[MS-CSOM]:

SharePoint Client Query Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
7/13/2009	0.1	Major	Initial Availability
8/28/2009	0.2	Editorial	Revised and edited the technical content
11/6/2009	0.3	Editorial	Revised and edited the technical content
2/19/2010	1.0	Editorial	Revised and edited the technical content
3/31/2010	1.01	Editorial	Revised and edited the technical content
4/30/2010	1.02	Minor	Updated the technical content
6/7/2010	1.03	Editorial	Revised and edited the technical content
6/29/2010	1.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	1.04	None	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	1.04	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	1.04	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.04	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	1.04	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	1.04	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.0	Major	Significantly changed the technical content.
4/11/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	2.0	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.1	Minor	Clarified the meaning of the technical content.
2/11/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.1	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.2	Minor	Clarified the meaning of the technical content.

Date	Revision History	Revision Class	Comments
7/31/2014	2.3	Minor	Clarified the meaning of the technical content.
10/30/2014	2.4	Minor	Clarified the meaning of the technical content.
2/26/2016	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The SharePoint Client Query Protocol allows a protocol client to call methods and access data on a protocol server. The actions to be executed are sent by a protocol client as part of a request, and the results are returned by a protocol server as part of a response.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

child item query: A set of filters and options for retrieving child objects in a collection of **CSOM Objects**.

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).

CSOM action: An individual method, property, or lookup operation that is performed by a protocol server in a request.

CSOM action list: A sequential list of **CSOM actions** that are defined in a CSOM request to be executed by a protocol server.

CSOM action response structure: A JavaScript Object Notation (JSON) data structure that contains an array of individual results from a protocol server in response to actions that were submitted by a protocol client.

CSOM array: An ordered collection of values that can be used in an **XML** request or **JSON** response text. The values are identified by their position and their position is determined by a zero-based integer index.

CSOM binary: An array of 8-bit, unsigned integers that can be used in an **XML** request or as a string in **JSON** response text.

CSOM Boolean: A Boolean value that can be used in an **XML** request or **JSON** response text. A CSOM Boolean value is either "true" or "false".

CSOM Byte: An 8-bit, unsigned integer value that represents the BYTE type, as described in [MS-DTYP]. The range of CSOM Byte values is 0-255 and it has different representations, depending on whether it is used in an **XML** request or **JSON** response text.

CSOM Char: A **Unicode** character value that can be used in an **XML** request or as a string in **JSON** response text.

CSOM DateTime: An Int64 value that represents the number of 100-nanosecond time intervals that have elapsed since 12:00:00, January 1, 0001. It can be used in an **XML** request or as a string in **JSON** response text. The value can represent time intervals through 23:59:59.9999999, December 31, 9999. It can also specify whether a local, **UTC**, or no time zone applies.

CSOM dictionary: An object that contains an unordered collection of key/value pairs that can be used in an **XML** request or **JSON** response text. Each key in a CSOM dictionary has a unique name.

CSOM Double: A 64-bit, double-precision, floating-point value, which is the DOUBLE type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM Double values is from "-1.79769313486232e308" to "1.79769313486232e308".

CSOM error: An object that contains information about an error that occurred on a protocol server when processing a request.

CSOM expando field: A field that stores data for an instance of a **CSOM Object** and is not defined explicitly in the corresponding **CSOM Object type**.

CSOM expression: A syntax that is used by protocol clients to express sets of actions to execute based on state data that is stored on a protocol server.

CSOM GUID: A **GUID**, as described in [MS-DTYP], that can be used in an **XML** request or as a string in **JSON** response text.

CSOM Int16: A 16-bit, signed integer value, which is the INT16 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM Int16 values is from "-32768" to "32767".

CSOM Int32: A 32-bit, signed integer value, which is the INT32 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM Int32 values is from "-2147483648" to "2147483647".

CSOM Int64: A 64-bit, signed integer value, which is the INT64 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM Int64 values is from "-9223372036854775808" to "9223372036854775807".

CSOM method: A procedure that is executed by a protocol server for a **CSOM Object**.

CSOM Object: An object that contains a set of members, which are named values and methods. It has a Unicode string value, which is referred to as a CSOM type name, that identifies its type.

CSOM Object type: A reference to a standard definition of methods, properties, and behavior for a logical object in the SharePoint Client-Side Object Model.

CSOM property: A representation of a field of data that is stored for a type of **CSOM Object**.

CSOM SByte: An 8-bit, signed integer value, which is the INT8 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM SByte values is from "-128" to "127".

CSOM Single: A 32-bit, single-precision, floating-point value, which is the FLOAT type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM Single values is from "-3.402823e38" to "3.402823e38".

CSOM String: A representation of text as a series of **Unicode** characters. It can be used in an **XML** request or **JSON** response text.

CSOM type: A predefined set of named values that enable a protocol client to access standard descriptions of exposed objects, members, and enumerations. A CSOM type can be a **CSOM Object type**, CSOM value object type, or CSOM enumeration.

CSOM type identifier: A **GUID** that is used to identify a **CSOM type**.

CSOM type name: A **Unicode** string that identifies the type of a **CSOM Object**.

CSOM UInt16: A 16-bit, unsigned integer value, which is the UINT16 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM UInt16 values is from "0" to "65535".

CSOM UInt32: A 32-bit, unsigned integer value, which is the UINT32 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM UInt32 values is from "0" to "4294967295".

CSOM UInt64: A 64-bit, unsigned integer value, which is the UINT64 type described in [MS-DTYP], that can be used in an **XML** request or as a number in **JSON** response text. The range of CSOM UInt64 values is from "0" to "18446744073709551615".

CSOM value object: An object that contains a set of named values, which are referred to as members. It has a Unicode string value, referred to as a CSOM type name, that identifies its type.

CSOM value object type: A CSOM type that contains a set of named values, which are referred to as members. It has type information, which is identified by a **Unicode** string, and is associated with a specific identifier, which is a **CSOM GUID**.

default scalar property set: A set of properties that are retrieved by default for an object. The properties map to fields in a storage schema.

expression: A combination of operators, symbols, constants, literal values, functions, names of fields or columns (2), controls, and properties that evaluates to a single value.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

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

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

JavaScript Object Notation (JSON): A text-based, data interchange format that is used to transmit structured data, typically in Asynchronous JavaScript + XML (AJAX) web applications, as described in [RFC4627]. The JSON format is based on the structure of ECMAScript (Jscript, JavaScript) objects.

Multipurpose Internet Mail Extensions (MIME): A set of extensions that redefines and expands support for various types of content in email messages, as described in [RFC2045], [RFC2046], and [RFC2047].

object identity: An optional, implementation-specific string that uniquely identifies a CSOM Object. It can be used in queries to retrieve a specific CSOM object.

object path: A string that is used to access namespaces, classes, objects, and instances. Each object on a system has a unique path that identifies it locally or over a network.

queryable expression: A syntax that is used by protocol clients to retrieve a set of **CSOM Objects** that meet a specific set of criteria, based on state data that is stored on a protocol server.

Request-URI: A URI in an **HTTP** request message, as described in [RFC2616].

Status-Code: A 3-digit integer result code in an HTTP response message, as described in [RFC2616].

Unicode: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

URL encode: The process of encoding characters that have reserved meanings for a Uniform Resource Locator (URL), as described in [RFC1738].

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

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

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

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

1.2 References

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

1.2.1 Normative References

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

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[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

[RFC2234] Crocker, D. and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997, http://www.ietf.org/rfc/rfc2234.txt

[RFC2387] Levinson, E., "The MIME Multipart/Related Content-type", RFC 2387, August 1998, http://www.rfc-editor.org/rfc/rfc2387.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006, http://www.ietf.org/rfc/rfc4627.txt

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

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

[XMLSCHEMA1] Thompson, H., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.V., Ed. and Malhotra, A., Ed., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

1.2.2 Informative References

[MS-CSOMSPT] Microsoft Corporation, "SharePoint Client-Side Object Model Protocol".

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

1.3 Overview

This protocol enables a protocol client to execute method calls in batches and to perform read/write operations on properties of logical objects or class types on a protocol server.

This protocol defines a system for locating instances of types, calling methods that are defined by those types, and performing read/write operations for properties of those types. This protocol defines two roles: protocol client and protocol server. A protocol client initiates communication by generating a list of actions. The protocol client then sends that list to the protocol server for processing. The protocol server executes the actions on the specified objects, and then returns a list of results for each of the specified actions.

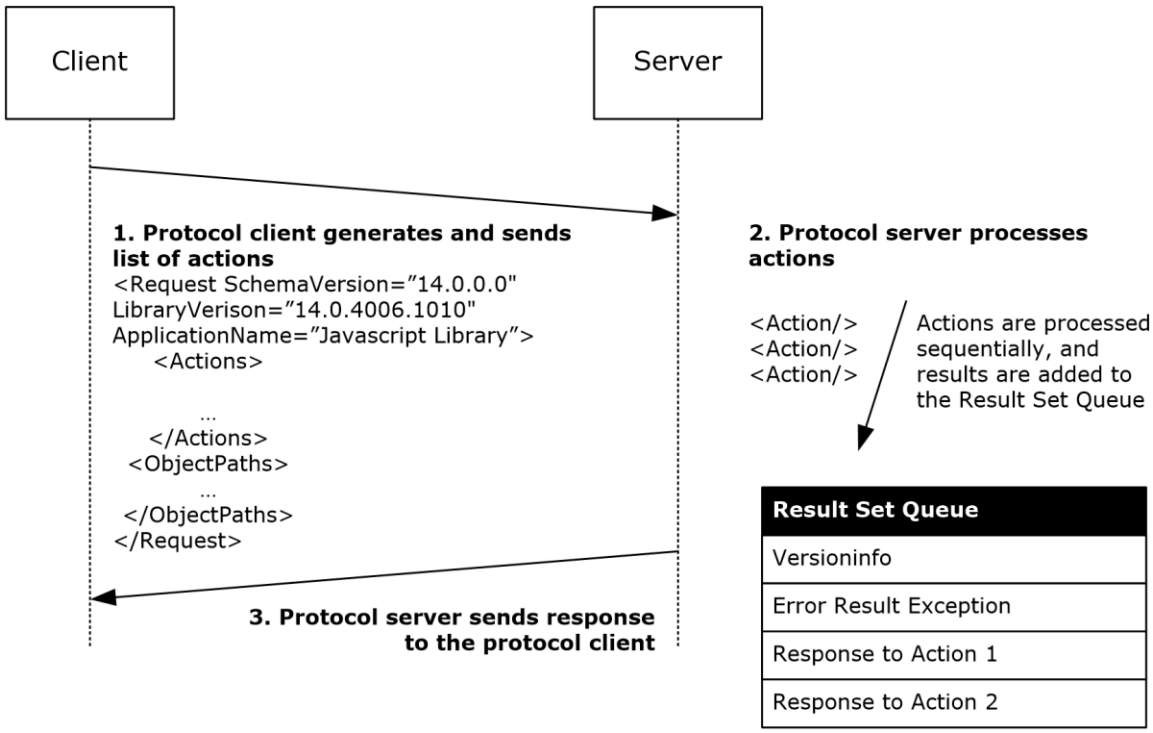

Figure 1: Overview of a request/response sequence

Actions can be performed against properties and methods of objects or class types. This protocol defines two types of properties and methods, scalar and object path. Scalar properties have values that are based on simple data types. Similarly, scalar methods have return values that are based on simple data types. Object path properties have values that are based on references to other logical objects. Similarly, object path methods return values that are based on references to other logical objects.

Actions can:

- Instantiate objects.
- Query the values of scalar properties for a specified object.
- Query the values of scalar properties for child objects of a collection object.
- Query the canonical identity of an object.
- Set the values of scalar properties.
- Call methods.
- Evaluate Boolean conditional expressions and execute a different set of actions based on evaluated values.
- Employ a try/catch/finally mechanism that executes different sets of actions, depending on whether an unhandled exception occurs.

The protocol server builds a queue of responses for each specified action. If the protocol server encounters an unhandled exception or executes all of the actions that are specified by the protocol client, the protocol server sends the appropriate response to the protocol client.

1.4 Relationship to Other Protocols

This protocol enables a protocol client to send a request that calls methods and accesses data on a protocol server, and then receive a corresponding set of results from the protocol server. This protocol depends on other structures and protocols to transport messages. Additional protocols, such as the SharePoint Client-Side Object Model protocol described in [MS-CSOMSPT], define the properties and methods that are exposed to protocol clients through this protocol. Applications are layered on top of this protocol, in combination with related protocols that define sets of logical types and related methods and properties.

The messages that are sent from the protocol client to the protocol server are encoded as **XML**. It transmits those messages by using **HTTP**, as described in [RFC2616], or **HTTPS**, as described in [RFC2818]. Responses from the protocol server are encoded by using **JavaScript Object Notation (JSON)**, as described in [RFC4627].

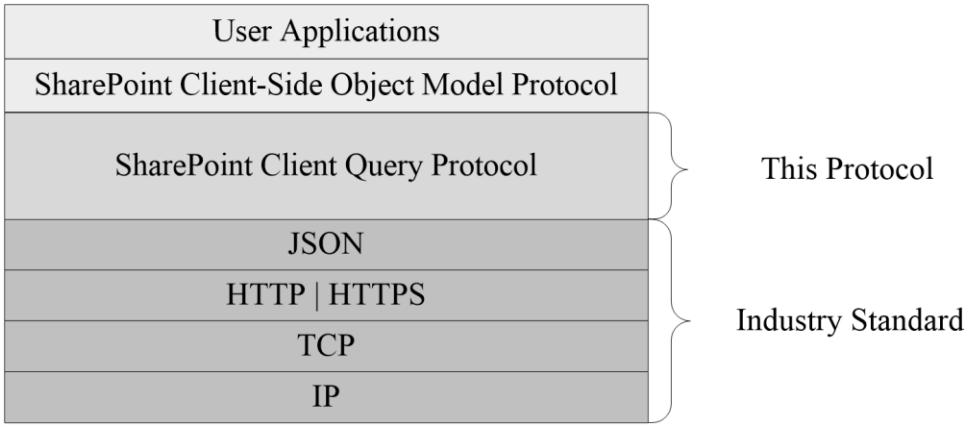

Figure 2: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a protocol server that is configured to listen for HTTP or HTTPS requests and a protocol client that knows the **Request-URI** of the protocol server.

This protocol additionally requires that the protocol client and protocol server agree to use the same version of types, methods, and properties. For more information about version validation, see section 1.7.

1.6 Applicability Statement

This protocol is useful for transferring information about object state and for performing method calls in a distributed environment. Due to the batch-processing aspects of this protocol, this protocol is well-suited for environments in which the time that is used to send messages from a protocol client to a protocol server is relatively high compared to the execution of the specified methods and properties.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol can use HTTP or HTTPS as a transport. For more information, see section 2.1.
- **Protocol Versions:** Currently, there are only two versions of this protocol. The version is specified by the **SchemaVersion** property and the value of that property is "14.0.0.0" or

"15.0.0.0". To interoperate, a protocol client includes the **SchemaVersion** property with a value of "14.0.0.0" or "15.0.0.0", as described in section 3.1.4.1.3.36, and a protocol server includes the **SchemaVersion** attribute with a value of "14.0.0.0" or "15.0.0.0" in its response, as described in section 3.1.4.1.8.1.5.

- **API Versions:** A protocol client uses the **LibraryVersion** attribute to specify the expected version of the API library. The protocol server can use the value of this attribute to indicate which library of types, methods, and properties is used.

1.8 Vendor-Extensible Fields

This protocol does not define any vendor-extensible fields. However, this protocol does not preclude implementers from adding custom HTTP headers, as specified in [RFC2616] section 4.2.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers MUST support HTTP. Protocol servers SHOULD additionally support HTTPS to help secure communications with protocol clients. Messages that are sent by a protocol client to a protocol server MUST be formatted as specified for **ProcessQueryIn** messages, as specified in section 3.1.4.1.1.1. Protocol clients MUST use the **POST** method to send messages to protocol servers. Protocol messages that are sent from the protocol server to the protocol client MUST be formatted as specified for **ProcessQueryOut** messages, as specified in section 3.1.4.1.1.2.

2.2 Message Syntax

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

The syntax of the definitions also uses JavaScript Object Notation (JSON), as defined in [RFC4627].

The JSON, **Augmented Backus-Naur Form (ABNF)** grammar, as specified in [RFC4627], for this protocol is as follows.

```
rfc4627-begin-array     = rfc4627-ws %x5B rfc4627-ws  ; [ left square bracket
rfc4627-begin-object    = rfc4627-ws %x7B rfc4627-ws  ; { left curly bracket
rfc4627-end-array       = rfc4627-ws %x5D rfc4627-ws  ; ] right square bracket
rfc4627-end-object      = rfc4627-ws %x7D rfc4627-ws  ; } right curly bracket
rfc4627-name-separator  = rfc4627-ws %x3A rfc4627-ws  ; : colon
rfc4627-value-separator = rfc4627-ws %x2C rfc4627-ws  ; , comma

rfc4627-ws = *(
                %x20 /              ; Space
                %x09 /              ; Horizontal tab
                %x0A /              ; Line feed or New line
                %x0D                ; Carriage return
              )

rfc4627-value = rfc4627-false / rfc4627-null / rfc4627-true / rfc4627-object / rfc4627-array
/ rfc4627-number / rfc4627-string
rfc4627-false = %x66.61.6c.73.65   ; false
rfc4627-null  = %x6e.75.6c.6c      ; null
rfc4627-true  = %x74.72.75.65      ; true

rfc4627-object = rfc4627-begin-object [ rfc4627-member *( rfc4627-value-separator rfc4627-
member ) ] rfc4627-end-object
rfc4627-member = rfc4627-string rfc4627-name-separator rfc4627-value

rfc4627-number = [ rfc4627-minus ] rfc4627-int [ rfc4627-frac ] [ rfc4627-exp ]
rfc4627-decimal-point = %x2E       ; .
rfc4627-digit1-9 = %x31-39         ; 1-9
rfc4627-e = %x65 / %x45            ; e E
rfc4627-exp = rfc4627-e [ rfc4627-minus / rfc4627-plus ] 1*DIGIT
rfc4627-frac = rfc4627-decimal-point 1*DIGIT
rfc4627-int = rfc4627-zero / ( rfc4627-digit1-9 *DIGIT )
rfc4627-minus = %x2D               ; -
rfc4627-plus = %x2B                ; +
rfc4627-zero = %x30                ; 0
rfc4627-array = rfc4627-begin-array [ rfc4627-value *( rfc4627-value-separator value ) ]
rfc4627-end-array

rfc4627-string = rfc4627-quotation-mark *rfc4627-char rfc4627-quotation-mark
rfc4627-char = rfc4627-unescaped /
               rfc4627-escape (
                    %x22 /          ; "    quotation mark  U+0022
```

```
                    %x5C /          ; \    reverse solidus U+005C
                    %x2F /          ; /    solidus         U+002F
                    %x62 /          ; b    backspace       U+0008
                    %x66 /          ; f    form feed       U+000C
                    %x6E /          ; n    line feed       U+000A
                    %x72 /          ; r    carriage return U+000D
                    %x74 /          ; t    tab             U+0009
                    %x75 4HEXDIG )  ; uXXXX                U+XXXX
rfc4627-escape = %x5C              ; \
rfc4627-quotation-mark = %x22      ; "
rfc4627-unescaped = %x20-21 / %x23-5B / %x5D-10FFFF
```

The following table summarizes the types that are defined in [RFC4627].

JSON value	ABNF grammar
JSON object	rfc4627-object ; ([RFC4627] section 2.2)
JSON array	rfc4627-array ; ([RFC4627] section 2.3)
JSON literal false	rfc4627-false ; ([RFC4627] section 2.1)
JSON literal true	rfc4627-true ; ([RFC4627] section 2.1)
JSON literal null	rfc4627-null ; ([RFC4627] section 2.1)
JSON string	rfc4627-string ; ([RFC4627] section 2.5)
JSON number	rfc4627-number ; ([RFC4627] section 2.4)

2.2.1 Namespaces

This specification defines and references various XML namespaces using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
mstns	http://schemas.microsoft.com/sharepoint/clientquery/2009	
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	

Prefix	Namespace URI	Reference
wsaw	http://www.w3.org/2006/05/addressing/wsdl	
tns	http://schemas.microsoft.com/sharepoint/soap/	
xsd	http://www.w3.org/2001/XMLSchema	
http	http://schemas.microsoft.com/ws/06/2004/policy/http	
wsdl	http://schemas.xmlsoap.org/wsdl/	
(none)	http://schemas.microsoft.com/sharepoint/clientquery/2009	

2.2.2 Messages

This specification does not define any common WSDL message definitions.

2.2.3 Elements

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

2.2.4 Complex Types

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

Complex type	Description
CSOM array	An ordered collection of values.
CSOM dictionary	An object that contains a set of key/value pairs.
CSOM null	A special value indicating that no instance is specified.
CSOM Object	An object that contains a set of named values and a set of methods that operates on those values.
CSOM Stream	A sequence of bytes.
CSOM value object	An object that contains a set of named values.

2.2.4.1 CSOM Array

The **CSOM array** complex type contains an ordered collection of values. The values are identified by position and position is determined by a zero-based **integer** index.

2.2.4.1.1 CSOM Array XML Value

If a **CSOM array** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:complexType name="CSOMArray">
  <xs:sequence>
    <xs:element name="Object" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:any />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Array" />
</xs:complexType>
```

It MUST also conform to the restricted format of a **MethodParameterArrayType** type, as specified in section 3.1.4.1.3.26.

2.2.4.1.2 CSOM Array JSON Value

If a **CSOM array** value is used in response JSON text, it is a JSON array and it MUST conform to the following format:

```
csom-array-value        = rfc4627-array
```

2.2.4.2 CSOM Dictionary

The **CSOM dictionary** complex type is an object that contains an unordered collection of key/value pairs, each of which is represented by a **Property** element. Each key MUST contain a unique name.

2.2.4.2.1 CSOM Dictionary XML Value

If a CSOM dictionary value is used in request XML, it MUST conform to the following XML schema:

```
<xs:complexType name="CSOMDictionary">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:any />
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:string" use="required" fixed=
  "Dictionary" />
</xs:complexType>
```

It MUST also conform to the restricted format of a **MethodParameterDictionaryType** type, as specified in section 3.1.4.1.3.26.10.

2.2.4.2.2 CSOM Dictionary JSON Value

If a CSOM dictionary value is used in response JSON text, it MUST conform to the format of a **CSOM dictionary JSON value** object, as specified in section 3.1.4.1.8.1.1.

2.2.4.3 CSOM Null

The **CSOM null** complex type is a special value that can be used in place of a **CSOM Object**, as specified in section 2.2.4.4, **CSOM value object**, as specified in section 2.2.4.5, **CSOM array**, as specified in section 2.2.4.1, **CSOM binary**, as specified in section 2.2.5.1, or **CSOM String**, as specified in section 2.2.5.14. The **CSOM null** type indicates that no instance is specified.

2.2.4.3.1 CSOM Null XML Value

If a **CSOM null** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:complexType name="CSOMNull">
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Null" />
</xs:complexType>
```

2.2.4.3.2 CSOM Null JSON Value

If a **CSOM null** value is used in response JSON text, it is a **JSON literal null**, as specified in section 2.2, and MUST conform to the following format:

```
csom-null-value         = rfc4627-null
```

2.2.4.4 CSOM Object

The **CSOM Object** complex type is an object that contains a set of named values and methods, which are referred to as members. In addition, a CSOM Object has a **Unicode string** value that identifies its type and is referred to as a **CSOM type name**.

2.2.4.4.1 Object Path XML Value

If a CSOM Object value is used in request XML, the request XML MUST specify both the **object path** identifier that is used to obtain the object path and the object path that is used to obtain the CSOM Object. The object path XML value MUST conform to one of the formats listed in the following table.

Format	Section
ObjectPathConstructorType	3.1.4.1.3.27
ObjectPathMethodType	3.1.4.1.3.28
ObjectPathObjectIdentityNameType	3.1.4.1.3.29
ObjectPathPropertyType	3.1.4.1.3.30
ObjectPathStaticMethodType	3.1.4.1.3.31
ObjectPathStaticPropertyType	3.1.4.1.3.32

2.2.4.4.2 CSOM Object JSON Value

If CSOM Object data is used in response JSON text, it MUST conform to the format of a **CSOM Object JSON value** type, as specified in section 3.1.4.1.8.1.3.

2.2.4.5 CSOM Value Object

The **CSOM value object** complex type is an object that contains a set of named values, which are referred to as members. A **CSOM value object** has a Unicode **string** value that identifies its type and is referred to as a CSOM type name.

2.2.4.5.1 CSOM Value Object XML Value

If a CSOM value object value is used in request XML, it MUST conform to the following XML schema:

```
<xs:complexType name="CSOMValueObject">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:any />
        </xs:sequence>
        <xs:attribute name="Name" type="xs:string" use="required" />
        <xs:anyAttribute />
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="TypeId" type="GuidType" use="required" />
</xs:complexType>
```

It MUST also conform to the restricted format of a **MethodParameterValueObjectType** type, as specified in section 3.1.4.1.3.26.27.

2.2.4.5.2 CSOM Value Object JSON Value

If a CSOM value object value is used in response JSON text, it MUST conform to the format of a **CSOM value object JSON value** type, as specified in section 3.1.4.1.8.1.4.

2.2.4.6 CSOM Stream

The **CSOM Stream**<1> complex type contains a sequence of bytes.

2.2.4.6.1 CSOM Stream XML Value

If a CSOM Stream value is used in a request, the **CSOM Stream** XML Value in the request XML message specifies the **MIME** part which contains the stream data. The **CSOM Stream** XML Value MUST conform to the following XML schema:

```
<xs:complexType name="CSOMStream">
  <xs:attribute name="href" type="xs:string" use="required" />
</xs:complexType>
```

href: The attribute that specifies the MIME part which contains the stream's data. The value of the **href** attribute MUST conform to the following format:

```
csom-stream-href-value  = csom-stream-id-prefix csom-stream-id
csom-stream-id-prefix   = %x63.69.64.3A  ; "cid:"
csom-stream-id          = 1*VCHAR
```

Where **csom-stream-id** is the identifier of the stream. The **csom-stream-id** MUST be unique within one request.

