two item identifiers (called lower-bound identifier and upper-bound identifier) paired with any change unit identifier, given that the knowledge does not contain a change unit override and item override covering that pair.

replica: A set of data together with associated synchronization metadata.

replica identifier: A unique identifier for a particular replica.

replica keymap: A data structure that provides one-to-one mapping between replica identifiers and unsigned integers, which are called replica keys.

scope clock vector: Part of the knowledge that associates any item identifier/change unit identifier pair with the clock vector, given that the knowledge does not contain a change unit override, item override, or range override covering that pair.

serialization: A mechanism by which an application converts an object into an XML representation.

synchronization metadata: Additional data associated with the ~~user's~~user's data for the purpose of enabling data synchronization.

synchronization version: Synchronization metadata associated with a particular item identifier/change unit identifier pair. The version consists of the replica key and an unsigned number, which is called a tick count.

tick count: A single per-replica integer associated with and maintained by a particular replica. A ~~replica's~~replica's tick count grows whenever the replica makes changes.

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

XML attribute: A name/value pair, separated by an equal sign (=) and included in a tagged element, that modifies features of an element. All XML attribute values are stored as strings enclosed in quotation marks.

XML document: A document object that is well formed, as described in [XML10/5], and might be valid. An XML document has a logical structure that is composed of declarations, elements, comments, character references, and processing instructions. It also has a physical structure that is composed of entities, starting with the root, or document, entity.

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 ~~described~~defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

~~1.2~~ References

1.2 ~~References~~

Links to a document in the Microsoft Open Specifications ~~documentation do not include a publishing year because links are to the latest~~library point to the correct section in the most recently published version of the referenced document. However, because individual documents~~, which~~ in the library are not updated ~~frequently. References to other documents include a publishing year when one is available.~~at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

~~1.1.1~~1.2.1 ~~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.

[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

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

~~[XML] See [XML10/5].~~

[XML10/5] Bray, T., Paoli, J., Sperberg-McQueen, C.M., et al., Eds., "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation, November 2008, http://www.w3.org/TR/2008/REC-xml-20081126/

~~1.1.2~~1.2.2 ~~1.2.2~~ Informative References

None.

~~1.2~~1.3 ~~1.3~~ Overview

The knowledge data structure is applicable to the scenarios that involve synchronization of two or more sets of data. In this document, the term **replica** is used to identify such sets of data together with the associated **synchronization metadata**. Each replica is uniquely identified by a **replica identifier**.

A **replica keymap** provides one-to-one mapping between the replica identifiers and unsigned integers. In this document, the term replica key is used to specify the concrete unsigned integer associated with the particular replica identifier. Every replica maintains the replica keymap, which contains entries for this replica and all replicas that this replica is ever synchronized with.

The data in every replica is represented in tabular form. Each row of data is uniquely identified by an **item identifier**. Each column of data is uniquely identified by a **change unit identifier**.

A set of all possible item identifiers is completely ordered.

Each element of the replica's data set, identified by an item identifier, change unit identifier pair, has a **synchronization version** associated with it. The synchronization version is a pair of replica key and unsigned integer that will be referred to as **tick count**. For every synchronization version, the replica key has a corresponding entry in the replica keymap attached to the replica.

In this document, the term **clock vector** is used to specify a list of zero or more sync versions, ordered by the replica key. Every replica key is present in the clock vector no more than once.

The knowledge data structure provides a mechanism to associate every possible item identifier, change unit identifier pair with a clock vector. The structure consists of the following parts:

- ~~•~~ Zero or more **change unit overrides**. Each change unit override is identified by a unique item identifier, change unit identifier pair and has a clock vector associated with it.
- ~~•~~ Zero or more **item overrides**. Each item override is identified by a unique item identifier and has a clock vector associated with it.
- ~~•~~ Zero or more **range overrides**. Each range override is identified by a (lower bound identifier, upper bound identifier) pair of item identifiers and has a clock vector associated with it. All ranges in the knowledge are not overlapping. That is, for any two ranges, the lower range identifier of one of them is always greater than upper range identifier of the other one.
- ~~•~~ A clock vector, which will be referred to as a **scope clock vector**.

