[MS-OCPROTO]:

Office Client Protocols Overview

This document provides an overview of the protocols in the Microsoft Office system. It is intended for use in conjunction with the Microsoft protocol technical documents, publicly available standard specifications, network programming art, and Microsoft Windows distributed systems concepts. It assumes that the reader is either familiar with the aforementioned material or has immediate access to it.

This system does not require use of Microsoft programming tools or programming environments to implement the protocols within it. Implementers who have access to Microsoft programming tools and environments are free to take advantage of them.

specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

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

Preliminary Documentation. This particular Open Specifications document provides documentation for past and current releases and/or for the pre-release version of this technology. This document provides final documentation for past and current releases and preliminary documentation, as applicable and specifically noted in this document, for the pre-release version. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. Because this documentation might change between the pre-release version and the final version of this technology, there are risks in relying on this preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.

Abstract

This document describes the intended functionality of the Microsoft Office system and how the protocols within this system interact. It provides examples of some common user scenarios. It does not restate the processing rules and other details that are specific to each protocol. Those details are described in the protocol specifications for each of the protocols and data structures that make up this system.

The Office system is designed to help users design, develop, collect, share, and manage documents, data, and other types of information. The system consists of a collection of components, services, protocols, structures, and supporting protocol clients and protocol servers.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.01	Major	Initial Availability
6/27/2008	1.0	Minor	Revised and edited technical content
10/6/2008	1.01	Editorial	Revised and edited technical content
12/12/2008	1.02	Editorial	Revised and edited technical content
7/13/2009	1.03	Major	Revised and edited the technical content
8/28/2009	1.04	Editorial	Revised and edited the technical content
11/6/2009	1.05	Editorial	Revised and edited the technical content
2/19/2010	2.0	Major	Updated and revised the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Major	Updated and revised the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
8/26/2010	2.05	Minor	Clarified the meaning of the technical content.
9/27/2010	2.06	Minor	Clarified the meaning of the technical content.
11/15/2010	2.06	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.06	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.7	Minor	Clarified the meaning of the technical content.
6/10/2011	2.7	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.8	Minor	Clarified the meaning of the technical content.
4/11/2012	2.8	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.9	Minor	Clarified the meaning of the technical content.
10/8/2012	2.10	Minor	Clarified the meaning of the technical content.
2/11/2013	2.11	Minor	Clarified the meaning of the technical content.
7/30/2013	2.11	None	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.12	Minor	Clarified the meaning of the technical content.
2/10/2014	2.12	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
4/30/2014	3.0	Major	Significantly changed the technical content.
7/31/2014	4.0	Major	Significantly changed the technical content.
10/30/2014	5.0	Major	Significantly changed the technical content.
9/4/2015	5.0	None	No changes to the meaning, language, or formatting of the technical content.
7/15/2016	5.0	None	No changes to the meaning, language, or formatting of the technical content.
9/14/2016	5.0	None	No changes to the meaning, language, or formatting of the technical content.
9/22/2016	5.0	None	No changes to the meaning, language, or formatting of the technical content.
4/27/2018	6.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

Microsoft Office is a client-based system that is designed to facilitate the design, development, and management of content and data by information workers, developers, and IT professionals. The system consists of protocol clients and client-based components that can function as stand-alone applications, integrated applications that communicate with each other, and integrated applications that communicate with each other and supporting protocol servers. Protocol clients use the data structures, file formats, and protocols that are described in section 2.2 and related documents.

The primary protocol clients in the Office system are:

- Access – Desktop database application that helps users track and report data, and share data more securely by using the Web.
- Excel – Spreadsheet application that helps users analyze, report, and manage data.
- InfoPath – Form development application that enables teams and organizations to gather, share, and reuse information by using electronic forms.
- OneNote – Digital notebook application that enables users to gather, organize, and search notes and other types of information, and to share those notes with others.
- Outlook – Internet messaging application that also provides a comprehensive time and information manager, enabling users to prioritize, organize, and search information.
- PowerPoint – Presentation application that enables users to create and broadcast presentations, and it offers extensive graphics and formatting capabilities.
- Publisher – Desktop publishing application that enables users to create, personalize, and distribute a wide range of publications and marketing materials in-house.
- SharePoint Workspace and Groove – Collaboration application that enables teams to work together from virtually any location.
- Word - Document authoring application that provides a comprehensive set of writing tools, and helps users design, create, and share documents.

1.1 Glossary

This document uses the following terms:

Active Directory Service Interfaces (ADSI): A directory service model and a set of Component Object Model (COM) interfaces. ADSI enables Windows applications and Active Directory Domain Services (AD DS) clients to gain access to several network directory services, including AD DS.

ActiveX control: A reusable software control, such as a check box or button, that uses ActiveX technology and provides options to users or runs macros or scripts that automate a task. See also ActiveX object.

alert: An **Internet message** that is sent to subscribers automatically to notify them when user-defined criteria are met. Alerts are generated automatically when items such as documents, webpages, list items, sites, or other resources on a server are changed.

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

blog: A website that contains a series of posts about a subject and is arranged in reverse chronological order. Also referred to as web log.

broadcast: A style of resource location or data transmission in which a client makes a request to all parties on a network simultaneously (a one-to-many communication). Also, a mode of resource location that does not use a name service.

broadcast session: A sharing session initiated by a presenter that is used for sharing the presenter's view of a document with one or more attendees.

browser-enable: The process of converting an InfoPath form template into a format that can be rendered in a web browser, and publishing it to and activating it on a protocol server that is running InfoPath Forms Services.

cabinet (.cab) file: A single file that stores multiple compressed files to facilitate storage or transmission.

check in: The process of placing a file or project into a source repository. This releases the lock for editing and enables other users to view the updated file or check out the file. See also **check out**.

check out: The process of retrieving a writable copy of a file or project from a source repository. This locks the file for editing to prevent other users from overwriting or editing it inadvertently.

Component Object Model (COM): An object-oriented programming model that defines how objects interact within a single process or between processes. In **COM**, clients have access to an object through interfaces implemented on the object. For more information, see [MS-DCOM].

cookie: A small data file that is stored on a user's computer and carries state information between participating protocol servers and protocol clients.

data connection: A connection between an InfoPath form template and an external data source, as specified by settings in an InfoPath **form template (.xsn) file** or a **Universal Data Connection (.udc, .udcx) file**.

data source: A database, web service, disk, file, or other collection of information from which data is queried or submitted. Supported data sources vary based on application and data provider.

digital certificate: See the "digital certificate definition standard," as described in [X509].

document library: A type of list that is a container for documents and folders.

document property: A name/value pair that serves as metadata for a document.

Document Workspace site: A SharePoint site that is based on a Document Workspace site template and has a template identifier value of "1". A Document Workspace site is used for planning, posting, and working together on a document or a set of related documents.

endpoint: A network-specific address of a remote procedure call (RPC) server process for remote procedure calls. The actual name and type of the endpoint depends on the RPC protocol sequence that is being used. For example, for RPC over TCP (RPC Protocol Sequence ncacn_ip_tcp), an endpoint might be TCP port 1025. For RPC over Server Message Block (RPC Protocol Sequence ncacn_np), an endpoint might be the name of a named pipe. For more information, see [C706].

form: A document with a set of controls into which users can enter information. Controls on a form can be bound to elements in the data source of the form, such as fields and groups. See also bind.

form file: An XML file that contains data that is entered into an InfoPath form by using a web browser or Microsoft InfoPath.

form server: A server that can host XML-based electronic forms and that supports rendering those forms in a web browser.

form template: A file or set of files that defines the data structure, appearance, and behavior of a **form**.

form template (.xsn) file: A **cabinet (.cab) file** with an .xsn file name extension that contains the files that comprise a form template.

forms authentication: An **authentication** method in which protocol clients redirect unauthenticated requests to an HTML form by using **HTTP**. If the protocol client authenticates the request, the system issues a **cookie** that stores the credentials or a key for reacquiring the identity. In subsequent requests, the cookie is submitted in request headers and the requests are authenticated and authorized by an ASP.NET event handler that uses the validation method that is specified by the protocol client.

front-end web server: A server that hosts webpages, performs processing tasks, and accepts requests from protocol clients and sends them to the appropriate back-end server for further processing.

HTTP GET: An HTTP method for retrieving a resource, as described in [RFC2616].

HTTP HEAD: An HTTP method for retrieving header information for a resource, as described in [RFC2616].

HTTP OPTIONS: An HTTP method for determining the options and requirements that are associated with a resource, or the capabilities of a protocol server, as described in [RFC2616].

HTTP POST: An HTTP method, as described in [RFC2616].

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 1.1 (HTTP/1.1): Version 1.1 of the Hypertext Transfer Protocol (HTTP), as described in [RFC2068].

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].

Input Method Editor (IME): An application that is used to enter characters in written Asian languages by using a standard 101-key keyboard. An IME consists of both an engine that converts keystrokes into phonetic and ideographic characters and a dictionary of commonly used ideographic words.

Internet message: A message, such as an email message, that conforms to the syntax that is described in [RFC2822].

Kerberos principal: A unique individual account known to the Key Distribution Center (KDC). Often a user, but it can be a service offering a resource on the network.

Lightweight Directory Access Protocol (LDAP): The primary access protocol for Active Directory. Lightweight Directory Access Protocol (LDAP) is an industry-standard protocol, established by the Internet Engineering Task Force (IETF), which allows users to query and update information in a directory service (DS), as described in [MS-ADTS]. The Lightweight Directory Access Protocol can be either version 2 [RFC1777] or version 3 [RFC3377].

list: (1) A container within a SharePoint site that stores list items. A list has a customizable schema that is composed of one or more fields.

(2) An organization of a region of cells into a tabular structure in a workbook.

list view: A named collection of settings for querying and displaying items in a SharePoint list. There are two types of views: Personal, which can be used only by the user who created the view; and Public, which can be used by all users who have permission to access to the site.

mail merge: The process of merging information into a document from a **data source**, such as an address book or database, to create customized documents, such as form letters or mailing labels.

minor version: An iteration of a software component, document, or list item that is in progress or has changed only slightly from the previous version. For an item on a SharePoint site, the minor version number is never "0" (zero) and is incremented for each new version of an item, unless a major version is explicitly published. When minor versioning is disabled on a SharePoint site, only major version numbers are incremented, and the minor version is always "0" (zero).

NT LAN Manager (NTLM) Authentication Protocol: A protocol using a challenge-response mechanism for **authentication** in which clients are able to verify their identities without sending a password to the server. It consists of three messages, commonly referred to as Type 1 (negotiation), Type 2 (challenge) and Type 3 (authentication). For more information, see [MS-NLMP].

Office Add-in: A cloud-enabled app that integrates rich, scenario-focused content and services into an Office application or equivalent protocol client.

OLE DB: A set of interfaces that are based on the Component Object Model (COM) programming model and expose data from a variety of sources. These interfaces support the amount of Database Management System (DBMS) functionality that is appropriate for a data store and they enable a data store to share data.

Online Analytical Processing (OLAP): A technology that uses multidimensional structures to provide access to data for analysis. The source data for OLAP is stored in data warehouses in a relational database. See also cube.

Open Database Connectivity (ODBC): A standard software API method for accessing data that is stored in a variety of proprietary personal computer, minicomputer, and mainframe databases. It is an implementation of [ISO/IEC9075-3:2008] and provides extensions to that standard.

picture library: A type of **document library** that is optimized for storing digital pictures or graphics.

PivotTable: An interactive table that summarizes large amounts of data from various sources by using format and calculation methods. Row and column headings can be rotated to view different summaries of the source data, filter the data, or display detail data for specific areas.

presence: A status indicator on a client device that is transmitted by using the Wide Area Network Device Presence Protocol (WAN DPP).

presentation: A collection of slides that are intended to be viewed by an audience.

presentation slide: A slide that contains the content that can be displayed during a slide show. A presentation slide can derive formatting and content from a main master slide or a title master slide.

Reading Layout view: A document view that displays a document as it will appear on a printed page and is optimized for reading a document on a computer screen. Two pages are displayed simultaneously, side-by-side.

rights: Tasks that a user is permitted to perform on a computer, site, domain, or other system resource. See also permission.

Server Message Block (SMB): A protocol that is used to request file and print services from server systems over a network. The SMB protocol extends the CIFS protocol with additional security, file, and disk management support. For more information, see [CIFS] and [MS-SMB].

shared space: A set of tools that is synchronized between different endpoints, as described in [MS-GRVDYNM].

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

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

slide: A frame that contains text, shapes, pictures, or other content. A slide is a digital equivalent to a traditional film slide.

Slide Library: A type of a document library that is optimized for storing and reusing presentation slides that conform to the format described in [ISO/IEC-29500:2008].

slide show: A delivery of a sequence of presentation slides, typically to an audience.

SOAP: A lightweight protocol for exchanging structured information in a decentralized, distributed environment. **SOAP** uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics. SOAP 1.2 supersedes SOAP 1.1. See [SOAP1.2-1/2003].

social rating: A user-defined value that indicates the perceived quality of a webpage or item on a SharePoint site or the Internet. Individual users create these ratings and, by default, share them with other users.

social tag: A user-defined keyword and hyperlink to a webpage or item on a SharePoint site or the Internet. Individual users create these tags and, by default, share them with other users.

subsite: A complete website that is stored in a named subdirectory of another website. The parent website can be the top-level site of a site collection or another subsite. Also referred to as subweb.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].

Universal Data Connection (.udc, .udcx) file: An XML file that has a .udc or .udcx file name extension that contains user credentials and other authentication information that is used to connect to a data source.

Universal Naming Convention (UNC): A string format that specifies the location of a resource. For more information, see [MS-DTYP] section 2.2.57.

Web Distributed Authoring and Versioning Protocol (WebDAV): The Web Distributed Authoring and Versioning Protocol, as described in [RFC2518] or [RFC4918].

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.

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.

[CIFS] Leach, P. and Naik, D., "A Common Internet File System (CIFS/1.0) Protocol", March 1997, http://www.microsoft.com/about/legal/protocols/BSTD/CIFS/draft-leach-cifs-v1-spec-02.txt

[MC-FPSEWM] Microsoft Corporation, "FrontPage Server Extensions: Website Management Protocol".

[MS-ADTS] Microsoft Corporation, "Active Directory Technical Specification".

[MS-ALERTSS] Microsoft Corporation, "Alerts Service Protocol".

[MS-ASWS] Microsoft Corporation, "Access Services Protocol".

[MS-CER2] Microsoft Corporation, "Corporate Error Reporting V.2 Protocol".

[MS-CER] Microsoft Corporation, "Corporate Error Reporting Version 1.0 Protocol".

[MS-COPYS] Microsoft Corporation, "Copy Web Service Protocol".

[MS-DPSP] Microsoft Corporation, "Digest Protocol Extensions".

[MS-DWSS] Microsoft Corporation, "Document Workspace Web Service Protocol".

[MS-ESURL] Microsoft Corporation, "Excel Services Publishing Protocol".

[MS-FORMS] Microsoft Corporation, "Forms Service Protocol".

[MS-FPSE] Microsoft Corporation, "FrontPage Server Extensions Remote Protocol".

[MS-FSDAP] Microsoft Corporation, "Forms Services Design and Activation Web Service Protocol".

[MS-FSFDP] Microsoft Corporation, "Forms Services Feature Detection Protocol".

[MS-FSPP] Microsoft Corporation, "Forms Services Proxy Web Service Protocol".

[MS-FSSHTTP] Microsoft Corporation, "File Synchronization via SOAP over HTTP Protocol".

[MS-GRVPROT] Microsoft Corporation, "Groove Protocols Overview".

[MS-IMESYN] Microsoft Corporation, "IMESync Syntax Structure".

[MS-INFODCF] Microsoft Corporation, "InfoPath Data Connection File Download Protocol".

[MS-IPFF2] Microsoft Corporation, "InfoPath Form Template Format Version 2".

[MS-IPFFX] Microsoft Corporation, "InfoPath Form File Format".

[MS-IPFF] Microsoft Corporation, "InfoPath Form Template Format".

[MS-KILE] Microsoft Corporation, "Kerberos Protocol Extensions".

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

[MS-MERX] Microsoft Corporation, "Microsoft Error Reporting Extension to Corporate Error Reporting Version 1.0 Protocol".

[MS-METAWEB] Microsoft Corporation, "MetaWeblog Extensions Protocol".