The **content-id** of the MIME part that contains the stream data MUST conform to the following format:

```
csom-mime-part-content-id  = "<" csom-stream-id ">"
```

The **CSOM Stream** XML value MUST also conform to the restricted format of a **MethodParameterBinaryType** type, as specified in section 3.1.4.1.3.26.28.

2.2.4.6.2 CSOM Stream JSON Value

If a CSOM Stream value is used in the response, the **CSOM Stream** JSON Value in the response JSON text message specifies the MIME part which contains the stream data. The **CSOM Stream** JSON value is a JSON **string** and it MUST conform to the following format:

```
csom-stream-json-value  = rfc4627-quotation-mark csom-stream-prefix
                          url-encoded-csom-stream-id csom-stream-suffix
                          rfc4627-quotation-mark
csom-stream-prefix      = %x5C.2F.42.69.6E.61.72.79.28  ; "\/Binary("
csom-stream-suffix      = %x29.5C.2F  ; ")\/"
csom-stream-id          = 1*VCHAR
url-encoded-csom-stream-id = rfc1738-url; The url defined in RFC 1738
```

The **url-encoded-csom-stream-id** is the **URL-encoded** value for **csom-stream-id**, and **csom-stream-id** is the identifier of the stream. The **csom-stream-id** MUST be unique within one response.

The content-id of the MIME part that contains stream data MUST conform to the following format:

```
csom-mime-part-content-id  = "<" csom-stream-id ">"
```

2.2.5 Simple Types

The following table summarizes the set of simple type definitions defined in [MS-DTYP].

Simple Type	Section
BYTE	[MS-DTYP] section 2.2.6
DOUBLE	[MS-DTYP] section 2.2.8
FLOAT	[MS-DTYP] section 2.2.15
INT8	[MS-DTYP] section 2.2.20
INT16	[MS-DTYP] section 2.2.21
INT32	[MS-DTYP] section 2.2.22
INT64	[MS-DTYP] section 2.2.23
UINT16	[MS-DTYP] section 2.2.48
UINT32	[MS-DTYP] section 2.2.49
UINT64	[MS-DTYP] section 2.2.50

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

Simple type	Description
CSOM binary	An array of unsigned 8-bit **integers**.
CSOM Boolean	A **Boolean** value.
CSOM Byte	An unsigned 8-bit **integer** value.
CSOM Char	A Unicode character value.
CSOM DateTime	A date-time value.
CSOM Decimal	A 128-bit, fixed precision, numeric value.
CSOM Double	A 64-bit, double-precision floating-point value.
CSOM Enum	An **integer** that is constrained to a specific set of **integers**.
CSOM GUID	A **GUID**.
CSOM Int16	A 16-bit signed **integer** value.
CSOM Int32	A 32-bit signed **integer** value.
CSOM Int64	A 64-bit signed **integer** value.
CSOM SByte	An 8-bit signed **integer** value.
CSOM Single	A 32-bit, single-precision floating-point value.
CSOM String	A string of Unicode characters.
CSOM TimeSpan	A duration of time.
CSOM UInt16	An unsigned 16-bit **integer** value.

Simple type	Description
CSOM UInt32	An unsigned 32-bit **integer** value.
CSOM UInt64	An unsigned 64-bit **integer** value.

2.2.5.1 CSOM Binary

The **CSOM binary** simple type is an array of unsigned 8-bit **integers**.

2.2.5.1.1 CSOM Binary XML Value

If a **CSOM binary** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMBinary">
  <xs:restriction base="xs:base64Binary"></xs:restriction>
</xs:simpleType>
```

For example, if a binary is of length 4, the first byte is "61", the second byte is "62", the third byte is "63", the fourth byte is "64", and it is specified as:

```
PT4/QA==
```

2.2.5.1.2 CSOM Binary JSON Value

If a **CSOM binary** value is used in response JSON text, it is a JSON **string** and it MUST conform to the following format:

```
csom-binary-json-value  = rfc4627-quotation-mark csom-binary-prefix
                          base64String csom-binary-suffix rfc4627-quotation-mark
csom-binary-prefix      = %x5C.2F.42.61.73.65.36.34.42.69.6E.61.72.79.28
                          ; "\/Base64Binary("
csom-binary-suffix      = %x29.5C.2F  ; ")\/"
base64String            = *base64Char
base64Char              = ALPHA / DIGIT / plus / solidus / pad
plus                    = "+"
solidus                  = "\u002f" / "\u002F"
pad                     = "="
```

The **base64String** is obtained by replacing the character solidus "'/' (U+002F)" in the **base64 encoding** result of the byte array with "\u002f" or "\u002F". For example, if a binary is of length 4, the first byte is "61", the second byte is "62", the third byte is "63", the fourth byte is "64", and the base64 encoding result is:

```
PT4/QA==
```

The **base64String** is either:

```
PT4\u002fQA==
```

Or:

```
PT4\u002FQA==
```

For example, if a binary is of length 4, the first byte is "61", the second byte is "62", the third byte is "63", and the fourth byte is "64", then the **CSOM binary** JSON value is specified as either:

```
"\/Base64Binary(PT4\u002fQA==)\/"
```

Or:

```
"\/Base64Binary(PT4\u002FQA==)\/"
```

2.2.5.2 CSOM Boolean

The **CSOM Boolean** simple type is a **Boolean** value of either "true" or "false".

2.2.5.2.1 CSOM Boolean XML Value

If a **CSOM Boolean** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMBoolean">
  <xs:restriction base="xs:boolean"></xs:restriction>
</xs:simpleType>
```

2.2.5.2.2 CSOM Boolean JSON Value

If a **CSOM Boolean** value is used in response JSON text, it MUST conform to the following format:

```
csom-boolean-json-value = rfc4627-true / rfc4627-false
```

In this format, "rfc4627-true" represents a value of "true" and "rfc4627-false" represents a value of "false".

2.2.5.3 CSOM Byte

The **CSOM Byte** simple type is an unsigned 8-bit **integer** value that represents the **BYTE** type specified in [MS-DTYP] section 2.2.6. The range of **CSOM Byte** values is 0–255.

2.2.5.3.1 CSOM Byte XML Value

If a **CSOM Byte** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMByte">
  <xs:restriction base="xs:unsignedByte"></xs:restriction>
</xs:simpleType>
```

2.2.5.3.2 CSOM Byte JSON Value

If a **CSOM Byte** value is used in response JSON text, it is a JSON number and it MUST be in the range 0–255. It MUST also conform to the following ABNF syntax, as specified in [RFC2234]:

```
csom-byte-json-value    = 1*3DIGIT
```

2.2.5.4 CSOM Char

The **CSOM Char** simple type is a Unicode character value.

2.2.5.4.1 CSOM Char XML Value

If a **CSOM Char** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMChar">
  <xs:restriction base="xs:string">
    <xs:length value="1" />
  </xs:restriction>
</xs:simpleType>
```

2.2.5.4.2 CSOM Char JSON Value

If a **CSOM Char** value is used in response JSON text, it is a JSON **string** and it MUST conform to the following ABNF syntax, as specified in [RFC2234]:

```
csom-char-value         = rfc4627-quotation-mark
                          restricted-char
                          rfc4627-quotation-mark
rfc4627-quotation-mark  = %x22                  ; "
restricted-char         = rfc4627-unescaped /
                          rfc4627-escape (
                                %x22 /          ; "    quotation mark  U+0022
                                %x5C /          ; \    reverse solidus U+005C
                                %x62 /          ; b    backspace       U+0008
                                %x66 /          ; f    form feed       U+000C
                                %x6E /          ; n    line feed       U+000A
                                %x72 /          ; r    carriage return U+000D
                                %x74 /          ; t    tab             U+0009
                                %x75 4HEXDIG )  ; uXXXX                U+XXXX
rfc4627-escape          = %x5C                  ; \
rfc4627-quotation-mark  = %x22                  ; "
rfc4627-unescaped       = %x20-21 / %x23-5B / %x5D-10FFFF
```

For example, the Unicode character with the value "0x2000" is specified as:

```
"\u2000"
```

The Unicode character solidus "'/' (U+002F)" is specified as either:

```
"\u002f"
```

Or:

```
"\u002F"
```

2.2.5.5 CSOM DateTime

The **CSOM DateTime** simple type is an instant of time that is represented as an **INT64** value and specifies the number of 100-nanosecond intervals that have elapsed since 12:00:00 A.M., January 1, 0001. The range of values for 100-nanosecond intervals begins at 12:00:00 A.M., January 1, 0001, and ends at 23:59:59.9999999, December 31, 9999. A **CSOM DateTime** value can also indicate time zone information, as follows.

Time zone value	Description
Unspecified	Time zone information is not specified.
UTC	The time is specified in the **Coordinated Universal Time (UTC)** time zone.
Local	The time is specified as a UTC time with a local time zone offset.

2.2.5.5.1 CSOM DateTime XML Value

If a **CSOM DateTime** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMDateTime">
  <xs:restriction base="xs:dateTime"/>
</xs:simpleType>
```

2.2.5.5.2 CSOM DateTime JSON Value

If a **CSOM DateTime** value is used in response JSON text, it MUST conform to the structure of a JSON **string**. It MUST also conform to the following format:

```
csom-datetime-value     = rfc4627-quotation-mark datetime-prefix
                          datetime-components
                          datetime-suffix
                          rfc4627-quotation-mark
datetime-prefix         = %x5C.2F.44.61.74.65.28  ; "\/Date("
datetime-suffix         = %x29.5C.2F  ; ")\/"
datetime-components     = datetime-unspecified /
                          datetime-utc /
                          datetime-local
datetime-utc            = ticks
datetime-local          = ticks ( "+" / "-" ) timeZoneOffset
datetime-unspecified    = year "," month "," day ","
                          hour "," minute "," second "," millisecond
year                    = int
month                   = int
day                     = int
hour                    = int
minute                  = int
second                  = int
millisecond             = int
ticks                   = int
timeZoneOffset          = int
int                     = [rfc4627-minus] 1*DIGIT
```

If the **CSOM DateTime** value is in the **datetime-unspecified** format, the value is relative to 12:00:00 A.M., January 1, 0001 and the time zone of the value is not specified:

```
"\/Date(year,month,day,hour,minute,second,millisecond)\/"
```

The following table summarizes the attributes of a **CSOM DateTime** value.

Name	Description
year	An **integer** value that represents the year component of a **CSOM DateTime** value. It MUST be in the range 1–9999, where a value such as "2009" means the 2009 calendar year.
month	An **integer** value that represents the month component of a **CSOM DateTime** value. It MUST be in the range 0–11, where January is "0", February is "1", and so forth.
day	An **integer** value that represents the day component of a **CSOM DateTime** value. It MUST be in the range 1–31, where "1" is the first day of the month, "2" is the second day of the month, and so forth. The year, month, and day components together MUST form a valid **date-time** value.
hour	An **integer** value that represents the hour component of a **CSOM DateTime** value. It MUST be in the range 0–23, where 12:00 A.M. is "0", 1:00 A.M. is "1", 12:00 P.M. is "12", 1:00 P.M. is "13", and so forth.
minute	An **integer** value that represents the minute component of a **CSOM DateTime** value. It MUST be in the range 0–59, where "0" is the starting point of an hour, "1" is the first minute of the hour, "2" is the second minute of the hour, and so forth.
second	An **integer** value that represents the second component of a **CSOM DateTime** value. It MUST be in the range 0–59, where "0" is the starting point of a minute, "1" is the first (that is, 1st) second of the minute, "2" is the second (that is, 2nd) second of the minute, and so forth.
millisecond	An **integer** value that represents the millisecond component of a **CSOM DateTime** value. It MUST be in the range 0–999, where "0" is the starting point of a second, "1" is the first millisecond of the second, "2" is the second millisecond of the second, and so forth.

For example, May 22, 2009, 05:12 P.M. is specified as:

```
"\/Date(2009,4,22,17,12,0,0)\/"
```

If the **CSOM DateTime** value is in the **datetime-utc** format, the value is:

```
"\/Date(ticks)\/"
```

In this value, **ticks** is an **integer** value that represents the number of ticks since January 1, 1970, 12:00:00 A.M. in the UTC time zone. A tick is 100 nanoseconds. If the value is negative, it represents an earlier time.

For example, the UTC date and time May 22, 2009, 05:12 P.M. is specified as:

```
"\/Date(1243012320000)\/"
```

The UTC date and time May 22, 1968, 05:12 P.M. is specified as:

```
"\/Date(-50827680000)\/"
```

If the **CSOM DateTime** value is in the **datetime-local** format, the value is:

```
"\/Date(ticks+timeZoneOffset)\/"
```

Or:

```
"\/Date(ticks-timeZoneOffset)\/"
```

The following table summarizes the parameters of the **datetime-local** format of a **CSOM DateTime** value.

Name	Description
ticks	An **integer** value that represents the number of ticks since January 1, 1970, 12:00:00 A.M. in the UTC time zone. A tick is 100 nanoseconds. If the value is negative, it represents an earlier time.
timeZoneOffset	This MUST be of the form *hhmm* where *hh* is an **integer** in the range 0–13 and *mm* is an **integer** in the range 0–59. This value represents the difference between a local time zone and the UTC time zone. If the date and time has +**timeZoneOffset**, the local time zone is *hh* hours and *mm* minutes after the UTC time zone. If the date and time has **–timeZoneOffset**, the local time zone is *hh* hours and *mm* minutes before the UTC time zone.

For example, the local date and time May 22, 2009, 05:12 P.M. Pacific Daylight Time is specified as:

```
"\/Date(1243037520000-0700)\/"
```

2.2.5.6 CSOM Double

The **CSOM Double** simple type is a 64-bit, double-precision, floating-point value and is the **DOUBLE** type specified in [MS-DTYP] section 2.2.8. A **CSOM Double** value MUST be in the range from –1.79769313486232e308 to 1.79769313486232e308.

2.2.5.6.1 CSOM Double XML Value

If a **CSOM Double** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMDouble">
  <xs:restriction base="xs:double" />
</xs:simpleType>
```

2.2.5.6.2 CSOM Double JSON Value

If a **CSOM Double** value is used in response JSON text, it is a JSON number and it MUST be in the range from –1.79769313486232e308 to 1.79769313486232e308. It MUST also conform to the following format:

```
csom-double-value = rfc4627-number
```

2.2.5.7 CSOM Enum

The **CSOM Enum** simple type is an **integer** whose values are constrained to a specific set of values.

2.2.5.7.1 CSOM Enum XML Value

If a **CSOM Enum** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMEnum">
  <xs:restriction base="xs:integer" />
</xs:simpleType>
```

2.2.5.7.2 CSOM Enum JSON Value

If a **CSOM Enum** value is used in response JSON text, it is a JSON number and it MUST conform to the following format:

```
csom-enum-value          = [rfc4627-minus] 1*DIGIT
```

2.2.5.8 CSOM GUID

The **CSOM GUID** simple type is a GUID and is the **GUID** type specified in [MS-DTYP] section 2.3.4.2.

2.2.5.8.1 CSOM Guid XML Value

If a **CSOM GUID** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMGUID">
  <xs:restriction base="xs:string">
    <xs:pattern value="\{[0-9A-Fa-f]{8}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-
    Fa-f]{4}-[0-9A-Fa-f]{12}\}" />
  </xs:restriction>
</xs:simpleType>
```

2.2.5.8.2 CSOM GUID JSON Value

If a **CSOM GUID** value is used in response JSON text, it is a JSON **string** and it MUST conform to the following format:

```
csom-guid-value         = rfc4627-quotation-mark guid-prefix
                          guid-string
                          guid-suffix rfc4627-quotation-mark
guid-prefix             = %x5C.2F.47.75.69.64.28  ; "\/Guid("
guid-suffix             = %x29.5C.2F  ; ")\/"
guid-string             = 4hexOctet "-" 2hexOctet "-" 2hexOctet
                          "-" 2hexOctet "-" 6hexOctet
hexOctet                = hexDigit hexDigit
hexDigit                = "0" / "1" / "2" / "3" / "4" /
                          "5" / "6" / "7" / "8" / "9" /
                          "a" / "b" / "c" / "d" / "e" / "f" /
                          "A" / "B" / "C" / "D" / "E" / "F"
```

For example, a **CSOM GUID** value of "af300e5f-ec35-4f39-998f-6c3694a04706" is specified as:

```
"\/Guid(af300e5f-ec35-4f39-998f-6c3694a04706)\/"
```

2.2.5.9 CSOM Int16

The **CSOM Int16** simple type is a 16-bit, signed **integer** value and is the **INT16** type specified in [MS-DTYP] section 2.2.21. The range of **CSOM Int16** values is from –32768 to 32767.

2.2.5.9.1 CSOM Int16 XML Value

If a **CSOM Int16** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMInt16">
  <xs:restriction base="xs:short" />
</xs:simpleType>
```

2.2.5.9.2 CSOM Int16 JSON Value

If a **CSOM Int16** value is used in response JSON text, it is a JSON number and it MUST be in the range from –32768 to 32767. It MUST also conform to the following format:

```
csom-int16-value        = [rfc4627-minus] 1*5DIGIT
```

2.2.5.10 CSOM Int32

The **CSOM Int32** simple type is a 32-bit, signed **integer** value and is the **INT32** type specified in [MS-DTYP] section 2.2.22. The range of **CSOM Int32** values is from –2147483648 to 2147483647.

2.2.5.10.1 CSOM Int32 XML Value

If a **CSOM Int32** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMInt32">
  <xs:restriction base="xs:int" />
</xs:simpleType>
```

2.2.5.10.2 CSOM Int32 JSON Value

If a **CSOM Int32** value is used in response JSON text, it is a JSON number and it MUST be in the range from –2147483648 to 2147483647. It MUST also conform to the following format:

```
csom-int32-value        = [rfc4627-minus] 1*10DIGIT
```

2.2.5.11 CSOM Int64

The **CSOM Int64** simple type is a 64-bit, signed **integer** value and is the **INT64** type specified in [MS-DTYP] section 2.2.23. The range of **CSOM Int64** values is from –9223372036854775808 to 9223372036854775807.

2.2.5.11.1 CSOM Int64 XML Value

If a **CSOM Int64** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMInt64">
  <xs:restriction base="xs:long" />
</xs:simpleType>
```

2.2.5.11.2 CSOM Int64 JSON Value

If a **CSOM Int64** value is used in response JSON text, it is a JSON number and it MUST conform to the following format:

```
csom-int64-value        = [rfc4627-minus] 1*19DIGIT
```

2.2.5.12 CSOM SByte

The **CSOM SByte** simple type is an 8-bit, signed **integer** value and is the **INT8** type specified in [MS-DTYP] section 2.2.20. The range of **CSOM SByte** values is from –128 to 127.

2.2.5.12.1 CSOM SByte XML Value

If a **CSOM SByte** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMSByte">
  <xs:restriction base="xs:byte" />
</xs:simpleType>
```

2.2.5.12.2 CSOM SByte JSON Value

If a **CSOM SByte** value is used in response JSON text, it is a JSON number and it MUST be in the range from –128 to 127. It MUST also conform to the following format:

```
csom-sbyte-value        = [rfc4627-minus] 1*3DIGIT
```

2.2.5.13 CSOM Single

The **CSOM Single** simple type is a 32-bit, single-precision, floating-point value and is the **FLOAT** type specified in [MS-DTYP] section 2.2.15. The range of **CSOM Single** values is from –3.402823e38 to 3.402823e38.

2.2.5.13.1 CSOM Single XML Value

If a **CSOM Single** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMSingle">
  <xs:restriction base="xs:float" />
</xs:simpleType>
```

2.2.5.13.2 CSOM Single JSON Value

If a **CSOM Single** value is used in response JSON text, it is a JSON number and it MUST be in the range from –3.402823e38 to 3.402823e38. It MUST also conform to the following format:

```
csom-single-value       = rfc4627-number
```

2.2.5.14 CSOM String

The **CSOM String** simple type is a representation of text as a series of Unicode characters.

2.2.5.14.1 CSOM String XML Value

If a **CSOM String** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMString">
  <xs:restriction base="xs:string" />
</xs:simpleType>
```

2.2.5.14.2 CSOM String JSON Value

If a **CSOM String** value is used in response JSON text, all of the characters except the Unicode character solidus "'/' (U+002F)" are represented by using the value "rfc4627-char". The Unicode character solidus "'/' (U+002F)" MUST be escaped as:

```
\u002f
```

Or:

```
\u002F
```

Not:

```
\/
```

The **CSOM String** value is a JSON **string** and it MUST conform to the following format:

```
csom-string-value       = rfc4627-quotation-mark
                          *restricted-char
                          rfc4627-quotation-mark
rfc4627-quotation-mark  = %x22                  ; "
restricted-char         = rfc4627-unescaped /
                          rfc4627-escape (
                                %x22 /          ; "    quotation mark  U+0022
                                %x5C /          ; \    reverse solidus U+005C
                                %x62 /          ; b    backspace       U+0008
                                %x66 /          ; f    form feed       U+000C
                                %x6E /          ; n    line feed       U+000A
                                %x72 /          ; r    carriage return U+000D
                                %x74 /          ; t    tab             U+0009
                                %x75 4HEXDIG )  ; uXXXX                U+XXXX
rfc4627-escape          = %x5C                  ; \
rfc4627-quotation-mark  = %x22                  ; "
rfc4627-unescaped       = %x20-21 / %x23-5B / %x5D-10FFFF
```

For example, the **string** literal value "example/bar" is specified as:

```
"example\u002fbar"
```

2.2.5.15 CSOM UInt16

The **CSOM UInt16** simple type is an unsigned, 16-bit **integer** value and is the **UINT16** type specified in [MS-DTYP] section 2.2.48. The range of **CSOM UInt16** values is 0–65535.

2.2.5.15.1 CSOM UInt16 XML Value

If a **CSOM UInt16** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMUInt16">
  <xs:restriction base="xs:unsignedShort" />
</xs:simpleType>
```

2.2.5.15.2 CSOM UInt16 JSON Value

If a **CSOM UInt16** value is used in response JSON text, it is a JSON number and it MUST be in the range 0–65535. It MUST also conform to the following format:

```
csom-uint16-value       = 1*5DIGIT
```

2.2.5.16 CSOM UInt32

The **CSOM UInt32** simple type is an unsigned, 32-bit **integer** value and is the **UINT32** type specified in [MS-DTYP] section 2.2.49. The range of **CSOM UInt32** values is 0–4294967295.

2.2.5.16.1 CSOM UInt32 XML Value

If a **CSOM UInt32** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMUInt32">
  <xs:restriction base="xs:unsignedInt" />
</xs:simpleType>
```

2.2.5.16.2 CSOM UInt32 JSON Value

If a **CSOM UInt32** value is used in response JSON text, it is a JSON number and it MUST be in the range 0–4294967295. It MUST also conform to the following format:

```
csom-uint32-value       = 1*10DIGIT
```

2.2.5.17 CSOM UInt64

The **CSOM UInt64** simple type is an unsigned, 64-bit **integer** value and is the **UINT64** type specified in [MS-DTYP] section 2.2.50. The range of **CSOM UInt64** values is 0–18446744073709551615.

2.2.5.17.1 CSOM UInt64 XML Value

If a **CSOM UInt64** value is used in request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMUInt64">
  <xs:restriction base="xs:unsignedLong" />
</xs:simpleType>
```

2.2.5.17.2 CSOM UInt64 JSON Value

If a **CSOM UInt64** value is used in response JSON text, it is a JSON number and it MUST be in the range 0–18446744073709551615. It MUST also conform to the following format:

```
csom-uint64-value       = 1*20DIGIT
```

2.2.5.18 CSOM Decimal

The **CSOM Decimal**<2> simple type is a 128-bit, fixed precision, numeric value. A **CSOM Decimal** value MUST be in the range from -79,228,162,514,264,337,593,543,950,335 through 79,228,162,514,264,337,593,543,950,335 with a maximum of 28 digits to the right of the decimal point to specify the fractional part of the number.

2.2.5.18.1 CSOM Decimal XML Value

If a **CSOM Decimal** value is used in the request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMDecimal">
  <xs:restriction base="xs:decimal" />
</xs:simpleType>
```

2.2.5.18.2 CSOM Decimal JSON Value

If a **CSOM Decimal** value is used in a response JSON text, it is a JSON string whose string value is the XML string representation of the decimal value. It MUST also conform to the following format:

```
csom-decimal-value = rfc4627-quotation-mark
                     csom-decimal
                     rfc4627-quotation-mark
csom-decimal       = [ rfc4627-minus ]
                     1*29DIGIT
                     [ "." 1*29DIGIT ]
```

2.2.5.19 CSOM TimeSpan

The **CSOM TimeSpan**<3> simple type represents a duration of time.

2.2.5.19.1 CSOM TimeSpan XML Value

If a **CSOM TimeSpan** value is used in the request XML, it MUST conform to the following XML schema:

```
<xs:simpleType name="CSOMTimeSpan">
  <xs:restriction base="xs:duration" />
</xs:simpleType>
```

2.2.5.19.2 CSOM TimeSpan JSON Value

If a **CSOM TimeSpan** value is used in a response JSON text, it is a JSON string and the JSON string value MUST be the XML representation of the duration data type specified by [XMLSCHEMA2]. It MUST also conform to the following format:

```
csom-timespan-value       = rfc4627-quotation-mark
                            csom-duration
                            rfc4627-quotation-mark
csom-duration             = <defined by the lexical representation for duration
                             in [XMLSCHEMA2] >
```

For example, a duration of 2 days 3 hours 4 minutes 5 seconds is specified as:

```
"P2DT3H4M5S"
```

2.2.6 Attributes

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

2.2.7 Groups

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

2.2.8 Attribute Groups

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

3 Protocol Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

This protocol allows protocol servers to perform implementation-specific authorization checks and to notify protocol clients of authorization faults by using either HTTP **Status-Codes** or **CSOM error** JSON values. Except where specified otherwise, protocol clients SHOULD interpret HTTP Status-Codes as specified in [RFC2616] section 10 and CSOM error JSON values as specified in section 3.1.4.1.8.1.2.

The **Flag** field in **CSOM** structures specifies whether the structure is an enumeration that can be used in combination. If the value is "true", it means the enumeration values can be combined.