The algorithm that finds a clock vector for a particular item identifier, change unit identifier pair is one of the following:

1. ~~1.~~ If the knowledge has a change unit override that is identified by the item identifier, change unit identifier pair equal to the one item identifier, change unit identifier in question, the answer is the clock vector associated with this change unit override.
2. ~~2.~~ Otherwise, if the knowledge has an item override that is identified by the item identifier equal to the item identifier in question, the answer is the clock vector associated with this item override.

3. ~~3.~~ Otherwise, if the knowledge has a range override, such as if its lower bound identifier is smaller than or equal to the item identifier in question, or its upper bound identifier is greater than or equal to the item identifier in question, the answer is the clock vector associated with this range override.

4. ~~4.~~ Otherwise, the answer is the scope clock vector.

In the data synchronization scenarios, the crucial question is whether the particular instance of the knowledge data structure covers the synchronization version associated with the specified item identifier, change unit identifier pair. To answer that question, do the following:

- ~~•~~ Find the clock vector associated with the specified item identifier, change unit identifier pair, as described earlier in this section.
- ~~•~~ If the clock vector doesn't have a synchronization version with a replica key equal to the replica key provided with the synchronization version in question, the answer is "not covered."
- ~~•~~ Otherwise, compare the tick count of the synchronization version in the clock vector with the one from the synchronization version that is provided. If the former tick count is greater than or equal to the latter one, the answer is "covered"; if it is smaller than the provided tick count, the answer is "not covered."

~~1.3~~1.4 ~~1.4~~ Relationship to Protocols and Other Structures

The knowledge data structure is not related to any other structures.

~~1.4~~1.5 ~~1.5~~ Applicability Statement

The knowledge data structure can be used for **data synchronization** scenarios in which the data is represented in tabular form. It serves to convey state to other replicas that use the knowledge data structure.

~~1.5~~1.6 ~~1.6~~ Versioning and Localization

This document covers versioning issues in the following areas:

- ~~•~~ Structure versions: This document specifies version 1 of all structures defined herein.

There are no localization-dependent structures in knowledge XML serialization.

~~1.6~~1.7 ~~1.7~~ Vendor-Extensible Fields

None.

2 ~~2~~ Structures

2.1 ~~2.1~~ XML Serialization of Synchronization Knowledge

This is a valid instance of an **XML document** as specified in [XML10/5].

2.1.1 ~~2.1.1~~ XML Namespace

All **XML elements** and **XML attributes** MUST be in the following **XML namespace**:

http://schemas.microsoft.com/2008/03/sync/

In addition, the root XML element of the document MUST define an empty XML namespace prefix with the preceding namespace. All child XML elements MUST NOT use nonempty XML namespace prefixes.

2.1.2 ~~2.1.2~~ Identifier Format

Replica identifiers, item identifiers, and change unit identifiers MUST be byte sequences of either variable length or fixed length.

Variable-length replica identifiers MUST begin with the 2-byte prefix, which MUST specify the length of the replica identifier, with the prefix included as an unsigned 16-bit integer.

The set of the item identifiers is completely sorted. Sorting is performed in dictionary (alphabetical) order on the sequence of bytes that constitutes the item identifier. For variable-length item identifiers, the 2-byte prefix is skipped during the sort.

2.1.3 ~~2.1.3~~ syncKnowledge

The **syncKnowledge** element is the root XML element of the Microsoft SQL Server XML Serialization Format of Knowledge. Its child elements are described in the following table.

Child element	Description
idFormatGroup	Describes the format for the replica identifiers, item identifiers, and change unit identifiers that are used in this knowledge instance.
replicaKeyMap	Specifies the replica keymap associated with this knowledge instance.
clockVector	Specifies the scope clock vector of this knowledge instance.
itemOverrides	Specifies the list of item overrides in this knowledge instance.
changeUnitOverrides	Specifies the list of change unit overrides in this knowledge instance.
rangeOverrides	Specifies the list of range overrides in this knowledge instance.

```
<xsd:element name="syncKnowledge">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="idFormatGroup" minOccurs="1" maxOccurs="1" />
            <xsd:element name="replicaKeyMap" minOccurs="1" maxOccurs="1" />
            <xsd:element name="clockVector" type="sync:clockVectorType" minOccurs="1"
maxOccurs="1" />

            <xsd:element name="itemOverrides" minOccurs="0" maxOccurs="1" />
            <xsd:element name="changeUnitOverrides" minOccurs="0" maxOccurs="1" />
```

```
            <xsd:element name="rangeOverrides" minOccurs="0" maxOccurs="1" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element
```

2.1.4 ~~2.1.4~~ idFormatGroup

The **idFormatGroup** element groups the descriptions of the format of replica identifiers, item identifiers, and change unit identifiers used in this XML document.