[MS-NLMP] Microsoft Corporation, "NT LAN Manager (NTLM) Authentication Protocol".

[MS-OBPAS] Microsoft Corporation, "Office Broadcast Participant Service".

[MS-OBPRS] Microsoft Corporation, "Office Broadcast Presentation Service".

[MS-OFBA] Microsoft Corporation, "Office Forms Based Authentication Protocol".

[MS-OMPWHP] Microsoft Corporation, "Office Mobile PowerPoint Web Handler Protocol".

[MS-OMWWH] Microsoft Corporation, "Office Mobile Word Web Handler Protocol".

[MS-OSALER] Microsoft Corporation, "Alerts Interoperability Protocol".

[MS-OUTSPS] Microsoft Corporation, "Lists Client Sync Protocol".

[MS-OWEMXML] Microsoft Corporation, "Office Web Extensibility Manifest Format".

[MS-OXPROTO] Microsoft Corporation, "Exchange Server Protocols System Overview".

[MS-PASS] Microsoft Corporation, "Passport Server Side Include (SSI) Version 1.4 Protocol".

[MS-PWBDPS] Microsoft Corporation, "PowerPoint Web Broadcast Discovery Protocol".

[MS-PWBHPS] Microsoft Corporation, "PowerPoint Web Broadcast Host Protocol".

[MS-PWBPS] Microsoft Corporation, "PowerPoint Web Broadcast Protocol".

[MS-PWEDPS] Microsoft Corporation, "PowerPoint Web Editor Data Protocol".

[MS-PWPHP] Microsoft Corporation, "PowerPoint Web Presentation Handler Protocol".

[MS-PWVPDP] Microsoft Corporation, "PowerPoint Web Viewer Presentation Data Protocol".

[MS-PWVRSC] Microsoft Corporation, "PowerPoint Web Viewer Rendered Static Content Structure".

[MS-RMPR] Microsoft Corporation, "Rights Management Services (RMS): Client-to-Server Protocol".

[MS-RMSI] Microsoft Corporation, "Rights Management Services (RMS): ISV Extension Protocol".

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

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

[MS-SLIDELI] Microsoft Corporation, "Slide Library Web Service Protocol".

[MS-SMB2] Microsoft Corporation, "Server Message Block (SMB) Protocol Versions 2 and 3".

[MS-SMB] Microsoft Corporation, "Server Message Block (SMB) Protocol".

[MS-SPNG] Microsoft Corporation, "Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) Extension".

[MS-SQMCS] Microsoft Corporation, "Software Quality Metrics (SQM) Client-to-Service Version 1 Protocol".

[MS-SSAS] Microsoft Corporation, "SQL Server Analysis Services Protocol".

[MS-STSSYN] Microsoft Corporation, "StsSync Data Structure".

[MS-TDS] Microsoft Corporation, "Tabular Data Stream Protocol".

[MS-UDCX] Microsoft Corporation, "Universal Data Connection 2.0 XML File Format".

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

[MS-UPSDWS] Microsoft Corporation, "User Profile Social Data Web Service Protocol".

[MS-VERSS] Microsoft Corporation, "Versions Web Service Protocol".

[MS-VIEWSS] Microsoft Corporation, "Views Web Service Protocol".

[MS-WDVME] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Microsoft Extensions".

[MS-WDVMODUU] Microsoft Corporation, "Office Document Update Utility Extensions".

[MS-WDVSE] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Server Extensions".

[MS-WDV] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions".

[MS-WEBDAVE] Microsoft Corporation, "Web Distributed Authoring and Versioning Error Extensions Protocol".

[MS-WEBSS] Microsoft Corporation, "Webs Web Service Protocol".

[MS-WOPI] Microsoft Corporation, "Web Application Open Platform Interface Protocol".

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

[MS-WSSCAP] Microsoft Corporation, "Windows SharePoint Services Collaborative Application Protocol".

[MS-WWSP] Microsoft Corporation, "Workflow Web Service Protocol".

[MSDN-OLEDB] Microsoft Corporation, "Microsoft OLE DB", http://msdn.microsoft.com/en-us/library/ms722784.aspx

[MSDN-OpenDBConnectivity] Microsoft Corporation, "Microsoft Open Database Connectivity (ODBC)", http://msdn.microsoft.com/en-us/library/ms710252.aspx

[MSDN-WSS3CLIENTSIDEAPI] Microsoft Corporation, "Client-Side API Reference", http://msdn.microsoft.com/en-us/library/ms440037.aspx

[ODMA 1.0] ODMA Interoperability Exchange, "Open Document Management 1.0 API", http://odma.info/

[RFC2068] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2068, January 1997, http://www.ietf.org/rfc/rfc2068.txt

[RFC2109] Kristol, D., and Montulli, L., "HTTP State Management Mechanism", RFC 2109, February 1997, http://www.rfc-editor.org/rfc/rfc2109.txt

[RFC2518] Goland, Y., Whitehead, E., Faizi, A., et al., "HTTP Extensions for Distributed Authoring - WebDAV", RFC 2518, February 1999, http://www.ietf.org/rfc/rfc2518.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

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., et al., "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999, http://www.rfc-editor.org/rfc/rfc2617.txt

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000, http://www.rfc-editor.org/rfc/rfc2743.txt

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

[RFC2831] Leach, P. and Newman, C., "Using Digest Authentication as a SASL Mechanism", RFC 2831, May 2000, http://www.ietf.org/rfc/rfc2831.txt

[RFC4120] Neuman, C., Yu, T., Hartman, S., and Raeburn, K., "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005, https://www.rfc-editor.org/rfc/rfc4120.txt

[RFC4287] Nottingham, M., and Sayre, R., Eds., "The Atom Syndication Format", RFC 4287, December 2005, http://www.rfc-editor.org/rfc/rfc4287.txt

[RFC4559] Jaganathan, K., Zhu, L., and Brezak, J., "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", RFC 4559, June 2006, http://www.rfc-editor.org/rfc/rfc4559.txt

[RFC5023] Gregorio, J., and de hOra, B., Eds., "The Atom Publishing Protocol", RFC 5023, October 2007, http://www.rfc-editor.org/rfc/rfc5023.txt

[RFC822] Crocker, D.H., "Standard for ARPA Internet Text Messages", STD 11, RFC 822, August 1982, http://www.ietf.org/rfc/rfc0822.txt

[RFC959] Postel, J., and Reynolds, J., "File Transfer Protocol (FTP)", RFC 959, October 1985, http://www.ietf.org/rfc/rfc959.txt

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", W3C Note, May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[UDDI 1.0] UDDI.org, "UDDI Programmer's API 1.0", UDDI Published Specification, June 2002, http://www.uddi.org/pubs/ProgrammersAPI-V1.01-Published-20020628.pdf

[UDDI 2.0] OASIS, "UDDI Version 2.04 API Specification", UDDI Committee Specification, July 2002, http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.pdf

[XRML] ContentGuard, Inc., "XrML: Extensible rights Markup Language Version 1.2", 2001, http://contentguard.com/contact-us

Note Contact the owner of the XrML specification for more information.

2 Functional Architecture

The following sections describe the functional architecture of the Microsoft Office system.

2.1 Overview

Microsoft Office is a client-based system that is designed to facilitate the design, development, and management of content and data by information workers, developers, and IT professionals. The system consists of protocol clients and client-based components that can function as stand-alone applications, integrated applications that communicate with each other, and integrated applications that communicate with each other and supporting protocol servers. The primary components of the system are client applications, such as Microsoft Word and Microsoft Excel. These applications use the data structures, file formats, and protocols that are described in this section, section 2.2, and related specifications.

The following sections describe features that are used by most or all protocol clients and components of the Microsoft Office system. Unless otherwise specified, these features are used by all protocol clients and components of the system.

2.1.1 Authentication

Protocol clients use the **authentication** services that are provided by the Microsoft Windows operating system. In some cases, protocol clients use **forms authentication**, as described in [MS-OFBA] and section 2.5.1 of this document. For information about which protocols are used, see section 2.2.1.1.

2.1.2 File Access

To gain access to files, protocol clients can communicate with protocol servers by using any of several protocols, depending on conditions such as the type of server, server capabilities, client and server operating system, and protocol client version.

2.1.2.1 Protocol Discovery and Feature Activation

To help ensure that the correct protocol is used and only supported features are enabled, protocol clients support discovery mechanisms that can be implemented on a protocol server or operating system. The most common system scenarios are gaining access to files that are stored in **Server Message Block (SMB)** file shares or on Web servers.

2.1.2.1.1 Server Message Block File Shares

Protocol clients can access files on **Server Message Block (SMB)** file shares by using the Server Message Block (SMB) Protocol, as described in [MS-SMB], or the Server Message Block (SMB) Version 2.0 Protocol, as described in [MS-SMB2]. These services are provided by the operating system, and the operating system determines which discovery mechanism and protocol version to use.

2.1.2.1.2 Web Servers

Protocol clients do one of the following when they attempt to open a file on a Web server:

- Open the file with read-only permission in browse mode.
- Open the file with write permission in edit mode and lock the file. The lock helps ensure that the protocol client has exclusive access to the file.

To determine the correct action, a protocol client evaluates the full path to the file, factors explicit user actions such as choosing edit mode, and investigates the capabilities of the protocol server that manages the path. The following steps describe this process.

1. The protocol client sends an **HTTP OPTIONS** request to the folder that contains the file, as described in [RFC2616].

2. The Web server does one of the following:

    - Responds with a list of supported methods for the folder with the specified **Uniform Resource Identifier (URI)**. The response also indicates whether the server supports various document management features, as described in section 2.1.3.

    - Rejects the request. If the request is rejected, the protocol client tries to open the file in browse mode by sending an **HTTP GET** request.

3. If the request is not rejected, the protocol client processes the response by evaluating the presence and value of the **X-MSFSSHTTP** header, and then the **MS-Author-Via** header, as described in the following sections.

X-MSFSSHTTP header

The protocol client evaluates the presence and value of the **X-MSFSSHTTP** header in the protocol server response as follows, depending on which protocol client sent the request:

- Microsoft Excel 2013, Microsoft Excel 2010, Microsoft PowerPoint 2013, Microsoft PowerPoint 2010, Microsoft Word 2013, and Microsoft Word 2010 examine the **X-MSFSSHTTP** header value to determine whether to use the File Synchronization via SOAP over HTTP Protocol. If the value is greater than or equal to "1.0", the protocol client uses the File Synchronization via SOAP over HTTP Protocol to access the file. If the value is less than "1.0" or the header is not present, the protocol client does not use the File Synchronization via SOAP over HTTP Protocol to access the file, but proceeds to evaluate the **MS-Author-Via** header, as described following.

- Microsoft OneNote 2013 and Microsoft OneNote 2010 use the File Synchronization via SOAP over HTTP Protocol to access the file, as described in section 2.1.14.

MS-Author-Via header

After the protocol client evaluates the presence and value of the **X-MSFSSHTTP** header, it evaluates the value of the **MS-Author-Via** header, as described in [MS-WDVSE].

If the protocol client uses the File Synchronization via SOAP over HTTP Protocol, as described in [MS-FSSHTTP], it uses the value of the **MS-Author-Via** header to determine whether the protocol server supports the FrontPage Server Extensions Remote Protocol, as described in [MS-FPSE], for data access calls. If the protocol client does not use the File Synchronization via SOAP over HTTP Protocol, it uses the value of the **MS-Author-Via** header to determine which Web authoring protocol to use. The following list describes the conditions under which these determinations are made:

- If the value of the **MS-Author-Via** header is "MS-FP/4.0,DAV", the protocol server supports the FrontPage Server Extensions Remote Protocol and the HTTP Extensions for Distributed Authoring WebDAV, as described in [RFC2518]. The determination of which protocol to use is then based on the version of the operating system and the protocol client that are in use:

    - If installed on Windows Vista operating system or Windows 7 operating system, Microsoft Office 2013, Microsoft Office 2010 suites, or the 2007 Microsoft Office system protocol clients use the HTTP Extensions for Distributed Authoring WebDAV to access the file.

    - If installed on Windows XP operating system, both Office 2010 and the 2007 Office system protocol clients use the FrontPage Server Extensions Remote Protocol to access the file.

- Microsoft Office 2003 protocol clients use the FrontPage Server Extensions Remote Protocol to access the file.

- If the value of the **MS-Author-Via** header is "DAV", the protocol server supports **WebDAV** and does not support the FrontPage Server Extensions Remote Protocol. The protocol client uses the HTTP Extensions for Distributed Authoring WebDAV to access the file.

- If the **MS-Author-Via** header is not present and a **DAV** header is present, the protocol client uses WebDAV to access the file and the following occurs, depending on the value of the **DAV** header:

  - If the value is "1", file locks are neither supported nor recommended. The protocol client does not notify the user that the protocol server does not support file locks. If there are conflicting write operations to the file from separate protocol clients, the changes that are made in the last write operation are used.

  - If the value is "1,2", file locks are supported.

- If the protocol server does not specify an authoring protocol, the protocol client sends an HTTP GET request to open the file as read-only in browse mode. This means that the protocol client can save only a copy of the file; it cannot save changes to the original file. Any data that is received by the HTTP GET request is considered file data unless the protocol server responds with an "Access Denied" or "405 Not Supported" error. An "Access Denied" response consists of one of the following codes: 401, 302 (with additional headers), or 403 (with additional headers).

2.1.2.1.2.1 Web View

Web view allows a user to browse folders and files that are stored on a Web server by using the Open dialog box or the Save dialog box. The following steps describe the process for opening Web view within a protocol client.

1. If the browse path is an **HTTP** path, the protocol client does one of the following, depending on the version of the operating system and the protocol client that are in use:

   - An Office 2013, Office 2010, or the 2007 Office system protocol client that is installed on Windows Vista or Windows 7 sends a HEAD request to the **URL** for the Web view. If the request is successful, Web view is enabled.

   - A Office 2003 protocol client that is installed on Windows Vista and Windows 7, or an Office 2010 or the 2007 Office system protocol client that is installed on Windows XP sends an **HTTP OPTIONS** request to the URL for the Web view. If the response from the protocol server contains the **MicrosoftOfficeWebServer: 5.0_Collab** header, Web view is enabled.

2. The protocol client sends a GET request to obtain the Web view of the specified folder, and then displays the HTML response in the dialog box.

After the user selects a file, the protocol client uses the process that is described in section 2.1.2.1.2 to access the file.

For information about responses from the protocol server, see [MS-WSSCAP] section 3.1.4.6.

2.1.2.1.2.2 Additional Considerations

To enable protocol clients to access files on Web servers, consider the following when configuring the server:

- Protocol clients use the **HTTP OPTIONS** request described in [RFC2616]. A Web server needs to handle this type of request to support full read/write access to a file from within a protocol client.

- When opening a file, a user can be prompted to provide credentials, such as a login name and password, before gaining access to the file. This behavior is expected and occurs if the Web server

requires **authentication** to process an HTTP OPTIONS request that is sent to the **URL** for the folder, or to provide access to a folder or a file. This behavior can be avoided by changing the server configuration to give anonymous users permission to browse the folder.

- If a user provides a **digital certificate** when connecting to a resource by using a Web browser, the user can be prompted to provide a digital certificate again when attempting to access that resource by using a protocol client. This behavior occurs because a protocol client creates a new session when it sends an HTTP OPTIONS request to the Web server or attempts to gain access to a file on that server. Each new session can produce additional security and other types of messages.
- When it sends an HTTP OPTIONS request, a protocol client does not include **cookie** information. If the Web server requires cookies for direct calls to a folder and those cookies are missing, HTTP OPTIONS and file-access requests can fail and the user can be asked repeatedly to provide authentication credentials. This problem is specific to server configurations that depend on cookie information instead of or in combination with authentication information.

2.1.2.1.2.3 HTTP Conversion for UNC Redirector Files

The Windows Web Client service in Windows operating systems allows **WebDAV**-enabled folders to appear as **Universal Naming Convention (UNC)** file shares to a protocol client. This means that a protocol client can be used to open, edit, and save a file in cases where the protocol client cannot interact directly with the HTTP server that hosts the file.

Document collaboration features of the Office system require more functionality than is provided by the Windows Web Client service. Therefore, protocol clients in the Office system determine if a file path is associated with the Windows Web Client service. If it is, the protocol client maps the path to a full Web **URL**, and then opens the file by using an appropriate protocol for the server that is handling requests for that URL. This enables the protocol client to provide all of the supported collaboration features for that document.