The **ShortName** field in **CSOM** structures specifies the alterative name of the CSOM Object type.

3.1 Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of a possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The SharePoint Client Query Data Model represents the data structures that are used by a protocol client and a protocol server. The following figure shows the communication sequence between a protocol client and a protocol server.

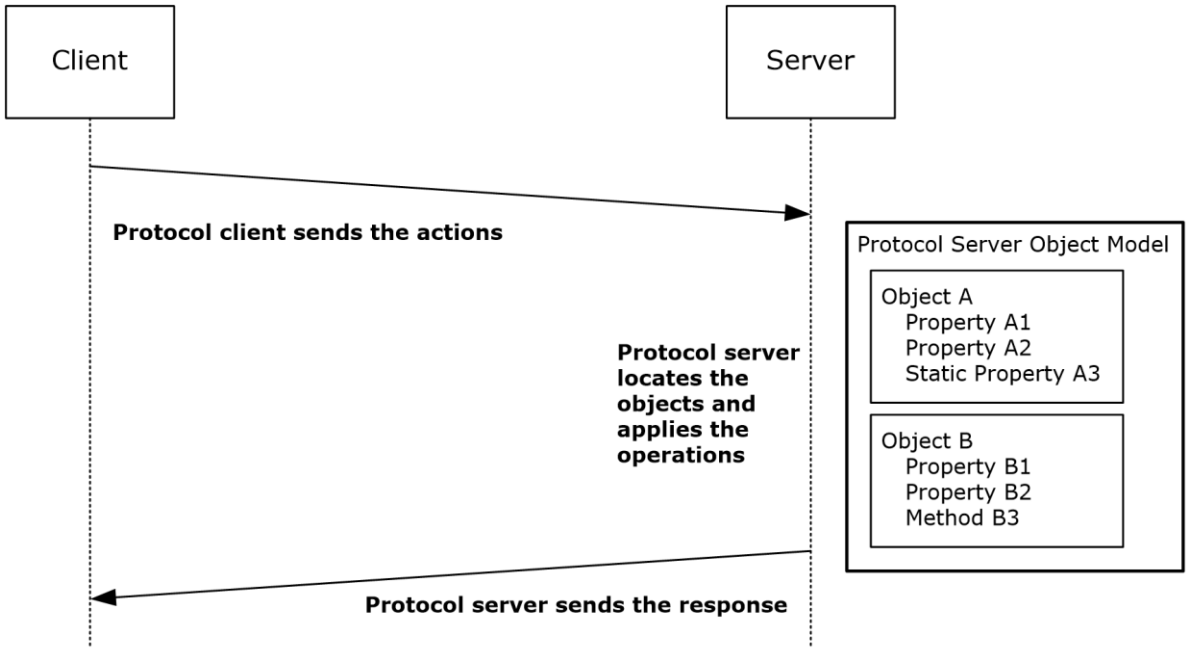

Figure 3: Communication sequence between a protocol client and a protocol server

3.1.1.1 Protocol Server Type Set

A protocol server exposes one or more sets of types that are accessible and can be called by a protocol client. Each type set contains a set of **CSOM value object types**, **CSOM Object types**, and CSOM **enumerations**.

CSOM Value Object Type

A CSOM value object type is a **CSOM type** that contains a set of named values, which are referred to as members. Each CSOM value object type has type information that is identified by a Unicode **string**, which is referred to as a CSOM type name. Each CSOM value object type is also associated with a **CSOM GUID**, which is referred to as a **CSOM type identifier**.

CSOM Object Type

A CSOM Object type is a CSOM type that contains a set of named values and methods, which are referred to as members. Each CSOM Object type has a type name that is identified by a Unicode **string**, which is referred to as a CSOM type name. Each CSOM Object type is also associated with a **CSOM GUID**, which is referred to as a CSOM type identifier.

CSOM Null

A CSOM **null** is a special value that can be used instead of a CSOM Object, CSOM value object, **CSOM array**, **CSOM binary**, or **CSOM String**. A CSOM **null** value indicates that no instance is specified.

CSOM Property

A **CSOM property** is a member that is declared in a CSOM value object type or a CSOM Object type. The value of a CSOM property can be read and, optionally, changed by a protocol client. A definition of a CSOM property includes the following:

- **Name:** A Unicode **string** that represents the name of the CSOM property.
- **Accessibility:** Whether the protocol client can only read or can also change the value of the CSOM property.
- **Type:** The CSOM type of the value that is contained by the CSOM property.

CSOM Expando Field

A protocol client can have predefined knowledge of the methods and properties that are exposed for types in a type set that is on a protocol server. A protocol server can expose additional properties for objects. These custom properties, which are referred to as CSOM expando fields, contain additional state information that applies to an instance of the type. These properties behave similarly to other instances of scalar properties for the type.

Protocol servers that expose types with CSOM expando fields can use a naming convention to help identify these properties as custom properties. For example, "PropertySetName$PropertyName" is one such convention, where a protocol client can assume that any property that is returned by the protocol server and starts with "PropertySetName$" is a custom property.

With the exception of server-defined, implementation-specific, naming conventions, this protocol does not specify a mechanism for discovering the exact set of CSOM expando fields for an object. A protocol client cannot define or specify additional CSOM expando fields for a type. It can only process the CSOM expando fields that are returned from the protocol server.

CSOM Method

A **CSOM method** is a method that is declared in a CSOM Object type and can be invoked by a protocol client. A definition of a CSOM method includes the following:

- **Name:** A Unicode **string** that represents the name of the CSOM method.
- **Arguments:** An ordered collection of parameters that can be passed when the method is called. Each parameter has a CSOM type.
- **Return type:** The CSOM type of the value that is returned by the CSOM method. If the CSOM method does not return any value, it does not have a return type.

The protocol server can provide access control restrictions to CSOM properties and CSOM methods. If a protocol server implements access control restrictions, all of the operations that are provided by this protocol respect those restrictions when calling methods and reading or writing property values.

CSOM Expression

A **CSOM expression** is an expression that is evaluated on a protocol server.

CSOM Error

A CSOM error is a type that contains information about an error that occurred on a protocol server. The error information consists of an error code, which is a **CSOM Int32**, an error message, which is a CSOM String, a call-stack trace, which is a **CSOM String**, a type name, which is a **CSOM String**, and an additional information value, which is a **CSOM String**.

CSOM Object Path

A CSOM Object path is the data that a protocol server uses to obtain the CSOM Object that a **CSOM action** is performed on.

CSOM Object Path List

A CSOM Object path list is a list of zero or more paths to a CSOM Object on a protocol server.

3.1.1.2 Client Actions

A protocol client sends one or more actions that execute against a type set that is exposed by a protocol server.

CSOM Action

A CSOM action is an action to be performed on a protocol server. The different types of actions are:

- **ObjectPath:** Instantiates a CSOM Object.
- **Query:** Queries the values of scalar properties for a specified CSOM Object or CSOM Object type.
- **ChildItemQuery:** Queries the values of scalar properties of CSOM Objects that are child objects of a specified CSOM Object collection.
- **ObjectIdentityQuery:** Queries the canonical identity of a specified CSOM Object.
- **SetProperty** or **SetStaticProperty:** Sets the value of a scalar property for a specified CSOM Object or CSOM Object type.
- **Method** or **StaticMethod:** Calls a method for a specified CSOM Object or CSOM Object type.

- **ConditionalScope:** Evaluates a **Boolean** conditional expression and executes a set of actions based on the evaluated value that results from the expression.
- **ExceptionHandlingScope:** Performs a try/catch/finally mechanism. The set of actions for the try block is executed. If an unhandled exception occurs, it is caught and the set of actions for the catch block is executed, followed by the set of actions for the finally block. If an unhandled exception does not occur, the set of actions for the finally block is executed after the set of actions for the try block.

CSOM Action List

A **CSOM action list** is an ordered list of zero or more CSOM actions to be processed by a protocol server.

CSOM Action Response Structure

A **CSOM action response structure** contains the data that results from processing a CSOM action list.

3.1.1.3 Request Processing

A protocol client initiates this protocol by sending a request to the protocol server. The request consists of a CSOM action list and a CSOM Object path list. The CSOM action list contains the CSOM actions to be executed by the protocol server. The CSOM Object path list contains the CSOM Object path that the protocol server can use to obtain the CSOM Object or CSOM Object type that a CSOM action is performed on.

After the protocol server finishes processing the CSOM action list, it sends a response to the protocol client. If no unhandled errors occurred during processing, the response contains a CSOM action response structure that contains the processing results for the CSOM action list. Otherwise, the response contains a CSOM error that provides details about the unhandled error.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

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

Operation	Description
ProcessQuery	Process the CSOM action list in the request message and respond with JSON text that contains the results of executing the specified CSOM actions

3.1.4.1 ProcessQuery

During this operation, the protocol server receives a message and extracts the XML request message that contains a CSOM action list, processes that CSOM action list, and then responds with JSON text that contains the results of processing the CSOM action list.

The protocol server endpoint for this operation is formed by appending "/_vti_bin/client.svc/ProcessQuery" to the URL of the **website**, for example "http://www.example.com/Repository/_vti_bin/client.svc/ProcessQuery".

The protocol client sends a **ProcessQueryIn** (section 3.1.4.1.1.1) message, and the protocol server responds with a **ProcessQueryOut(**section 3.1.4.1.1.2) message, as follows:

1. The protocol client sends a **ProcessQueryIn** request message that contains a CSOM action list and a CSOM Object path list. The CSOM action list contains the CSOM actions to be executed by the protocol server. The CSOM Object path list contains the CSOM Object paths that the protocol server can use to obtain the CSOM Object or CSOM Object type on which a CSOM action is to be performed.

2. The protocol server receives the **ProcessQueryIn** request message and processes the CSOM action list.

3. After the protocol server finishes processing the CSOM action list, it responds to the protocol client with a **ProcessQueryOut** response message. If no unhandled errors occur during processing, the response contains a CSOM action response structure that contains the processing results for the CSOM action list. Otherwise, the response contains a CSOM error that provides details about the unhandled error.

The protocol server processes a CSOM action list as follows:

1. The CSOM action response structure is initially empty.

2. If all of the CSOM actions in the CSOM action list have been processed, the protocol server MUST stop processing the CSOM action list.

3. If one or more of the CSOM actions in the CSOM action list have not been processed, the protocol server MUST execute the first unprocessed CSOM action in the CSOM action list.

4. If the CSOM action is executed successfully, the CSOM action is considered processed, and the result of the CSOM action execution MUST be appended to the CSOM action response structure when the CSOM action has result. The protocol server then repeats steps 2 and 3.

5. If an unhandled exception occurs when executing the CSOM action, the protocol server MUST stop processing the CSOM action list.

The operation ends when the protocol server stops processing the CSOM action list.

If no unhandled exceptions occur while processing the CSOM action list, the protocol server MUST respond with JSON output data in the form of a CSOM action response structure. If an unhandled exception occurs while processing the CSOM action list, the protocol server MUST respond with JSON output data in the form of a CSOM error that provides details about the unhandled error.

3.1.4.1.1 Message

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

Message	Description
ProcessQueryIn	The request message for the **ProcessQuery** Web

Message	Description
	service method.
ProcessQueryOut	The response message for the **ProcessQuery** Web service method.

3.1.4.1.1.1 ProcessQueryIn

The **ProcessQueryIn** message is the request message for the **ProcessQuery** Web service method.

When there is no CSOM Stream data involved in the request, the **ProcessQueryIn** message is an XML document that MUST adhere to the request XML schema specified in section 6.1.

When there is CSOM Stream data involved in the request, the **ProcessQueryIn** message is a MIME multipart/related message as specified by [RFC2387], and the first MIME part is an XML document that MUST adhere to the request XML schema specified in section 6.1. The other MIME parts contain **CSOM Stream** data used by the request. Each MIME part in the MIME multipart/related message MUST have a Content-Length MIME header.

For example, the following is a request message that contains three MIME parts where the first MIME part is the XML document and the other two MIME parts contain CSOM Stream data used by the request.

```
--D1995E12-4610-4C2B-AB08-C1A0ECE8CE4D+id=1
Content-ID: <http://sharepoint.microsoft.com/client/634551683559778320>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml;charset=utf-8;type="application/xml"
Content-Length: 1203

<Request AddExpandoFieldTypeSuffix="true" SchemaVersion="15.0.0.0" LibraryVersion="15.0.0.0"
ApplicationName=".NET Library"
xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009"><Actions><ObjectPath Id="2"
ObjectPathId="1" /><ObjectPath Id="4" ObjectPathId="3" /><ObjectPath Id="6" ObjectPathId="5"
/><Method Name="UpdateSampleStream" Id="7" ObjectPathId="5"><Parameters><Parameter
Type="Binary"><Include href="cid:http://sharepoint.microsoft.com/8"
/></Parameter></Parameters></Method><ObjectPath Id="10" ObjectPathId="9" /><Method
Name="UpdateSampleStream" Id="11" ObjectPathId="9"><Parameters><Parameter
Type="Binary"><Include href="cid:http://sharepoint.microsoft.com/12"
/></Parameter></Parameters></Method></Actions><ObjectPaths><StaticProperty Id="1"
TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" /><Property Id="3"
ParentId="1" Name="Books" /><Method Id="5" ParentId="3" Name="GetById"><Parameters><Parameter
Type="Guid">{3387ac63-e73d-421f-bff7-359a4aa2bc38}</Parameter></Parameters></Method><Method
Id="9" ParentId="3" Name="GetById"><Parameters><Parameter Type="Guid">{704655a3-c136-469c-
a578-f79652a93f9b}</Parameter></Parameters></Method></ObjectPaths></Request>
--D1995E12-4610-4C2B-AB08-C1A0ECE8CE4D+id=1
Content-ID: <http://sharepoint.microsoft.com/8>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 26

New sample content of book
--D1995E12-4610-4C2B-AB08-C1A0ECE8CE4D+id=1
Content-ID: <http://sharepoint.microsoft.com/12>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 28

New sample content of book 2
--D1995E12-4610-4C2B-AB08-C1A0ECE8CE4D+id=1--
```

3.1.4.1.1.2 ProcessQueryOut

The **ProcessQueryOut** message is the response message for the **ProcessQuery** Web service method.

If no **CSOM Stream** is returned to the protocol client, the message is JSON text that MUST adhere to the JSON response structure specified in section 3.1.4.1.8.2.

If a **CSOM Stream** is returned to the protocol client, the message is a MIME multipart/related message as specified by [RFC2387], and the first MIME part is the JSON text that MUST adhere to the JSON response structure specified in section 3.1.4.1.8.2. The other MIME parts contain the **CSOM Stream** data returned by the protocol server. Each MIME part MUST have a Content-Length MIME header.

For example, the following is a response message that contains three MIME parts where the first MIME part is the JSON text and the other two MIME parts contain **CSOM Stream** data returned by the protocol server.

```
--1599120F-20CE-4389-9371-BDC377356FEF+id=1
Content-ID: <http://sharepoint.microsoft.com/client/634551729566307786>
Content-Transfer-Encoding: eightbit
Content-Type: application/jop+json;charset=utf-8;type="application/json"
Content-Length: 653

[
{
"SchemaVersion":"15.0.0.0","LibraryVersion":"15.0.3421.3000","ErrorInfo":null
},2,{
"IsNull":false
},4,{
"IsNull":false
},6,{
"IsNull":false
},7,{
"_ObjectType_":"SampleCode.Book","Author":"Soha Kamal","Id":"\/Guid(3387ac63-e73d-421f-bff7-
359a4aa2bc38)\/","PublishDate":"\/Date(2008,2,1,0,0,0,0)\/","Status":0,"Title":"How to Cook
Chinese Food"
},8,"\/Binary(http%3a%2f%2fsharepoint.microsoft.com%2fclient%2fattachment%2f0%2f6345517295662
57776)\/",10,{
"IsNull":false
},11,{
" ObjectType ":"SampleCode.Book","Author":"Patrick Hines","Status":0
},12,"\/Binary(http%3a%2f%2fsharepoint.microsoft.com%2fclient%2fattachment%2f1%2f634551729566
307786)\/"
]
--1599120F-20CE-4389-9371-BDC377356FEF+id=1
Content-ID: <http://sharepoint.microsoft.com/client/attachment/0/634551729566257776>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 48

Sample Content of book How to Cook Chinese Food.
--1599120F-20CE-4389-9371-BDC377356FEF+id=1
Content-ID: <http://sharepoint.microsoft.com/client/attachment/1/634551729566307786>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 37

Sample Content of book Family Recipe.
--1599120F-20CE-4389-9371-BDC377356FEF+id=1--
```

3.1.4.1.2 Elements

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

Element	Description
Request	Contains the input data for a **ProcessQuery** operation.

3.1.4.1.2.1 Request

The **Request** element contains the input data for a **ProcessQuery** operation. The definition of the **Request** element is:

```
<xs:element name="Request" type="RequestType" xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

3.1.4.1.3 Complex Types

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

Complex type	Description
ActionInstantiateObjectPathType	A CSOM action that instantiates the CSOM Object associated with the specified CSOM Object path identifier.
ActionInvokeMethodType	A CSOM action that calls a method on a CSOM Object, which is obtained from a specified CSOM Object path identifier, passing any parameters for the method.
ActionInvokeStaticMethodType	A CSOM action that invokes a static method on a CSOM Object type, which is obtained from a specified CSOM Object path, passing any parameters.
ActionObjectIdentityQueryType	A CSOM action that obtains the **object identity** of a CSOM Object.
ActionQueryType	A CSOM action that obtains the data resulting from a **child item query** on a CSOM Object.
ActionSetPropertyType	A CSOM action that sets the property of a CSOM Object to a specified parameter value.
ActionSetStaticPropertyType	A CSOM action that sets the static property of a CSOM Object type to a specified parameter value.
ActionsType	An ordered list of CSOM actions to execute.
ChildItemQueryType	The child item query that is applied to a CSOM Object collection.
ConditionalScopeType	A CSOM action that evaluates a conditional expression that does not have any input parameters.
ExceptionHandlingScopeSimpleType	A CSOM action that attempts to execute an ordered list of CSOM actions.
ExceptionHandlingScopeType	a CSOM action that attempts to execute an ordered list of CSOM actions specified by a **TryScope** element.
ExpressionConvertExpressionType	A component of an **expression** that takes the evaluated value of an expression and converts that value to a specified type.

Complex type	Description
ExpressionLeftRightOperandExpressionType	A component of an expression that represents the left and right operands of a binary expression.
ExpressionMethodExpressionType	A component of an expression that represents a method to call, including any applicable parameters, on the CSOM Object specified by its first immediate child element.
ExpressionParameterExpressionType	A component of an expression that represents the input parameter of an expression.
ExpressionPropertyExpressionType	A component of an expression that represents a property of a CSOM Object or CSOM value object.
ExpressionQueryableExpressionType	A component of an expression that represents the **queryable expression** component of a child item query that is applied to a CSOM Object collection to retrieve a filtered subset of child CSOM Objects.
ExpressionQueryableTakeType	A component of an expression that represents a filtered set of CSOM Objects that are returned from an inner queryable expression that returns a specified number of CSOM Objects.
ExpressionQueryableWhereType	A component of an expression that represents a filtered set of CSOM Objects that are returned from an inner queryable expression.
ExpressionStaticMethodExpressionType	A component of an expression that represents a static method to call, including any parameters, on a CSOM Object type.
ExpressionStaticPropertyExpressionType	A component of an expression that represents a static property of a CSOM Object type.
ExpressionType	An expression, which is a combination of operators, symbols, constants, literal values, functions, names of fields or columns, controls, and properties that evaluates to a single value.
ExpressionTypeIsExpressionType	A component of an expression that represents a test result indicating whether the value is a specific type.
MethodParameterType	A parameter that can be passed when calling a method or setting a property.
ObjectPathConstructorType	A CSOM Object path that specifies the CSOM Object that is obtained by invoking a constructor method, including any parameters, of a specified type.
ObjectPathMethodType	A CSOM Object path that specifies a CSOM Object that is obtained by calling a method, including any parameters, of a CSOM Object.
ObjectPathObjectIdentityNameType	A CSOM Object path that specifies the CSOM Object that is obtained from a specified CSOM Object identity.
ObjectPathPropertyType	A CSOM Object path that specifies the CSOM Object that is obtained by accessing a property of a CSOM Object.
ObjectPathStaticMethodType	A CSOM Object path that specifies the CSOM Object that is obtained by calling a static method, including

Complex type	Description
	any parameters, of a CSOM Object type.
ObjectPathStaticPropertyType	A CSOM Object path that specifies the CSOM Object that is obtained by accessing a static property of a CSOM Object type.
ObjectPathsType	A CSOM Object path list.
QueryPropertyType	A child item query component that retrieves property data for a CSOM Object.
QueryType	A child item query that returns data about a specified CSOM Object.
RequestType	The root element of a **ProcessQueryIn** XML request, as specified in section 3.1.4.1.1.1.

3.1.4.1.3.1 ActionInstantiateObjectPathType

The **ActionInstantiateObjectPathType** complex type specifies a CSOM action that instantiates the CSOM Object associated with the specified CSOM Object path identifier. The protocol server MUST instantiate the CSOM Object with the specified CSOM Object path identifier only once. If the CSOM Object with the specified CSOM Object path identifier is already instantiated, this CSOM action MUST NOT instantiate the object again.

If the CSOM action executes successfully, the protocol server MUST append the CSOM action identifier and an **ObjectPathResponse** JSON value, as specified in section 3.1.4.1.8.2.1.6, to the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.1.1 Schema

```
<xs:complexType name="ActionInstantiateObjectPathType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.1.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

ObjectPathId: The identifier of the CSOM Object path, which is used to obtain the CSOM Object on which the CSOM action is performed.

3.1.4.1.3.1.3 Child Elements

None.

3.1.4.1.3.2 ActionInvokeMethodType

The **ActionInvokeMethodType** complex type specifies a CSOM action that calls a method on a CSOM Object, which is obtained from a specified CSOM Object path identifier, and passes any parameters for the method.

If the CSOM action executes successfully and the method that is called does not have a void return type, the protocol server MUST append the CSOM action identifier and a **MethodResponse** JSON value, as specified in section 3.1.4.1.8.2.1.4, to the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action executes successfully and the method that is called has a void return type, the protocol server MUST NOT modify the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.2.1 Schema

```
<xs:complexType name="ActionInvokeMethodType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
  <xs:attribute name="Version" type="xs:string" use="optional"/>
</xs:complexType>
```

3.1.4.1.3.2.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

Name: The name of the method to call on the CSOM Object that is specified by the **ObjectPathId** attribute. It MUST be the name of a valid method of the CSOM Object.

ObjectPathId: The identifier of the CSOM Object path, which is used to obtain the CSOM Object on which the CSOM action is performed.

Version: The version of the CSOM Object. If this attribute exists and the specified method requires that the object version is checked, the protocol server MUST first check this value against the current version of the CSOM Object. If the specified version does not match the actual version, the server MUST fail the method call and return an error with both the error code "-2130575339" and the error type name "Microsoft.SharePoint.Client.VersionConflictException".

3.1.4.1.3.2.3 Child Elements

Parameter: A parameter in the parameter list that is passed into the method call.

Parameters: The list of parameters that is passed into the method call.

3.1.4.1.3.3 ActionInvokeStaticMethodType

The **ActionInvokeStaticMethodType** complex type specifies a CSOM action that invokes a static method on a CSOM Object type, which is obtained from a specified CSOM Object path, and passes any parameters.

If the CSOM action executes successfully and the static method that is called does not have a void return type, the protocol server MUST append the CSOM action identifier and a **MethodResponse** JSON value, as specified in section 3.1.4.1.8.2.1.4, to the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action executes successfully and the static method that is called has a void return type, the protocol server MUST NOT modify the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.3.1 Schema

```
<xs:complexType name="ActionInvokeStaticMethodType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.3.2 Attributes

Id: The identifier of this CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

Name: The name of the static method to call on the CSOM Object type that is specified by the **TypeId** attribute. It MUST be the name of a valid static method of the CSOM Object type.

TypeId: A GUID that identifies the CSOM Object type on which to call the static method.

3.1.4.1.3.3.3 Child Elements

Parameter: A parameter in the parameter list to be passed into the static method call.

Parameters: The list of parameters that is passed into the static method call.

3.1.4.1.3.4 ActionObjectIdentityQueryType

The **ActionObjectIdentityQueryType** complex type specifies a CSOM action that obtains the object identity of a CSOM Object.

If the CSOM action executes successfully, the protocol server MUST append the CSOM action identifier and an **ObjectIdentityQueryResponse** JSON value, as specified in section 3.1.4.1.8.2.1.5, to the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, , as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.4.1 Schema

```
<xs:complexType name="ActionObjectIdentityQueryType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.4.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

ObjectPathId: The identifier of a CSOM Object path, which is used to obtain the CSOM Object on which the CSOM action is performed.

3.1.4.1.3.4.3 Child Elements

None.

3.1.4.1.3.5 ActionQueryType

The **ActionQueryType** complex type specifies a CSOM action that obtains the data resulting from a child item query on a CSOM Object.

If the CSOM action executes successfully, the protocol server MUST append the CSOM action identifier and a **QueryResponse** JSON value, as specified in section 3.1.4.1.8.2.1.7, to the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.5.1 Schema

```
<xs:complexType name="ActionQueryType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Query" type="mstns:QueryType"/>
    <xs:element name="ChildItemQuery" type="mstns:ChildItemQueryType" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.5.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

ObjectPathId: The identifier of a CSOM Object path, which is used to obtain the CSOM Object on which the action is performed.

3.1.4.1.3.5.3 Child Elements

ChildItemQuery: The child item query that is applied to the CSOM Object specified by the **ObjectPathId** attribute. If the **ChildItemQuery** element is present, the CSOM Object specified by the **ObjectPathId** attribute MUST be a CSOM Object collection.

Query: The child item query that is applied to the CSOM Object specified by the **ObjectPathId** attribute.

3.1.4.1.3.6 ActionSetPropertyType

The **ActionSetPropertyType** complex type specifies a CSOM action that sets the property of a CSOM Object to a specified parameter value.

If the CSOM action executes successfully, the protocol server MUST NOT modify the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.6.1 Schema

```
<xs:complexType name="ActionSetPropertyType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameter" type="mstns:MethodParameterType"/>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.6.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

Name: The name of the property to set. It MUST be the name of a valid property of the CSOM Object specified by the **ObjectPathId** attribute.