Child elements	Description
replicaIdFormat	Describes the format for replica identifiers.
itemIdFormat	Describes the format for item identifiers.
changeUnitIdFormat	Describes the format for change unit identifiers.

Parent element
syncKnowledge

```
<xsd:element name="idFormatGroup">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="replicaIdFormat" minOccurs="1" maxOccurs="1" />
            <xsd:element name="itemIdFormat" minOccurs="1" maxOccurs="1" />
            <xsd:element name="changeUnitIdFormat" minOccurs="1" maxOccurs="1" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

2.1.5 ~~2.1.5~~ replicaIdFormat

The **replicaIdFormat** element describes the format of the replica identifiers used in this XML document.

Parent element
idFormatGroup

The **replicaIdFormat** element has the following **attributes**:

isVariable: If the value of the attribute is "true", all replica identifiers in the document are of variable length, as specified in section 2.1.2. If the value of the attribute is "false", all replica identifiers in the document are of fixed length, as specified in section 2.1.2.

maxLength: For fixed-length replica identifiers, this attribute specifies the length of the identifiers; for variable-length replica identifiers, it specifies the maximum length of the replica identifier, including the 2-byte prefix.

For fixed-length replica identifiers, the value of the **maxLength** attribute MUST be 1 or greater. For variable-length replica identifiers, the value of this attribute MUST be 3 or greater.

```
<xsd:element name="replicaIdFormat">
    <xsd:complexType>
```

```
            <xsd:attribute name="isVariable" type="xsd:boolean" use="required" />
            <xsd:attribute name="maxLength" type="xsd:unsignedInt" use="required" />
        </xsd:complexType>
    </xsd:element
```

2.1.6 ~~2.1.6~~ itemIdFormat

The **itemIdFormat** element describes the format of the item identifiers used in this XML document.

Parent element
idFormatGroup

The **itemIdFormat** element has the following attributes:

isVariable: If the value of the attribute is "true", all item identifiers in the document are of variable length, as specified in section 2.1.2.

If the value of the attribute is "false", all item identifiers in the document are of fixed length, as specified in section 2.1.2.

maxLength: For fixed-length item identifiers, this attribute specifies the length of the identifiers; for variable-length item identifiers, it specifies the maximum length of the item identifier, including the 2-byte prefix.

For fixed-length item identifiers, the value of the **maxLength** attribute MUST be 1 or greater. For variable-length item identifiers, the value of the **maxLength** attribute MUST be 3 or greater.

```
<xsd:element name=“=”itemIdFormat“>”>
    <xsd:complexType>
        <xsd:attribute name=“=”isVariable“” type=“=”xsd:boolean“” use=“=”required“” />
        <xsd:attribute name=“=”maxLength“” type=“=”xsd:unsignedInt“” use=“=”required“” />
    </xsd:complexType>
</xsd:element>
```

2.1.7 ~~2.1.7~~ changeUnitIdFormat

The **changeUnitIdFormat** element describes the format of the change unit identifiers used in this XML document.

Parent element
idFormatGroup

The **changeUnitIdFormat** element has the following attributes:

isVariable: If the value of the attribute is “”true“”, all change unit identifiers in the document are of variable length, as specified in section 2.1.2.

If the value of the attribute is “”false“”, all change unit identifiers in the document are of fixed length, as specified in section 2.1.2.

maxLength: For fixed-length change unit identifiers, this attribute specifies the length of the identifiers; for variable-length change unit identifiers, it specifies the maximum length of the change unit identifier, including the 2-byte prefix.

For fixed-length change unit identifiers, the value of the **maxLength** attribute MUST be 1 or greater. For variable-length change unit identifiers, the value of the **maxLength** attribute MUST be 3 or greater.