2.1.3 Document Management

Protocol clients support a variety of document management features, if those features are implemented on the protocol server that stores the document file. Examples of document management features are document recovery, workflow, version history, **check in** and **check out** operations, **document properties**, and a Document Management task pane. The following sections describe how a protocol client determines whether a document management feature is implemented on a protocol server and the conditions in which a feature is enabled in a protocol client.

2.1.3.1 Protocol Discovery and Feature Activation

When it initiates communications with a protocol server, a protocol client first determines which type of protocol server it is communicating with and which document management features are implemented on the protocol server. The protocol client obtains this information by sending an **HTTP OPTIONS** request to the protocol server, and then evaluating the presence and values of the **MicrosoftSharePointTeamServices** and **DocumentManagementServer** headers in the response. These are single-value and multiple-value HTTP headers, respectively, that conform to the format that is described in [RFC2616]. The following is an example of a **DocumentManagementServer** header in a response:

```
DocumentManagementServer: Version History;Source Control;\r\n
```

The following sections describe the features that can be enabled in a protocol client, depending on which document management features are supported on a protocol server. In general, document

management features of protocol clients require a protocol server to support the FrontPage Server Extensions Web Site Management Protocol, as described in [MC-FPSEWM].

2.1.3.1.1 Document Recovery and Workflow

When it receives a response to an **HTTP OPTIONS** request, an Office 2013, Office 2010, or the 2007 Office system protocol client checks for the presence of the **DocumentManagementServer** header in the response. If the header exists, the protocol client automatically enables the following features and does not assume that the protocol server supports these features:

- Document recovery
- **Reading Layout view**
- Workflow

To support workflow, the protocol server needs to return data by using the Workflow Web Service Protocol, as described in [MS-WWSP].

An Office 2003 protocol client does not check for the **DocumentManagementServer** header in the response and does not enable these features.

2.1.3.1.2 Version History

By default, version history features are enabled in a protocol client only if a protocol server response contains the correct header and header value. In addition, version history features work correctly in a protocol client only if the protocol server supports the FrontPage Server Extensions Web Site Management Protocol, as described in [MC-FPSEWM].

The following table describes additional conditions in which a protocol client enables version history features.

Protocol client version	MicrosoftSharePointTeamServices OPTIONS header	DocumentManagementServer OPTIONS header	Enabled
Office 2003	Exists	Any (exists or not) - not checked	Yes
The 2007 Microsoft Office system	Exists and contains no value or a value that is less than "12"	Any (exists or not)	Yes
The 2007 Office system	Any (exists or not)	Exists and contains the value "Version History"	Yes
Microsoft Office 2010 suites	Exists and contains no value or a value that is less than "12"	Any (exists or not)	Yes
Office 2010	Any (exists or not)	Exists and contains the value "Version History"	Yes
Office 2013	Exists and contains no value or a value that is less than "12"	Any (exists or not)	Yes
Office 2013	Any (exists or not)	Exists and contains the value "Version History"	Yes

2.1.3.1.3 Check In and Check Out

By default, **check in** and **check out** features are enabled in a protocol client only if a protocol server response contains the correct header and header value. In addition, check in and check out features work correctly in a protocol client only if the protocol server supports the FrontPage Server Extensions Web Site Management Protocol, as described in [MC-FPSEWM].

The following table describes additional conditions in which a protocol client enables check in and check out features.

Protocol client version	MicrosoftSharePointTeamServices OPTIONS header	DocumentManagementServer OPTIONS header	Enabled
Office 2003	Exists	Any (exists or not) – not checked	Yes
The 2007 Office system	Exists	Any (exists or not)	Yes
The 2007 Office system	Any (exists or not)	Exists and contains the value "Source Control"	Yes
Office 2010	Exists	Any (exists or not)	Yes
Office 2010	Any (exists or not)	Exists and contains the value "Source Control"	Yes
Office 2013	Exists	Any (exists or not)	Yes
Office 2013	Any (exists or not)	Exists and contains the value "Source Control"	Yes

In addition to these conditions, protocol clients enable check in and check out features only if the current user is permitted to check out the file. A protocol client uses the FrontPage Server Extensions Web Site Management Protocol, as described in [MC-FPSEWM], to examine permission settings for a file on a protocol server.

2.1.3.1.4 Document Properties

Protocol clients provide an integrated property panel that can be used to view and collect property information about a document and to save that information on a protocol server. When a user enters property information for a document by using this panel, the information is stored in the document file. When the file is subsequently saved to a protocol server, the protocol server can extract property information from the file and use that information for document management purposes.

The integrated property panel can be enabled for the following file types: doc, .docm, .docx, .dot, .dotm, .dotx, .pot, .potm .potx, .pps, .ppsm, .ppsx, .ppt, .pptm, .pptx, .vsd, .vst, .xls, .xlsb, .xlsm, .xlsx, .xlt, .xltm, or .xltx. In addition, it can be enabled only if a protocol server response contains the correct header and header value, and the protocol server supports use of the Lists Web Service Protocol, as described in [MS-LISTSWS], for retrieving and setting document properties.

The following table describes additional conditions in which a protocol client enables the integrated property panel.

Protocol client version	MicrosoftSharePointTeamServices OPTIONS header	DocumentManagementServer OPTIONS header	Enabled
Office 2003	Does not apply	Does not apply	Not supported
The 2007 Office system	Exists and contains a value that is greater than "12"	Does not apply	Yes
The 2007 Office system	Does not apply	Exists and contains the value "Properties Schema"	Yes
Office 2010	Exists and contains a value that is greater than "12"	Does not apply	Yes
Office 2010	Does not apply	Exists and contains the value "Properties Schema"	Yes
Office 2013	Exists and contains a value that is greater than "12"	Does not apply	Yes
Office 2013	Does not apply	Exists and contains the value "Properties Schema"	Yes

If the integrated property panel cannot be enabled because the preceding conditions cannot be met, a Web-form property panel can be used instead. The Web-form property panel is a Web **form** that is displayed in a protocol client, typically when a document is saved to a protocol server for the first time by using a protocol client.

By default, the Web-form property panel is enabled only if a protocol server response contains the correct header and header value. In addition, it works correctly only if the protocol server supports use of the FrontPage Server Extensions Web Site Management Protocol, as described in [MC-FPSEWM], to retrieve and set document properties.

The following table describes additional conditions in which a protocol client enables the Web-form property panel.

Protocol client version	MicrosoftOfficeWebServer OPTIONS header	Integrated property panel	Web-form property panel
Office 2003	Exists and contains the value "5.0_Collab"	Does not apply	Yes
The 2007 Office system	Exists and contains the value "5.0_Collab"	Not enabled by the preceding conditions	Yes
Office 2010	Exists and contains the value "5.0_Collab"	Not enabled by the preceding conditions	Yes
Office 2013	Exists and contains the value "5.0_Collab"	Not enabled by the preceding conditions	Yes

2.1.3.1.5 Document Management Task Pane

The Document Management task pane displays property information about a document that is stored on a **Document Workspace site**. It displays information such as document status and related tasks, a list of other **site** members and **presence** information for each member, and a list of other documents in the Document Workspace site.

By default, Office 2013, Office 2010, the 2007 Office system, and Office 2003 enable the Document Management task pane only if a protocol server includes the **MicrosoftSharePointTeamServices OPTIONS** header in its response to an **HTTP OPTIONS** request from the protocol client. In addition, the task pane works correctly only if the protocol server supports the Document Workspace Web Service Protocol, as described in [MS-DWSS].

2.1.4 Data Access

Protocol clients provide various features that can be used to access data that is stored locally or remotely. Such features enable tasks such as importing data into a table or **PivotTable** report in Excel 2013, Excel 2010 or Microsoft Office Excel 2007, linking to and displaying tabular data in Microsoft Access 2013, Microsoft Access 2010 or Microsoft Office Access 2007, and using data for mail-merge operations in Word 2013, Word 2010 or Microsoft Office Word 2007. Many of these features use **OLE DB**, as described in [MSDN-OLEDB], and **Open Database Connectivity (ODBC)**, as described in [MSDN-OpenDBConnectivity], which are client-side APIs for accessing data in local and remote locations by using ODBC drivers.

Typically, these features can be used to access a **data source** by installing an ODBC driver that is appropriate for the type of data source to be accessed, and then configuring a connection to the data source from within a protocol client. ODBC drivers are available from various providers because ODBC is a client-side API that is used broadly across many products and technologies.

To communicate over the wire with a Microsoft SQL Server data source, the OLE DB drivers that are included in the Office system use the Tabular Data Stream Protocol, as described in [MS-TDS].

2.1.5 Information Rights Management

To help users restrict access to document content and e-mail messages, protocol clients support rights management functionality, which is referred to as Information Rights Management (IRM) in the Office system. This functionality enables a user to assign specific **rights** to content. Those rights specify whether content can be accessed, printed, forwarded, or copied by specific people or groups of people. Rights management information is stored by using the eXtensible Rights Markup Language (XrML), as described in [XRML], and that information travels with a document file or e-mail message.

When it processes an IRM-protected document or e-mail message, a protocol client communicates with the Rights Management Client component of the Windows operating system. The operating system in turn communicates with a Rights Management Services (RMS) server, as described in [MS-RMPR], to determine the current user's rights for the document or e-mail message, and then shares that information with the protocol client. Depending on the rights that are specified for a user, the protocol client can then decrypt the document or e-mail message, and enable and disable features that correspond to the user's rights. For example, if content can be copied but not printed, the protocol client enables controls for copying and pasting content and disables controls for printing content.

To support IRM functionality, protocol clients require access to a protocol server that supports the Rights Management Services (RMS) Client-to-Server Protocol, as described in [MS-RMPR].

2.1.6 Active Directory Domain Services

To perform tasks such as finding printers and looking up user information, protocol clients communicate with Active Directory Domain Services (AD DS), and they do so by using **Lightweight**

Directory Access Protocol (LDAP) APIs and **Active Directory Service Interfaces (ADSI)** APIs. When a protocol client calls ADSI APIs, it uses an ADSI LDAP Provider to communicate with AD DS. An ADSI LDAP Provider is a set of **Component Object Model (COM)** objects that implement ADSI, make LDAP calls, and run locally on the computer where an Office protocol client is installed. The LDAP protocols that communicate with AD DS are described in [MS-ADTS].

Protocol clients use ADSI, and therefore AD DS, to find printers based on specified search criteria and to install the appropriate printer drivers.

Microsoft Outlook 2013, Microsoft Outlook 2010, Microsoft Office Outlook 2007, Microsoft InfoPath 2013, Microsoft InfoPath 2010, Microsoft Office InfoPath 2007, Microsoft SharePoint Workspace 2010, and Microsoft Office Groove 2007 use ADSI and LDAP APIs, and consequently the protocols that are described in [MS-ADTS], to find user information, such as e-mail address and group membership, create and delete user accounts or passwords in AD DS, manage personal **sites** in Microsoft SharePoint Server, and integrate with instant messaging services.

2.1.7 Microsoft Error Reporting

Protocol clients support use of the Microsoft Error Reporting service to collect information about events and errors that occur, such as application exceptions, kernel faults, or generic events that are defined by an application. This information can be stored as an error report in a **cabinet (.cab) file**, as described in [MS-MERX].

By using the Corporate Error Reporting Version 1.0 Protocol, as described in [MS-CER], an organization can copy error reports from a set of client computers to a file share on a protocol server. An organization can also configure a protocol client to collect additional information for error reports by using both the Corporate Error Reporting Version 1.0 Protocol and the Microsoft Error Reporting Extension to Corporate Error Reporting Version 1.0 Protocol, as described in [MS-MERX].

As described in [MS-CER], policy settings can be used to specify the location of a file share that contains configuration files for report operations. These configuration files, if present, specify settings for protocol clients, such as whether report information is copied to the file share and additional data to be included in an error report. For more information, see [MS-CER].

If a protocol client is installed on Windows XP or Windows Server 2003 operating system, the files are copied by using the Server Message Block (SMB) Protocol, as described in [MS-SMB], and the Corporate Error Reporting Version 1.0 Protocol, as described in [MS-CER]. If the protocol client is installed on Windows 7 or Windows Vista, the files are copied by using the **HTTP POST** method and the Corporate Error Reporting Version 2.0 Protocol, as described in [MS-CER2].

2.1.8 Customer Experience Improvement Program

Protocol clients support use of the Customer Experience Improvement Program (CEIP) service to collect instrumentation data, which is written to files on a client computer. By using the Corporate CEIP Protocol, an organization can redirect instrumentation files from a set of client computers to a specific protocol server.

Protocol clients collect instrumentation data only for users who choose to participate in the CEIP. The instrumentation data is written to files on the client computer. If the protocol client is configured with policy settings that specify an upload **URL** for the files, as described in [MS-SQMCS], the files are uploaded by using the **HTTP POST** method. The upload URL specifies a service URL as a query string parameter and the protocol redirects the files to the service URL.

2.1.9 ActiveX Controls

To enable additional functionality and system integration when documents and data are viewed by using a Web browser, the Office system installs several **ActiveX controls** on a user's computer. If a Web browser supports use of ActiveX controls, these controls can be invoked the same way any other

type of ActiveX control is invoked from a Web page. Although these controls do not define any additional wire protocols, they are described in this document to facilitate interoperability. For more information about the ActiveX controls that are installed by the system, see [MSDN-WSS3CLIENTSIDEAPI].

The following table identifies and describes ActiveX controls that can be used to access and manage content on SharePoint **sites**, without requiring a protocol client to start.

ActiveX control	Description	Related protocols
CopyCtl	Enables users to copy files, including associated metadata such as creation date and author, to one or more locations on a single protocol server or between different protocol servers.	Copy Web Service Protocol, as described in [MS-COPYS]
ListNet	Enables users to view, edit, and manipulate the data in a SharePoint **list (1)** by using a format that is similar to a table in a database or spreadsheet.	Lists Web Service Protocol, as described in [MS-LISTSWS]
UploadCtl	Enables users to upload multiple documents simultaneously to a **document library** on a SharePoint site.	Windows SharePoint Services Collaborative Application Protocol, as described in [MS-WSSCAP] FrontPage Server Extensions Remote Protocol, as described in [MS-FPSE]

The following table identifies and describes ActiveX controls that start a protocol client, interact with a protocol client by using client-side APIs, or specify settings that are subsequently used by a protocol client.

ActiveX control	Description
ExportDatabase	Enables users to create or open a database that contains data from a SharePoint list (1), by using an application such as Access 2013, Access 2010 or Office Access 2007.
NameCtrl	Enables a Web page to display **presence** information for people and enables users to take various related actions by using an on-object user interface in Windows SharePoint Services. This control integrates with Microsoft Lync Client 2013/Skype for Business, Microsoft Lync 2010 and Microsoft Office Communicator 2007 through client-side APIs.
OISClientLauncher	Starts the Microsoft Office Picture Manager and passes command line parameters that enable users to download, edit, or upload pictures to **picture libraries**.
OpenDocuments	Starts a protocol client and enables users to create a document or edit an existing document. Also enables users to create a document that is based on a specific template, choose to open a document with read-only or read/write permission, and check out a document.
OpenXMLDocuments	Starts InfoPath 2013, InfoPath 2010 or Office InfoPath 2007 and enables users to create or edit XML documents or **forms** by using InfoPath.

ActiveX control	Description
PersonalSite	Configures the location of a user's My Site, which is used by protocol clients to discover and suggest locations of SharePoint sites and libraries where users can save or open files. Also retrieves colleague suggestions that are generated from protocol clients.
PPActiveX	Starts PowerPoint 2013, PowerPoint 2010 or Microsoft Office PowerPoint 2007 and enables users to open **presentations** from a **Slide Library** and publish individual **presentation slides** to a Slide Library.
SharePoint.OfflineClient	Enables users to synchronize lists (2) and document libraries with **shared spaces** in Microsoft SharePoint Foundation 2013, SharePoint Workspace 2010 or Office Groove 2007, and to verify whether an installation of SharePoint Foundation 2013, SharePoint Workspace 2010 or Office Groove 2007 is enabled to synchronize lists (2) and document libraries.
SpreadSheetLauncher	Enables users to import lists (2) from Windows SharePoint Services to Excel 2013, Excel 2010, or Office Excel 2007, and to verify whether an installation of Excel 2013, Excel 2010 or Office Excel 2007 is enabled to export and import lists (2) from Windows SharePoint Services.
StssyncHandler	Returns the name of the application that is used to synchronize event and contact lists (2) between Windows SharePoint Services and a messaging application such as Outlook 2013, Outlook 2010 or Office Outlook 2007.