ObjectPathId: The identifier of a CSOM Object path, which is used to obtain the CSOM Object on which the CSOM action is performed.

3.1.4.1.3.6.3 Child Elements

Parameter: The parameter value to set the property to.

3.1.4.1.3.7 ActionSetStaticPropertyType

The **ActionSetStaticPropertyType** complex type specifies a CSOM action that sets the static property of a CSOM Object type to a specified parameter value.

If the CSOM action executes successfully, the protocol server MUST NOT modify the JSON response structure, as specified in section 3.1.4.1.8.2.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.7.1 Schema

```
<xs:complexType name="ActionSetStaticPropertyType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameter" type="mstns:MethodParameterType"/>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.7.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

Name: The name of the static property to set. It MUST be the name of a valid static property of the CSOM Object type specified by the **TypeId** attribute.

TypeId: A GUID that identifies the CSOM Object type that contains the static property to set.

3.1.4.1.3.7.3 Child Elements

Parameter: The parameter value to set the static property to.

3.1.4.1.3.8 ActionsType

The **ActionsType** complex type specifies an ordered list of CSOM actions to execute.

3.1.4.1.3.8.1 Schema

```
<xs:complexType name="ActionsType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
      <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
      <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
      <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
      <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
      <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
      <xs:element name="Query" type="mstns:ActionQueryType"/>
      <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
      <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
      <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.8.2 Attributes

None.

3.1.4.1.3.8.3 Child Elements

ConditionalScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScopeSimple: See section 3.1.4.1.6.1.3.

Method: See section 3.1.4.1.6.1.3.

ObjectIdentityQuery: See section 3.1.4.1.6.1.3.

ObjectPath: See section 3.1.4.1.6.1.3.

Query: See section 3.1.4.1.6.1.3.

SetProperty: See section 3.1.4.1.6.1.3.

SetStaticProperty: See section 3.1.4.1.6.1.3.

StaticMethod: See section 3.1.4.1.6.1.3.

3.1.4.1.3.9 ChildItemQueryType

The **ChildItemQueryType** complex type specifies the child item query that is applied to a CSOM Object collection.

3.1.4.1.3.9.1 Schema

```
<xs:complexType name="ChildItemQueryType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexContent>
    <xs:extension base="mstns:QueryType">
      <xs:sequence>
        <xs:element name="QueryableExpression" type="mstns:ExpressionQueryableExpressionType"
minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

3.1.4.1.3.9.2 Attributes

None.

3.1.4.1.3.9.3 Child Elements

QueryableExpression: The queryable expression that is applied to a CSOM Object collection. This results in a filtered subset of child CSOM Objects.

3.1.4.1.3.10 ConditionalScopeType

The **ConditionalScopeType** complex type specifies a CSOM action that evaluates a conditional expression that does not have any input parameters. Depending on the outcome of the evaluation, it executes an ordered list of CSOM actions for either the "true" or "false" case, if a list of such actions is present.

If the CSOM action executes successfully, the protocol server MUST append the CSOM action identifier and a **ConditionalScopeResponse** JSON value, as specified in section 3.1.4.1.8.2.1.1, to the JSON response structure, as specified in section 3.1.4.1.8.2, before it executes the ordered list of CSOM actions specified inside the **IfTrueScope** or **IfFalseScope** element.

If the CSOM action fails and the exception is not handled by **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.10.1 Schema

```
<xs:complexType name="ConditionalScopeType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Test">
      <xs:complexType>
        <xs:all>
          <xs:element name="Body" type="mstns:ExpressionType"/>
        </xs:all>
      </xs:complexType>
    </xs:element>
    <xs:element name="IfTrueScope" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
            <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
            <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
            <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
            <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
            <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
            <xs:element name="Query" type="mstns:ActionQueryType"/>
            <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
            <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
            <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
          </xs:choice>
        </xs:sequence>
        <xs:attribute name="Id" type="mstns:IdType" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="IfFalseScope" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
            <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
            <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
            <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
            <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
            <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
            <xs:element name="Query" type="mstns:ActionQueryType"/>
            <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
            <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
            <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
          </xs:choice>
        </xs:sequence>
        <xs:attribute name="Id" type="mstns:IdType" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.10.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

IfFalseScope::Id: The identifier for the "false" scope. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request. Use of this attribute is implementation-specific.

IfTrueScope::Id: The identifier for the "true" scope. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request. Use of this attribute is implementation-specific.

3.1.4.1.3.10.3 Child Elements

Body: The conditional expression to evaluate.

ConditionalScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScopeSimple: See section 3.1.4.1.6.1.3.

IfFalseScope: The ordered list of CSOM actions to execute if the conditional expression specified by the **Body** attribute evaluates to "false".

IfTrueScope: The ordered list of CSOM actions to execute if the conditional expression specified by the **Body** attribute evaluates to "true".

Method: See section 3.1.4.1.6.1.3.

ObjectIdentityQuery: See section 3.1.4.1.6.1.3.

ObjectPath: See section 3.1.4.1.6.1.3.

Query: See section 3.1.4.1.6.1.3.

SetProperty: See section 3.1.4.1.6.1.3.

SetStaticProperty: See section 3.1.4.1.6.1.3.

StaticMethod: See section 3.1.4.1.6.1.3.

Test: An element that contains the conditional expression to evaluate.

3.1.4.1.3.11 ExceptionHandlingScopeSimpleType

The **ExceptionHandlingScopeSimpleType** complex type specifies a CSOM action that attempts to execute an ordered list of CSOM actions.

If an unhandled exception does not occur, all of the CSOM actions in the list are executed.

If an unhandled exception occurs while executing a CSOM action in the list, the protocol server MUST catch the exception and MUST NOT execute any of the remaining CSOM actions in the list.

If this CSOM action executes successfully, even if the protocol catches the exception while executing the CSOM action list, the protocol server MUST append the CSOM action identifier of this CSOM action and an **ExceptionHandlingScopeSimpleResponse** JSON value, as specified in section 3.1.4.1.8.2.1.3, to the JSON response structure, as specified in section 3.1.4.1.8.2, after executing the CSOM actions specified inside this CSOM action.

If this CSOM action fails and the exception is not handled by an outer **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.11.1 Schema

```
<xs:complexType name="ExceptionHandlingScopeSimpleType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
      <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
      <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
      <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
      <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
      <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
      <xs:element name="Query" type="mstns:ActionQueryType"/>
      <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
      <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
      <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.11.2 Attributes

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

3.1.4.1.3.11.3 Child Elements

ConditionalScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScopeSimple: See section 3.1.4.1.6.1.3.

Method: See section 3.1.4.1.6.1.3.

ObjectIdentityQuery: See section 3.1.4.1.6.1.3.

ObjectPath: See section 3.1.4.1.6.1.3.

Query: See section 3.1.4.1.6.1.3.

SetProperty: See section 3.1.4.1.6.1.3.

SetStaticProperty: See section 3.1.4.1.6.1.3.

StaticMethod: See section 3.1.4.1.6.1.3.

3.1.4.1.3.12 ExceptionHandlingScopeType

The **ExceptionHandlingScopeType** complex type specifies a CSOM action that attempts to execute an ordered list of CSOM actions specified by a **TryScope** element.

If an unhandled exception does not occur, all of the CSOM actions in the list are executed. In addition, the ordered list of CSOM actions specified in the **FinallyScope** element, if present, is executed.

If an unhandled exception occurs while executing an action specified in a **TryScope** element, the protocol server MUST catch the exception and MUST NOT execute the remaining CSOM actions specified in the **TryScope** element. In addition, the ordered list of CSOM actions specified in the **CatchScope** element, if present, is executed, and then the ordered list of CSOM actions specified in the **FinallyScope** element, if present, is executed.

If this CSOM action executes successfully, even if the protocol server catches the exception while executing actions inside **TryScope** element, the protocol server MUST append the CSOM action identifier of this CSOM action and an **ExceptionHandlingScopeResponse** JSON value, as specified in section 3.1.4.1.8.2.1.2, to the JSON response structure, as specified in section 3.1.4.1.8.2, after executing the CSOM actions specified inside this CSOM action.

If this CSOM action fails and the exception is not handled by an outer **ExceptionHandlingScopeSimpleType**, as specified in section 3.1.4.1.3.11, or **ExceptionHandlingScopeType**, as specified in section 3.1.4.1.3.12, the protocol server MUST set the exception information in the **ErrorInfo** member of the **Response Header JSON Value**, as specified in section 3.1.4.1.8.1.5.

3.1.4.1.3.12.1 Schema

```
<xs:complexType name="ExceptionHandlingScopeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="TryScope">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
            <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
            <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
            <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
            <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
            <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
            <xs:element name="Query" type="mstns:ActionQueryType"/>
            <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
            <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
            <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
          </xs:choice>
        </xs:sequence>
        <xs:attribute name="Id" type="mstns:IdType" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="CatchScope" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
            <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
            <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
            <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
            <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
            <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
            <xs:element name="Query" type="mstns:ActionQueryType"/>
            <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
            <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
            <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
          </xs:choice>
        </xs:sequence>
        <xs:attribute name="Id" type="mstns:IdType" use="required"/>
      </xs:complexType>
    </xs:element>

    <xs:element name="FinallyScope" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
```

```
            <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
            <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
            <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
            <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
            <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
            <xs:element name="Query" type="mstns:ActionQueryType"/>
            <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
            <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
            <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
          </xs:choice>
        </xs:sequence>
        <xs:attribute name="Id" type="mstns:IdType" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.12.2 Attributes

CatchScope::Id: The identifier of the catch scope. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request. Use of this attribute is implementation-specific.

FinallyScope::Id: The identifier of the finally scope. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request. Use of this attribute is implementation-specific.

Id: The identifier of the CSOM action. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request.

TryScope::Id: The identifier of the try scope. This value MUST be unique from all other **IdType** values, as specified in section 3.1.4.1.4.3, in the request. Use of this attribute is implementation-specific.

3.1.4.1.3.12.3 Child Elements

CatchScope: Optional. The list of CSOM actions to execute if an unhandled exception occurs while executing the list of CSOM actions that are specified by the **TryScope** element. If an unhandled exception does not occur, the protocol server MUST NOT execute the list of CSOM actions specified by this element.

ConditionalScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScope: See section 3.1.4.1.6.1.3.

ExceptionHandlingScopeSimple: See section 3.1.4.1.6.1.3.

FinallyScope: Optional. The list of CSOM actions to execute after executing the CSOM actions specified by the **TryScope** and **CatchScope** elements.

Method: See section 3.1.4.1.6.1.3.

ObjectIdentityQuery: See section 3.1.4.1.6.1.3.

ObjectPath: See section 3.1.4.1.6.1.3.

Query: See section 3.1.4.1.6.1.3.

SetProperty: See section 3.1.4.1.6.1.3.

SetStaticProperty: See section 3.1.4.1.6.1.3.

StaticMethod: See section 3.1.4.1.6.1.3.

TryScope: The ordered list of CSOM actions to execute. If an unhandled exception occurs while executing a CSOM action, the protocol server MUST catch the exception and MUST NOT execute the remaining CSOM actions.

3.1.4.1.3.13 ExpressionConvertExpressionType

The **ExpressionConvertExpressionType** complex type is a component of an expression. It takes the evaluated value of an expression and converts that value to a specified type. Its inner element is the value to convert.

3.1.4.1.3.13.1 Schema

```
<xs:complexType name="ExpressionConvertExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Type" use="optional">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Boolean"/>
        <xs:enumeration value="Byte"/>
        <xs:enumeration value="SByte"/>
        <xs:enumeration value="Char"/>
        <xs:enumeration value="Int16"/>
        <xs:enumeration value="UInt16"/>
        <xs:enumeration value="Int32"/>
        <xs:enumeration value="UInt32"/>
        <xs:enumeration value="Int64"/>
        <xs:enumeration value="UInt64"/>
        <xs:enumeration value="DateTime"/>
        <xs:enumeration value="Single"/>
        <xs:enumeration value="Double"/>
        <xs:enumeration value="Decimal"/>
        <xs:enumeration value="TimeSpan"/>

        <xs:enumeration value="String"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="TypeId" use="optional" type="mstns:TypeIdGuidType"/>
```

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

3.1.4.1.3.13.2 Attributes

Type: The type to convert the value to. The conversion behavior and the validity of a specific conversion are implementation-specific.

TypeId: A GUID that identifies the CSOM Object type.

3.1.4.1.3.13.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs: See section 3.1.4.1.6.2.3

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

3.1.4.1.3.14 ExpressionLeftRightOperandExpressionType

The **ExpressionLeftRightOperandExpressionType** complex type is a component of an expression. It represents the left and right operands of a binary expression. Its first immediate child element MUST contain the left operand of the binary expression. Its second immediate child element MUST contain the right operand of the binary expression.

3.1.4.1.3.14.1 Schema

```
<xs:complexType name="ExpressionLeftRightOperandExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
```

```
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.14.2 Attributes

None.

3.1.4.1.3.14.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs: See section 3.1.4.1.6.2.3

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

3.1.4.1.3.15 ExpressionMethodExpressionType

The **ExpressionMethodExpressionType** complex type is a component of an expression. It represents a method to call, including any applicable parameters, on the CSOM Object specified by its first immediate child element.

3.1.4.1.3.15.1 Schema

```
<xs:complexType name="ExpressionMethodExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="NOT" type="mstns:ExpressionType"/>
            <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>

            <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
            <xs:element name="ExpressionProperty"
type="mstns:ExpressionPropertyExpressionType"/>
```

```
            <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
            <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
            <xs:element name="ExpressionConvert"
type="mstns:ExpressionConvertExpressionType"/>
            <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
            <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
          </xs:choice>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.15.2 Attributes

Name: The name of the method to call. It MUST be the name of a valid method of the CSOM Object.

3.1.4.1.3.15.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs:

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

Parameters: The list of parameters to be passed when calling the method. Each parameter is an **ExpressionGroup** element, as specified in section 3.1.4.1.6.2.

3.1.4.1.3.16 ExpressionParameterExpressionType

The **ExpressionParameterExpressionType** complex type is a component of an expression. It represents the input parameter of an expression.

3.1.4.1.3.16.1 Schema

```
<xs:complexType name="ExpressionParameterExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>
```

3.1.4.1.3.16.2 Attributes

Name: The name of the input parameter of an expression.

3.1.4.1.3.16.3 Child Elements

None.

3.1.4.1.3.17 ExpressionPropertyExpressionType

The **ExpressionPropertyExpressionType** complex type is a component of an expression. It represents a property of a CSOM Object or CSOM value object. Its first immediate child element specifies the CSOM Object or CSOM value object that the property applies to.

3.1.4.1.3.17.1 Schema

```
<xs:complexType name="ExpressionPropertyExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.17.2 Attributes

Name: The name of the property of the CSOM Object or CSOM value object that is specified by its first immediate child element. It MUST be the name of a valid property of the CSOM Object or CSOM value object.

3.1.4.1.3.17.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

3.1.4.1.3.18 ExpressionQueryableExpressionType

The **ExpressionQueryableExpressionType** complex type is a component of an expression. It represents the queryable expression component of a child item query that is applied to a CSOM Object collection for retrieving a filtered subset of child CSOM Objects.

3.1.4.1.3.18.1 Schema

```
<xs:complexType name="ExpressionQueryableExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
      <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
      <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="QueryableObject">
        <xs:complexType/>
      </xs:element>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.18.2 Attributes

None.

3.1.4.1.3.18.3 Child Elements

OfType: See section 3.1.4.1.6.3.3

OrderBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies the sorted set of CSOM Objects in ascending order.

OrderByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies the sorted set of CSOM Objects in descending order.

QueryableObject: See section 3.1.4.1.6.3.3.

Take: See section 3.1.4.1.6.3.3.

ThenBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies the sorted set of CSOM Objects in ascending order.

ThenByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified section 3.1.4.1.3.21, which specifies the sorted set of CSOM Objects in descending order.

Where: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies the filtered set of CSOM Objects.

3.1.4.1.3.19 ExpressionQueryableOfTypeType

The **ExpressionQueryableOfTypeType** complex type is a component of an expression. It represents a filtered set of CSOM Objects that are of a specific type from an inner queryable expression. Its first immediate child element specifies the inner queryable expression

3.1.4.1.3.19.1 Schema

```
<xs:complexType name="ExpressionQueryableOfTypeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
      <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
      <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="QueryableObject">
        <xs:complexType/>
      </xs:element>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Type" type="mstns:TypeFunctionTypeType" use="optional"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="optional"/>
</xs:complexType>
```

3.1.4.1.3.19.2 Attributes

Type: The name of the type.

TypeId: A GUID that identifies the CSOM Object type

3.1.4.1.3.19.3 Child Elements

OfType: See section 3.1.4.1.6.3.3

OrderBy: See section 3.1.4.1.6.3.3

OrderByDescending: See section 3.1.4.1.6.3.3

QueryableObject: See section 3.1.4.1.6.3.3

Take: See section 3.1.4.1.6.3.3

ThenBy: See section 3.1.4.1.6.3.3

ThenByDescending: See section 3.1.4.1.6.3.3

Where: See section 3.1.4.1.6.3.3

3.1.4.1.3.20 ExpressionQueryableTakeType

The **ExpressionQueryableTakeType** complex type is a component of an expression. It represents a filtered set of CSOM Objects that are returned from an inner queryable expression that returns a specified number of CSOM Objects. Its first immediate child element specifies the inner queryable expression.

3.1.4.1.3.20.1 Schema

```
<xs:complexType name="ExpressionQueryableTakeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
      <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
      <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="QueryableObject">
        <xs:complexType/>
      </xs:element>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Count" type="xs:nonNegativeInteger"/>
</xs:complexType>
```

3.1.4.1.3.20.2 Attributes

Count: The maximum number of child CSOM Objects to return. If the total number of CSOM Objects that are returned from the inner queryable expression is larger than this value, the protocol server MUST return only the first *n* CSOM Objects, where *n* is the value specified by the **Count** attribute. Otherwise, the protocol server MUST return all of the CSOM Objects.

3.1.4.1.3.20.3 Child Elements

OfType: See section 3.1.4.1.6.3.3

OrderBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specified the sorted set of CSOM Objects in ascending order.

OrderByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies the sorted set of CSOM Objects in descending order.

QueryableObject: See section 3.1.4.1.6.3.3.

Take: See section 3.1.4.1.6.3.3.

ThenBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specified the sorted set of CSOM Objects in ascending order.

ThenByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specified the sorted set of CSOM Objects in descending order.

Where: An expression of the **ExpressionQueryableWhereType** type, as specified section 3.1.4.1.3.21, which specified the filtered set of CSOM Objects.

3.1.4.1.3.21 ExpressionQueryableWhereType

The **ExpressionQueryableWhereType** complex type is a component of an expression. It represents a filtered or sorted set of CSOM Objects that are returned from an inner queryable expression. When it is a filtered set of CSOM Objects, the protocol server MUST return only those CSOM Objects that meet the conditions that are defined by the expression. When it is a sorted set of CSOM Objects, the protocol server MUST return the CSOM Objects sorted by the value specified by the expression.

3.1.4.1.3.21.1 Schema

```
<xs:complexType name="ExpressionQueryableWhereType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Test">
      <xs:complexType>
        <xs:all>
          <xs:element name="Body" type="mstns:ExpressionType"/>
          <xs:element name="Parameters">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Parameter">
                  <xs:complexType>
                    <xs:attribute name="Name" type="xs:string"/>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:all>
      </xs:complexType>
    </xs:element>
    <xs:element name="Object" type="mstns:ExpressionQueryableExpressionType"/>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.21.2 Attributes

Name: The name of the input parameter in the expression.

3.1.4.1.3.21.3 Child Elements

Body: The expression to evaluate, the type of this element is **ExpressionType** as specified in section 3.1.4.1.3.24.

Object: The set of child CSOM Objects to filter.

Parameter: The input parameter in the expression. Inside the expression **Body** element, when the input parameter is referenced by an element whose type is **ExpressionParameterExpressionType**, as specified in section 3.1.4.1.3.16, and whose name attribute is same as the name of the input parameter, the value of the input parameter is the child object in the set of CSOM Objects that are returned from an inner queryable expression.

Parameters: The collection of input parameters for the expression.

Test: An element that contains the expression to evaluate.

3.1.4.1.3.22 ExpressionStaticMethodExpressionType

The **ExpressionStaticMethodExpressionType** complex type is a component of an expression. It represents a static method to call, including any parameters, on a CSOM Object type.

3.1.4.1.3.22.1 Schema

```
<xs:complexType name="ExpressionStaticMethodExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
            <xs:element name="NOT" type="mstns:ExpressionType"/>
            <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
            <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
            <xs:element name="ExpressionProperty"
type="mstns:ExpressionPropertyExpressionType"/>
            <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
            <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
            <xs:element name="ExpressionConvert"
type="mstns:ExpressionConvertExpressionType"/>
            <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
            <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
          </xs:choice>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.22.2 Attributes

Name: The name of the static method to call on the CSOM Object type that is specified by the **TypeId** attribute. It MUST be the name of a valid static method of the specified CSOM Object type.

TypeId: A GUID that identifies the CSOM Object type on which to call the static method.

3.1.4.1.3.22.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs: See section 3.1.4.1.6.2.3.

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

Parameters: The list of parameters to be passed when calling the static method. Each parameter is an **ExpressionGroup** element, as specified in section 3.1.4.1.6.2.

3.1.4.1.3.23 ExpressionStaticPropertyExpressionType

The **ExpressionStaticPropertyExpressionType** complex type is a component of an expression. It represents a static property of a CSOM Object type.

3.1.4.1.3.23.1 Schema

```
<xs:complexType name="ExpressionStaticPropertyExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence/>
  <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.23.2 Attributes

Name: The name of the static property of the CSOM Object type that is specified by the **TypeId** attribute. It MUST be the name of a valid static property of the CSOM Object type.

TypeId: A GUID that identifies the CSOM Object type.

3.1.4.1.3.23.3 Child Elements

None.

3.1.4.1.3.24 ExpressionType

The **ExpressionType** complex type represents an expression, which is a combination of operators, symbols, constants, literal values, functions, names of fields or columns, controls, and properties that evaluates to a single value.

3.1.4.1.3.24.1 Schema

```
<xs:complexType name="ExpressionType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.24.2 Attributes

None.

3.1.4.1.3.24.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs: See section 3.1.4.1.6.2.3.

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

3.1.4.1.3.25 ExpressionTypeIsExpressionType

The **ExpressionTypeIsExpressionType** complex type is a component of an expression. It represents a test result indicating whether the value is a specific type.

3.1.4.1.3.25.1 Schema

```
<xs:complexType name="ExpressionTypeIsExpressionType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="Type" use="optional" type="mstns:TypeFunctionTypeType"/>
  <xs:attribute name="TypeId" use="optional" type="mstns:TypeIdGuidType"/>
</xs:complexType>
```

3.1.4.1.3.25.2 Attributes

Type: A name that identifiers the type. The value will be checked whether it is that type.

TypeId: A GUID that identifies the CSOM Object type. The value will be checked whether it is that type.

3.1.4.1.3.25.3 Child Elements

AND: See section 3.1.4.1.6.2.3.

EQ: See section 3.1.4.1.6.2.3.

ExpressionConstant: See section 3.1.4.1.6.2.3.

ExpressionConvert: See section 3.1.4.1.6.2.3.

ExpressionMethod: See section 3.1.4.1.6.2.3.

ExpressionParameter: See section 3.1.4.1.6.2.3.

ExpressionProperty: See section 3.1.4.1.6.2.3.

ExpressionStaticMethod: See section 3.1.4.1.6.2.3.

ExpressionStaticProperty: See section 3.1.4.1.6.2.3.

ExpressionTypeIs: See section 3.1.4.1.6.2.3.

GE: See section 3.1.4.1.6.2.3.

GT: See section 3.1.4.1.6.2.3.

LE: See section 3.1.4.1.6.2.3.

LT: See section 3.1.4.1.6.2.3.

NE: See section 3.1.4.1.6.2.3.

NOT: See section 3.1.4.1.6.2.3.

OR: See section 3.1.4.1.6.2.3.

3.1.4.1.3.26 MethodParameterType

The **MethodParameterType** complex type is a parameter that can be passed when calling a method or setting a property. The **MethodParameterType** complex type MUST contain only one instance of the **ObjectPathId**, **TypeId**, and **Type** attributes. Additionally, it MUST conform to one of the following XML types:

- **MethodParameterArrayType**
- **MethodParameterBooleanType**
- **MethodParameterByteArrayType**
- **MethodParameterByteType**
- **MethodParameterCharType**
- **MethodParameterDateTimeType**
- **MethodParameterDictionaryType**
- **MethodParameterDoubleType**
- **MethodParameterEnumType**
- **MethodParameterGuidType**
- **MethodParameterInt16Type**
- **MethodParameterInt32Type**
- **MethodParameterInt64Type**
- **MethodParameterNullType**
- **MethodParameterNumberType**
- **MethodParameterObjectPathType**
- **MethodParameterSByteType**
- **MethodParameterSingleType**

- **MethodParameterStringType**
- **MethodParameterUInt16Type**
- **MethodParameterUInt32Type**
- **MethodParameterUInt64Type**
- **MethodParameterUnspecifiedType**
- **MethodParameterValueObjectType**

3.1.4.1.3.26.1 Schema

```
<xs:complexType name="MethodParameterType" mixed="true"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="mstns:MethodParameterType">
            <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
    <xs:element name="Object" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
    <xs:element name="Include" minOccurs="0" maxOccurs="1">
      <xs:complexType>
        <xs:attribute name="href" use="required" type="xs:string"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="optional"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="optional"/>
  <xs:attribute name="Type" type="mstns:MethodParameterTypeType" use="optional"/>
</xs:complexType>
```

3.1.4.1.3.26.2 Attributes

href: Specifies the MIME part that contains the stream data. The format of the value is specified in section 2.2.4.6.1.

Name: The name of a property of the object. If the **Property** element is present, this value MUST be a CSOM value object.