```
<xsd:element name="changeUnitIdFormat" >
    <xsd:complexType>
        <xsd:attribute name="isVariable" type="xsd:boolean" use="required" />
        <xsd:attribute name="maxLength" type="xsd:unsignedInt" use="required" />
    </xsd:complexType>
</xsd:element>
```

2.1.8 ~~2.1.8~~ replicaKeyMap

The **replicaKeyMap** element specifies the replica keymap associated with a particular knowledge instance. The **replicaKey** attribute values in the **replicaKeyMap** child elements MUST be consecutive unsigned integers, starting with 0 (zero).

Child elements	Description
replicaKeyMapEntry	Individual entry of the replica keymap.

Parent element
syncKnowledge

```
<xsd:element name="replicaKeyMap" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="replicaKeyMapEntry" minOccurs="1"
maxOccurs="unbounded"/>"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

2.1.9 ~~2.1.9~~ replicaKeyMapEntry

The **replicaKeyMapEntry** element specifies a single entry in a replica keymap.

Parent element
replicaKeyMap

The **replicaKeyMapEntry** element has the following attributes:

replicaId: Specifies the replica identifier in base64 encoding as specified in [RFC4648]. The replica identifier MUST conform to the replica identifier format specified by the replicaIdFormat element.

replicaKey: Specifies an unsigned integer that specifies the replica key associated with the replica identifier.

```
<xsd:element name="replicaKeyMapEntry" >
    <xsd:complexType>
        <xsd:attribute name="replicaId" type="xsd:base64binary" use="required" />
        <xsd:attribute name="replicaKey" type="xsd:unsignedInt" use="required" />
    </xsd:complexType>
```

```
</xsd:element>
```

2.1.10 ~~2.1.10~~ clockVector

The **clockVector** element specifies the scope clock vector of a particular knowledge instance. The type of this XML element is ClockVectorType.

Parent element
syncKnowledge

```
<xsd:element name="clockVector" type="sync:clockVectorType" />
```

2.1.11 ~~2.1.11~~ itemOverrides

The **itemOverrides** element specifies a set of the item overrides in a particular knowledge instance. The values of the **itemId** attribute in the itemOverride child elements MUST be distinct.

Child element	Description
itemOverride	An individual item override.

Parent element
syncKnowledge

```
<xsd:element name="itemOverrides" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="itemOverride" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

2.1.12 ~~2.1.12~~ itemOverride

The **itemOverride** element specifies a single item override.

Child element	Description
clockVector	The clock vector associated with a specified item override. The type of this element is ClockVectorType.

Parent element
itemOverrides

The **itemOverride** element has the following attribute:

itemId: The item identifier of the item override in base64 encoding, as specified in [RFC4648]. The item identifier MUST conform to the item identifier format, as described in section 2.1.6.

```
<xsd:element name="itemOverride" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="clockVector" minOccurs="1" maxOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="itemId" type="xsd:base64binary" use="required" />
    </xsd:complexType>
</xsd:element>
```

2.1.13 ~~2.1.13~~ changeUnitOverrides

The **changeUnitOverrides** element specifies a set of the change unit overrides in this knowledge instance. The pairs of **itemId** and **changeUnitId** attributes in the changeUnitOverrides child elements MUST be distinct.

Child element	Description
changeUnitOverride	Individual change unit override

Parent element
syncKnowledge

```
<xsd:element name="changeUnitOverrides" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="changeUnitOverride" minOccurs="0"
maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

2.1.14 ~~2.1.14~~ changeUnitOverride

The **changeUnitOverride** element specifies a single change unit override.

Child element	Description
clockVector	The clock vector associated with a specified change unit override. The type of this element is ClockVectorType.

Parent element
changeUnitOverrides

The **changeUnitOverride** element has the following attributes:

itemId: Specifies the item identifier of the change unit override in base64 encoding, as specified in [RFC4648]. The item identifier MUST conform to the item identifier format specified by the itemIdFormat element.

changeUnitId: Specifies the change unit identifier of the change unit override in base64 encoding, as specified in [RFC4648]. The change unit identifier MUST conform to the change unit identifier format specified by the changeUnitIdFormat element.

```
<xsd:element name="changeUnitOverride" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="clockVector" minOccurs="1" maxOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="itemId" type="xsd:base64binary" use="required" />
        <xsd:attribute name="changeUnitId" type="xsd:base64binary" use="required" />
    </xsd:complexType>
</xsd:element>
```

2.1.15 ~~2.1.15~~ rangeOverrides

The **rangeOverrides** element specifies a set of the range overrides in a particular knowledge instance. The individual range overrides MUST NOT overlap.