2.1.10 Microsoft Word

Microsoft Word is a document authoring application that provides a comprehensive set of writing tools, and helps users design, create, and share documents. For common operations such as authenticating users, accessing files, and managing documents and content, Word 2013, Word 2010 and Office Word 2007 use the protocols that are identified and described in section 2.2.1. To support application-specific operations, Word 2013, Word 2010 and Office Word 2007 use additional protocols.

For **mail merge** operations, Word 2013, Word 2010, and Office Word 2007 use the data access protocols that are described in section 2.2.1.4.

To support coauthoring, which is a feature that enables users to simultaneously edit and save changes to a document that is stored on a SharePoint Foundation 2013 or Microsoft SharePoint Foundation 2010 server, Word 2013 and Word 2010 use the File Synchronization via SOAP over HTTP Protocol, as described in [MS-FSSHTTP] and discussed in section 2.2.1.2. Office Word 2007 does not provide this feature.

To support the publication of content to a **blog**, Word 2013, Word 2010 and Office Word 2007 use the ATOM Publishing Protocol, as described in [RFC5023], and the MetaWeblog Extensions Protocol, as described in [MS-METAWEB]. When communicating with blogs that are hosted on protocol servers running SharePoint Foundation 2013 or SharePoint Foundation 2010, Word 2013, Word 2010, and Office Word 2007 use the MetaWeblog Extensions Protocol. For all scenarios, Word 2013, Word 2010 and Office Word 2007 determine which protocol to use based on the service provider that is specified by the user.

To facilitate integration with document management systems, Word 2013, Word 2010 and Office Word 2007 support the Open Document Management 1.0 API, as described in [ODMA 1.0]. This is a client-side API that can be used if an ODMA provider is installed on the client computer.

For a complete list of the protocols that are used for application-specific operations, see section 2.2.2.

2.1.11 Microsoft Excel

Microsoft Excel is a spreadsheet application that helps users analyze, report, and manage data. For common operations such as authenticating users and managing documents, Excel 2013, Excel 2010 and Office Excel 2007 use the protocols that are identified and described in section 2.2.1. To support application-specific operations, Excel 2013, Excel 2010 and Office Excel 2007 use additional protocols.

To connect to SQL Server Analysis Services and access **Online Analytical Processing (OLAP)** data, Excel 2013, Excel 2010 and Office Excel 2007 use the SQL Server Analysis Services Version 8.0 Protocol, as described in [MS-SSAS]. To connect to other types of external **data sources**, Excel 2013, Excel 2010 and Office Excel 2007 use the protocols that are described in section 2.1.4.

To publish a workbook to a protocol server that is running Microsoft Office SharePoint Server 2007, Microsoft SharePoint Server 2010, or SharePoint Foundation 2013 and providing support for Excel Calculation Services, Excel 2013, Excel 2010 and Office Excel 2007 use the Excel Services Publishing Protocol, as described in [MS-ESURL], to form the correct URL and associated query string parameters.

To import and synchronize data with SharePoint **lists (1)**, Excel 2013, Excel 2010 and Office Excel 2007 use the Lists Web Services Protocol, as described in [MS-LISTSWS]. Excel 2013, Excel 2010 and Office Excel 2007 also use the Lists Web Service Protocol to create lists (1) from existing tables in Excel worksheets.

For a complete list of the protocols that are used for application-specific operations, see section 2.2.3.

2.1.12 Microsoft PowerPoint

Microsoft PowerPoint is a presentation application that enables users to create and broadcast **presentations**, and it offers extensive graphics and formatting capabilities. For common operations such as authenticating users, accessing data and files, and managing documents and content, PowerPoint 2013, PowerPoint 2010, and Office PowerPoint 2007 use the protocols that are identified and described in section 2.2.1. To support application-specific operations, PowerPoint 2013, PowerPoint 2010, and Office PowerPoint 2007 use additional protocols.

PowerPoint 2013, PowerPoint 2010, and Office PowerPoint 2007 users can share and reuse individual **presentation slides** by storing them in a **Slide Library** that is hosted on a protocol server running SharePoint Foundation 2013, Microsoft SharePoint Server 2013, SharePoint Server 2010, or Office SharePoint Server 2007. After a presentation slide is added to a Slide Library, any instances of that **slide** in a presentation are associated with the original slide in the Slide Library. If a user opens a presentation that contains the slide, PowerPoint 2013, PowerPoint 2010, and Office PowerPoint 2007 notify the user if the slide has been updated and prompts the user to ignore the update, append the new version of the slide, or update the slide. To query and retrieve information about the content in a Slide Library, PowerPoint 2013, PowerPoint 2010, and Office PowerPoint 2007 use the Slide Library Web Service Protocol, as described in [MS-SLIDELI].

To support coauthoring, which is a feature that enables users to simultaneously edit and save changes to a presentation that is stored on a SharePoint Foundation 2013 or SharePoint Server 2010 server, PowerPoint 2013 or PowerPoint 2010 use the File Synchronization via SOAP over HTTP Protocol, as described in [MS-FSSHTTP] and discussed in section 2.2.1.2. Office PowerPoint 2007 does not provide this feature.

By using a Web browser and Microsoft PowerPoint Online, users can directly view and edit presentations that are stored on a protocol server, without using PowerPoint 2013, PowerPoint 2010, or Office PowerPoint 2007. To support these tasks, PowerPoint Online uses the PowerPoint Web Viewer Presentation Data Protocol, as described in [MS-PWVPDP], to retrieve information about a presentation and display presentation content, and it uses the PowerPoint Web Editor Data Protocol, as described in [MS-PWEDPS], to enable editing of presentation content.

By using PowerPoint 2010, users can broadcast a **slide show** for remote viewers to watch in a Web browser. To initiate and end a broadcast, PowerPoint 2010 uses the PowerPoint Web Broadcast Host Protocol, as described in [MS-PWBHPS]. During a **broadcast**, PowerPoint 2010 communicates with a protocol server by using the PowerPoint Web Broadcast Discovery Protocol, as described in [MS-PWBDPS]. To send updates about the state of the slide show, PowerPoint 2010 uses the PowerPoint Web Broadcast Protocol, as described in [MS-PWBPS]. These three protocols are implemented in SharePoint Foundation 2010.

In PowerPoint 2013, broadcasts are implemented in Microsoft Office Online using Lync Client 2013/Skype for Business and the Office Broadcast Presentation Service protocols, as described in section 2.1.20. Microsoft Lync Server 2013 uses the same protocols to communicate with the protocol server to provide backward capability for Lync 2010 clients. A feature in OneNote 2013 enables users to take notes linked to the presentation using the Office Broadcast Presentation Service Protocol, as described in [MS-OBPAS], as well as the PowerPoint Web Presentation Handler Protocol, as described in [MS-PWPHP].

For a complete list of the protocols that are used for application-specific operations, see section 2.2.4.

2.1.13 Microsoft Access

Microsoft Access is a desktop database application that helps users track and report data, and share data more securely by using the Web. For common operations such as authenticating users, accessing data and files, and managing database files and data, Access 2013, Access 2010 and Office Access 2007 use the protocols that are identified and described in section 2.2.1. To support application-specific operations, Access 2013, Access 2010 and Office Access 2007 use additional protocols.

To access external **data sources**, Access 2013, Access 2010 and Office Access 2007 use the data access protocols that are described in section 2.2.1.4 in addition to **OLE DB**, as described in [MSDN-OLEDB], and **Open Database Connectivity (ODBC)**, as described in [MSDN-OpenDBConnectivity]. ODBC provides client-side APIs for accessing data by using ODBC drivers, and these APIs use the Tabular Data Stream Protocol, as described in [MS-TDS], to communicate with SQL Server.

To add, modify, and synchronize data between an Access database and a SharePoint **list (1)**, Access 2013, Access 2010 and Office Access 2007 use the Lists Web Services Protocol, as described in [MS-LISTSWS]. Access 2013, Access 2010 and Office Access 2007 also use the Lists Web Services Protocol to create lists (1) from tables in existing Access 2013, Access 2010 or Office Access 2007 databases. After a list (1) is created from an existing table, Access 2013, Access 2010 and Office Access 2007 use the Views Web Service Protocol, as described in [MS-VIEWSS], to modify the default view of the list (1) by hiding and displaying specific columns.

Access 2013, Access 2010 and Office Access 2007 also use the Web Part Pages Web Service Protocol, as described in [MS-WPPS], to provide a feature that enables users to open a list (1) as an Access database and display a specific **form**, instead of opening the list (1) in a Web browser.

For a complete list of the protocols that are used for application-specific operations, see section 2.2.5.

2.1.14 Microsoft OneNote

Microsoft OneNote is a digital notebook application that enables users to gather, organize, and search notes and other types of information, and to share those notes with others. For common operations such as authenticating users, OneNote 2013, OneNote 2010 and Microsoft Office OneNote 2007 use the protocols that are identified and described in section 2.2.1, except the data access protocols that are described in section 2.2.1.4 and the document management protocols that are described in section 2.2.1.3. To support application-specific operations, OneNote 2013, OneNote 2010 and Office OneNote 2007 use additional protocols.

To transfer and synchronize files that are stored on a protocol server running SharePoint Products and Technologies, OneNote 2013 or OneNote 2010 determine which Web authoring protocol to use by

sending a **SOAP** call to the protocol server by using the File Synchronization via SOAP over HTTP Protocol, as described in [MS-FSSHTTP], and then processing the response as follows.

1. If the protocol server responds in the format that is defined by the File Synchronization via SOAP over HTTP Protocol, OneNote 2013 or OneNote 2010 use the File Synchronization via SOAP over HTTP Protocol to access the file.

2. If the protocol server does not respond in the format that is defined by the File Synchronization via SOAP over HTTP Protocol or it responds incorrectly, OneNote 2013 or OneNote 2010 use either of the following, depending on the version of the operating system on which it is running:

    - If it is running on Windows Vista or Windows 7, OneNote 2013 or OneNote 2010 use the Web Distributed Authoring and Versioning (WebDAV) Protocol Client Extensions, as described in [MS-WDV].

    - If it is running on Windows XP, OneNote 2013 or OneNote 2010 uses the FrontPage Server Extensions Remote Protocol, as described in [MS-FPSE].

In addition to using the File Synchronization via SOAP over HTTP Protocol, the Web Distributed Authoring and Versioning (WebDAV) Protocol Client Extensions, or the FrontPage Server Extensions Remote Protocol, OneNote 2013 or OneNote 2010 use the protocols that are described in section 2.2.6 to synchronize files that are stored on a protocol server running SharePoint Products and Technologies.

For a complete list of the protocols that are used for application-specific operations, see section 2.2.6.

2.1.15 Microsoft Publisher

Microsoft Publisher is a desktop publishing application that enables users to create, personalize, and distribute a wide range of publications and marketing materials in-house. For common operations such as authenticating users, accessing files, and managing documents and content, Microsoft Publisher 2010 and Microsoft Office Publisher 2007 use the protocols that are identified and described in section 2.2.1.

To support application-specific operations, Publisher 2010 and Office Publisher 2007 use additional protocols.

For **mail merge** and other types of data access operations, Publisher 2010 and Office Publisher 2007 use the protocols that are described in section 2.2.1.4 in addition to **OLE DB**, as described in [MSDN-OLEDB], and **Open Database Connectivity (ODBC)**, as described in [MSDN-OpenDBConnectivity].

Neither Publisher 2010 nor Office Publisher 2007 uses any additional protocols that are specific to it.

2.1.16 Microsoft InfoPath

Microsoft InfoPath is a **form** development application that enables teams and organizations to gather, share, and reuse information by using electronic forms. For common operations such as authenticating users, and accessing and saving files, InfoPath 2013, InfoPath 2010 and Office InfoPath 2007 use the protocols that are identified in section 2.2.1. To support application-specific operations, InfoPath 2013, InfoPath 2010 and Office InfoPath 2007 use additional protocols and custom file formats.

When a user designs a form, InfoPath 2013, InfoPath 2010 and Office InfoPath 2007 store the data structure, appearance, and behavior of the form in a **form template (.xsn) file**. InfoPath 2013 and InfoPath 2010 store this information in a **form template** file that conforms to the file format that is described in [MS-IPFF2]. Office InfoPath 2007 stores this information in a form template (.xsn) file that conforms to the file format that is described in [MS-IPFF]. When a user enters data into a form, InfoPath 2013, InfoPath 2010 and Office InfoPath 2007 store that data in a **form file**, which is a file that conforms to the file format that is described in [MS-IPFFX].

To support broader-scale data collection and reuse, and by using InfoPath 2013 or InfoPath 2010 and SharePoint Server 2013 or Microsoft SharePoint Server 2010, data that has been entered into a form can be used in a **list (1)**. In addition, users can publish form templates to a protocol server that is running SharePoint Server 2013, SharePoint Server 2010 or Office SharePoint Server 2007. SharePoint Server 2013, SharePoint Server 2010 and Office SharePoint Server 2007 provide services that can be used to define the behavior and display of an InfoPath form. SharePoint Server 2013, SharePoint Server 2010 and Office SharePoint Server 2007 support the InfoPath Form Template Format Structure, as described in [MS-IPFF]. SharePoint Server 2013 and SharePoint Server 2010 also support the InfoPath Form Template Format Version 2 Structure, as described in [MS-IPFF2], but Office SharePoint Server 2007 does not.

For a complete list of the protocols that are used for application-specific operations, see section 2.2.7.

2.1.17 Microsoft Outlook

Microsoft Outlook is an Internet messaging application that also provides a comprehensive time and information manager, enabling users to prioritize, organize, and search information. For common operations such as authenticating users and accessing files, Outlook 2013, Outlook 2010 and Office Outlook 2007 use the protocols that are identified and described in section 2.2.1, except the data access protocols that are described in section 2.2.1.4. To support application-specific operations and integration with protocol server technologies, Outlook 2013, Outlook 2010 and Office Outlook 2007 use additional protocols.

To communicate with protocol servers that are running Microsoft Exchange Server, Outlook 2013, Outlook 2010 and Office Outlook 2007 use the protocols that are listed and described in the Exchange Server Protocols System Overview document ([MS-OXPROTO]).

Outlook 2013, Outlook 2010 and Office Outlook 2007 can also integrate with protocol servers that are running SharePoint Server 2013, SharePoint Server 2010 or Office SharePoint Server 2007, primarily by receiving **alerts** about content on SharePoint **sites**, and storing and synchronizing data with SharePoint **lists (1)**. A user of SharePoint Server 2013, SharePoint Server 2010 or Office SharePoint Server 2007 can choose to be notified when a document, Web page, or other type of resource changes on a site. These notifications are referred to as alerts. An alert is a standard Internet message, as described in [RFC2822], that uses X-header fields, as described in [RFC822], to store additional information about that alert. To parse the information in those X-header fields, display the appropriate icons for alerts, and to catalog alert subscriptions, Outlook 2013, Outlook 2010 and Office Outlook 2007 use the Alerts Interoperability Protocol, as described in [MS-OSALER]. To enable users to manage alert subscriptions, Outlook 2013, Outlook 2010 and Office Outlook 2007 use the Alerts Service Protocol, as described in [MS-ALERTSS].

To store and synchronize data with SharePoint lists (1), Outlook 2013, Outlook 2010 and Office Outlook 2007 use the StsSync Data Structure, as described in [MS-STSSYN], and the Lists Web Service Protocol, as described in [MS-LISTSWS]. If Outlook 2013, Outlook 2010 or Office Outlook 2007 is installed on a client computer, the **Actions** menu in the user interface of a SharePoint list (1) provides an option to connect the list (1) to Outlook. When a user selects this option, an stssync URL (stssync://) is opened, Outlook 2013, Outlook 2010 or Office Outlook 2007 is registered with the operating system as a handler for that protocol, and Outlook 2013, Outlook 2010 or Office Outlook 2007 use the parameters specified in the initial stssync message to connect to the list (1), as described in [MS-STSSYN]. After it connects to the list (1), the protocol client creates a local copy of the list (1) data by using the Lists Web Service Protocol, as described in [MS-LISTSWS]. Thereafter, any changes to the local or server copies of the list (1) data are transmitted by using the Lists Web Service Protocol. For more information about these interactions, including how Outlook 2013, Outlook 2010 and Office Outlook 2007 handle the list (1) schema, see [MS-OUTSPS].