ObjectPathId: The identifier of a CSOM Object path to the CSOM Object that is used as a parameter.

Type: The type of parameter.

TypeId: A **TypeIdGuidType** simple type, as specified in section 3.1.4.1.4.11, that represents the GUID of the CSOM type, if the parameter is a CSOM value object or CSOM Object.

3.1.4.1.3.26.3 Child Elements

Include: Specifies the MIME part that contains the stream data.

Object: An array element. If the **Object** element is present, this value MUST be a CSOM array.

Property: The name of a property of the parameter object. If the **Property** element is present, this value MUST be a CSOM value object.

3.1.4.1.3.26.4 MethodParameterArrayType

The **MethodParameterArrayType** XML type is a parameter that represents a CSOM array.

3.1.4.1.3.26.4.1 Schema

```
<xs:complexType name="MethodParameterArrayType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Object" minOccurs="0" maxOccurs="unbounded"
type="mstns:MethodParameterType"/>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Array"/>
</xs:complexType>
```

3.1.4.1.3.26.4.2 Attributes

Type: The type of the parameter. This value MUST be "Array".

3.1.4.1.3.26.4.3 Child Elements

Object: A value in an array of parameter types.

3.1.4.1.3.26.5 MethodParameterBooleanType

The **MethodParameterBooleanType** XML type is a parameter that represents a **CSOM Boolean** value.

3.1.4.1.3.26.5.1 Schema

```
<xs:complexType name="MethodParameterBooleanType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:boolean">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Boolean"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.5.2 Attributes

Type: The type of the parameter. This value MUST be "Boolean".

3.1.4.1.3.26.5.3 Child Elements

None.

3.1.4.1.3.26.6 MethodParameterByteArrayType

The **MethodParameterByteArrayType** XML type is a parameter that represents a CSOM binary value.

3.1.4.1.3.26.6.1 Schema

```
<xs:complexType name="MethodParameterByteArrayType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:base64Binary">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Base64Binary"/>
    </xs:extension>
  </xs:simpleContent>
```

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

3.1.4.1.3.26.6.2 Attributes

Type: The type of the parameter. This value MUST be "Base64Binary".

3.1.4.1.3.26.6.3 Child Elements

None.

3.1.4.1.3.26.7 MethodParameterByteType

The **MethodParameterByteType** XML type is a parameter that represents a **CSOM Byte** value.

3.1.4.1.3.26.7.1 Schema

```
<xs:complexType name="MethodParameterByteType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedByte">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Byte"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.7.2 Attributes

Type: The type of the parameter. This value MUST be "Byte".

3.1.4.1.3.26.7.3 Child Elements

None.

3.1.4.1.3.26.8 MethodParameterCharType

The **MethodParameterCharType** XML type is a parameter that represents a **CSOM Char** value.

3.1.4.1.3.26.8.1 Schema

```
<xs:complexType name="MethodParameterCharType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Char"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.8.2 Attributes

Type: The type of the parameter. This value MUST be "Char".

3.1.4.1.3.26.8.3 Child Elements

None.

3.1.4.1.3.26.9 MethodParameterDateTimeType

The **MethodParameterDateTimeType** XML type is a parameter that represents a **CSOM DateTime** value.

3.1.4.1.3.26.9.1 Schema

```
<xs:complexType name="MethodParameterDateTimeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:dateTime">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="DateTime"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.9.2 Attributes

Type: The type of the parameter. This value MUST be "DateTime".

3.1.4.1.3.26.9.3 Child Elements

None.

3.1.4.1.3.26.10 MethodParameterDictionaryType

The **MethodParameterDictionaryType** XML type is a parameter that represents a CSOM dictionary, which is a collection of key/value pairs.

3.1.4.1.3.26.10.1 Schema

```
<xs:complexType name="MethodParameterDictionaryType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="mstns:MethodParameterType">
            <xs:attribute name="Name" type="xs:string" use="required"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Dictionary"/>
</xs:complexType>
```

3.1.4.1.3.26.10.2 Attributes

Name: The name of the key/value pair.

Type: The type of the parameter. This value MUST be "Dictionary".

3.1.4.1.3.26.10.3 Child Elements

Property: An element that contains one key/value pair.

3.1.4.1.3.26.11 MethodParameterDoubleType

The **MethodParameterDoubleType** XML type is a parameter that represents a **CSOM Double** value.

3.1.4.1.3.26.11.1 Schema

```
<xs:complexType name="MethodParameterDoubleType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:double">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Double"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.11.2 Attributes

Type: The type of the parameter. This value MUST be "Double".

3.1.4.1.3.26.11.3 Child Elements

None.

3.1.4.1.3.26.12 MethodParameterEnumType

The **MethodParameterEnumType** XML type is a parameter that represents a CSOM enumeration.

3.1.4.1.3.26.12.1 Schema

```
<xs:complexType name="MethodParameterEnumType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:integer">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Enum"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.12.2 Attributes

Type: The type of the parameter. This value MUST be "Enum".

3.1.4.1.3.26.12.3 Child Elements

None.

3.1.4.1.3.26.13 MethodParameterGuidType

The **MethodParameterGuidType** XML type is a parameter that represents a CSOM GUID.

3.1.4.1.3.26.13.1 Schema

```
<xs:complexType name="MethodParameterGuidType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="mstns:GuidType">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Guid"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.13.2 Attributes

Type: The type of the parameter. This value MUST be "Guid".

3.1.4.1.3.26.13.3 Child Elements

None.

3.1.4.1.3.26.14 MethodParameterInt16Type

The **MethodParameterInt16Type** XML type is a parameter that represents a **CSOM Int16** value.

3.1.4.1.3.26.14.1 Schema

```
<xs:complexType name="MethodParameterInt16Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:short">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int16"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.14.2 Attributes

Type: The type of the parameter. This value MUST be "Int16".

3.1.4.1.3.26.14.3 Child Elements

None.

3.1.4.1.3.26.15 MethodParameterInt32Type

The **MethodParameterInt32Type** XML type is a parameter that represents a CSOM Int32 value.

3.1.4.1.3.26.15.1 Schema

```
<xs:complexType name="MethodParameterInt32Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:int">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int32"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.15.2 Attributes

Type: The type of the parameter. This value MUST be "Int32".

3.1.4.1.3.26.15.3 Child Elements

None.

3.1.4.1.3.26.16 MethodParameterInt64Type

The **MethodParameterInt64Type** XML type is a parameter that represents a **CSOM Int64** value.

3.1.4.1.3.26.16.1 Schema

```
<xs:complexType name="MethodParameterInt64Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:long">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int64"/>
    </xs:extension>
  </xs:simpleContent>
```

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

3.1.4.1.3.26.16.2 Attributes

Type: The type of the parameter. This value MUST be "Int64".

3.1.4.1.3.26.16.3 Child Elements

None.

3.1.4.1.3.26.17 MethodParameterNullType

The **MethodParameterNullType** XML type is a parameter that represents a CSOM null value.

3.1.4.1.3.26.17.1 Schema

```
<xs:complexType name="MethodParameterNullType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Null"/>
</xs:complexType>
```

3.1.4.1.3.26.17.2 Attributes

Type: The type of the parameter. This value MUST be "Null".

3.1.4.1.3.26.17.3 Child Elements

None.

3.1.4.1.3.26.18 MethodParameterNumberType

The **MethodParameterNumberType** XML type is a parameter that represents a CSOM Byte, CSOM Double, CSOM Int16, CSOM Int32, CSOM Int64, **CSOM SByte**, **CSOM Single**, **CSOM UInt16**, **CSOM UInt32**, or **CSOM UInt64** value.

3.1.4.1.3.26.18.1 Schema

```
<xs:complexType name="MethodParameterNumberType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Number"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.18.2 Attributes

Type: The type of the parameter. This value MUST be "Number".

3.1.4.1.3.26.18.3 Child Elements

None.

3.1.4.1.3.26.19 MethodParameterObjectPathType

The **MethodParameterObjectPathType** XML type is a parameter that represents the CSOM Object path for a CSOM Object.

3.1.4.1.3.26.19.1 Schema

```
<xs:complexType name="MethodParameterObjectPathType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.26.19.2 Attributes

ObjectPathId: The identifier of a CSOM Object path for a CSOM Object.

3.1.4.1.3.26.19.3 Child Elements

None.

3.1.4.1.3.26.20 MethodParameterSByteType

The **MethodParameterSByteType** XML type is a parameter that represents a CSOM SByte value.

3.1.4.1.3.26.20.1 Schema

```
<xs:complexType name="MethodParameterSByteType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:byte">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="SByte"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.20.2 Attributes

Type: The type of the parameter. This value MUST be "SByte".

3.1.4.1.3.26.20.3 Child Elements

None.

3.1.4.1.3.26.21 MethodParameterSingleType

The **MethodParameterSingleType** XML type is a parameter that represents a CSOM Single value.

3.1.4.1.3.26.21.1 Schema

```
<xs:complexType name="MethodParameterSingleType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:float">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Single"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.21.2 Attributes

Type: The type of the parameter. This value MUST be "Single".

3.1.4.1.3.26.21.3 Child Elements

None.

3.1.4.1.3.26.22 MethodParameterStringType

The **MethodParameterStringType** XML type is a parameter that represents a CSOM String value.

3.1.4.1.3.26.22.1 Schema

```
<xs:complexType name="MethodParameterStringType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="String"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.22.2 Attributes

Type: The type of the parameter. This value MUST be "String".

3.1.4.1.3.26.22.3 Child Elements

None.

3.1.4.1.3.26.23 MethodParameterUInt16Type

The **MethodParameterUInt16Type** XML type is a parameter that represents a CSOM UInt16 value.

3.1.4.1.3.26.23.1 Schema

```
<xs:complexType name="MethodParameterUInt16Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedShort">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt16"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.23.2 Attributes

Type: The type of the parameter. This value MUST be "UInt16".

3.1.4.1.3.26.23.3 Child Elements

None.

3.1.4.1.3.26.24 MethodParameterUInt32Type

The **MethodParameterUInt32Type** XML type is a parameter that represents a CSOM UInt32 value.

3.1.4.1.3.26.24.1 Schema

```
<xs:complexType name="MethodParameterUInt32Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedInt">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt32"/>
    </xs:extension>
  </xs:simpleContent>

</xs:complexType>
```

3.1.4.1.3.26.24.2 Attributes

Type: The type of the parameter. This value MUST be "UInt32".

3.1.4.1.3.26.24.3 Child Elements

None.

3.1.4.1.3.26.25 MethodParameterUInt64Type

The **MethodParameterUInt64Type** XML type is a parameter that represents a CSOM UInt64 value.

3.1.4.1.3.26.25.1 Schema

```
<xs:complexType name="MethodParameterUInt64Type" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedLong">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt64"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.25.2 Attributes

Type: The type of parameter. This value MUST be "UInt64".

3.1.4.1.3.26.25.3 Child Elements

None.

3.1.4.1.3.26.26 MethodParameterUnspecifiedType

The **MethodParameterUnspecifiedType** XML type is a parameter that represents a value that does not have a specified type; the protocol client did not specify the type of value that is being passed. The protocol server MUST determine the type of the parameter, based on the method definition, and then parse the parameter from the **string**.

3.1.4.1.3.26.26.1 Schema

```
<xs:complexType name="MethodParameterUnspecifiedType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Unspecified"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.26.2 Attributes

Type: A fixed string value that MUST be "Unspecified".

3.1.4.1.3.26.26.3 Child Elements

None.

3.1.4.1.3.26.27 MethodParameterValueObjectType

The **MethodParameterValueObjectType** XML type is a parameter that represents a CSOM value object.

3.1.4.1.3.26.27.1 Schema

```
<xs:complexType name="MethodParameterValueObjectType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="mstns:MethodParameterType">
            <xs:attribute name="Name" type="xs:string" use="required"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.26.27.2 Attributes

Name: The name of the property of the CSOM value object.

TypeId: A GUID that identifies the type of the CSOM value object.

3.1.4.1.3.26.27.3 Child Elements

Property: The value of the property that is specified by the **Property.Name** attribute.

3.1.4.1.3.26.28 MethodParameterBinaryType

The **MethodParameterBinaryType**<4> XML type is a parameter that represents a **CSOM Stream** value.

3.1.4.1.3.26.28.1 Schema

```
<xs:complexType name="MethodParameterBinaryType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Include" minOccurs="1" maxOccurs="1">
      <xs:complexType>
        <xs:attribute name="href" use="required" type="xs:string"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.26.28.2 Attributes

href: Specifies the MIME part that contains the stream data. The format of the value is specified in section 2.2.4.6.1.

3.1.4.1.3.26.28.3 Child Elements

Include: Specifies the MIME part that contains the stream data.

3.1.4.1.3.26.29 MethodParameterDecimalType

The **MethodParameterDecimalType**<5> XML type is a parameter that represents a CSOM Decimal value.

3.1.4.1.3.26.29.1 Schema

```
<xs:complexType name="MethodParameterDecimalType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Decimal"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.29.2 Attributes

Type: A fixed string value that MUST be "Decimal".

3.1.4.1.3.26.29.3 Child Elements

None.

3.1.4.1.3.26.30 MethodParameterTimeSpanType

The **MethodParameterTimeSpanType**<6> XML type is a parameter that represents a CSOM TimeSpan value.

3.1.4.1.3.26.30.1 Schema

```
<xs:complexType name="MethodParameterTimeSpanType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleContent>
    <xs:extension base="xs:duration">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="TimeSpan"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

3.1.4.1.3.26.30.2 Attributes

Type: A fixed string value that MUST be "TimeSpan".

3.1.4.1.3.26.30.3 Child Elements

None.

3.1.4.1.3.27 ObjectPathConstructorType

The **ObjectPathConstructorType** complex type is a CSOM Object path. It specifies the CSOM Object that is obtained by invoking a constructor method, including any parameters, of a specified type.

3.1.4.1.3.27.1 Schema

```
<xs:complexType name="ObjectPathConstructorType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
```

```
          <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.27.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

TypeId: The GUID of the CSOM Object type, which is used to identify the constructor method to call.

3.1.4.1.3.27.3 Child Elements

Parameter: A parameter in the parameter list that is passed to the constructor method of the type specified by the **TypeId** attribute.

Parameters: The list of parameters that are passed to the constructor method of the type specified by the **TypeId** attribute.

3.1.4.1.3.28 ObjectPathMethodType

The **ObjectPathMethodType** complex type is a CSOM Object path. It specifies a CSOM Object that is obtained by calling a method, including any parameters, of a CSOM Object.

3.1.4.1.3.28.1 Schema

```
<xs:complexType name="ObjectPathMethodType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="ParentId" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.28.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

Name: The name of the method that is invoked on the CSOM Object specified by the **ParentId** attribute. It MUST be the name of a valid method of the CSOM Object.

ParentId: The identifier of the CSOM Object path of the CSOM Object that defines the method to call.

3.1.4.1.3.28.3 Child Elements

Parameter: A parameter in the parameter list that is passed in the method call.

Parameters: The list of parameters that are passed in the method call.

3.1.4.1.3.29 ObjectPathObjectIdentityNameType

The **ObjectPathObjectIdentityNameType** complex type is a CSOM Object path. It specifies the CSOM Object that is obtained from a specified CSOM Object identity, which is an implementation-specific string that uniquely identifies a CSOM Object.

3.1.4.1.3.29.1 Schema

```
<xs:complexType name="ObjectPathObjectIdentityNameType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="Name" type="xs:string" use="required"/>
</xs:complexType>
```

3.1.4.1.3.29.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

Name: An implementation-specific **string** that uniquely identifies a CSOM Object.

3.1.4.1.3.29.3 Child Elements

None.

3.1.4.1.3.30 ObjectPathPropertyType

The **ObjectPathPropertyType** complex type is a CSOM Object path. It specifies the CSOM Object that is obtained by accessing a property of a CSOM Object.

3.1.4.1.3.30.1 Schema

```
<xs:complexType name="ObjectPathPropertyType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="ParentId" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.30.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

Name: The name of the property that is used to access the CSOM Object. It MUST be the name of a valid property of the CSOM Object that is specified by the **ParentId** attribute.

ParentId: The identifier of the CSOM Object path of the CSOM Object that defines the property to access.

3.1.4.1.3.30.3 Child Elements

None.

3.1.4.1.3.31 ObjectPathStaticMethodType

The **ObjectPathStaticMethodType** complex type is a CSOM Object path. It specifies the CSOM Object that is obtained by calling a static method, including any parameters, of a CSOM Object type.

3.1.4.1.3.31.1 Schema

```
<xs:complexType name="ObjectPathStaticMethodType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Parameters" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.31.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

Name: The name of the static method that is called on the CSOM Object type specified by the **TypeId** attribute. It MUST be the name of a valid static method of the CSOM Object type.

TypeId: The GUID that identifies the CSOM Object type on which to call the static method.

3.1.4.1.3.31.3 Child Elements

Parameter: A parameter in the parameter list that is passed in the static method call.

Parameters: The list of parameters that are passed in the static method call.

3.1.4.1.3.32 ObjectPathStaticPropertyType

The **ObjectPathStaticPropertyType** complex type is a CSOM Object path. It specifies the CSOM Object that is obtained by accessing a static property of a CSOM Object type.

3.1.4.1.3.32.1 Schema

```
<xs:complexType name="ObjectPathStaticPropertyType"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
  <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
</xs:complexType>
```

3.1.4.1.3.32.2 Attributes

Id: The identifier of the CSOM Object path. This value MUST be unique from all other **IdType** values, as specified section 3.1.4.1.4.3, in the request.

Name: The name of the static property that is used to access the CSOM Object. It MUST be the name of a valid static property of the CSOM Object type that is specified by the **TypeId** attribute.

TypeId: The GUID that identifies the CSOM Object type that defines the static property.

3.1.4.1.3.32.3 Child Elements

None.

3.1.4.1.3.33 ObjectPathsType

The **ObjectPathsType** complex type is a CSOM Object path list.

3.1.4.1.3.33.1 Schema

```
<xs:complexType name="ObjectPathsType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="Property" type="mstns:ObjectPathPropertyType"/>
      <xs:element name="StaticProperty" type="mstns:ObjectPathStaticPropertyType"/>
      <xs:element name="Method" type="mstns:ObjectPathMethodType"/>
      <xs:element name="StaticMethod" type="mstns:ObjectPathStaticMethodType"/>
      <xs:element name="Constructor" type="mstns:ObjectPathConstructorType"/>
      <xs:element name="Identity" type="mstns:ObjectPathObjectIdentityNameType"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

3.1.4.1.3.33.2 Attributes

None.

3.1.4.1.3.33.3 Child Elements

Constructor: The CSOM Object path that is created by invoking the constructor of a CSOM Object type.

Identity: The CSOM Object path that is obtained by its opaque CSOM Object identity, which is an implementation-specific string that uniquely identifies a CSOM Object.

Method: The CSOM Object path that is returned by a method call.

Property: The CSOM Object path that is accessed as a property of another CSOM Object.

StaticMethod: The CSOM Object path that is returned by a static method call.

StaticProperty: The CSOM Object path that is accessed as a static property of a CSOM Object type.

3.1.4.1.3.34 QueryPropertyType

The **QueryPropertyType** complex type is a child item query component that retrieves property data for a CSOM Object.

3.1.4.1.3.34.1 Schema

```
<xs:complexType name="QueryPropertyType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Query" type="mstns:QueryType" minOccurs="0"/>
    <xs:element name="ChildItemQuery" type="mstns:ChildItemQueryType" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
  <xs:attribute name="ScalarProperty" type="xs:boolean" use="optional"/>
  <xs:attribute name="SelectAll" type="xs:boolean" use="optional"/>
</xs:complexType>
```

3.1.4.1.3.34.2 Attributes

Name: The name of the property or **CSOM expando field**. It MUST be a valid property or CSOM expando field that is defined by the CSOM Object.

ScalarProperty: A **Boolean** value that specifies whether the property is a scalar property.

If this value is "true" and the property is not a scalar property, the protocol server MUST return an error whose error type name is "Microsoft.SharePoint.Client.InvalidClientQueryException".

If this value is "false" and the property is a scalar property, the protocol server MUST return an error whose error type name is "Microsoft.SharePoint.Client.InvalidClientQueryException".

SelectAll: If this value is "true", the property MUST be a CSOM Object collection or a CSOM Object.

If this value is "true" and the property or CSOM expando field is a CSOM Object collection, the protocol server MUST return all of the child objects of the CSOM Object collection. Additionally, for each child object of the CSOM Object collection, the protocol server MUST return all of the default scalar properties and fields, and the query specified by the **ChildItemQuery** element, if that element is present.

Otherwise, if this value is "false" and the property or CSOM expando field is a CSOM Object collection, the protocol server MUST return the query specified by the **ChildItemQuery** element, if that element is present.

If this value is "true" and the property or CSOM expando field is a CSOM Object, the protocol server MUST return all of the default scalar properties of the CSOM Object and the query specified by the **Query** element, if that element is present.

Otherwise, if this value is "false" and the property or CSOM expando field is a CSOM Object, the protocol server MUST return the query specified by the **Query** element, if that element is present.

3.1.4.1.3.34.3 Child Elements

ChildItemQuery: The child item query that is applied to the child CSOM Objects if the property or CSOM expando field is a CSOM Object collection.

Query: The child item query that is applied to the property if the property or CSOM expando field is a CSOM Object.

3.1.4.1.3.35 QueryType

The **QueryType** complex type is a child item query that returns data about a specified CSOM Object.

3.1.4.1.3.35.1 Schema

```
<xs:complexType name="QueryType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Properties">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Property" type="mstns:QueryPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="SelectAllProperties" type="xs:boolean" use="optional"/>
</xs:complexType>
```

3.1.4.1.3.35.2 Attributes

SelectAllProperties: A **Boolean** value that specifies whether the protocol server returns data from the **default scalar property set** of the CSOM Object. If this value is "true", the protocol server MUST return data from the default scalar property set of the CSOM Object and any of the properties that are specified by the **Properties** element. If one scalar property is already included in the **Properties** element, the protocol server MUST return only one copy of the property. Otherwise, if this value is "false", the protocol server MUST return data from the properties specified by the **Properties** element.

3.1.4.1.3.35.3 Child Elements

Properties: A list of properties to return data from.

Property: A property to be returned. It MUST be a valid property that is defined by the CSOM Object being queried.

3.1.4.1.3.36 RequestType

The **RequestType** complex type is the root element of a **ProcessQueryIn** XML request, as specified section 3.1.4.1.1.1.

3.1.4.1.3.36.1 Schema

```
<xs:complexType name="RequestType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:sequence>
    <xs:element name="Actions" type="mstns:ActionsType"/>
    <xs:element name="ObjectPaths" type="mstns:ObjectPathsType"/>
  </xs:sequence>
  <xs:attribute name="AddExpandoFieldTypeSuffix" type="xs:boolean" use="optional"/>
  <xs:attribute name="SchemaVersion" type="mstns:VersionStringType" use="required"/>
  <xs:attribute name="LibraryVersion" type="mstns:VersionStringType" use="required"/>
  <xs:attribute name="ApplicationName" use="optional">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="128"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
```

3.1.4.1.3.36.2 Attributes

AddExpandoFieldTypeSuffix: A **Boolean** value that specifies whether the protocol server MUST append type information to the name of the JSON member for a CSOM expando field, as specified section 3.1.4.1.8.1.3.1. The default value is "false".

ApplicationName: The name of the client application. Use of this attribute is implementation-specific.

LibraryVersion: The version of the CSOM library. Use of this attribute is implementation-specific.

SchemaVersion: The version of the CSOM protocol. This value MUST be "14.0.0.0" or "15.0.0.0".

3.1.4.1.3.36.3 Child Elements

Actions: The CSOM action list to be executed.

ObjectPaths: The CSOM Object path list that is used when processing the CSOM action list specified by the **Actions** element.

3.1.4.1.4 Simple Types

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

Simple type	Description
ActionIdType	An **integer** that identifies, and is associated with, a CSOM action.
GuidType	A GUID.
IdType	An **integer** identifier that MUST be a unique value in a request.
MethodNameType	The name of a method that is defined in a CSOM Object type.
MethodParameterTypeType	Specifies which types can be used as method parameters.
ObjectPathIdType	The integer identifier of a CSOM Object path.
PropertyNameType	The name of a valid property that is defined by a CSOM Object type or CSOM value object type.
StaticMethodNameType	The name of a valid static method that is defined by a CSOM Object type.
StaticPropertyNameType	The name of a valid static property that is defined by a CSOM Object type.
TypeIdGuidType	The GUID of a CSOM type.
VersionStringType	A version **string**.

3.1.4.1.4.1 ActionIdType

The **ActionIdType** simple type is an **integer** that identifies and is associated with a CSOM action.

```
<xs:simpleType name="ActionIdType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="mstns:IdType"/>
</xs:simpleType>
```

3.1.4.1.4.2 GuidType

The **GuidType** simple type is a GUID.

```
<xs:simpleType name="GuidType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:pattern value="\{[0-9A-Fa-f]{8}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-
Fa-f]{12}\}"/>
  </xs:restriction>
</xs:simpleType>
```

3.1.4.1.4.3 IdType

The **IdType** simple type is an **integer** identifier that MUST be a unique value in a request.

```
<xs:simpleType name="IdType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

```
  <xs:restriction base="xs:integer"/>
</xs:simpleType>
```

3.1.4.1.4.4 MethodNameType

The **MethodNameType** simple type specifies the name of a method that is defined in a CSOM Object type. It MUST be the name of a valid method for the CSOM Object type.

```
<xs:simpleType name="MethodNameType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

3.1.4.1.4.5 MethodParameterTypeType

The **MethodParameterTypeType** simple type specifies which types can be used as method parameters.

```
<xs:simpleType name="MethodParameterTypeType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Null"/>
    <xs:enumeration value="Boolean"/>
    <xs:enumeration value="Char"/>
    <xs:enumeration value="Byte"/>
    <xs:enumeration value="SByte"/>
    <xs:enumeration value="Int16"/>
    <xs:enumeration value="UInt16"/>
    <xs:enumeration value="Int32"/>
    <xs:enumeration value="UInt32"/>
    <xs:enumeration value="Int64"/>
    <xs:enumeration value="UInt64"/>
    <xs:enumeration value="DateTime"/>
    <xs:enumeration value="Single"/>
    <xs:enumeration value="Double"/>
    <xs:enumeration value="Decimal"/>
    <xs:enumeration value="TimeSpan"/>
    <xs:enumeration value="Guid"/>
    <xs:enumeration value="String"/>
    <xs:enumeration value="Base64Binary"/>
    <xs:enumeration value="Dictionary"/>
    <xs:enumeration value="Array"/>
    <xs:enumeration value="Enum"/>
    <xs:enumeration value="Number"/>
    <xs:enumeration value="Unspecified"/>
    <xs:enumeration value="Binary"/>
  </xs:restriction>
</xs:simpleType>
```

Array: The parameter is a CSOM array.