Child element	Description
rangeOverride	An individual range override.

Parent element
syncKnowledge

```
<xsd:element name="rangeOverrides" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="rangeOverride" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

2.1.16 ~~2.1.16~~ rangeOverride

The **rangeOverride** element specifies a single range override. The value of the **closedUpperBound** attribute MUST be greater than or equal to the value of the **closedLowerBound** attribute, according to the sorting order described in section 2.1.2.

Child element	Description
clockVector	The clock vector associated with a specified range override. The type of this element is ClockVectorType.

Parent element
rangeOverrides

The **rangeOverride** element has the following attributes:

closedLowerBound: Specifies the lower bound identifier of the range override in base64 encoding, as specified in [RFC4648]. The lower bound identifier attribute value MUST conform to the item identifier format specified by the itemIdFormat element.

closedUpperBound: Specifies the upper bound identifier of the range override in base64 encoding, as specified in [RFC4648]. The upper bound identifier attribute value MUST conform to the item identifier format specified by the **itemIdFormat** element.

```
<xsd:element name="rangeOverride" >
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="clockVector" minOccurs="1" maxOccurs="1" />
        </xsd:sequence>
        <xsd:attribute name="closedLowerBound" type="xsd:base64binary"
use="required" />
        <xsd:attribute name="closedUpperBound" type="xsd:base64binary"
use="required" />
    </xsd:complexType>
</xsd:element>
```

2.1.17 ~~2.1.17~~ clockVectorElement

The **clockVectorElement** element specifies a single synchronization version. The following table specifies all possible parent elements of the **clockVectorElement** element.

Parent element	Description
clockVector	A child of the ~~syncKnowledgeelement~~syncKnowledge element.
clockVector	A child of the ~~itemOverrideelement~~itemOverride element.
clockVector	A child of the ~~changeUnitOverrideelement~~changeUnitOverride element.
clockVector	A child of the ~~rangeOverrideelement~~rangeOverride element.

The **clockVectorElement** element has the following attributes:

replicaKey: Specifies the replica key of a specified synchronization version. The value of this attribute MUST have the corresponding replicaKeyMapEntry element.

tickCount: Specifies the tick count of the specified synchronization version.

```
<xsd:element name="clockVectorElement" >
    <xsd:complexType>
        <xsd:attribute name="replicaKey" type="xsd:unsignedInt" use="required" />
        <xsd:attribute name="tickCount" type="xsd:unsignedLong" use="required" />
    </xsd:complexType>
</xsd:element>
```

2.1.18 ~~2.1.18~~ ClockVectorType

The **ClockVectorType** type is a custom XML type used to describe the clock vectors that are referred to in sections 2.1.10, 2.1.11, 2.1.12, 2.1.13, 2.1.14, 2.1.15, and 2.1.16. The clockVectorElement child elements MUST be sorted by the value of their **replicaKey** attribute. The **replicaKey** attribute values of the child elements MUST be distinct within the single instance of the **ClockVectorType** type.

Child element	Description
clockVectorElement	An individual clock vector element.

```
<xsd:complexType name=”="clockVectorType”" >
    <xsd:sequence>
        <xsd:element name=”="clockVectorElement”" minOccurs=”="0”" maxOccurs=”="unbounded />
    </xsd:sequence>
</xsd:element>
```

3 ~~3~~ Structure Examples

The following is an example of the knowledge data structure serialized in the XML format.

Knowledge in this example contains only a scope clock vector; it does not contain any range overrides, item overrides, or change unit overrides.