For a complete list of the protocols that are used for application-specific operations, see section 2.2.8.

2.1.18 Microsoft SharePoint Workspace and Groove

SharePoint Workspace 2010 and Office Groove 2007 are collaboration applications that enable teams to work together from virtually any location, primarily through the use of **shared spaces**. For common operations such as authenticating users, SharePoint Workspace 2010 and Office Groove 2007 use the protocols that are identified and described in section 2.2.1, except the data access protocols that are described in section 2.2.1.4 and the information rights management protocols that are described in section 2.2.1.5. To support application-specific operations, SharePoint Workspace 2010 and Office Groove 2007 use additional protocols.

To communicate with other protocol clients and protocol servers in the Groove and SharePoint Workspace system, SharePoint Workspace 2010 and Office Groove 2007 use the protocols and structures that are described in the Groove Protocols Overview ([MS-GRVPROT]).

When setting up connections to and synchronizing data with **lists (1)** and **document libraries** on protocol servers that are running Windows SharePoint Services, Office Groove 2007, and SharePoint Workspace 2010 use the following protocols:

- To detect protocol server configurations and capabilities, the FrontPage Server Extensions Remote Protocol, as described in [MS-FPSE].
- To determine which **sites** and lists (1) exist on a protocol server, the Webs Web Service Protocol, as described in [MS-WEBSS].
- To obtain information about a site and its users, the Site Data Web Service Protocol, as described in [MS-SITEDATS], and the UserGroup Web Service Protocol, as described in [MS-UGS].
- To obtain details about **list views** for each list (1) or document library, the Views Web Service Protocol, as described in [MS-VIEWSS].
- To collect information about the **forms** that are associated with each list (1), the Forms Service Protocol, as described in [MS-FORMS].
- To obtain copies of data and files in lists (1) and document libraries, the Lists Web Service Protocol, as described in [MS-LISTSWS].

To synchronize changes between copies of data and files on the protocol client and protocol server, Office Groove 2007 and SharePoint Workspace 2010 use several protocols:

- To synchronize data for list (1) items, the Lists Web Service Protocol, as described in [MS-LISTSWS].
- To synchronize files in document libraries, the SharePoint Files Tool feature in SharePoint Workspace 2010, and Office Groove 2007 use **WebDAV**, as described in [RFC2518].
- To synchronize file attachments for list (1) items and documents and document templates in document libraries, SharePoint Workspace 2010 use the File Synchronization via SOAP over HTTP Protocol, as described in [MS-FSSHTTP], if the protocol server supports that protocol. If the protocol server does not support the File Synchronization via SOAP over HTTP Protocol, files will not be able to be synchronized with SharePoint Workspace 2010.

2.1.19 Microsoft Office Mobile

Microsoft Word Mobile 2010 enables users to open and display documents by using a mobile device. It does so by using the Office Mobile Word Web Handler Protocol, as described in [MS-OMWWH], to communicate with Microsoft Word Online on a protocol server that is running SharePoint Foundation 2013 or SharePoint Foundation 2010. The Office Mobile Word Web Handler Protocol can be used to create a mobile-ready version of a document, and to access the pages, text, and metadata about that version of a document. For more information about these communications and processes, see [MS-OMWWH].

Similarly, Microsoft PowerPoint Mobile 2010 enables users to open and display **presentations** by using a mobile device. It does so by communicating with Microsoft PowerPoint Online on a protocol server that is running SharePoint Foundation 2013 or SharePoint Foundation 2010. To retrieve information about a presentation and images of **presentation slides**, PowerPoint Mobile 2010 uses the Office Mobile PowerPoint Web Handler Protocol, as described in [MS-OMPWHP]. Some of the data that is used by this protocol conforms to the PowerPoint Web Viewer Rendered Static Content structure, as described in [MS-PWVRSC].

2.1.20 Office Broadcast Presentation Service

The Office Broadcast Presentation Service<1> allows hosts to broadcast Microsoft Word and Microsoft PowerPoint documents over the web using web viewers using PowerPoint 2010, PowerPoint 2013, and Word 2013. The presentation service has presenter and attendee protocols, as described in section 2.2.10.

To present to a protocol server running the presentation service, use the Office Broadcast Presentation Service protocol, as described in [MS-OBPRS].

To attend a **broadcast session** from a protocol server running the presentation service, use the Office Broadcast Participant Service protocol, as described in [MS-OBPAS].

2.1.21 Web Application Open Platform Interface

The Web Application Open Platform Interface (WOPI)<2>, as described in [MS-WOPI], defines a set of operations that enables a client to access and change files stored by a protocol server. This allows the client to render files and provide file editing functionality for files stored by the protocol server.

One example of how a client might use WOPI is by providing a browser-based viewer for a specific type of file. That client uses WOPI to get the contents of the file in order to present that content to the user as a web page in a browser.

2.1.22 Apps for Office

Office 2013 includes a new extensibility model for Office clients where web developers can create **Office Add-ins**. An app for Office is an area inside an Office application, containing a Web page that can interact with a document to augment content and provide new interactive content types and functionality. An app for Office consists of both Web page and a manifest file, the structure of which is described in Office Web Extensibility Manifest Format [MS-OWEMXML]).

Collections of apps for Office manifest files, called app catalogs, can be made available through the Office Store, a SharePoint App Catalog (in the form of stand-alone apps for Office or components of a document template solution), or in a shared folder on a local network.

The following Office 2013 applications support apps for Office: Excel 2013, Microsoft Excel Online, Word 2013, PowerPoint 2013, Outlook 2013, Microsoft Outlook Web App, and Microsoft Project 2013.

2.2 Protocol Summary

The following tables provide a comprehensive list of the member protocols of the Microsoft Office system. The member protocols are grouped according to their primary purpose.

2.2.1 Common Protocols

The following tables provide a comprehensive list of the protocols that are used by all or most protocol clients in the Office system for common operations such as authenticating users, accessing files, and managing documents and content. The member protocols are grouped according to their primary purpose.

2.2.1.1 Authentication

Protocols in this table enable protocol clients in the Office system to authenticate users.

Protocol name	Description	Short name
Digest Protocol Extensions	Supports client **authentication** to protocol servers, based on user name and password, and server authentication to protocol clients. These are Windows extensions to the Digest Authentication standard, as described in [RFC2617], "HTTP Authentication: Basic and Digest Access Authentication," and [RFC2831], "Using Digest Authentication as a SASL Mechanism."	[MS-DPSP]
HTTP Authentication: Basic and Digest Access Authentication	Defines an authentication (2) scheme that can be used to verify that both parties to a communication know a shared secret (password). If Digest Access Authentication is used, this verification can be done without sending a password as cleartext.	[RFC2617]
Kerberos Protocol Extensions	Extends the Kerberos Network Authentication Service (V5) protocol, as described in [RFC4120]. These extensions provide additional capabilities for authorization information, including group memberships, interactive logon information and integrity levels, and constrained delegation and encryption supported by **Kerberos principals**.	[MS-KILE]
NT LAN Manager (NTLM) Authentication Protocol	Provided by the Windows operating system, enables authentication between protocol clients and protocol servers when the Kerberos Protocol Extensions, as described in [MS-KILE], cannot be used. In Windows Server 2008 operating system with Service Pack 2 (SP2), Windows Vista, Windows Server 2003, Windows XP, and Windows 2000 Server operating system, Kerberos authentication replaces **NT LAN Manager (NTLM) Authentication Protocol** as the preferred authentication protocol.	[MS-NLMP]
Office Forms Based Authentication Protocol	Implemented by the Office system, establishes a user's identity by using HTTP-based **forms authentication** when other authentication mechanisms are not available.	[MS-OFBA]
Passport Server Side Include (SSI) Version 1.4 Protocol	An HTTP-based protocol that enables protocol clients to authenticate to a partner server with the assistance of an authentication (2) server. Also referred to as the "Passport Tweener" protocol.	[MS-PASS]
Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) Extensions	Defines a negotiation mechanism for the Generic Security Service Application Program Interface (GSS-API), as described in [RFC2743]. SPNEGO provides a framework for two parties that are engaged in authentication (2) to select from a set of possible authentication (2) mechanisms, in a manner that preserves the opaque nature of the security protocols to the application protocol that uses SPNEGO.	[MS-SPNG]
SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows	Provided by Windows 2000 operating system, enhances the security of Web-based transactions by using Kerberos.	[RFC4559]

2.2.1.2 File Access

Protocols in this table enable protocol clients in the Office system to access files in local and remote locations.

Protocol name	Description	Short name
File Synchronization via SOAP over HTTP	Implemented by the Office system, enables protocol clients to synchronize changes to files, and related metadata, that are stored on a protocol server. It also enables a protocol server to process requests that it receives from protocol clients for different types of file-lock operations, which helps prevent merge conflicts. This protocol uses **SOAP**-based request/response message sequences.	[MS-FSSHTTP]
File Transfer Protocol (FTP)	Enables users to open and save files by using protocol clients. Protocol clients do not support file-lock operations when they access files by using FTP.	[RFC959]
FrontPage Server Extensions Remote Protocol	Provides file server functionality that is similar to **WebDAV** by presenting the content of a **website** as a file share to users.	[MS-FPSE]
FrontPage Server Extensions Website Management Protocol	Provides a set of methods that enables users to upload, download, lock, and move files, and to create directories and listings on a protocol server by using protocol clients.	[MC-FPSEWM]
HTTP Extensions for Distributed Authoring -- WebDAV	Extends the **HTTP/1.1** Protocol by providing a set of methods and defining headers and body formats for messages. These extensions enable users to create, query, and manage resources on a Web server by using protocol clients.	[RFC2518]
Hypertext Transfer Protocol -- HTTP/1.1	An application-level protocol for distributed, collaborative, and hypermedia information systems. It is a generic, stateless, protocol that can be used for many tasks, including file access. If other protocols are not supported, protocol clients use the **HTTP GET** method that is defined by this protocol to provide users with read-only access to files.	[RFC2616]
Office Document Update Utility Extensions Protocol	Provides a set of extensions to WebDAV, including header updates, a property that helps optimize protocol interaction during synchronization operations, and a property that allows a protocol server to send virus-infection information about a file to protocol clients. These extensions are used primarily during synchronization operations between protocol clients and **Document Workspace sites**.	[MS-WDVMODUU]
Server Message Block (SMB) Version 1.0 Protocol	Provides a set of extensions to the Common Internet File System (CIFS/1.0) Protocol, as described in [CIFS], and enables protocol clients to connect to protocol servers, establish authenticated contexts for those connections, and issue requests to access files, printers, and named pipes for inter-process communication.	[MS-SMB]
Server Message Block (SMB) Version 2.0 Protocol	Provides a set of extensions and enhancements to the SMB Version 1.0 Protocol, as described in [MS-SMB], and the Common Internet File System (CIFS/1.0) Protocol, as described in [CIFS]. The enhancements generally improve scalability and load balancing.	[MS-SMB2]
Web Distributed Authoring and Versioning Error Extensions Protocol	Extends WebDAV by describing the extended error codes that protocol clients can receive in responses from protocol servers, such as document checked out, **minor version** limit extended, and missing required **document properties**.	[MS-WEBDAVE]

Protocol name	Description	Short name
Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions	Extends WebDAV, as described in [RFC2518], by introducing new headers that enable protocol clients to support management of additional file types and to optimize processing and related interactions with supporting protocol servers.	[MS-WDV]
Web Distributed Authoring and Versioning (WebDAV) Protocol: Microsoft Extensions	Provides a set of Microsoft–specific extensions to WebDAV. These extensions are implemented in all versions of Windows SharePoint Services and are therefore used during interactions between protocol clients and protocol servers that are running a version of Windows SharePoint Services. Most of the extensions are designed to enhance support for authoring and managing documents that are stored on a protocol server. For example, a protocol server can use these extensions to indicate whether it supports various capabilities for document management through other protocols based on the value of a Document Management Server header.	[MS-WDVME]
Web Distributed Authoring and Versioning (WebDAV) Protocol: Server Extensions	Extends WebDAV, as described in [RFC2518], by introducing new headers that enable protocol servers to support management of additional file types and to optimize processing and related interactions with supporting protocol clients.	[MS-WDVSE]
Windows SharePoint Services Collaborative Application Protocol	Enables communications between protocol clients and a **front-end Web server** to retrieve and manipulate various types of content that is stored on a protocol server.	[MS-WSSCAP]

2.2.1.3 Document Management

Protocols in this table enable protocol clients in the Office system to perform document management operations such as checking files in or out, and displaying and saving document properties.

Protocol name	Description	Short name
Document Workspace Web Service Protocol	Defines operations for creating, editing, and deleting files, folders, and **subsites** on a **Document Workspace site** by using the Document Management task pane in a protocol client.	[MS-DWSS]
FrontPage Server Extensions Remote Protocol	Defines operations for accessing, checking in, and checking out files that are stored on a protocol server.	[MS-FPSE]
FrontPage Server Extensions Website Management Protocol	Defines operations for uploading, downloading, locking, and moving files, and for creating directories and listings on a protocol server.	[MC-FPSEWM]
Hypertext Transfer Protocol -- HTTP/1.1	An application-level protocol for distributed, collaborative, and hypermedia information systems. It is a generic, stateless, protocol that can be used for many tasks, including file access. Protocol clients use various **HTTP** methods to perform operations such as determining the type and document management capabilities of a protocol server.	[RFC2616]
Lists Web Service Protocol	Enumerates, retrieves changes to, and retrieves and sets properties for **list (1)** items and files.	[MS-LISTSWS]

Protocol name	Description	Short name
User Profile Social Data Web Service Protocol	Enables a protocol client to add, update, remove, and retrieve **social tags**, notes, and **social ratings** for files and Web pages that are stored on a protocol server. This protocol allows a protocol client to pass criteria to a protocol server and receive a response that lists tags, notes, and ratings.	[MS-UPSDWS]
Versions Web Service Protocol	Provides programmatic access to a view and enables a protocol client to view, delete, and restore a version of a file on a protocol server. It also allows a protocol client to retrieve of all versions of a file.	[MS-VERSS]
Webs Web Service Protocol	Enables a protocol client to retrieve the parent **site** of a file. The **WebUrlFromPageUrl** method, which is defined by this protocol, is commonly used as a precursor to method calls for other protocols.	[MS-WEBSS]
Workflow Web Service Protocol	Defines the communication sequences that protocol clients use to query, start, and manipulate workflows on a document. For a specified document, protocol clients can: query information about existing workflow instances and workflow tasks that are emitted by those instances, query for available workflows that are associated with a list (1), create workflow instances from the workflow association for a document, and modify workflow tasks that are related to a document.	[MS-WWSP]

2.2.1.4 Data Access

Protocols in this table enable protocol clients in the Office system to access data in local and remote locations.

Protocol name	Description	Short name
Tabular Data Stream Protocol	Protocol used for remote data access to SQL Server.	[MS-TDS]
Open Database Connectivity (ODBC)	Client-side API for accessing data in local and remote locations using ODBC drivers.	[MSDN-OpenDBConnectivity]
OLE DB	Client-side COM-based API for accessing data in local and remote locations.	[MSDN-OLEDB]

2.2.1.5 Information Rights Management

Protocols in this table enable protocol clients in the Office system to manage access to document content and e-mail messages.

Protocol name	Description	Short name
Rights Management Services (RMS): Client-to-Server Protocol	Used to obtain and issue **digital certificates** and licenses for creating and working with protected content.	[MS-RMPR]

Protocol name	Description	Short name
Rights Management Services (RMS): ISV Extension Protocol	Facilitates the creation of applications that extend the capabilities of RMS-enabled applications or bridge capabilities of different software systems by allowing for direct communication between applications and RMS servers without the use of the RMS client. This protocol enables applications to decommission protected content and to retrieve a recipient's public key digital certificate.	[MS-RMSI]

2.2.1.6 Active Directory Domain Services

Protocols in this table enable protocol clients in the Office system to communicate with AD DS.

Protocol name	Description	Short name
Active Directory Technical	The primary specification for AD DS, including both Active Directory Domain Services (AD DS) and Active Directory Lightweight Directory Services (AD LDS).	[MS-ADTS]

2.2.1.7 Microsoft Error Reporting

Protocols in this table enable protocol clients in the Office system to collect and report data about events and errors that occur, such as application exceptions, kernel faults, or generic events that are defined by an application.