Base64Binary: The parameter is a CSOM binary.

Binary: The parameter is a CSOM Stream.

Boolean: The parameter is a CSOM Boolean.

Byte: The parameter is a CSOM Byte.

Char: The parameter is a CSOM Char.

DateTime: The parameter is a CSOM DateTime.

Decimal: The parameter is a CSOM Decimal.

Dictionary: The parameter is a CSOM dictionary.

Double: The parameter is a CSOM Double.

Enum: The parameter is a CSOM enumeration.

Guid: The parameter is a CSOM GUID.

Int16: The parameter is a CSOM Int16.

Int32: The parameter is a CSOM Int32.

Int64: The parameter is a CSOM Int64.

Null: The parameter is a CSOM null.

Number: The parameter is a **CSOM Byte**, CSOM SByte, **CSOM Int16**, CSOM UInt16, **CSOM Int32**, CSOM UInt32, **CSOM Int64**, CSOM UInt64, CSOM Single, or **CSOM Double**.

SByte: The parameter is a **CSOM SByte**.

Single: The parameter is a **CSOM Single**.

String: The parameter is a CSOM String.

TimeSpan: The parameter is a CSOM TimeSpan.

UInt16: The parameter is a **CSOM UInt16**.

UInt32: The parameter is a **CSOM UInt32**.

UInt64: The parameter is a **CSOM UInt64**.

Unspecified: The parameter type was not specified by the protocol client. An implementation MUST determine the correct parameter type based on the method definition.

3.1.4.1.4.6 ObjectPathIdType

The **ObjectPathIdType** simple type specifies the **integer** identifier of a CSOM Object path. It MUST be a unique value in a request.

```
<xs:simpleType name="ObjectPathIdType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="mstns:IdType"/>
</xs:simpleType>
```

3.1.4.1.4.7 PropertyNameType

The **PropertyNameType** simple type specifies the name of a valid property that is defined by a CSOM Object type or CSOM value object type.

```
<xs:simpleType name="PropertyNameType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

3.1.4.1.4.8 StaticMethodNameType

The **StaticMethodNameType** simple type specifies the name of a valid static method that is defined by a CSOM Object type.

```
<xs:simpleType name="StaticMethodNameType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

3.1.4.1.4.9 StaticPropertyNameType

The **StaticPropertyNameType** simple type specifies the name of a valid static property that is defined by a CSOM Object type.

```
<xs:simpleType name="StaticPropertyNameType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

3.1.4.1.4.10 TypeFunctionTypeType

The name of a type that could be used in type function.

```
<xs:simpleType name="TypeFunctionTypeType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Boolean"/>
    <xs:enumeration value="Byte"/>
    <xs:enumeration value="SByte"/>
    <xs:enumeration value="Char"/>
    <xs:enumeration value="Int16"/>
    <xs:enumeration value="UInt16"/>
    <xs:enumeration value="Int32"/>
    <xs:enumeration value="UInt32"/>
    <xs:enumeration value="Int64"/>
    <xs:enumeration value="UInt64"/>
    <xs:enumeration value="DateTime"/>
    <xs:enumeration value="Single"/>
    <xs:enumeration value="Double"/>
    <xs:enumeration value="Decimal"/>
    <xs:enumeration value="TimeSpan"/>
    <xs:enumeration value="String"/>
  </xs:restriction>
</xs:simpleType>
```

Boolean: CSOM Boolean, see section 2.2.5.2

Byte: CSOM Byte, see section 2.2.5.3

Char: CSOM Char, see section 2.2.5.4

DateTime: CSOM DateTime, see section 2.2.5.5

Decimal: CSOM Decimal, see section 2.2.5.18

Double: CSOM Double, see section 2.2.5.6

Int16: CSOM Int16, see section 2.2.5.9

Int32: CSOM Int32, see section 2.2.5.10

Int64: CSOM Int64, see section 2.2.5.11

SByte: CSOM SByte, see section 2.2.5.12

Single: CSOM Single, see section 2.2.5.13

String: CSOM String, see section 2.2.5.14

TimeSpan: CSOM TimeSpan, see section 2.2.5.19

UInt16: CSOM UInt16, see section 2.2.5.15

UInt32: CSOM UInt32, see section 2.2.5.16

UInt64: CSOM UInt64, see section 2.2.5.17

3.1.4.1.4.11 TypeIdGuidType

The **TypeIdGuidType** simple type specifies the GUID of a CSOM type.

```
<xs:simpleType name="TypeIdGuidType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="mstns:GuidType"/>
</xs:simpleType>
```

3.1.4.1.4.12 VersionStringType

The **VersionStringType** simple type specifies a version string.

```
<xs:simpleType name="VersionStringType" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:restriction base="xs:string">
    <xs:pattern value="[0-9]{1,8}\.[0-9]{1,8}\.[0-9]{1,8}\.[0-9]{1,8}"/>
  </xs:restriction>
</xs:simpleType>
```

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

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

Group	Description
ActionGroup	A set of possible CSOM actions.
ExpressionGroup	A set of possible expression components.
ExpressionQueryableExpressionGroup	A set of possible queryable expressions.

3.1.4.1.6.1 ActionGroup

The **ActionGroup** group contains a set of possible CSOM actions.

3.1.4.1.6.1.1 Schema

```
<xs:group name="ActionGroup" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:choice>
    <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
    <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
    <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
```

```
    <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
    <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
    <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
    <xs:element name="Query" type="mstns:ActionQueryType"/>
    <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
    <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
    <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
  </xs:choice>
</xs:group>
```

3.1.4.1.6.1.2 Attributes

None.

3.1.4.1.6.1.3 Child Elements

ConditionalScope: A CSOM action of the **ConditionalScopeType** type, as specified section 3.1.4.1.3.10.

ExceptionHandlingScope: A CSOM action of the **ExceptionHandlingScopeType** type, as specified section 3.1.4.1.3.12.

ExceptionHandlingScopeSimple: A CSOM action of the **ExceptionHandlingScopeSimpleType** type, as specified section 3.1.4.1.3.11.

Method: A CSOM action of the **ActionInvokeMethodType** type, as specified section 3.1.4.1.3.2.

ObjectIdentityQuery: A CSOM action of the **ActionObjectIdentityQueryType** type, as specified section 3.1.4.1.3.4.

ObjectPath: A CSOM action of the **ActionInstantiateObjectPathType** type, as specified section 3.1.4.1.3.1.

Query: A CSOM action of the **ActionQueryType** type, as specified section 3.1.4.1.3.5.

SetProperty: A CSOM action of the **ActionSetPropertyType** type, as specified section 3.1.4.1.3.6.

SetStaticProperty: A CSOM action of the **ActionSetStaticPropertyType** type, as specified section 3.1.4.1.3.7.

StaticMethod: A CSOM action of the **ActionInvokeStaticMethodType** type, as specified section 3.1.4.1.3.3.

3.1.4.1.6.2 ExpressionGroup

The **ExpressionGroup** group contains a set of possible expression components.

3.1.4.1.6.2.1 Schema

```
<xs:group name="ExpressionGroup" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:choice>
    <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
    <xs:element name="NOT" type="mstns:ExpressionType"/>
    <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
```

```
    <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
    <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
    <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
    <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
    <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
    <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
    <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
  </xs:choice>
</xs:group>
```

3.1.4.1.6.2.2 Attributes

None.

3.1.4.1.6.2.3 Child Elements

AND: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a logical "and" operation that is performed on the left and right operands. The left and right operands MUST evaluate to one of the **Boolean** values listed in the following table.

Left operand	Right operand	Result
true	true	true
true	false	false
false	true/false	false

If the left operand is "false", the protocol server MUST NOT evaluate the right operand.

EQ: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of an equality comparison between the left and right operands. If the left operand and right operand have different CSOM types, the value is "false"<7>. If the left operand is equal to the right operand, the value is "true". Otherwise, the value is "false".

ExpressionConstant: An expression component of the **MethodParameterType** type, as specified section 3.1.4.1.3.26. It represents a constant value.

ExpressionConvert: An expression component of the **ExpressionConvertExpressionType** type, as specified section 3.1.4.1.3.13. It represents the result of a type conversion from a source expression component.

ExpressionMethod: An expression component of the **ExpressionMethodExpressionType** type, as specified section 3.1.4.1.3.15. It represents the result of a method call.

ExpressionParameter: An expression component of the **ExpressionParameterExpressionType** type, as specified section 3.1.4.1.3.16. It represents a parameter.

ExpressionProperty: An expression component of the **ExpressionPropertyExpressionType** type, as specified section 3.1.4.1.3.17. It represents the value of a property of a CSOM Object or CSOM value object.

ExpressionStaticMethod: An expression component of the **ExpressionStaticMethodExpressionType** type, as specified section 3.1.4.1.3.22. It represents the result of a static method call.

ExpressionStaticProperty: An expression component of the **ExpressionStaticPropertyExpressionType** type, as specified section 3.1.4.1.3.23. It represents the value of a static property of a CSOM Object type.

ExpressionTypeIs: An expression component of **ExpressionTypeIsExpression** type, as specified section 3.1.4.1.3.25. It represents the result of a test whether a value is a specific type.

GE: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a greater-than-or-equal-to comparison between the left and right operands. If the left operand is greater than or equal to the right operand, the value is "true". Otherwise, the value is "false". The left operand and the right operand MUST have the same CSOM type.

GT: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a greater-than comparison between the left and right operands. If the left operand is greater than the right operand, the value is "true". Otherwise, the value is "false". The left operand and the right operand MUST have the same CSOM type.

LE: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a less-than-or-equal-to comparison between the left and right operands. If the left operand is less than or equal to the right operand, the value is "true". Otherwise, the value is "false". The left operand and the right operand MUST have the same CSOM type.

LT: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a less-than comparison between the left and right operands. If the left operand is less than the right operand, the value is "true". Otherwise, the value is "false". The left operand and the right operand MUST have the same CSOM type.

NE: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a not-equal-to comparison between the left and right operands. If the left operand and right operand have different CSOM types, the value is "true"<8>. If the value of the left and right operands is the same, the value is "false". Otherwise, the value is "true".

NOT: An expression component of the **ExpressionType** type, as specified section 3.1.4.1.3.24. It represents the result of a logical "not" operation. The operand MUST evaluate to one of the **Boolean** values in the following table.

Operand	Result
true	false
false	true

OR: An expression component of the **ExpressionLeftRightOperandExpressionType** type, as specified section 3.1.4.1.3.14. It represents the result of a logical "or" operation performed on a left and right operand. The left operand and right operand MUST evaluate to one of the **Boolean** values in the following table.

Left operand	Right operand	Result
false	true	true
false	false	false
true	true/false	true

If the left operand is evaluated to be "true", the protocol server MUST NOT evaluate the right operand.

3.1.4.1.6.3 ExpressionQueryableExpressionGroup

The **ExpressionQueryableExpressionGroup** group contains a set of possible queryable expressions.

3.1.4.1.6.3.1 Schema

```
<xs:group name="ExpressionQueryableExpressionGroup"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:choice>
    <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
    <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
    <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
    <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
    <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
    <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
    <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
    <xs:element name="QueryableObject">
      <xs:complexType/>
    </xs:element>
  </xs:choice>
</xs:group>
```

3.1.4.1.6.3.2 Attributes

None.

3.1.4.1.6.3.3 Child Elements

OfType: An expression of the **ExpressionQueryableOfTypeType** type, as specified in section 3.1.4.1.3.19, which specified a filtered set of CSOM Objects that are a specific type.

OrderBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies a sorted set of CSOM Objects in ascending order.

OrderByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies a sorted set of CSOM Objects in descending order.

QueryableObject: An expression that references a collection of CSOM Objects.

Take: An expression of the **ExpressionQueryableTakeType** type, as specified section 3.1.4.1.3.20.

ThenBy: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies a subsequent ordering of the sorted set of CSOM Objects in ascending order.

ThenByDescending: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies a subsequent ordering of the sorted set of CSOM Objects in descending order.

Where: An expression of the **ExpressionQueryableWhereType** type, as specified in section 3.1.4.1.3.21, which specifies a filtered set of CSOM Objects.

3.1.4.1.7 Attribute Groups

None.

3.1.4.1.8 JSON Types

The **ProcessQueryOut** response message, as specified section 3.1.4.1.1.2, consists of UTF-8 text that contains JSON values.

3.1.4.1.8.1 JSON Object Types

3.1.4.1.8.1.1 CSOM Dictionary JSON Value

The **CSOM dictionary JSON value** object is a JSON object that represents an unordered collection of key/value pairs. Each JSON object name/value member represents a key/value entry in a CSOM dictionary. The name of the JSON member is the dictionary key and the value of the JSON member is the dictionary value.

3.1.4.1.8.1.2 CSOM Error JSON Value

The **CSOM error JSON value** object is a JSON object that contains information about an error that occurred on the protocol server while processing a request. It contains the following JSON members listed in the following table.

Member name	Member value
ErrorCode	A CSOM Int32 JSON value that indicates the error code for the error. This member MUST be present.
ErrorDetails	A CSOM value object JSON value that contains detailed information about the error. This member is optional. If there isn't any detailed information for the error, this member is not present.
ErrorMessage	If an error message is generated while executing the request, this value is a CSOM String JSON value that is set to the value of the error message. If no error message is generated, this value MUST be a CSOM null JSON value. This member MUST be present.
ErrorStackTrace	If a call-stack trace is generated during the execution of the request, this value is a **CSOM String** JSON value that contains the call-stack trace information for the error. If no call-stack trace is generated, this value MUST be a CSOM null JSON value. This member is optional.
ErrorTypeName	If there is type name information, this value is a **CSOM String** JSON value with the type name of the error. If there is no type name information, this value MUST be a CSOM null JSON value. This member MUST be present.
ErrorValue	If there is additional error information, this value is a **CSOM String** JSON value with additional information about the error. If there isn't any additional error information, this MUST be a CSOM null JSON value. This member MUST be present.

The protocol server MUST return the errors listed in the following table when the specified error condition occurs.

ErrorCode	ErrorTypeName	ErrorValue	Description
-2147024891	System.UnauthorizedAccessException	CSOM null JSON value	The user does not have permission to perform the specified actions or access one or more of the specified resources.

ErrorCode	ErrorTypeName	ErrorValue	Description
-2130575339	Microsoft.SharePoint.Client.VersionConflictException	CSOM null JSON value	The version of the CSOM Object that was passed from the protocol client does not match the version of that object on the protocol server.
-2147024846	Microsoft.SharePoint.Client.ApiBlockedException	CSOM null JSON value	The property or method is not allowed to be accessed.
-2130575151	Microsoft.SharePoint.Client.NotSupportedRequestVersionException	A CSOM String JSON value that contains a comma-delimited list of the supported schema versions.	The schema version specified by the **SchemaVersion** attribute of the **Request** element is not supported by the protocol server.
-2130575152	Microsoft.SharePoint.Client.RedirectException	A **CSOM String** JSON value that contains the URL that the request is redirected to.	The request was redirected to another URL by the protocol server.

3.1.4.1.8.1.3 CSOM Object JSON Value

The **CSOM Object JSON value** object is a JSON object that contains information about a CSOM Object. It has the JSON members listed in the following table.

Member name	Member value
ObjectType	A CSOM String JSON value that identifies the type of a CSOM Object. This member MUST be the first member in the JSON object. This member MUST be present.
ObjectIdentity	A **CSOM String** JSON value that contains implementation-specific information that uniquely identifies a CSOM Object. By using the **ObjectPathObjectIdentityNameType** type, as specified section 3.1.4.1.3.29, this value can be used in subsequent **ProcessQuery** requests, as specified section 3.1.4.1, to obtain the CSOM Object. If the CSOM Object has an object identity, this member MUST be present. Otherwise, this member MUST NOT be present.
ObjectVersion	A **CSOM String** JSON value that contains implementation-specific information about the version of a CSOM Object. By using the **ActionInvokeMethodType** type, as specified section 3.1.4.1.3.2, this value can be used in subsequent **ProcessQuery** requests, as specified section 3.1.4.1, to verify that a method call is valid, based on the protocol client version of the CSOM Object and the protocol server version of the CSOM Object. If the CSOM Object has a version, this member MUST be present. Otherwise, this member MUST NOT be present.
propName#	The member name is the CSOM Object property name. The member value is the value of the CSOM Object property. If a query is specified in the request XML for the CSOM Object, the protocol server MUST include the properties that are specified by the query in the CSOM Object JSON value. If a query is not specified in the request XML for the CSOM Object, the protocol server MUST include only the scalar properties of the CSOM Object in the CSOM Object JSON value. The **propName** member can have a numeric suffix, which is indicated by a pound sign (#), and that sign is replaced with a numeral to represent one CSOM Object property. For example, if there are three properties whose names are Title,

Member name	Member value
	Description, and Address, the **propName1**, **propName2** and **propName3** represent those three properties.
expandoProp#	The member name is the name of a CSOM expando field. The member name optionally has a suffix if the value of the **AddExpandoFieldTypeSuffix** attribute is "true" in the **RequestType** type, as specified section 3.1.4.1.3.36. For more information see section 3.1.4.1.8.1.3.1. The member value is the value of the CSOM expando field. The **expandoProp** member can have a numeric suffix, which is indicated by a pound sign (#), and that sign is replaced with numeral to represent one CSOM expando field. For example, if there are three CSOM expando fields whose names are Title, Description, and Address, the **expandoProp1**, **expandoProp2** and **expandoProp3** represent those three CSOM expando fields.
_Child_Items_	A CSOM array JSON value that contains child objects of the CSOM Object. This member is optional. The protocol server MUST include this member only if the CSOM Object is a CSOM Object collection and the protocol client queries for child objects of that collection.

3.1.4.1.8.1.3.1 JSON Member Name for Expando Field

If a protocol client specifies that the value of the **AddExpandoFieldTypeSuffix** attribute in a **RequestType** type, as specified section 3.1.4.1.3.36, is "true", the protocol server MUST append a suffix to the name of the CSOM expando field and use the resulting name as the name of the JSON member. Otherwise, the protocol server MUST use only the name of the CSOM expando field as the name of the JSON member. The following is the Augmented Backus-Naur Form (ABNF) grammar for defining the name of the JSON member for a CSOM expando field, if a protocol client specifies that value of the **AddExpandoFieldTypeSuffix** attribute is "true" in the **RequestType** type.

```
csom-expando-member-name    = rfc4627-quotation-mark csom-expando-property-name
                              [csom-expando-suffix] rfc4627-quotation-mark
csom-expando-property-name  = 1*restricted-char
csom-expando-suffix         = csom-expando-suffix-char /
                              csom-expando-suffix-byte /
                              csom-expando-suffix-int16 /
                              csom-expando-suffix-uint16 /
                              csom-expando-suffix-int32 /
                              csom-expando-suffix-uint32 /
                              csom-expando-suffix-single /
                              csom-expando-suffix-double /
                              csom-expando-suffix-array
csom-expando-suffix-char    = %x24.20.20.20.43.68.61.72   ; "$   Char"
csom-expando-suffix-byte    = %x24.20.20.20.42.79.74.65   ; "$   Byte"
csom-expando-suffix-int16   = %x24.20.20.49.6E.74.31.36   ; "$  Int16"
csom-expando-suffix-uint16  = %x24.20.55.49.6E.74.31.36   ; "$ UInt16"
csom-expando-suffix-int32   = %x24.20.20.49.6E.74.33.32   ; "$  Int32"
csom-expando-suffix-uint32  = %x24.20.55.49.6E.74.33.32   ; "$ UInt32"
csom-expando-suffix-single  = %x24.20.53.69.6E.67.6C.65   ; "$ Single"
csom-expando-suffix-double  = %x24.20.44.6F.75.62.6C.65   ; "$ Double"
csom-expando-suffix-array   = %x24 csom-expando-array-itemtype
                              %x24.20.20.41.72.72.61.79   ; "$  Array"
csom-expando-array-itemtype = %x42.6F.6F.6C.65.61.6E /    ; "Boolean"
                              %x43.68.61.72 /             ; "Char"
                              %x53.42.79.74.65 /          ; "SByte"
                              %x49.6E.74.31.36 /          ; "Int16"
                              %x55.49.6E.74.31.36 /       ; "UInt16"
                              %x49.6E.74.33.32 /          ; "Int32"
                              %x55.49.6E.74.33.32 /       ; "UInt32"
                              %x49.6E.74.36.34 /          ; "Int64"
                              %x55.49.6E.74.36.34 /       ; "UInt64"
                              %x44.61.74.65.54.69.6D.65 / ; "DateTime"
                              %x53.69.6E.67.6C.65 /       ; "Single"
```

```
                                %x44.6F.75.62.6C.65 /       ; "Double"
                                %x47.75.69.64 /             ; "Guid"
                                %x53.74.72.69.6E.67 /       ; "String"
                                csom-expando-array-itemtypeid
csom-expando-array-itemtypeid = 4hexOctet "-" 2hexOctet "-" 2hexOctet
                                "-" 2hexOctet "-" 6hexOctet

hexOctet                    = hexDigit hexDigit
hexDigit                    = "0" / "1" / "2" / "3" / "4" /
                              "5" / "6" / "7" / "8" / "9" /
                              "a" / "b" / "c" / "d" / "e" / "f" /
                              "A" / "B" / "C" / "D" / "E" / "F"
```

In the preceding ABNF, the **restricted-char** rule is specified in the CSOM String JSON value, as specified section 2.2.5.14.2.

The protocol server MUST define and append a suffix based on the type of value that is stored in the CSOM expando field, according to the rules in the following table.

Type of value	Suffix
CSOM Char	csom-expando-suffix-char
CSOM Byte	csom-expando-suffix-byte
CSOM Int16	csom-expando-suffix-int16
CSOM UInt16	csom-expando-suffix-uint16
CSOM Int32	csom-expando-suffix-int32
CSOM UInt32	csom-expando-suffix-uint32
CSOM Single	csom-expando-suffix-single
CSOM Double	csom-expando-suffix-double
Array of CSOM Char	csom-expando-suffix-array and csom-expando-array-itemtype is Char.
Array of CSOM SByte	csom-expando-suffix-array and csom-expando-array-itemtype is SByte.
Array of CSOM Int16	csom-expando-suffix-array and csom-expando-array-itemtype is Int16.
Array of CSOM UInt16	csom-expando-suffix-array and csom-expando-array-itemtype is UInt16.
Array of CSOM UInt32	csom-expando-suffix-array and csom-expando-array-itemtype is Int32.
Array of CSOM Int64	csom-expando-suffix-array and csom-expando-array-itemtype is Int64.
Array of CSOM UInt64	csom-expando-suffix-array and csom-expando-array-itemtype is UInt64.
Array of CSOM DateTime	csom-expando-suffix-array and csom-expando-array-itemtype is DateTime.
Array of CSOM Single	csom-expando-suffix-array and csom-expando-array-itemtype is Single.

Type of value	Suffix
Array of CSOM Double	csom-expando-suffix-array and csom-expando-array-itemtype is Double.
Array of CSOM GUID	csom-expando-suffix-array and csom-expando-array-itemtype is Guid.
Array of CSOM object	csom-expando-suffix-array and csom-expando-array-itemtype is csom-expando-array-itemtypeid, which is the CSOM Object type identifier.
Array of CSOM value object	csom-expando-suffix-array and csom-expando-array-itemtype is csom-expando-array-itemtypeid, which is the CSOM value object type identifier.
Other types	The protocol server MUST NOT append any suffix.

For example, if the **AddExpandoFieldTypeSuffix** attribute value of a **RequestType** type is "true", the name of a CSOM expando field is "example", and the property value is a CSOM Int32, the name of the JSON member for the CSOM expando field MUST be:

```
"example$  Int32"
```

In the preceding code, there are two space characters (U+0020) between "$" and "Int32".

For a CSOM expando field with the name "examples" and a property value that is an array of CSOM Strings, the name of the JSON member for the CSOM expando field MUST be:

```
"examples$String$  Array"
```

In the preceding code, there are two space characters (U+0020) between "$" and "Array".

For a CSOM expando field with the name "test" and a property value that is an array of CSOM value objects with the type identifier is "3e28ad77-9cf0-4557-8223-6609053f89b8", the name of the JSON member for the CSOM expando field MUST be:

```
"test$3e28ad77-9cf0-4557-8223-6609053f89b8$  Array"
```

In the preceding code, there are two space characters (U+0020) between "$" and "Array".

3.1.4.1.8.1.4 CSOM Value Object JSON Value

The **CSOM value object JSON value** object is a JSON object that contains information about a CSOM value object. It has the JSON members listed in the following table.

Member name	Member value
ObjectType	A CSOM String JSON value that identifies the type of the CSOM value object. This member MUST be the first member in the JSON object and it MUST be present in the JSON object.
propName#	Represents a property of a CSOM value object and the value of that property. The name of the member is the name of the property. The value of the member is the value of the property. The protocol server MUST include in the JSON object all of the properties of the CSOM value object. The **propName** member can have a numeric suffix, which is indicated by a pound sign (#), and that sign is replaced with a numeral if the property is a custom property. For example, if there

Member name	Member value
	are three properties whose names are Title, Description, and Address, the **propName1**, **propName2** and **propName3** represent those three properties.

3.1.4.1.8.1.5 Response Header JSON Value

The **response header JSON value** object is a JSON object that contains header information for a response. It has the JSON members listed in the following table.

Member name	Member value
SchemaVersion	A CSOM String JSON value that represents the version of the protocol scheme. This value MUST conform to the **VersionStringType** simple type, as defined in section 3.1.4.1.4.12. It MUST be "14.0.0.0" or "15.0.0.0". This member MUST be present.
LibraryVersion	A **CSOM String** JSON value that represents the product version on the protocol server. This value MUST conform to the **VersionStringType** simple type, as defined in section 3.1.4.1.4.12. This member MUST be present.
ErrorInfo	If an unhandled error occurs on the protocol server while processing a request, this value is a CSOM error JSON value that contains information about the error. Otherwise, this value is a CSOM null JSON value. This member MUST be present.

3.1.4.1.8.2 JSON Response Structure

The JSON response structure consists of UTF-8 text in JSON format. It represents a response from a **ProcessQuery** request message, as specified section 3.1.4.1. The top-level JSON value MUST be a JSON array.

A JSON response structure MUST have at least one element, which is the response header, followed by a "0" (zero), and one or more **actionId** and **actionResponse** value pairs. These values MUST correspond to the CSOM actions that were executed and resulted in a response.

The following table lists the elements in a response JSON array.

Index	Name	Description
0	responseHeader	A response header JSON value, as specified in section 3.1.4.1.8.1.5.
i + 1	actionId*i*	A JSON number that identifies the CSOM action that the **actionResponse*i*** applies to.
i + 2	actionResponse*i*	An **ActionResponseType** result, as specified in section 3.1.4.1.8.2.1, that results from the protocol server executing the CSOM action specified by the **actionId*i*** element.

3.1.4.1.8.2.1 ActionResponseTypes

The **ActionResponseTypes** values in a JSON response structure represent the results of a CSOM action that was executed by a protocol server.

3.1.4.1.8.2.1.1 ConditionalScopeResponse JSON Value

The **ConditionalScopeResponse** JSON value is the response for a CSOM action **ConditionalScopeType** complex type, as specified in section 3.1.4.1.3.10.