```
<syncKnowledge xmlns="http://schemas.microsoft.com/2008/03/sync/"
xmlns:sync="http://schemas.microsoft.com/2008/03/sync/">
  <idFormatGroup>
    <replicaIdFormat sync:isVariable="false" sync:maxLength="16" />
    <itemIdFormat sync:isVariable="false" sync:maxLength="24"/>
    <changeUnitIdFormat sync:isVariable="false" sync:maxLength="1" />
  </idFormatGroup>
  <replicaKeyMap>
    <replicaKeyMapEntry sync:replicaId="zaun9erpTKCRxvHzTngj4w==" sync:replicaKey="0" />
    <replicaKeyMapEntry sync:replicaId="71J30mgqQ6K/wjnSqEIKYg==" sync:replicaKey="1" />
    <replicaKeyMapEntry sync:replicaId="nQh3j4ExQluKail5dm1YaA==" sync:replicaKey="2" />
  </replicaKeyMap>
  <clockVector>
    <clockVectorElement sync:replicaKey="0" sync:tickCount="10" />
    <clockVectorElement sync:replicaKey="2" sync:tickCount="20" />
  </clockVector>
</syncKnowledge>
```

The following is another example of the knowledge data structure serialized in the XML format.

Knowledge in this example contains a scope clock vector, one range override, two item overrides, and three change unit overrides.

```
<syncKnowledge xmlns="http://schemas.microsoft.com/2008/03/sync/"
xmlns:sync="http://schemas.microsoft.com/2008/03/sync/">
  <idFormatGroup>
    <replicaIdFormat sync:isVariable="false" sync:maxLength="16" />
    <itemIdFormat sync:isVariable="false" sync:maxLength="24"/>
    <changeUnitIdFormat sync:isVariable="false" sync:maxLength="1" />
  </idFormatGroup>
  <replicaKeyMap>
    <replicaKeyMapEntry sync:replicaId="zaun9erpTKCRxvHzTngj4w==" sync:replicaKey="0" />
    <replicaKeyMapEntry sync:replicaId="71J30mgqQ6K/wjnSqEIKYg==" sync:replicaKey="1" />
  </replicaKeyMap>
  <clockVector>
    <clockVectorElement sync:replicaKey="0" sync:tickCount="10" />
    <clockVectorElement sync:replicaKey="2" sync:tickCount="20" />
  </clockVector>
  <itemOverrides>
    <itemOverride sync:itemId="="AAAAAAAAAAARVFBb7zBEmJCiSPPioeuL=>">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="5" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="5" />
      </clockVector>
    </itemOverride>
    <itemOverride sync:itemId="="AAAAAAAAB9AiNPqZB/pB7p3TXWo3VrZ0=>">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="6" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="4" />
      </clockVector>
    </itemOverride>
  </itemOverrides>
  <changeUnitOverrides>
    <changeUnitOverride sync:itemId="="AAAAAAAAB9Ddesz1YFtE9r8QN7JEg4ZQ="
sync:changeUnitId="="FA==">==">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="15" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="2" />
```

```
      </clockVector>
    </changeUnitOverride>
    <changeUnitOverride sync:itemId="="AAAAAAAAD6AXfz97akZByL01Lj96GlFL"
sync:changeUnitId="="KA==">==">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="16" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="12" />
      </clockVector>
    </changeUnitOverride>
    <changeUnitOverride sync:itemId="="AAAAAAAAAADUaRgYm21PjIEqSfh+SI1/"/"
sync:changeUnitId="="AA==">==">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="17" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="22" />
      </clockVector>
    </changeUnitOverride>
  </changeUnitOverrides>
  <rangeOverrides>
    <rangeOverride sync:closedLowerBound="="AAAAAAAAAGTIXlJ1VXBP2Kqk6mGiuwvL"
sync:closedUpperBound="="AAAAAAAAAMjIXlJ1VXBP2Kqk6mGiuwvL">">
      <clockVector>
        <clockVectorElement sync:replicaKey="="0" sync:tickCount="="18" />
        <clockVectorElement sync:replicaKey="="1" sync:tickCount="="28" />
      </clockVector>
    </rangeOverride>
  </rangeOverrides>
</syncKnowledge>
```

4 ~~4~~ Security Considerations

None.

5 ~~5~~ Appendix A: Full XML Schema

For ease of implementation, the following is the full XML ~~Schema~~schema for this protocol.