Protocol name	Description	Short name
Corporate Error Reporting Version 1.0 Protocol	Multi-staged protocol used by Windows XP and Windows Server 2003 clients that allows components to send user mode and kernel mode error reports to a corporation's servers instead of to Microsoft servers. The reports are stored on a configurable shared folder location. This protocol uses SMB as a transport. All versions of the Windows operating system implement basic server-side configurations, but the CER tool is required to implement the full range of server-side configuration options. The CER tool is available only under the software assurance application.	[MS-CER]
Corporate Error Reporting Version 2.0 Protocol	Multi-staged client service protocol that allows components to send user mode and kernel mode error reports to a corporation's servers. This protocol allows components to send failure information to the corporate servers instead of to Microsoft servers. Developers can configure failure report buckets to collect additional failure specific information and also send responses back to the client computers. The server side of the protocol is implemented by System Center Operations Manager 2007. This protocol is used by Windows Vista and Windows 7 clients.	[MS-CER2]

Protocol name	Description	Short name
Microsoft Error Reporting Extension to Corporate Error Reporting Version 1.0 Protocol	Extends the original Corporate Error Reporting V.1 Protocol to support additional kinds of error reporting, additional options for existing protocol details, and more specific requirements about error report contents. This protocol is used by Windows XP and Windows Server 2003 clients.	[MS-MERX]

2.2.1.8 Customer Experience Improvement Program

Protocols in this table enable protocol clients in the Office system to collect instrumentation data for users who choose to participate in the Customer Experience Improvement Program (CEIP).

Protocol name	Description	Short name
Software Quality Metrics (SQM) Client-to-Service Version 1 Protocol	Allows components to send instrumentation files to a corporation's servers instead of to Microsoft servers. This protocol uses Hypertext Transfer Protocol -- HTTP/1.1 [RFC2068] as a transport. This protocol is used by Windows XP and Windows Vista clients.	[MS-SQMCS]

2.2.1.9 IMESync Structure

Protocols in this table enable protocol clients in the Office system to locate and use a custom word **list (1)** for an **Input Method Editor (IME)**.

Protocol name	Description	Short name
IMESync Syntax Structure	Used to locate a remote list (1), which is accessible through the Lists Web Service Protocol [MS-LISTSWS]. More specifically, this structure does the following: ▪ Enables a Web page to specify the location of a remote list (1), which is used to define a custom word list (1) for an IME. ▪ Enables the IME to locate the remote list (1). ▪ Enables the IME to download the list (1) content that defines a custom word list (1).	[MS-IMESYN]
Lists Web Service Protocol	Enumerates, retrieves changes to, and retrieves and sets properties for list (1) items and files.	[MS-LISTSWS]

2.2.2 Microsoft Word

The following table describes the protocols used by Word 2013, Word 2010, or Office Word 2007 to publish to **blogs**.

Protocol name	Description	Short name
MetaWeblog Extensions Protocol	API for publishing blogs (1).	[MS-METAWEB]
The Atom Publishing Protocol	Simple HTTP-based protocol. Creates and updates Web resources.	[RFC5023]
The Atom Syndication Format	Describes the format of the data packet used by the Atom Publishing Protocol.	[RFC4287]

2.2.3 Microsoft Excel

The following table describes the protocols used by Excel 2013, Excel 2010, or Office Excel 2007, in addition to the common protocols covered in section 2.2.1.

Protocol name	Description	Short name
Excel Services Publishing Protocol	Enables a protocol client to form the protocol server URL and associated query string parameters to display the workbook in the browser after the workbook is published to the protocol server.	[MS-ESURL]
Lists Web Service Protocol	Enumerates items, gets changes, gets and sets properties, and checks in or checks out **list (2)** items.	[MS-LISTSWS]
SQL Server Analysis Services Protocol	Enables remote access to **Online Analytical Processing (OLAP)** data from SQL Server Analysis Services.	[MS-SSAS]

2.2.4 Microsoft PowerPoint

The following table describes the protocol used by PowerPoint 2013, PowerPoint 2010, or Office PowerPoint 2007 for communicating with **Slide Libraries**.

Protocol name	Description	Short name
Slide Library Web Service Protocol	Obtains information about **slides** in a PowerPoint Slide Library on a SharePoint **site**.	[MS-SLIDELI]

The following table describes the protocols used by PowerPoint Online for viewing and editing **presentations** stored on a protocol server through a Web browser.

Protocol name	Description	Short name
PowerPoint Web Editor Data Protocol	This protocol enables a protocol client to send a request to modify presentation content to the protocol server and then receive from the protocol server information about the result of the modification. To facilitate this, the protocol allows the protocol client to send a request to the protocol server and then receive from the protocol server information about the existence of **presentation slides**, main master slides, and slide layouts. The protocol client can also send a request to the protocol server and then receive from the protocol server images of slide content.	[MS-PWEDPS]
PowerPoint Web Viewer Presentation Data Protocol	This protocol enables a protocol client to send a request to retrieve presentation content from the protocol server. To facilitate this, the protocol allows the protocol client to request specific pieces of content from a presentation stored on the protocol server. In a presentation slide contained within a presentation, a protocol client can retrieve information describing the presentation slide contents as well as images of the presentation slide contents.	[MS-PWVPDP]

The following table describes the protocols used by PowerPoint 2010 and PowerPoint Online for creating and viewing a **broadcast slide show**.

Protocol name	Description	Short name
PowerPoint Web Broadcast Discovery Protocol	This protocol enables a protocol client to send a request containing client information to the protocol server and then receive information about the endpoint (2) that is most suited for future communication with the protocol client. The protocol server uses the information provided by the protocol client such as the list of supported protocol versions to decide a suitable **endpoint**.	[MS-PWBDPS]
PowerPoint Web Broadcast Host Protocol	This protocol enables a protocol client to upload a presentation to a protocol server, delete a presentation from a protocol server, and retrieve information that is necessary to successfully start a presentation broadcast session. The protocol allows the protocol client to request information such as the protocol server settings, URL to upload the presentation to, location of the PowerPoint Web Broadcast service, and the URL to view the Presentation Broadcast session.	[MS-PWBHPS]
PowerPoint Web Broadcast Protocol	This protocol enables a protocol client to send requests to a protocol server allowing the client to start or end a broadcast session, and to store broadcast state data on the protocol server.	[MS-PWBPS]

In addition to the protocols used in the Office Broadcast Presentation Service<3> (section 2.1.20), the following table describes the protocols used by PowerPoint 2013 and PowerPoint Online for creating and viewing a broadcast slide show.

Protocol name	Description	Short name
PowerPoint Web Presentation Handler Protocol	This protocol enables a protocol client to send a request to retrieve information about presentation content from the protocol server. To facilitate this, the protocol allows the protocol client to request specific pieces of content from a presentation stored on the protocol server.	[MS-PWPHP]

2.2.5 Microsoft Access

The following table describes the protocols used by Access 2013, Access 2010, or Office Access 2007 for syncing, adding, and modifying **list (1)** data and for modifying the default view of a list.

Protocol name	Description	Short name
Lists Web Service Protocol	Enumerates items, gets changes, gets and sets properties, and checks in or checks out list (1) items.	[MS-LISTSWS]
Tabular Data Stream Protocol	Used for remote data access to a protocol server running SQL Server.	[MS-TDS]
Views Web Service Protocol	Provides methods to manage a list (1) view.	[MS-VIEWSS]
Web Part Pages Web Service Protocol	Enables a client to gather the information needed to author Web pages that use protocol server resources and author execution logic that reacts to changes in the protocol server state.	[MS-WPPS]
Webs Web Service Protocol	Maps file path URLs to document libraries for actions such as subsequent calls to the Lists Web Service Protocol [MS-LISTSWS].	[MS-WEBSS]

2.2.6 Microsoft OneNote

The following table describes the additional protocols used by OneNote 2013, OneNote 2010, or Office OneNote 2007 when synchronizing files with protocol servers running Windows SharePoint Services.

Protocol name	Description	Short name
Hypertext Transfer Protocol -- HTTP/1.1	Sends **HTTP OPTIONS** requests to the protocol server to identify the protocol server version and supported protocols. Sends **HTTP HEAD** requests to files and folders to validate their existence, invoke authentication (2), and check file modified times.	[RFC2616]
Lists Web	Enumerates Office OneNote 2007, OneNote 2010, or OneNote 2013 files	[MS-

Protocol name	Description	Short name
Service Protocol	in the document library folder, and efficiently enumerates changed files.	LISTSWS]
Webs Web Service Protocol	Maps file path URLs to document libraries for subsequent calls to the Lists Web Service Protocol [MS-LISTSWS].	[MS-WEBSS]

2.2.7 Microsoft InfoPath

The following table describes the additional protocols used by InfoPath 2013, InfoPath 2010, and Office InfoPath 2007 (unless otherwise noted) when communicating with a **form server** for publishing **form template** files, saving **form files**, and using **data connections**.

Protocol name	Description	Short name
Active Directory Technical	The primary specification for AD DS, including both AD DS and Active Directory Lightweight Directory Services (AD LDS).	[MS-ADTS]
Forms Services Design and Activation Web Service Protocol	Allows a client to communicate with a form server to validate, **browser-enable**, or manipulate settings of form template files in supported scenarios.	[MS-FSDAP]
Forms Services Feature Detection Protocol	Allows a client to detect whether InfoPath Forms Services is present and enabled on the protocol server. InfoPath Forms Services is required to publish a browser-enabled **form template (.xsn) file**.	[MS-FSFDP]
Forms Services Proxy Web Service Protocol	Forwards **SOAP** messages for a client, and returns the targeted Web service response.	[MS-FSPP]
HTTP Over TLS	Enables secure transmission of HTTP traffic.	[RFC2818]
Hypertext Transfer Protocol -- HTTP/1.1	Enables navigation to documents on the Internet.	[RFC2616]
InfoPath Data Connection File Download Protocol	Allows a client to retrieve a **Universal Data Connection (.udc, .udcx) file**.	[MS-INFODCF]
InfoPath Form File Format	Enables a form server, with an associated form template (.xsn) file, to render and edit **form** data from a form file in a Web browser.	[MS-IPFFX]
InfoPath Form Template Format	Enables a form server to render and edit form data in a Web browser.	[MS-IPFF]
InfoPath Form Template Format	Enables a form server to render and edit form data in a Web browser.	[MS-IPFF2]

Protocol name	Description	Short name
Version 2	Supported only by InfoPath 2013 and InfoPath 2010.	
Lists Web Service Protocol	Manipulates the schema and contents of SharePoint **lists (1)**.	[MS-LISTSWS]
Simple Object Access Protocol (SOAP) 1.1	Makes remote procedure calls to a protocol server.	[SOAP1.1]
Site Data Web Service Protocol	Specifies a set of protocol server extensions used to augment a basic HTTP server so that it supports full and incremental indexing. Indexing, in this context, is defined as the process of exploring **website** content and building an index to use for search, systematic cataloging, content auditing, or similar purposes.	[MS-SITEDATS]
Tabular Data Stream Protocol	Enables remote access to data on a protocol server running SQL Server.	[MS-TDS]
UDDI Programmer's API 1.0 UDDI Version 2.04 API	Enables discovery of Web services exposed by a remote server.	[UDDI 1.0] [UDDI 2.0]
Universal Data Connection 2.0 XML File Format	Provides a container for data connection information.	[MS-UDCX]
Webs Web Service Protocol	Provides methods for modifying **sites** in the **site collection**.	[MS-WEBSS]
Windows SharePoint Services Collaborative Application Protocol	Enables a client to retrieve and manipulate various types of content on a protocol server.	[MS-WSSCAP]

2.2.8 Microsoft Outlook

The following table describes the additional protocols used by Outlook 2013, Outlook 2010, and Office Outlook 2007.

Protocol name	Description	Short name
Alerts Interoperability Protocol	Identifies and interprets **Internet messages** that can be sent to protocol clients when a document, web page, or other type of resource is changed on a protocol server. This protocol also specifies the syntax and semantics of user-defined fields in message headers of those messages.	[MS-OSALER]

Protocol name	Description	Short name
Alerts Service Protocol	Extracts data from an alert (1) sent from a protocol server running Windows SharePoint Services to Office Outlook 2007, Outlook 2010, or Outlook 2013.	[MS-ALERTSS]
Exchange Server Protocols System Overview	An overview of the Exchange system. Provides a rich set of interfaces with which messaging clients can interoperate.	[MS-OXPROTO]
Lists Client Sync Protocol	Determines the correct processing and logic to correctly host local copies of SharePoint **list (1)** items in Office Outlook 2007, Outlook 2010, and Outlook 2013. Uses the Lists Web Service Protocol [MS-LISTSWS] for the actual transfers.	[MS-OUTSPS]
Lists Web Service Protocol	Transfers list (1) items from the protocol server to the client, and from the client to the protocol server.	[MS-LISTSWS]
StsSync Data Structure	URL format that instantiates a connection between Office Outlook 2007, Outlook 2010, or Outlook 2013 and a given SharePoint list (1).	[MS-STSSYN]

2.2.9 Microsoft Office Mobile

The following table describes the mobility protocols used by Word Mobile 2010 and PowerPoint Mobile 2010.

Protocol name	Description	Short name
Office Mobile PowerPoint Web Handler Protocol	This protocol enables a protocol client to send a request to retrieve presentation content from the protocol server. To facilitate this, the protocol allows the protocol client to request specific pieces of content from a **presentation** stored on the protocol server. In a **presentation slide** contained within a presentation, a protocol client can retrieve information describing the presentation slide contents as well as images of the presentation slide contents.	[MS-OMPWHP]
Office Mobile Word Web Handler Protocol	This protocol specifies the communication between the client and the **front-end Web server** to get the mobile rendition information for a Word document. Each method is an **HTTP GET** request, as described in Hypertext Transfer Protocol -- HTTP/1.1 [RFC2616], that accepts a set of parameters and returns an HTTP response depending upon the method called. The parameters to the method are sent as query parameters as part of the URL as described in Hypertext Transfer Protocol -- HTTP/1.1. All communication is transported over HTTP or Hypertext Transfer Protocol over Secure Sockets Layer (**HTTPS**).	[MS-OMWWH]
PowerPoint Web Viewer Rendered Static Content Structure	A presentation consists of slides, pictures, and other content. A protocol client can use the Office Mobile PowerPoint Web Handler Protocol [MS-OMPWHP] to request specific pieces of presentation content stored on a protocol server. This structure specifies a set of data types that enable the transfer of such presentation content. It also specifies data types that	[MS-PWVRSC]

Protocol name	Description	Short name
	encapsulate error information that enables the protocol server to signal the protocol client that it is unable to serve specific requests.	

2.2.10 Office Broadcast Presentation Service

The following table describes the protocols used by the Office Broadcast Presentation Service<4>.

Protocol name	Description	Short name
Office Broadcast Presentation Service	This protocol enables a protocol client to send requests to a protocol server allowing the client to begin or end a document **broadcast session**, and to store data about the state of a broadcast session on the protocol server.	[MS-OBPRS]
Office Broadcast Participant Service	This protocol enables a protocol client to send requests to a protocol server allowing the client to join an in-progress document broadcast session, and to retrieve data about the state of a broadcast session on the protocol server.	[MS-OBPAS]

2.2.11 Web Application Open Platform Interface

The following table describes the protocols used by the Web Application Open Platform Interface (WOPI)<5>.

Protocol name	Description	Short name
Web Application Open Platform Interface Protocol	This protocol defines a set of operations that enables a client to access and change files stored by a protocol server, allowing the client to render files and provide file editing functionality.	[MS-WOPI]

2.2.12 Apps for Office

The following table describes the structure used by **Office Add-ins**.

Protocol name	Description	Short name
Office Web Extensibility Manifest Format	This structure specifies the schema for Office Add-ins manifest files.	[MS-OWEMXML]

2.3 Environment

The following sections identify the context in which the system exists, including the systems that use the interfaces provided by this system of protocols, other systems that depend on this system, and, as appropriate, how components of the system communicate.

2.3.1 Dependencies on This System

2.3.1.1 Authentication

All client applications covered by this overview document, which are listed in section 1, depend on the authentication protocols covered to access files from protocol servers that require authenticated access.