```
<xs:element name="ConditionalScope" type="ConditionalScopeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

It is a JSON object and it MUST have a **Test** JSON member. The value of the **Test** member is a CSOM Boolean JSON value that represents the result of evaluating a conditional expression. If the conditional expression is evaluated as "true", the value MUST be a JSON literal "true". Otherwise, the value MUST be a JSON literal "false".

3.1.4.1.8.2.1.2 ExceptionHandlingScopeResponse JSON Value

The **ExceptionHandlingScopeResponse** JSON value is the response for a CSOM action **ExceptionHandlingScopeType** complex type, as specified in section 3.1.4.1.3.12.

```
<xs:element name="ExceptionHandlingScope" type="ExceptionHandlingScopeType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

It is a JSON object and it has the JSON members listed in the following table.

Member name	Member value
HasException	A CSOM Boolean JSON value that indicates whether an unhandled exception occurred on the protocol server while executing the CSOM actions specified by the **TryScope** element of the **ExceptionHandlingScopeType** complex type. If an unhandled exception occurred, this value MUST be a JSON literal "true". Otherwise, this value MUST be a JSON literal "false". This member MUST be present.
ErrorInfo	Either a CSOM error JSON value that contains information about the error or, if there is no error message, a CSOM null JSON value. This member MUST be present.

3.1.4.1.8.2.1.3 ExceptionHandlingScopeSimpleResponse JSON Value

The **ExceptionHandlingScopeSimpleResponse** JSON value is the response for a CSOM action **ExceptionHandlingScopeSimpleType** complex type, as specified in section 3.1.4.1.3.11.

```
<xs:element name="ExceptionHandlingScopeSimple" type="ExceptionHandlingScopeSimpleType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

It is a JSON object and it has the JSON members listed in the following table.

Member name	Member value
HasException	A CSOM Boolean JSON value that indicates whether an unhandled exception occurred on the protocol server while executing the CSOM actions specified by the **ExceptionHandlingScopeSimpleType** complex type. If an unhandled exception occurred, this value MUST be a JSON literal "true". Otherwise, this value MUST be a JSON literal "false". This member MUST be present.

Member name	Member value
ErrorInfo	Either a CSOM error JSON value that contains information about the error or, if there is no error message, a CSOM null JSON value. This member MUST be present.

3.1.4.1.8.2.1.4 MethodResponse JSON Value

The **MethodResponse** JSON value is the response of either CSOM action **ActionInvokeMethodType** complex type or CSOM action **ActionInvokeStaticMethodType** complex type when the method does not have a void response type. The **ActionInvokeMethodType** complex type is specified in section 3.1.4.1.3.2, as follows:

```
<xs:element name="Method" type="ActionInvokeMethodType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

and the **ActionInvokeStaticMethodType** complex type is specified in section 3.1.4.1.3.3, as follows:

```
<xs:element name="StaticMethod" type="ActionInvokeStaticMethodType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

The **MethodResponse** JSON value represents the response for a CSOM action. If the value that is returned while calling the method is null, the protocol server MUST return a CSOM null JSON value. Otherwise, the protocol server MUST return one of the data types listed in the following table, based on the type that is returned by the method.

Method return type	JSON response type
CSOM array	CSOM array JSON value
CSOM binary	CSOM binary JSON value
CSOM Byte	CSOM Byte JSON value
CSOM Char	CSOM Char JSON value
CSOM DateTime	CSOM DateTime JSON value
CSOM dictionary	CSOM dictionary JSON value
CSOM Double	CSOM Double JSON value
CSOM Enum	CSOM Enum JSON value
CSOM GUID	CSOM GUID JSON value
CSOM Int16	CSOM Int16 JSON value
CSOM Int32	CSOM Int32 JSON value
CSOM Int64	CSOM Int64 JSON value
CSOM SByte	CSOM SByte JSON value
CSOM Single	CSOM Single JSON value
CSOM String	CSOM String JSON value

Method return type	JSON response type
CSOM UInt16	CSOM UInt16 JSON value
CSOM UInt32	CSOM UInt32 JSON value
CSOM UInt64	CSOM UInt64 JSON value
CSOM Stream	CSOM Stream JSON value
CSOM Object	CSOM Object JSON value
CSOM value object	CSOM value object JSON value

3.1.4.1.8.2.1.5 ObjectIdentityQueryResponse JSON Value

The **ObjectIdentityQueryResponse** JSON value is an **ActionObjectIdentityQueryType** complex type, as specified in section 3.1.4.1.3.4, that represents the response for a CSOM action.

```
<xs:element name="ObjectIdentityQuery" type="ActionObjectIdentityQueryType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

If the CSOM Object obtained from the specified CSOM Object path identifier is null, it is a JSON literal null. Otherwise, it is a JSON object and it has the **_ObjectIdentity_** JSON member. The value of the **_ObjectIdentity_** member is a CSOM String JSON value that contains implementation-specific information that uniquely identifies a CSOM Object. By using the **ObjectPathObjectIdentityNameType** complex type, as specified section 3.1.4.1.3.29, this value can be used in subsequent **ProcessQuery** request messages, as specified section 3.1.4.1, to obtain the CSOM Object.

3.1.4.1.8.2.1.6 ObjectPathResponse JSON Value

The **ObjectPathResponse** JSON value is an **ActionInstantiateObjectPathType** complex type, as specified in section 3.1.4.1.3.1, that represents the response for CSOM action.

```
<xs:element name="ObjectPath" type="ActionInstantiateObjectPathType"
xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

It is a JSON object and it MUST have an **IsNull** JSON member. The value of the **IsNull** member is a CSOM Boolean JSON value that indicates whether the CSOM Object obtained from the specified CSOM Object path identifier is null. If the CSOM Object is null, this value MUST be a JSON literal "true". Otherwise, this value MUST be a JSON literal "false".

3.1.4.1.8.2.1.7 QueryResponse JSON Value

The **QueryResponse** JSON value is an **ActionQueryType** complex type, as specified in section 3.1.4.1.3.5, that represents a response for a CSOM action.

```
<xs:element name="Query" type="ActionQueryType" xmlns:xs="http://www.w3.org/2001/XMLSchema"/>
```

If the CSOM Object obtained from the specified CSOM Object path identifier is null, it is a JSON literal null. Otherwise, it is a CSOM Object JSON value, as specified in section 3.1.4.1.8.1.3.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

The following examples illustrate how a protocol client retrieves and updates data on a protocol server by using this protocol. The example scenario is a book store that has a catalog of books. Each book within that catalog has an identifier, title, author, and status.

In this scenario, the CSOM types listed in the following table are defined on the protocol server.

CSOM type	CSOM type name (identifier)	Description	Properties	Methods
BookStore	SampleCode.BookStore (acc57e47-24b0-4400-b1c7-aa1cf3c9542d)	A CSOM Object type that defines the book store, which contains a catalog.	**Catalog**– A **Catalog** type that contains a static property specifying a catalog of books.	None.
Catalog	SampleCode.Catalog (04a81e1b-a5b9-445d-97c0-681fe3f61189)	A CSOM Object type that defines the catalog, which contains a collection of books.	**Books**– A list of books.	None.
BookCollection	SampleCode.BookCollection (4c456811-3967-4021-8d9f-237ebd9c1170)	A CSOM Object type that defines a collection of books within a catalog.	None.	**Add**– Returns a **Book** type and adds a book to the collection. The only parameter is a **BookCreationInformation** object, which contains the title, author, publish date, and status of a book. **GetById**– Returns a **Book** type that represents the book with the specified book identifier. If the book does not exist, the protocol server returns CSOM null. The only parameter is a CSOM GUID that contains the book identifier of the book.
Book	SampleCode.Book (030f9ac0-5f2b-4422-9e32-bcdfc1a0c93a)	A CSOM Object type that defines a specific book in a collection of books.	**Author**– A **CSOM String** that contains the name of the author of the book. **Id**– A **CSOM String** value that contains the identification number for	**Update** – Updates information about the book by saving the property values that are currently set for the book. This method does not define any parameters. **GetSampleStream** – Get the sample content of the book. This method does not define any parameters..

CSOM type	CSOM type name (identifier)	Description	Properties	Methods
			the book. **PublishDate**– A **CSOM DateTime** value that contains the date when the book was published. **Status**– A **Status** type that contains the status of the book. **Title**– A **CSOM String** that contains the title of the book.	**UpdateSampleStream** – Update the sample content of the book. The only parameter is a CSOM Stream that contains the sample content of the book.
BookCreationInformation	SampleCode.BookCreationInformation (dda98aeb-f87d-490f-9a61-be08644ad461)	A CSOM value object type that contains information about a book to be added to the catalog for the book store.	**Author**– A **CSOM String** that contains the name of the author of the book. **PublishDate**– A **CSOM DateTime** value that contains the date when the book was published. **Status**– A **Status** type that contains the status of the book. **Title**– A **CSOM String** that contains the title of the book.	None.
Status		A CSOM enumeration type that specifies	None.	None.

CSOM type	CSOM type name (identifier)	Description	Properties	Methods
		whether a book is in stock. If a book is in stock, this value is "0" (zero). If a book is out of stock, this value is "1". If a book is on back order, this value is "2".		

In the examples, the catalog contains the following books.

Identifier	Title	Author	Status	Publish date
3387ac63-e73d-421f-bff7-359a4aa2bc38	How to Cook Chinese Food	Soha Kamal	0	March 1, 2008
f6a265ab-86e5-4fbf-937c-49923604b91d	How to Cook Japanese Food	Soha Kamal	1	May 5, 2007
2e80eb25-b64a-4506-b87b-2fff6ddb3f57	Best Recipes	Lisa Andrews	0	January 3, 2009
704655a3-c136-469c-a578-f79652a93f9b	Family Recipes	Patrick Hines	0	December 1, 2005

4.1 Retrieve Book Information

In this example, a protocol client requests information about a book by using the book identifier "3387ac63-e73d-421f-bff7-359a4aa2bc38". The protocol client also requests author and status information for the book that is associated with identifier "704655a3-c136-469c-a578-f79652a93f9b".

The protocol client sends the following message:

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="2" ObjectPathId="1" />
    <ObjectPath Id="4" ObjectPathId="3" />
    <ObjectPath Id="6" ObjectPathId="5" />
    <Query Id="7" ObjectPathId="5">
      <Query SelectAllProperties="true">
        <Properties />
      </Query>
    </Query>
    <ObjectPath Id="9" ObjectPathId="8" />
    <Query Id="10" ObjectPathId="8">
      <Query SelectAllProperties="false">
        <Properties>
          <Property Name="Author" ScalarProperty="true" />
          <Property Name="Status" ScalarProperty="true" />
        </Properties>
      </Query>
    </Query>
  </Actions>

  <ObjectPaths>
```

```
    <StaticProperty Id="1" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="3" ParentId="1" Name="Books" />
    <Method Id="5" ParentId="3" Name="GetById">
      <Parameters>
        <Parameter Type="Guid">{3387ac63-e73d-421f-bff7-359a4aa2bc38}</Parameter>
      </Parameters>
    </Method>
    <Method Id="8" ParentId="3" Name="GetById">
      <Parameters>
        <Parameter Type="Guid">{704655a3-c136-469c-a578-f79652a93f9b}</Parameter>
      </Parameters>
    </Method>
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : null
  },
  2,
  {
    "IsNull" : false
  },
  4,
  {
    "IsNull" : false
  },
  6,
  {
    "IsNull" : false
  },
  7,
  {
    "_ObjectType_" : "SampleCode.Book",
    "Author" : "Soha Kamal",
    "Id" : "\/Guid(3387ac63-e73d-421f-bff7-359a4aa2bc38)\/",
    "PublishDate" : "\/Date(2008,2,1,0,0,0,0)\/",
    "Status" : 0,
    "Title" : "How to Cook Chinese Food"
  },
  9,
  {
    "IsNull" : false
  },
  10,
  {
    "_ObjectType_" : "SampleCode.Book",
    "Author" : "Patrick Hines",
    "Status" : 0
  }
]
```

4.2 Retrieve Books by a Specific Author

In this example, a protocol client requests information about all of the books that are associated with the author named Soha Kamal.

The protocol client sends the following message:

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="12" ObjectPathId="11" />
    <ObjectPath Id="14" ObjectPathId="13" />
    <Query Id="15" ObjectPathId="13">
      <Query SelectAllProperties="false">
        <Properties />
      </Query>
      <ChildItemQuery SelectAllProperties="true">
        <Properties />
        <QueryableExpression>
          <Where>
            <Test>
              <Parameters>
                <Parameter Name="bk" />
              </Parameters>
              <Body>
                <EQ>
                  <ExpressionProperty Name="Author">
                    <ExpressionParameter Name="bk" />
                  </ExpressionProperty>
                  <ExpressionConstant Type="String">Soha Kamal</ExpressionConstant>
                </EQ>
              </Body>
            </Test>
            <Object>
              <QueryableObject />
            </Object>
          </Where>
        </QueryableExpression>
      </ChildItemQuery>
    </Query>
  </Actions>
  <ObjectPaths>
    <StaticProperty Id="11" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="13" ParentId="11" Name="Books" />
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : null
  },
  12,
  {
    "IsNull" : false
  },
  14,
  {
    "IsNull" : false
  },
  15,
  {
    "_ObjectType_" : "SampleCode.BookCollection",
    "_Child_Items_" : [
      {
        "_ObjectType_" : "SampleCode.Book",
```

```
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(3387ac63-e73d-421f-bff7-359a4aa2bc38)\/",
        "PublishDate" : "\/Date(2008,2,1,0,0,0,0)\/",
        "Status" : 0,
        "Title" : "How to Cook Chinese Food"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(f6a265ab-86e5-4fbf-937c-49923604b91d)\/",
        "PublishDate" : "\/Date(2007,4,4,0,0,0,0)\/",
        "Status" : 1,
        "Title" : "How to Cook Japanese Food"
      }
    ]
  }
]
```

4.3 Update Book Information

In this example, a protocol client sends a request to update author and status information for the book that is associated with the identifier "2e80eb25-b64a-4506-b87b-2fff6ddb3f57", and to retrieve information about all of the books in the catalog.

The protocol client sends the following message:

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="18" ObjectPathId="17" />
    <ObjectPath Id="20" ObjectPathId="19" />
    <ObjectPath Id="22" ObjectPathId="21" />
    <SetProperty Id="23" ObjectPathId="21" Name="Author">
      <Parameter Type="String">Lisa Andrews</Parameter>
    </SetProperty>
    <SetProperty Id="24" ObjectPathId="21" Name="Status">
      <Parameter Type="Enum">1</Parameter>
    </SetProperty>
    <Method Name="Update" Id="25" ObjectPathId="21" />
    <Query Id="26" ObjectPathId="19">
      <Query SelectAllProperties="true">
        <Properties />
      </Query>
      <ChildItemQuery SelectAllProperties="true">
        <Properties />
      </ChildItemQuery>
    </Query>
  </Actions>
  <ObjectPaths>
    <StaticProperty Id="17" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="19" ParentId="17" Name="Books" />
    <Method Id="21" ParentId="19" Name="GetById">
      <Parameters>
        <Parameter Type="Guid">{2e80eb25-b64a-4506-b87b-2fff6ddb3f57}</Parameter>
      </Parameters>
    </Method>
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : null
  },
  18,
  {
    "IsNull" : false
  },
  20,
  {
    "IsNull" : false
  },
  22,
  {
    "IsNull" : false
  },
  26,
  {
    " ObjectType " : "SampleCode.BookCollection",
    "_Child_Items_" : [
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(3387ac63-e73d-421f-bff7-359a4aa2bc38)\/",
        "PublishDate" : "\/Date(2008,2,1,0,0,0,0)\/",
        "Status" : 0,
        "Title" : "How to Cook Chinese Food"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(f6a265ab-86e5-4fbf-937c-49923604b91d)\/",
        "PublishDate" : "\/Date(2007,4,4,0,0,0,0)\/",
        "Status" : 1,
        "Title" : "How to Cook Japanese Food"
      },
      {
        " ObjectType " : "SampleCode.Book",
        "Author" : "Lisa Andrews",
        "Id" : "\/Guid(2e80eb25-b64a-4506-b87b-2fff6ddb3f57)\/",
        "PublishDate" : "\/Date(2009,0,3,0,0,0,0)\/",
        "Status" : 1,
        "Title" : "Best Recipe"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Patrick Hines",
        "Id" : "\/Guid(704655a3-c136-469c-a578-f79652a93f9b)\/",
        "PublishDate" : "\/Date(2005,11,1,0,0,0,0)\/",
        "Status" : 0,
        "Title" : "Family Recipe"
      }
    ]
  }
]
```

4.4 Add a Book to a Catalog

In this example, the protocol client submits a request to add a book to the catalog and to retrieve a list of all of the books in the catalog.

The protocol client sends the following message**:**

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="29" ObjectPathId="28" />
    <ObjectPath Id="31" ObjectPathId="30" />
    <ObjectPath Id="33" ObjectPathId="32" />
    <Query Id="34" ObjectPathId="30">
      <Query SelectAllProperties="true">
        <Properties />
      </Query>
      <ChildItemQuery SelectAllProperties="true">
        <Properties />
      </ChildItemQuery>
    </Query>
  </Actions>
  <ObjectPaths>
    <StaticProperty Id="28" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="30" ParentId="28" Name="Books" />
    <Method Id="32" ParentId="30" Name="Add">
      <Parameters>
        <Parameter TypeId="{dda98aeb-f87d-490f-9a61-be08644ad461}">
          <Property Name="Author" Type="String">Neil Black</Property>
          <Property Name="PublishDate" Type="DateTime">2009-08-01T00:00:00.0000000</Property>
          <Property Name="Status" Type="Enum">2</Property>
          <Property Name="Title" Type="String">Simple Cookbook</Property>
        </Parameter>
      </Parameters>
    </Method>
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : null
  },
  29,
  {
    "IsNull" : false
  },
  31,
  {
    "IsNull" : false
  },
  33,
  {
    "IsNull" : false
  },
  34,
  {
    " ObjectType " : "SampleCode.BookCollection",
    "_Child_Items_" : [
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(3387ac63-e73d-421f-bff7-359a4aa2bc38)\/",
        "PublishDate" : "\/Date(2008,2,1,0,0,0,0)\/",
        "Status" : 0,
```

```
        "Title" : "How to Cook Chinese Food"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Soha Kamal",
        "Id" : "\/Guid(f6a265ab-86e5-4fbf-937c-49923604b91d)\/",
        "PublishDate" : "\/Date(2007,4,4,0,0,0,0)\/",
        "Status" : 1,
        "Title" : "How to Cook Japanese Food"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Lisa Andrews",
        "Id" : "\/Guid(2e80eb25-b64a-4506-b87b-2fff6ddb3f57)\/",
        "PublishDate" : "\/Date(2009,0,3,0,0,0,0)\/",
        "Status" : 1,
        "Title" : "Best Recipe"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Patrick Hines",
        "Id" : "\/Guid(704655a3-c136-469c-a578-f79652a93f9b)\/",
        "PublishDate" : "\/Date(2005,11,1,0,0,0,0)\/",
        "Status" : 0,
        "Title" : "Family Recipe"
      },
      {
        "_ObjectType_" : "SampleCode.Book",
        "Author" : "Neil Black",
        "Id" : "\/Guid(63a687c6-921b-f898-4204-22f09533f28e)\/",
        "PublishDate" : "\/Date(2009,7,1,0,0,0,0)\/",
        "Status" : 2,
        "Title" : "Simple Cookbook"
      }
    ]
  }
]
```

4.5 Unsuccessfully Add a Book to a Catalog

In this example, the protocol client submits a request to add a book to the catalog. The protocol server returns a CSOM error JSON value, as specified section 3.1.4.1.8.1.2, because the book already exists in the catalog.

The protocol client sends the following message:

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="37" ObjectPathId="36" />
    <ObjectPath Id="39" ObjectPathId="38" />
    <ObjectPath Id="41" ObjectPathId="40" />
    <Query Id="42" ObjectPathId="38">
      <Query SelectAllProperties="true">
        <Properties />
      </Query>
      <ChildItemQuery SelectAllProperties="true">
        <Properties />
      </ChildItemQuery>
```

```
    </Query>
  </Actions>
  <ObjectPaths>
    <StaticProperty Id="36" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="38" ParentId="36" Name="Books" />
    <Method Id="40" ParentId="38" Name="Add">
      <Parameters>
        <Parameter TypeId="{dda98aeb-f87d-490f-9a61-be08644ad461}">
          <Property Name="Author" Type="String">Lisa Andrews</Property>
          <Property Name="PublishDate" Type="DateTime">2006-07-20T00:00:00.0000000</Property>
          <Property Name="Status" Type="Enum">0</Property>
          <Property Name="Title" Type="String">Best Recipe</Property>
        </Parameter>
      </Parameters>
    </Method>
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : {
      "ErrorMessage" : "The book with title 'Best Recipe' already exits in the book store.",
      "ErrorValue" : null,
      "ErrorCode" : -2147024809,
      "ErrorTypeName" : "System.ArgumentException"
    }
  }
]
```

4.6 Retrieve Book Sample Content

In this example, a protocol client sends a request to retrieve book's sample content stream.

The protocol client sends the following message:

```
<Request
  AddExpandoFieldTypeSuffix="true"
  SchemaVersion="15.0.0.0"
  LibraryVersion="15.0.0.0"
  ApplicationName=".NET Library"
  xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009">
  <Actions>
    <ObjectPath Id="45" ObjectPathId="44" />
    <ObjectPath Id="47" ObjectPathId="46" />
    <ObjectPath Id="49" ObjectPathId="48" />
    <Query Id="50" ObjectPathId="48">
      <Query SelectAllProperties="true">
        <Properties />
      </Query>
    </Query>
    <Method Name="GetSampleStream" Id="51" ObjectPathId="48" />
    <ObjectPath Id="53" ObjectPathId="52" />
    <Query Id="54" ObjectPathId="52">
      <Query SelectAllProperties="false">
        <Properties>
          <Property Name="Author" ScalarProperty="true" />
          <Property Name="Status" ScalarProperty="true" />
        </Properties>
      </Query>
```

```
    </Query>
    <Method Name="GetSampleStream" Id="55" ObjectPathId="52" />
  </Actions>
  <ObjectPaths>
    <StaticProperty Id="44" TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" />
    <Property Id="46" ParentId="44" Name="Books" />
    <Method Id="48" ParentId="46" Name="GetById">
      <Parameters>
        <Parameter Type="Guid">{3387ac63-e73d-421f-bff7-359a4aa2bc38}</Parameter>
      </Parameters>
    </Method>
    <Method Id="52" ParentId="46" Name="GetById">
      <Parameters>
        <Parameter Type="Guid">{704655a3-c136-469c-a578-f79652a93f9b}</Parameter>
      </Parameters>
    </Method>
  </ObjectPaths>
</Request>
```

The protocol server responds with the following message including the HTTP headers:

```
HTTP/1.1 200 OK
Content-Type: multipart/related;type="application/jop+json";boundary="742FC209-4650-4E0E-
8BBF-
F47E18AD3028+id=1";start="<http://sharepoint.microsoft.com/client/634551857588785297>";start-
info="application/json"
Server: Microsoft-IIS/7.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Wed, 26 Oct 2011 00:29:18 GMT
Content-Length: 1506


--742FC209-4650-4E0E-8BBF-F47E18AD3028+id=1
Content-ID: <http://sharepoint.microsoft.com/client/634551857588785297>
Content-Transfer-Encoding: eightbit
Content-Type: application/jop+json;charset=utf-8;type="application/json"
Content-Length: 658

[
{
"SchemaVersion":"15.0.0.0","LibraryVersion":"15.0.3421.3000","ErrorInfo":null
},45,{
"IsNull":false
},47,{
"IsNull":false
},49,{
"IsNull":false
},50,{
"_ObjectType_":"SampleCode.Book","Author":"Soha Kamal","Id":"\/Guid(3387ac63-e73d-421f-bff7-
359a4aa2bc38)\/","PublishDate":"\/Date(2008,2,1,0,0,0,0)\/","Status":0,"Title":"How to Cook
Chinese Food"
},51,"\/Binary(http%3a%2f%2fsharepoint.microsoft.com%2fclient%2fattachment%2f0%2f634551857588
775295)\/",53,{
"IsNull":false
},54,{
"_ObjectType_":"SampleCode.Book","Author":"Patrick Hines","Status":0
},55,"\/Binary(http%3a%2f%2fsharepoint.microsoft.com%2fclient%2fattachment%2f1%2f634551857588
780296)\/"
]
--742FC209-4650-4E0E-8BBF-F47E18AD3028+id=1
Content-ID: <http://sharepoint.microsoft.com/client/attachment/0/634551857588775295>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 48
```