```
<?xml version='1.0'?>
<xs:schema targetNamespace='http://schemas.microsoft.com/2008/03/sync/'
           xmlns:sync='http://schemas.microsoft.com/2008/03/sync/'
           xmlns:xs='http://www.w3.org/2001/XMLSchema'
           attributeFormDefault='qualified'
           elementFormDefault='qualified'>
    <xs:element name='syncKnowledge'>
        <xs:complexType>
            <xs:sequence>               <xs:element name='idFormatGroup' minOccurs='1'
maxOccurs='1'>
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name='replicaIdFormat' minOccurs='1' maxOccurs='1'>
                                <xs:complexType>
                                    <xs:attribute name='isVariable' type='xs:boolean'
use='required' />
                                    <xs:attribute name='maxLength' type='xs:unsignedInt'
use='required' />
                                </xs:complexType>
                            </xs:element>
                            <xs:element name='itemIdFormat' minOccurs='1' maxOccurs='1'>
                                <xs:complexType>
                                    <xs:attribute name='isVariable' type='xs:boolean'
use='required' />
                                    <xs:attribute name='maxLength' type='xs:unsignedInt'
use='required' />
                                </xs:complexType>
                            </xs:element>
                            <xs:element name='changeUnitIdFormat' minOccurs='1'
maxOccurs='1'>
                                <xs:complexType>
                                    <xs:attribute name='isVariable' type='xs:boolean'
use='required' />
                                    <xs:attribute name='maxLength' type='xs:unsignedInt'
use='required' />
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <!-- -->
                <xs:element name='replicaKeyMap' minOccurs='1' maxOccurs='1'>
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name='replicaKeyMapEntry' minOccurs='1'
maxOccurs='unbounded'>
                                <xs:complexType>
                                    <xs:attribute name='replicaId' type='xs:base64Binary'
use='required' />
                                    <xs:attribute name='replicaKey' type='xs:unsignedInt'
use='required' />
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <!-- -->
                <xs:element name='clockVector' type='sync:clockVectorType' minOccurs='1'
maxOccurs='1' />
                <!-- -->
                <xs:element name='itemOverrides' minOccurs='0' maxOccurs='1'>
                    <xs:complexType>
                        <xs:sequence>
```

```
                            <xs:element name='itemOverride' minOccurs='0'
maxOccurs='unbounded'>
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name='clockVector'
type='sync:clockVectorType' minOccurs='1' maxOccurs='1' />
                                    </xs:sequence>
                                <xs:attribute name='itemId' type='xs:base64Binary'
use='required' />
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <!-- -->
                <xs:element name='changeUnitOverrides' minOccurs='0' maxOccurs='1'>
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name='changeUnitOverride' minOccurs='0'
maxOccurs='unbounded'>
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name='clockVector'
type='sync:clockVectorType' minOccurs='1' maxOccurs='1' />
                                    </xs:sequence>
                                    <xs:attribute name='itemId' type='xs:base64Binary'
use='required' />
                                    <xs:attribute name='changeUnitId' type='xs:base64Binary'
use='required' />
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <!-- -->
                <xs:element name='rangeOverrides' minOccurs='0' maxOccurs='1'>
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name='rangeOverride' minOccurs='0'
maxOccurs='unbounded'>
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name='clockVector'
type='sync:clockVectorType' minOccurs='1' maxOccurs='1' />
                                    </xs:sequence>
                                    <xs:attribute name='closedLowerBound'
type='xs:base64Binary' use='required' />
                                    <xs:attribute name='closedUpperBound'
type='xs:base64Binary' use='required' />
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <!-- -->
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:complexType name='clockVectorType'>
        <xs:sequence>
            <xs:element name='clockVectorElement' minOccurs='0' maxOccurs='unbounded'>
                <xs:complexType>
                    <xs:attribute name='replicaKey' type='xs:unsignedInt' use='required' />
                    <xs:attribute name='tickCount' type='xs:unsignedLong' use='required' />
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
```

```
</xs:schema>
```

6 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 released service packs.

- Microsoft Office Groove 2010
- OneDrive for Business
- Microsoft SharePoint 2010
- Microsoft SharePoint 2013

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.

7 ~~7~~ Change Tracking

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

8 ~~8~~ Index

S

T

V

X