2.3.1.2 File Access

All client applications covered in this overview document, which are listed in section 1, depend on the File Access protocols to access files from protocol servers that support those protocols.

2.3.1.3 Document Management

All client applications that support document management features covered in this document depend on the document management protocols to enable those features.

2.3.1.4 Data Access

All client applications that support data access features covered in this document depend on the data access protocols to enable those features.

2.3.1.5 Information Rights Management

All client applications covered by this overview document, which are listed in section 1, depend on the rights management protocols to enable rights management features.

2.3.1.6 Mobility

- Microsoft PowerPoint Mobile 2010
- Microsoft Word Mobile 2010
- Microsoft Excel Mobile 2010

2.3.2 Dependencies on Other Systems/Components

2.3.2.1 Authentication

Authentication (2) protocols depend on the following components:

- Internet protocols (HTTP, TCP/IP) and TCP/IP networks
- Domain Name System (DNS) servers
- Local area network (LAN) and Internet routers
- Computer operating systems supporting TCP/IP networking

Forms authentication over HTTP depends on the following:

- The Web server is configured such that the user's identity is established by using forms authentication. The user's identity is transferred between the protocol client and the server by using the HTTP State Management Mechanism [RFC2109].
- The protocol client is configured to store and transmit **cookies** as described in the HTTP State Management Mechanism.

2.3.2.2 Mobility

- Office SharePoint Server 2007 or SharePoint Server 2010
- Microsoft SQL Server 2005
- Microsoft SQL Server 2008
- Windows Server 2008 operating system or Windows Server 2008 R2 operating system
- HTTP 1.1
- Mobile Viewers for Microsoft Office

2.4 Assumptions and Preconditions

For information about assumptions and preconditions that apply to a specific protocol in this system, see the specification for that protocol.

2.4.1 All Client/Server Protocols

The user can connect to the protocol server.

- The user has permissions to access the protocol server.

2.4.2 Mobility

Administrative access to the protocol server.

- If the installation is on a farm, the SQL Server database can be installed on a separate computer.

2.5 Use Cases

Client and server protocol interactions differ by application and scenario. The following use cases are provided to facilitate an understanding of the Office Client Protocols system overall:

- Authenticate Against a Web Server That Is Gated by Forms Authentication
- Download a Document from a Web Server
- Open a Historical Version of a File from a Web Server
- Use Information Rights Management
- Open a Document by Using an ActiveX Control
- Synchronize a SharePoint List with Outlook
- Receive E-mail **Alerts** in Outlook from a SharePoint Server
- Publish an Access Database Application to a Web Server
- Publish an InfoPath Form to a Server
- View the First Slide of a Broadcast Presentation in a Web Browser
- Start a Broadcast Slide Show
- Synchronize IME with a Remote List

- Publish an Excel Workbook to a SharePoint Library

These use cases are not intended to provide a thorough and complete model of the system for any implementation.

2.5.1 Authenticate Against a Web Server That Is Gated by Forms Authentication

Use Case Diagram

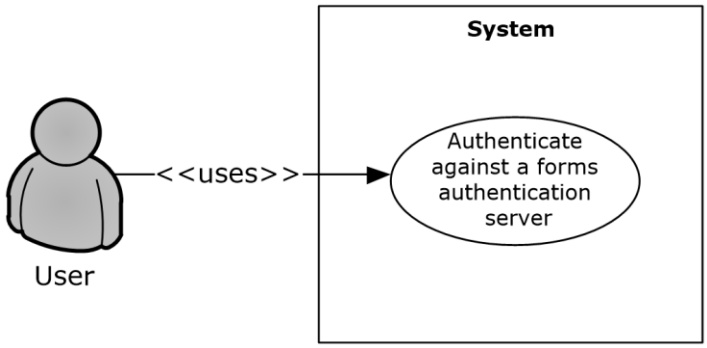

Figure 1: Process for authenticating against a forms authentication server

Preconditions

- The protocol client can connect to the server that hosts the document.
- The user has permissions to access the document on the server.
- The Web server and the protocol client both support the Office Forms Based Authentication Protocol [MS-OFBA].
- The Web server is configured such that the user's identity is established by using **forms authentication**. The user's identity is transferred between the protocol client and the server by using the HTTP State Management Mechanism [RFC2109].
- The protocol client is configured to store and transmit **cookies**, as described in [RFC2109].

Main Flow

1. The user opens the client application and brings up the file open dialog box.
2. The user navigates to the Web server in the file open dialog box.
3. The user is presented with a forms authentication logon prompt.
4. The user enters credentials and authenticates with the server.
5. The user selects a document from the Web server and chooses to open it in the client application.

Alternate Scenarios

- The user chooses to cancel out of the forms authentication logon prompt. This cancels the file open action.

Error Scenarios

- The user enters incorrect credentials in the forms authentication prompt three times. This cancels the file open action.

2.5.2 Download a Document from a Web Server

Use Case Diagram

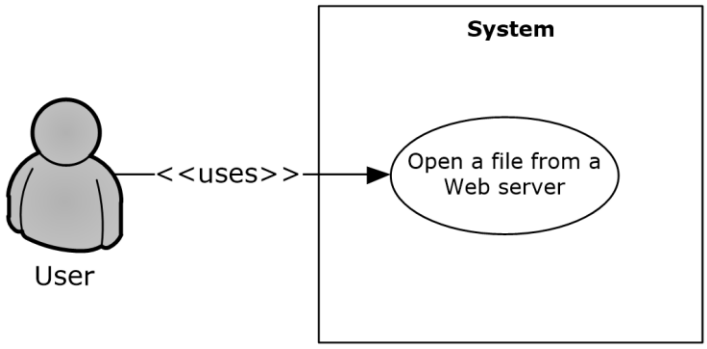

Figure 2: Process for downloading a document from a Web server

Preconditions

- The protocol client can connect to the server that hosts the document.
- The user has permissions to access the document on the Web server.

Main Flow

1. The user opens the client application and brings up the file open dialog box.
2. The user navigates to the Web server in the file open dialog box.
3. The user selects a document from the Web server and chooses to open it in the client application.
4. The client application issues an **HTTP OPTIONS** request to the Web server to determine the capabilities of the server.
5. The client application examines the response from the server and determines:
   - The Web authoring protocol to use (as described in **Alternate Scenarios** following).
   - The document management features to turn on.
6. The client application uses the File Synchronization via SOAP over HTTP Protocol [MS-FSSHTTP] to download the document.
7. The client application determines that the Web server supports the FrontPage Server Extensions: Website Management Protocol [MC-FPSEWM] and turns on the relevant document management features as described in section 2.1.3.

Alternate Scenarios

- The Web server does not support the File Synchronization via SOAP over HTTP Protocol, but does support the **WebDAV** protocol described in HTTP Extensions for Distributed Authoring WebDAV [RFC2518] and the FrontPage Server Extensions Remote Protocol [MS-FPSE]. In this case the document is downloaded by using the WebDAV protocol described in HTTP Extensions for Distributed Authoring WebDAV and the relevant document management features are turned on. The file is downloaded in read/write mode.
- The Web server does not support the File Synchronization via SOAP over HTTP Protocol or the FrontPage Server Extensions: Web Site Management Protocol, but does support the WebDAV protocol described in HTTP Extensions for Distributed Authoring WebDAV. In this case the

document is downloaded by using the WebDAV protocol; however, none of the document management features are turned on. The file is downloaded in read/write mode.

- The Web server does not support the File Synchronization via SOAP over HTTP Protocol, FrontPage Server Extensions: Web Site Management Protocol, or the WebDAV protocol described in HTTP Extensions for Distributed Authoring WebDAV. However, the Web server does support the Hypertext Transfer Protocol -- HTTP/1.1 [RFC2068]. In this case the file is downloaded in read-only mode and none of the document management features are turned on.

2.5.3 Open a Historical Version of a File from a Web Server

Use Case Diagram

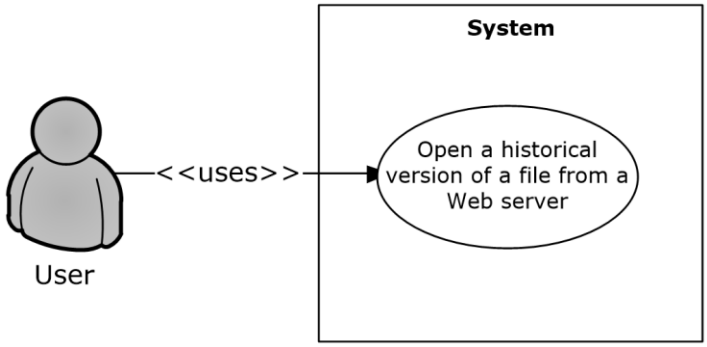

Figure 3: Process for opening a historical version of a file from a Web server

Preconditions

- The protocol client can connect to the server that hosts the document.
- The user has permissions to access the document on the Web server.
- The server supports the FrontPage Server Extensions: Web Site Management Protocol [MC-FPSEWM].
- The server responds with an OPTIONS header described in the following table.

Client version	MicrosoftSharePointTeamServices OPTIONS header	DocumentManagementServer OPTIONS header	Enabled
Office 2003	Exists	Any (exists or not); not checked	Yes
The 2007 Office system	Exists with no value or a value less than 12	Any (exists or not)	Yes
The 2007 Office system	Any (exists or not)	Exists with value **Version History**	Yes
Office 2010	Exists with no value or a value less than 12	Any (exists or not)	Yes
Office 2010	Any (exists or not)	Exists with value **Version History**	Yes
Office 2013	Exists with no value or a value less than 12	Any (exists or not)	Yes
Office 2013	Any (exists or not)	Exists with value **Version History**	Yes

Main Flow

1. The user chooses to view the historical versions that are available on the Web server for a document that is already open from an entry point in the client application.
2. The user selects a historical version of the document to open.
3. The historical version is downloaded from the Web server and opened in read-only mode in the client application.

2.5.4 Use Information Rights Management

Use Case Diagram

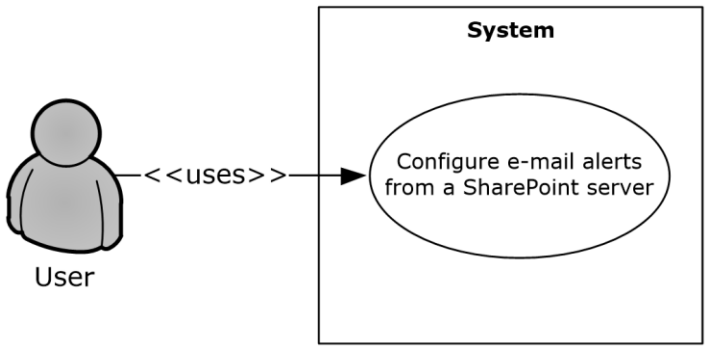

Figure 4: Process for protecting a document with rights management

Preconditions

- The protocol client can connect to a server that supports the rights management feature.

Main Flow

1. The user creates a document in the client application.
2. The user selects the option in the client application that enables the use of rights management to protect the document.
3. The client application protects the document, and this process is finished by using the Rights Management Services (RMS): Client-to-Server Protocol [MS-RMPR].

2.5.5 Open a Document by Using an ActiveX Control

Use Case Diagram

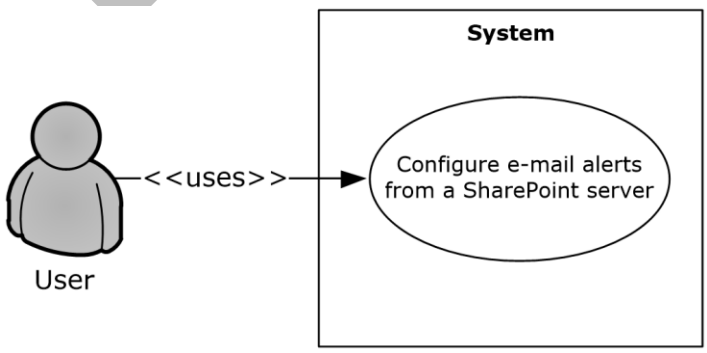

Figure 5: Process for opening a document by using an ActiveX control

Preconditions

- The protocol client can connect to the Web server that hosts the document.
- The protocol client has **ActiveX controls** enabled in the browser.
- The user has permissions to access the document on the Web server.

Main Flow

1. The user navigates to the Web server in the browser.
2. The user selects the document to open in the client application.
3. The user clicks the hyperlink that points to the document.
4. The user is presented with a dialog box, prompting a choice between opening the document in edit or read-only mode.
5. The user chooses to open the document in edit mode.
6. The client application opens the document by using a file access protocol. Refer to section 2.1.2.1.2 for more details.

Alternate Scenario

- The user chooses to open the document in read-only mode. The client application opens the document by using a file access protocol.

2.5.6 Synchronize a SharePoint List with Outlook

Use Case Diagram

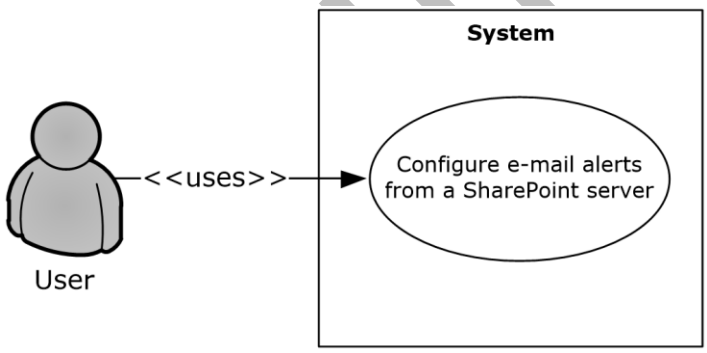

Figure 6: Process for synchronizing a SharePoint list (1) with Outlook

Preconditions

- The protocol client can connect to the server that hosts the **list (1)**.
- The user has permissions to access the list (1) on the Web server.

Main Flow

1. The user navigates to the Web server and chooses to connect the list (1) to the protocol client.

2. The browser generates the protocol URL, which it hands off to the operating system that is registered to the protocol client.

3. The user reads the client prompt and accepts syncing the list (1) to the protocol client. In this scenario the following protocols for syncing the list (1) data are used:

   - The StsSync Data Structure [MS-STSSYN] for generating the protocol URL.
   - The Lists Web Service Protocol [MS-LISTSWS] for obtaining the client copies of the server data.
   - The Lists Client Sync Protocol [MS-OUTSPS] for determining how the protocol client handles the list (1) schema.

2.5.7 Receive E-mail Alerts in Outlook from a SharePoint Server

Use Case Diagram

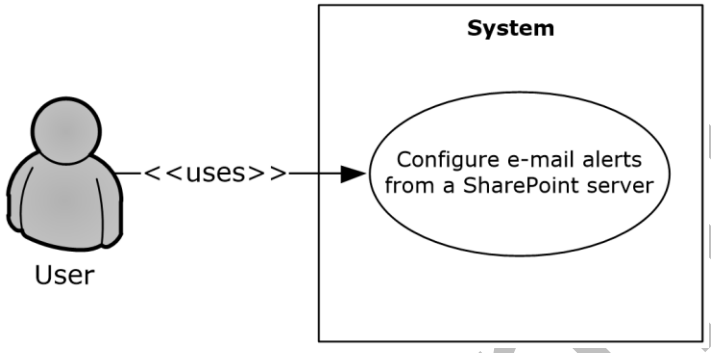

Figure 7: Process for receiving e-mail alerts from a SharePoint server

Preconditions

- The Web server is able to send Internet messages.
- The protocol client is able to receive Internet messages and interpret X-header fields.

Main Flow

1. The user navigates to the Web server, supplies an e-mail address, and configures **alerts** to be sent by e-mail.
2. The user receives an alert e-mail in the client application. In this scenario the following protocols are used:

   - The Internet Message Format [RFC2822] for generating a standard Internet message and storing additional alert information in the X-header fields on the Web server.
   - The Alerts Interoperability Protocol [MS-OSALER] for parsing this extra information, displaying icons for this message type, and cataloging the alert subscription in the client application.
   - The Alerts Service Protocol [MS-ALERTSS] for allowing the user to manage the alert subscriptions in the client application.