```
Sample Content of book How to Cook Chinese Food.
--742FC209-4650-4E0E-8BBF-F47E18AD3028+id=1
Content-ID: <http://sharepoint.microsoft.com/client/attachment/1/634551857588780296>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 37

Sample Content of book Family Recipe.
--742FC209-4650-4E0E-8BBF-F47E18AD3028+id=1--
```

4.7 Update Book Sample Content

In this example, a protocol client sends a request to update book's sample content stream.

The protocol client sends the following message including the HTTP headers:

```
POST http://www.example.com/ vti bin/client.svc/ProcessQuery HTTP/1.1
Content-Type: multipart/related;type="application/xop+xml";boundary="8F66EEF4-511A-4328-B8F7-
B25B75D7A236+id=1";start="<http://sharepoint.microsoft.com/client/634551857589435427>";start-
info="application/xml"
Content-Length: 1961


--8F66EEF4-511A-4328-B8F7-B25B75D7A236+id=1
Content-ID: <http://sharepoint.microsoft.com/client/634551857589435427>
Content-Transfer-Encoding: 8bit
Content-Type: application/xop+xml;charset=utf-8;type="application/xml"
Content-Length: 1221

<Request AddExpandoFieldTypeSuffix="true" SchemaVersion="15.0.0.0" LibraryVersion="15.0.0.0"
ApplicationName=".NET Library"
xmlns="http://schemas.microsoft.com/sharepoint/clientquery/2009"><Actions><ObjectPath Id="57"
ObjectPathId="56" /><ObjectPath Id="59" ObjectPathId="58" /><ObjectPath Id="61"
ObjectPathId="60" /><Method Name="UpdateSampleStream" Id="62"
ObjectPathId="60"><Parameters><Parameter Type="Binary"><Include
href="cid:http://sharepoint.microsoft.com/63" /></Parameter></Parameters></Method><ObjectPath
Id="65" ObjectPathId="64" /><Method Name="UpdateSampleStream" Id="66"
ObjectPathId="64"><Parameters><Parameter Type="Binary"><Include
href="cid:http://sharepoint.microsoft.com/67"
/></Parameter></Parameters></Method></Actions><ObjectPaths><StaticProperty Id="56"
TypeId="{acc57e47-24b0-4400-b1c7-aa1cf3c9542d}" Name="Catalog" /><Property Id="58"
ParentId="56" Name="Books" /><Method Id="60" ParentId="58"
Name="GetById"><Parameters><Parameter Type="Guid">{3387ac63-e73d-421f-bff7-
359a4aa2bc38}</Parameter></Parameters></Method><Method Id="64" ParentId="58"
Name="GetById"><Parameters><Parameter Type="Guid">{704655a3-c136-469c-a578-
f79652a93f9b}</Parameter></Parameters></Method></ObjectPaths></Request>
--8F66EEF4-511A-4328-B8F7-B25B75D7A236+id=1
Content-ID: <http://sharepoint.microsoft.com/63>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 26

New sample content of book
--8F66EEF4-511A-4328-B8F7-B25B75D7A236+id=1
Content-ID: <http://sharepoint.microsoft.com/67>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream
Content-Length: 28

New sample content of book 2
--8F66EEF4-511A-4328-B8F7-B25B75D7A236+id=1--
```

The protocol server responds with the following message:

```
[
  {
    "SchemaVersion" : "15.0.0.0",
    "LibraryVersion" : "15.0.3421.3000",
    "ErrorInfo" : null
  },
  57,
  {
    "IsNull" : false
  },
  59,
  {
    "IsNull" : false
  },
  61,
  {
    "IsNull" : false
  },
  65,
  {
    "IsNull" : false
  }
]
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: XML Schema

6.1 Request XML Schema

For ease of implementation, the full XML schema for request XML is provided in the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="clientquery"
targetNamespace="http://schemas.microsoft.com/sharepoint/clientquery/2009"
elementFormDefault="qualified"
xmlns:mstns="http://schemas.microsoft.com/sharepoint/clientquery/2009"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="GuidType">
    <xs:restriction base="xs:string">
      <xs:pattern value="\{[0-9A-Fa-f]{8}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-Fa-f]{4}-[0-9A-
Fa-f]{12}\}"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="ObjectPathIdType">
    <xs:restriction base="mstns:IdType"/>
  </xs:simpleType>
  <xs:simpleType name="IdType">
    <xs:restriction base="xs:integer"/>
  </xs:simpleType>
  <xs:simpleType name="ActionIdType">
    <xs:restriction base="mstns:IdType"/>
  </xs:simpleType>
  <xs:simpleType name="TypeIdGuidType">
    <xs:restriction base="mstns:GuidType"/>
  </xs:simpleType>
  <xs:simpleType name="TypeFunctionTypeType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Boolean"/>
      <xs:enumeration value="Byte"/>
      <xs:enumeration value="SByte"/>
      <xs:enumeration value="Char"/>
      <xs:enumeration value="Int16"/>
      <xs:enumeration value="UInt16"/>
      <xs:enumeration value="Int32"/>
      <xs:enumeration value="UInt32"/>
      <xs:enumeration value="Int64"/>
      <xs:enumeration value="UInt64"/>
      <xs:enumeration value="DateTime"/>
      <xs:enumeration value="Single"/>
      <xs:enumeration value="Double"/>
      <xs:enumeration value="Decimal"/>
      <xs:enumeration value="TimeSpan"/>
      <xs:enumeration value="String"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="PropertyNameType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="StaticPropertyNameType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="MethodNameType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="StaticMethodNameType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:simpleType name="MethodParameterTypeType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Null"/>
      <xs:enumeration value="Boolean"/>
```

```
        <xs:enumeration value="Char"/>
        <xs:enumeration value="Byte"/>
        <xs:enumeration value="SByte"/>
        <xs:enumeration value="Int16"/>
        <xs:enumeration value="UInt16"/>
        <xs:enumeration value="Int32"/>
        <xs:enumeration value="UInt32"/>
        <xs:enumeration value="Int64"/>
        <xs:enumeration value="UInt64"/>
        <xs:enumeration value="DateTime"/>
        <xs:enumeration value="Single"/>
        <xs:enumeration value="Double"/>
        <xs:enumeration value="Decimal"/>
        <xs:enumeration value="TimeSpan"/>
        <xs:enumeration value="Guid"/>
        <xs:enumeration value="String"/>
        <xs:enumeration value="Base64Binary"/>
        <xs:enumeration value="Dictionary"/>
        <xs:enumeration value="Array"/>
        <xs:enumeration value="Enum"/>
        <xs:enumeration value="Number"/>
        <xs:enumeration value="Unspecified"/>
        <xs:enumeration value="Binary"/>
      </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="MethodParameterType" mixed="true">
      <xs:sequence>
        <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:complexContent>
              <xs:extension base="mstns:MethodParameterType">
                <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
              </xs:extension>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>
        <xs:element name="Object" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="Include" minOccurs="0" maxOccurs="1">
          <xs:complexType>
            <xs:attribute name="href" use="required" type="xs:string"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="optional"/>
      <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="optional"/>
      <xs:attribute name="Type" type="mstns:MethodParameterTypeType" use="optional"/>
    </xs:complexType>
    <xs:complexType name="MethodParameterNullType">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Null"/>
    </xs:complexType>
    <xs:complexType name="MethodParameterBooleanType">
      <xs:simpleContent>
        <xs:extension base="xs:boolean">
          <xs:attribute name="Type" type="xs:string" use="required" fixed="Boolean"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="MethodParameterCharType">
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="Type" type="xs:string" use="required" fixed="Char"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="MethodParameterByteType">
      <xs:simpleContent>
        <xs:extension base="xs:unsignedByte">
          <xs:attribute name="Type" type="xs:string" use="required" fixed="Byte"/>
```

```
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterSByteType">
  <xs:simpleContent>
    <xs:extension base="xs:byte">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="SByte"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterInt16Type">
  <xs:simpleContent>
    <xs:extension base="xs:short">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int16"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterUInt16Type">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedShort">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt16"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterInt32Type">
  <xs:simpleContent>
    <xs:extension base="xs:int">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int32"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterUInt32Type">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedInt">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt32"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterInt64Type">
  <xs:simpleContent>
    <xs:extension base="xs:long">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Int64"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterUInt64Type">
  <xs:simpleContent>
    <xs:extension base="xs:unsignedLong">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="UInt64"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterSingleType">
  <xs:simpleContent>
    <xs:extension base="xs:float">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Single"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterDoubleType">
  <xs:simpleContent>
    <xs:extension base="xs:double">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Double"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterDecimalType">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
```

```
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Decimal"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterTimeSpanType">
  <xs:simpleContent>
    <xs:extension base="xs:duration">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="TimeSpan"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterDateTimeType">
  <xs:simpleContent>
    <xs:extension base="xs:dateTime">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="DateTime"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterGuidType">
  <xs:simpleContent>
    <xs:extension base="mstns:GuidType">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Guid"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterStringType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="String"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterByteArrayType">
  <xs:simpleContent>
    <xs:extension base="xs:base64Binary">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Base64Binary"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterDictionaryType">
  <xs:sequence>
    <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:complexContent>
          <xs:extension base="mstns:MethodParameterType">
            <xs:attribute name="Name" type="xs:string" use="required"/>
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="Type" type="xs:string" use="required" fixed="Dictionary"/>
</xs:complexType>
<xs:complexType name="MethodParameterNumberType">
  <xs:simpleContent>
    <xs:extension base="xs:decimal">
      <xs:attribute name="Type" type="xs:string" use="required" fixed="Number"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="MethodParameterObjectPathType">
  <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
</xs:complexType>
<xs:complexType name="MethodParameterBinaryType">
  <xs:sequence>
    <xs:element name="Include" minOccurs="1" maxOccurs="1">
      <xs:complexType>
        <xs:attribute name="href" use="required" type="xs:string"/>
      </xs:complexType>
```

```
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="MethodParameterValueObjectType">
    <xs:sequence>
      <xs:element name="Property" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:complexContent>
            <xs:extension base="mstns:MethodParameterType">
              <xs:attribute name="Name" type="xs:string" use="required"/>
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  </xs:complexType>
  <xs:complexType name="MethodParameterArrayType">
    <xs:sequence>
      <xs:element name="Object" minOccurs="0" maxOccurs="unbounded"
type="mstns:MethodParameterType"/>
    </xs:sequence>
    <xs:attribute name="Type" type="xs:string" use="required" fixed="Array"/>
  </xs:complexType>
  <xs:complexType name="MethodParameterEnumType">
    <xs:simpleContent>
      <xs:extension base="xs:integer">
        <xs:attribute name="Type" type="xs:string" use="required" fixed="Enum"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:complexType name="MethodParameterUnspecifiedType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="Type" type="xs:string" use="required" fixed="Unspecified"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:simpleType name="VersionStringType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]{1,8}\.[0-9]{1,8}\.[0-9]{1,8}\.[0-9]{1,8}"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="RequestType">
    <xs:sequence>
      <xs:element name="Actions" type="mstns:ActionsType"/>
      <xs:element name="ObjectPaths" type="mstns:ObjectPathsType"/>
    </xs:sequence>
    <xs:attribute name="AddExpandoFieldTypeSuffix" type="xs:boolean" use="optional"/>
    <xs:attribute name="SchemaVersion" type="mstns:VersionStringType" use="required"/>
    <xs:attribute name="LibraryVersion" type="mstns:VersionStringType" use="required"/>
    <xs:attribute name="ApplicationName" use="optional">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:maxLength value="128"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="ObjectPathsType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="Property" type="mstns:ObjectPathPropertyType"/>
        <xs:element name="StaticProperty" type="mstns:ObjectPathStaticPropertyType"/>
        <xs:element name="Method" type="mstns:ObjectPathMethodType"/>
        <xs:element name="StaticMethod" type="mstns:ObjectPathStaticMethodType"/>
        <xs:element name="Constructor" type="mstns:ObjectPathConstructorType"/>
        <xs:element name="Identity" type="mstns:ObjectPathObjectIdentityNameType"/>
      </xs:choice>
```

```
      </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ObjectPathPropertyType">
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="ParentId" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
    </xs:complexType>
    <xs:complexType name="ObjectPathStaticPropertyType">
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
      <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
    </xs:complexType>
    <xs:complexType name="ObjectPathMethodType">
      <xs:sequence>
        <xs:element name="Parameters" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="ParentId" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
    </xs:complexType>
    <xs:complexType name="ObjectPathStaticMethodType">
      <xs:sequence>
        <xs:element name="Parameters" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
      <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
    </xs:complexType>
    <xs:complexType name="ObjectPathConstructorType">
      <xs:sequence>
        <xs:element name="Parameters" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
    </xs:complexType>
    <xs:complexType name="ObjectPathObjectIdentityNameType">
      <xs:attribute name="Id" type="mstns:ObjectPathIdType" use="required"/>
      <xs:attribute name="Name" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:complexType name="ActionsType">
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
          <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
          <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
          <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
          <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
          <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
```

```
        <xs:element name="Query" type="mstns:ActionQueryType"/>
        <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
        <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
        <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:group name="ActionGroup">
    <xs:choice>
      <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
      <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
      <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
      <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
      <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
      <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
      <xs:element name="Query" type="mstns:ActionQueryType"/>
      <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
      <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
      <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
    </xs:choice>
  </xs:group>
  <xs:complexType name="ActionSetPropertyType">
    <xs:sequence>
      <xs:element name="Parameter" type="mstns:MethodParameterType"/>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
    <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ActionSetStaticPropertyType">
    <xs:sequence>
      <xs:element name="Parameter" type="mstns:MethodParameterType"/>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
    <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ActionInvokeMethodType">
    <xs:sequence>
      <xs:element name="Parameters" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
    <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
    <xs:attribute name="Version" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="ActionInvokeStaticMethodType">
    <xs:sequence>
      <xs:element name="Parameters" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Parameter" type="mstns:MethodParameterType" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
    <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
```

```
  </xs:complexType>
  <xs:complexType name="ActionInstantiateObjectPathType">
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ActionObjectIdentityQueryType">
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ActionQueryType">
    <xs:sequence>
      <xs:element name="Query" type="mstns:QueryType"/>
      <xs:element name="ChildItemQuery" type="mstns:ChildItemQueryType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
    <xs:attribute name="ObjectPathId" type="mstns:ObjectPathIdType" use="required"/>
  </xs:complexType>
  <xs:complexType name="QueryType">
    <xs:sequence>
      <xs:element name="Properties">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Property" type="mstns:QueryPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="SelectAllProperties" type="xs:boolean" use="optional"/>
  </xs:complexType>
  <xs:complexType name="ChildItemQueryType">
    <xs:complexContent>
      <xs:extension base="mstns:QueryType">
        <xs:sequence>
          <xs:element name="QueryableExpression"
type="mstns:ExpressionQueryableExpressionType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="QueryPropertyType">
    <xs:sequence>
      <xs:element name="Query" type="mstns:QueryType" minOccurs="0"/>
      <xs:element name="ChildItemQuery" type="mstns:ChildItemQueryType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
    <xs:attribute name="ScalarProperty" type="xs:boolean" use="optional"/>
    <xs:attribute name="SelectAll" type="xs:boolean" use="optional"/>
  </xs:complexType>
  <xs:complexType name="ExceptionHandlingScopeSimpleType">
    <xs:sequence>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
        <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
        <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
        <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
        <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
        <xs:element name="ObjectIdentityQuery" type="mstns:ActionObjectIdentityQueryType"/>
        <xs:element name="Query" type="mstns:ActionQueryType"/>
        <xs:element name="ExceptionHandlingScope" type="mstns:ExceptionHandlingScopeType"/>
        <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
        <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExceptionHandlingScopeType">
    <xs:sequence>
```

```
      <xs:element name="TryScope">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
              <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
              <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
              <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
              <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
              <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
              <xs:element name="Query" type="mstns:ActionQueryType"/>
              <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
              <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
              <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
            </xs:choice>
          </xs:sequence>
          <xs:attribute name="Id" type="mstns:IdType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="CatchScope" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
              <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
              <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
              <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
              <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
              <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
              <xs:element name="Query" type="mstns:ActionQueryType"/>
              <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
              <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
              <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
            </xs:choice>
          </xs:sequence>
          <xs:attribute name="Id" type="mstns:IdType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="FinallyScope" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
              <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
              <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
              <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
              <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
              <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
              <xs:element name="Query" type="mstns:ActionQueryType"/>
              <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
              <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
              <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
            </xs:choice>
          </xs:sequence>
          <xs:attribute name="Id" type="mstns:IdType" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  </xs:complexType>
```

```
  <xs:complexType name="ConditionalScopeType">
    <xs:sequence>
      <xs:element name="Test">
        <xs:complexType>
          <xs:all>
            <xs:element name="Body" type="mstns:ExpressionType"/>
          </xs:all>
        </xs:complexType>
      </xs:element>
      <xs:element name="IfTrueScope" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
              <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
              <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
              <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
              <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
              <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
              <xs:element name="Query" type="mstns:ActionQueryType"/>
              <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
              <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
              <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
            </xs:choice>
          </xs:sequence>
          <xs:attribute name="Id" type="mstns:IdType" use="required"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="IfFalseScope" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="SetProperty" type="mstns:ActionSetPropertyType"/>
              <xs:element name="SetStaticProperty" type="mstns:ActionSetStaticPropertyType"/>
              <xs:element name="Method" type="mstns:ActionInvokeMethodType"/>
              <xs:element name="StaticMethod" type="mstns:ActionInvokeStaticMethodType"/>
              <xs:element name="ObjectPath" type="mstns:ActionInstantiateObjectPathType"/>
              <xs:element name="ObjectIdentityQuery"
type="mstns:ActionObjectIdentityQueryType"/>
              <xs:element name="Query" type="mstns:ActionQueryType"/>
              <xs:element name="ExceptionHandlingScope"
type="mstns:ExceptionHandlingScopeType"/>
              <xs:element name="ExceptionHandlingScopeSimple"
type="mstns:ExceptionHandlingScopeSimpleType"/>
              <xs:element name="ConditionalScope" type="mstns:ConditionalScopeType"/>
            </xs:choice>
          </xs:sequence>
          <xs:attribute name="Id" type="mstns:IdType" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Id" type="mstns:ActionIdType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
```

```
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:group name="ExpressionGroup">
    <xs:choice>
      <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
      <xs:element name="NOT" type="mstns:ExpressionType"/>
      <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
      <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
      <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
      <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
      <xs:element name="ExpressionParameter" type="mstns:ExpressionParameterExpressionType"/>
      <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
      <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
      <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
    </xs:choice>
  </xs:group>
  <xs:complexType name="ExpressionLeftRightOperandExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
```

```
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ExpressionPropertyExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Name" type="mstns:PropertyNameType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExpressionStaticPropertyExpressionType">
    <xs:sequence/>
    <xs:attribute name="Name" type="mstns:StaticPropertyNameType" use="required"/>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExpressionMethodExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
```

```
      <xs:element name="Parameters" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="NOT" type="mstns:ExpressionType"/>
              <xs:element name="ExpressionMethod"
type="mstns:ExpressionMethodExpressionType"/>
              <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
              <xs:element name="ExpressionProperty"
type="mstns:ExpressionPropertyExpressionType"/>
              <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
              <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
              <xs:element name="ExpressionConvert"
type="mstns:ExpressionConvertExpressionType"/>
              <xs:element name="ExpressionTypeIs"
type="mstns:ExpressionTypeIsExpressionType"/>
              <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
            </xs:choice>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Name" type="mstns:MethodNameType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExpressionStaticMethodExpressionType">
    <xs:sequence>
      <xs:element name="Parameters" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
              <xs:element name="NOT" type="mstns:ExpressionType"/>
              <xs:element name="ExpressionMethod"
type="mstns:ExpressionMethodExpressionType"/>
              <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
              <xs:element name="ExpressionProperty"
type="mstns:ExpressionPropertyExpressionType"/>
              <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
              <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
              <xs:element name="ExpressionConvert"
type="mstns:ExpressionConvertExpressionType"/>
              <xs:element name="ExpressionTypeIs"
type="mstns:ExpressionTypeIsExpressionType"/>
              <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
            </xs:choice>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
```

```
    </xs:sequence>
    <xs:attribute name="Name" type="mstns:StaticMethodNameType" use="required"/>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="required"/>
  </xs:complexType>
  <xs:complexType name="ExpressionConvertExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Type" use="optional">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="Boolean"/>
          <xs:enumeration value="Byte"/>
          <xs:enumeration value="SByte"/>
          <xs:enumeration value="Char"/>
          <xs:enumeration value="Int16"/>
          <xs:enumeration value="UInt16"/>
          <xs:enumeration value="Int32"/>
          <xs:enumeration value="UInt32"/>
          <xs:enumeration value="Int64"/>
          <xs:enumeration value="UInt64"/>
          <xs:enumeration value="DateTime"/>
          <xs:enumeration value="Single"/>
          <xs:enumeration value="Double"/>
          <xs:enumeration value="Decimal"/>
          <xs:enumeration value="TimeSpan"/>
          <xs:enumeration value="String"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="TypeId" use="optional" type="mstns:TypeIdGuidType"/>
  </xs:complexType>
  <xs:complexType name="ExpressionTypeIsExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="GT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LT" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="EQ" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="GE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="LE" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="AND" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="OR" type="mstns:ExpressionLeftRightOperandExpressionType"/>
        <xs:element name="NOT" type="mstns:ExpressionType"/>
        <xs:element name="ExpressionMethod" type="mstns:ExpressionMethodExpressionType"/>
        <xs:element name="ExpressionStaticMethod"
type="mstns:ExpressionStaticMethodExpressionType"/>
        <xs:element name="ExpressionProperty" type="mstns:ExpressionPropertyExpressionType"/>
```

```
        <xs:element name="ExpressionStaticProperty"
type="mstns:ExpressionStaticPropertyExpressionType"/>
        <xs:element name="ExpressionParameter"
type="mstns:ExpressionParameterExpressionType"/>
        <xs:element name="ExpressionConvert" type="mstns:ExpressionConvertExpressionType"/>
        <xs:element name="ExpressionTypeIs" type="mstns:ExpressionTypeIsExpressionType"/>
        <xs:element name="ExpressionConstant" type="mstns:MethodParameterType"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Type" use="optional" type="mstns:TypeFunctionTypeType"/>
    <xs:attribute name="TypeId" use="optional" type="mstns:TypeIdGuidType"/>
  </xs:complexType>
  <xs:complexType name="ExpressionParameterExpressionType">
    <xs:attribute name="Name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:group name="ExpressionQueryableExpressionGroup">
    <xs:choice>
      <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
      <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
      <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
      <xs:element name="QueryableObject">
        <xs:complexType/>
      </xs:element>
    </xs:choice>
  </xs:group>
  <xs:complexType name="ExpressionQueryableExpressionType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
        <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
        <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="QueryableObject">
          <xs:complexType/>
        </xs:element>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ExpressionQueryableTakeType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
        <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
        <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="QueryableObject">
          <xs:complexType/>
        </xs:element>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Count" type="xs:nonNegativeInteger"/>
  </xs:complexType>
  <xs:complexType name="ExpressionQueryableOfTypeType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="Take" type="mstns:ExpressionQueryableTakeType"/>
        <xs:element name="OfType" type="mstns:ExpressionQueryableOfTypeType"/>
        <xs:element name="Where" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="OrderBy" type="mstns:ExpressionQueryableWhereType"/>
```

```
        <xs:element name="OrderByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenBy" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="ThenByDescending" type="mstns:ExpressionQueryableWhereType"/>
        <xs:element name="QueryableObject">
          <xs:complexType/>
        </xs:element>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="Type" type="mstns:TypeFunctionTypeType" use="optional"/>
    <xs:attribute name="TypeId" type="mstns:TypeIdGuidType" use="optional"/>
  </xs:complexType>
  <xs:complexType name="ExpressionQueryableWhereType">
    <xs:sequence>
      <xs:element name="Test">
        <xs:complexType>
          <xs:all>
            <xs:element name="Body" type="mstns:ExpressionType"/>
            <xs:element name="Parameters">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Parameter">
                    <xs:complexType>
                      <xs:attribute name="Name" type="xs:string"/>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:all>
        </xs:complexType>
      </xs:element>
      <xs:element name="Object" type="mstns:ExpressionQueryableExpressionType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Request" type="mstns:RequestType"/>
</xs:schema>
```

7 Appendix B: Product Behavior

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

- Microsoft SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

<1> Section 2.2.4.6: This type is available only in SharePoint Foundation 2013.

<2> Section 2.2.5.18: This type is available only in SharePoint Foundation 2013.

<3> Section 2.2.5.19: This type is available only in SharePoint Foundation 2013.

<4> Section 3.1.4.1.3.26.28: This type is available only in SharePoint Foundation 2013.

<5> Section 3.1.4.1.3.26.29: This type is available only in SharePoint Foundation 2013.

<6> Section 3.1.4.1.3.26.30: This type is available only in SharePoint Foundation 2013.

<7> Section 3.1.4.1.6.2.3: For SharePoint Foundation 2013, if the left operand and right operand have different CSOM types and both of their types are one of these numeric types: **CSOM Decimal, CSOM Double, CSOM Single, CSOM UInt64, CSOM Int64, CSOM UInt32, CSOM Int32, CSOM UInt16, CSOM Int16, CSOM Byte, CSOM SByte** (these types are listed with priority from high to low), the server will first convert them to the same type before comparison. The type used after the conversion is the one with the higher priority of the two operands' types.

<8> Section 3.1.4.1.6.2.3: For SharePoint Foundation 2013, if the left operand and right operand have different CSOM types and both of their types are one of these numeric types: **CSOM Decimal, CSOM Double, CSOM Single, CSOM UInt64, CSOM Int64, CSOM UInt32, CSOM Int32, CSOM UInt16, CSOM Int16, CSOM Byte, CSOM SByte** (these types are listed with priority from high to low), the server will first convert them to the same type before comparison. The type used after the conversion is the one with the higher priority of the two operands' types.

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

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

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

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

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

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

9 Index

T

U

V

X