2.5.8 Publish an Access Database Application to a Web Server

Use Case Diagram

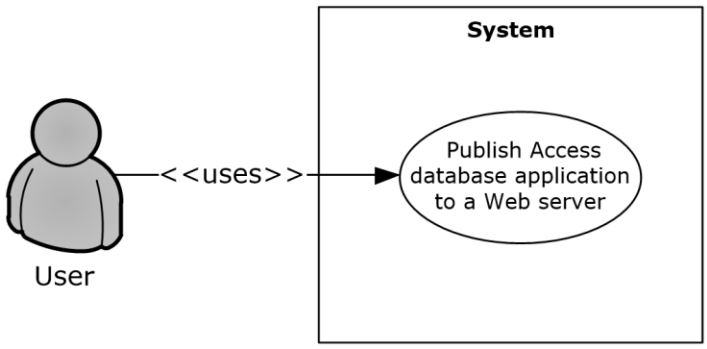

Figure 8: Process for publishing an Access database application to a Web server

Preconditions

- The protocol client can connect to the Web server that hosts the published Access 2013, Access 2010, or Office Access 2007 database application.
- The user has permissions to create the Access 2013, Access 2010, or Office Access 2007 database application on the Web server.

Main Flow

1. The user opens the Access 2013, Access 2010, or Office Access 2007 client application.
2. The user creates the Access 2013, Access 2010, or Office Access 2007 database in the client application.
3. The user types the **URL** to the Access 2013, Access 2010, or Office Access 2007 application in the Publish Form dialog box. In this scenario the following protocols are used:
   - The Sites Web Service Protocol [MS-SITESS] for Site Template instantiation.
   - The Lists Web Service Protocol [MS-LISTSWS] for creating lists (1) for each of the tables in the database application.
   - The Access Services Protocol [MS-ASWS] for moving data from the client cache to the lists (1) on the Web server.

Browser-enabled database applications do not support all available Access 2013, Access 2010, or Office Access 2007 features. Application users who need to use database objects that are not supported in the Web browser can publish the database application, but are required to use the Access 2013, Access 2010, or Office Access 2007 client application to use those objects.

Updating a previously published database application uses the same set of protocols and a very similar process.

2.5.9 Publish an InfoPath Form to a Server

For a user to be able to fill out an InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form, the **form template** is published to a network server where the user can access it. Servers running SharePoint Server frequently act as these servers.

When using **form files** to store the data for an electronic **form**, a form template can be filled out by users in one of two ways: in the Web browser or in a client application such as InfoPath 2013, InfoPath 2010, or Office InfoPath 2007. If the form template is published as a browser-enabled form template, users can fill it out in a Web browser by using InfoPath Forms Services or by using the

InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 client application. If the form template is not published as a browser-enabled form template, users can complete it only if they have the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 client application installed. Some InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 features are not available for browser-enabled form templates.

When using **lists (1)** to store the data for an electronic form, a form template can be filled out by users in one of two ways: in the Web browser using SharePoint Server 2013 or SharePoint Server 2010, or in a client such as Microsoft SharePoint Workspace 2010.

The following sections provide an overview of the primary ways that an InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form template can be published.

2.5.9.1 Publish a Non-Browser-Enabled Form Template to the Server

Use Case Diagram

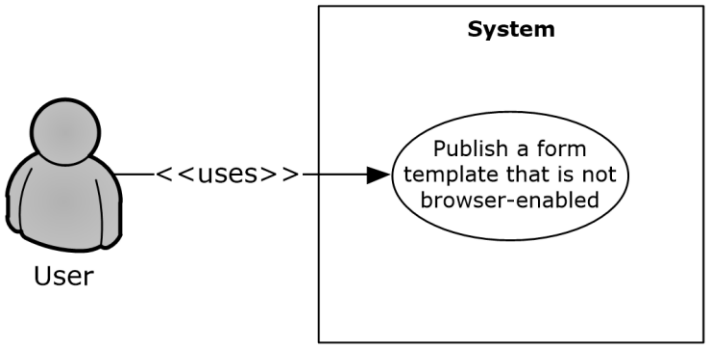

Figure 9: Process for publishing a non-browser-enabled form template

Preconditions

- The protocol client can connect to the protocol server that hosts the published InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 **form template**.
- The user has permissions to create the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form template on the protocol server.

Main Flow

1. The user opens the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 client application.
2. The user creates the form template in the client application.
3. The user types the **URL** to the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form template in the Publish Form dialog box. In this scenario the following protocols are used:
    - The FrontPage Server Extensions Remote Protocol [MS-FPSE] for getting **website** metadata, setting file properties, and performing file operations such as renaming.
    - The Windows SharePoint Services Collaborative Application Protocol [MS-WSSCAP] for performing the document library operations.
    - The Webs Web Service Protocol [MS-WEBSS] for getting **site** URLs and column fields.
    - The Lists Web Service Protocol [MS-LISTSWS] for managing content types.
4. The form template is uploaded to the server. In this scenario the Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions [MS-WDV] is used to upload the file.

Browser-enabled form templates do not support all available InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 features. Form designers who need to use features that are not supported in the Web browser can publish a form template without enabling it for Web browsers. As a result, users are required to use the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 client application to complete it.

Updating a previously published form template uses the same set of protocols and a very similar process.

2.5.9.2 Publish a Browser-Enabled Form Template to the Server

Use Case Diagram

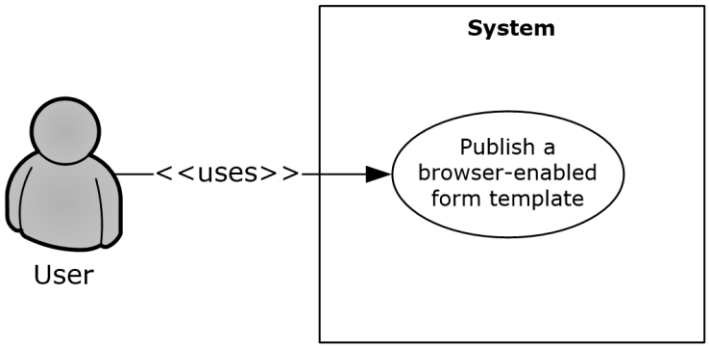

Figure 10: Process for publishing a browser-enabled form template

Preconditions

- The protocol client can connect to the protocol server that hosts the published InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 **form template**.
- The user has permissions to create the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form template on the protocol server.

Main Flow

1. The user opens the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 client application.
2. The user creates the form template in the client application.
3. The user types the **URL** to the InfoPath 2013, InfoPath 2010, or Office InfoPath 2007 form template in the Publish Form dialog box. In this scenario the following protocols are used:
    - The FrontPage Server Extensions Remote Protocol [MS-FPSE] for getting **website** metadata, setting file properties, and performing file operations such as renaming.
    - The Windows SharePoint Services Collaborative Application Protocol [MS-WSSCAP] for document library operations.
    - The Webs Web Service Protocol [MS-WEBSS] for getting **site** URLs and column fields.
    - The Lists Web Service Protocol [MS-LISTSWS] to manage content types.
    - The Forms Services Feature Detection Protocol [MS-FSFDP] to detect the presence of InfoPath Forms Services on the server.
4. The form template is uploaded to the server. In this scenario the Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions [MS-WDV] is used to upload the file.

5. The Forms Services Design and Activation Web Service Protocol [MS-FSDAP] is used to **browser-enable** the form template.

Updating a previously published form template uses the same set of protocols and a very similar process.

List content types are used to publish a form template to a **list (1)** on servers running SharePoint Server 2013 or SharePoint Server 2010. For more information, see [MS-LISTSWS].

2.5.10 View the First Slide of a Broadcast Presentation in a Web Browser

Use Case Diagram

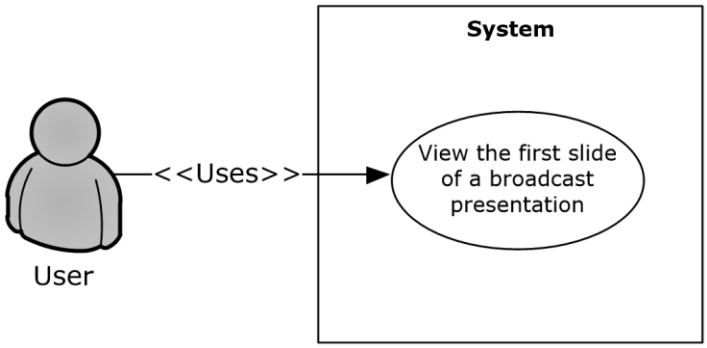

Figure 11: Process for viewing the first slide of a broadcast presentation in a Web browser

Preconditions

- The protocol client is a Web application.
- The protocol client can connect to the protocol server that hosts the document.
- The user has permissions to access the protocol server.
- The protocol client has obtained the identifier of a **broadcast presentation** that was uploaded by using the PowerPoint Web Broadcast Host Protocol [MS-PWBHPS].

Main Flow

1. A user navigates to a Web page to view a presentation.
2. The protocol client sends the URL of the presentation document by using the PowerPoint Web Editor Data Protocol [MS-PWEDPS].
3. The protocol server finds the document and issues a response that includes an identifier for the document.
4. The protocol client uses the PowerPoint Web Viewer Presentation Data Protocol [MS-PWVPDP] to retrieve the **URL** of an image of the first slide of the presentation.
5. The protocol client displays the image at this URL.

2.5.11 Start a Broadcast Slide Show

Use Case Diagram

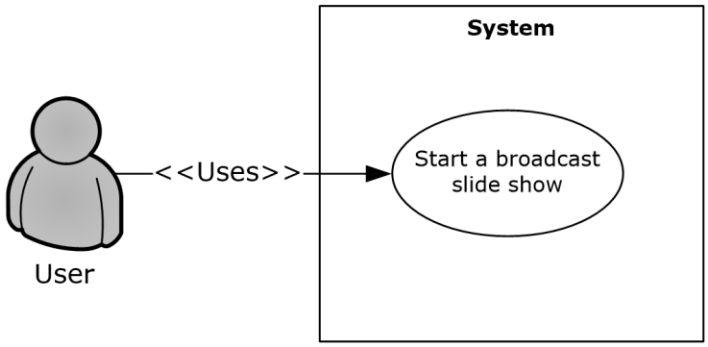

Figure 12: Process for starting a broadcast slide show

Preconditions

- The protocol client can connect to the **broadcast** server.
- The user has permissions to upload files to the server.

Main Flow

1. The user opens a **presentation** in the client application and chooses to start a broadcast **slide show**.
2. The user selects a broadcast service to connect to.
3. PowerPoint 2010 or PowerPoint 2013 connects to the service and starts the broadcast. It uses the following protocols:
    - The PowerPoint Web Broadcast Discovery Protocol [MS-PWBDPS] to retrieve the **URL** of a compatible PowerPoint Web Broadcast Host Protocol [MS-PWBHPS] service.
    - The PowerPoint Web Broadcast Host Protocol to retrieve the URL to upload the file, retrieve the URL of a PowerPoint Web Broadcast Protocol [MS-PWBPS] service, and retrieve an attendee URL.
    - The HTTP Extensions for Distributed Authoring WebDAV [RFC2518] to upload the file.
    - The PowerPoint Web Broadcast Protocol [MS-PWBPS] to create a session and set the initial state of the broadcast.
4. The user shares the broadcast attendee URL provided by the PowerPoint Web Broadcast Host Protocol.
5. The user begins presenting.

Alternate Scenarios

- The server does not support **WebDAV**. In this case, PowerPoint 2010 or PowerPoint 2013 uploads the file using an **HTTP POST** request with the file data included in the message body.

2.5.12 Synchronize IME with a Remote List

Use Case Diagram

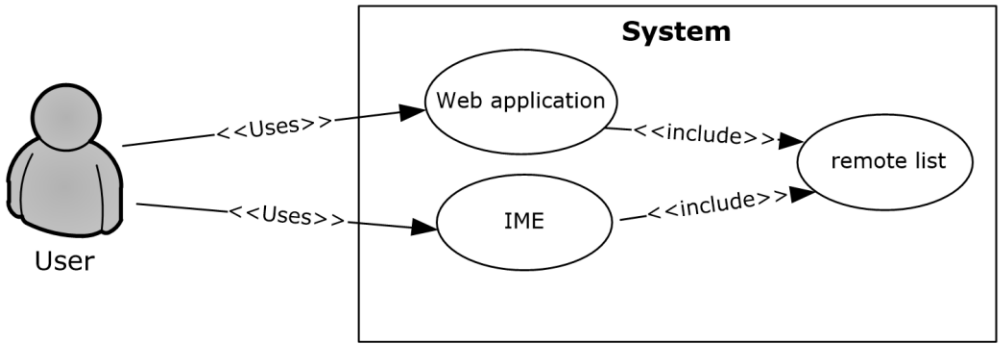

Figure 13: Process for synchronizing with a remote list

Preconditions

- The protocol client can connect to the remote **list (1)**.
- The user has permissions to download list (1) contents from the remote list (1) by using the Lists Web Service Protocol [MS-LISTSWS].
- The remote list (1) has the item scheme described in [MS-IMESYN].

Main Flow

1. The user clicks a link or button on a Web page.
2. The Web page issues the URI protocol string described in the IMESync Syntax Structure [MS-IMESYN].
3. An **IME** component is launched and passed the URI protocol string.
4. The IME component parses the URI protocol string and extracts the location of the remote list (1).
5. The IME component downloads the list (1) contents in the located remote list (1) by using the Lists Web Service Protocol [MS-LISTSWS].
6. The IME component parses the list (1) contents and builds an IME custom dictionary from it by using these list (1) contents as custom words.
7. IME starts using this newly created custom dictionary with other existing IME dictionaries.

2.5.13 Publish an Excel Workbook to a SharePoint Library

Use Case Diagram

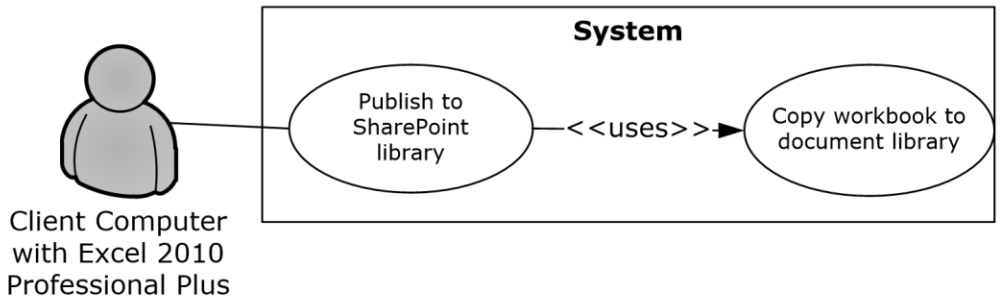

Figure 14: Process for publishing an Excel workbook to a SharePoint library

Preconditions:

- The user can connect to the **site** that includes the SharePoint library for the published Excel 2013, Excel 2010, or Office Excel 2007 workbook.
- The user has permissions to save files to a SharePoint **document library**.

Main flow:

1. The user opens the Excel 2013, Excel 2010, or Office Excel 2007 client application.
2. The user creates an Excel 2013, Excel 2010, or Office Excel 2007 workbook.
3. The user initiates publishing from Excel 2013, Excel 2010, or Office Excel 2007.
4. Excel 2013, Excel 2010, or Office Excel 2007 creates a file in a SharePoint document library. The web authoring protocol used to create the file is selected by the processes defined in section 2.1.2.1.2 above.
5. If requested, the workbook opens in a browser by using a **URL** that is specified according to the Excel Services Publishing Protocol, as described in [MS-ESURL].

2.6 Versioning, Capability Negotiation, and Extensibility

None.

2.7 Error Handling

None.

2.8 Coherency Requirements

This system has no special coherency requirements.

2.9 Security

Please refer directly to the protocols listed in section 2.2 for detailed information about protocol security.

2.10 Additional Considerations

None.

3 Examples

The examples in the following sections provide more information about the use and operation of the Office Client Protocols system. Protocol-level examples can be found in the individual protocol documents. The following system-level examples are provided in this document:

- Open a document from a Web server gated by forms authentication
- Check out and check in a document
- Synchronize a SharePoint list (1) with Outlook
- View the first slide of a presentation in a Web browser
- Start a broadcast slide show
- Synchronize IME with a remote list (1)

3.1 Example 1: Open a Document from a Web Server Gated by Forms Authentication

This example demonstrates how a document is opened from a Web server that is gated by **forms authentication**. This example builds on the use cases covered in "Authenticate Against a Web Server That Is Gated by Forms Authentication" (section 2.5.1) and "Download a Document from a Web Server" (section 2.5.2).

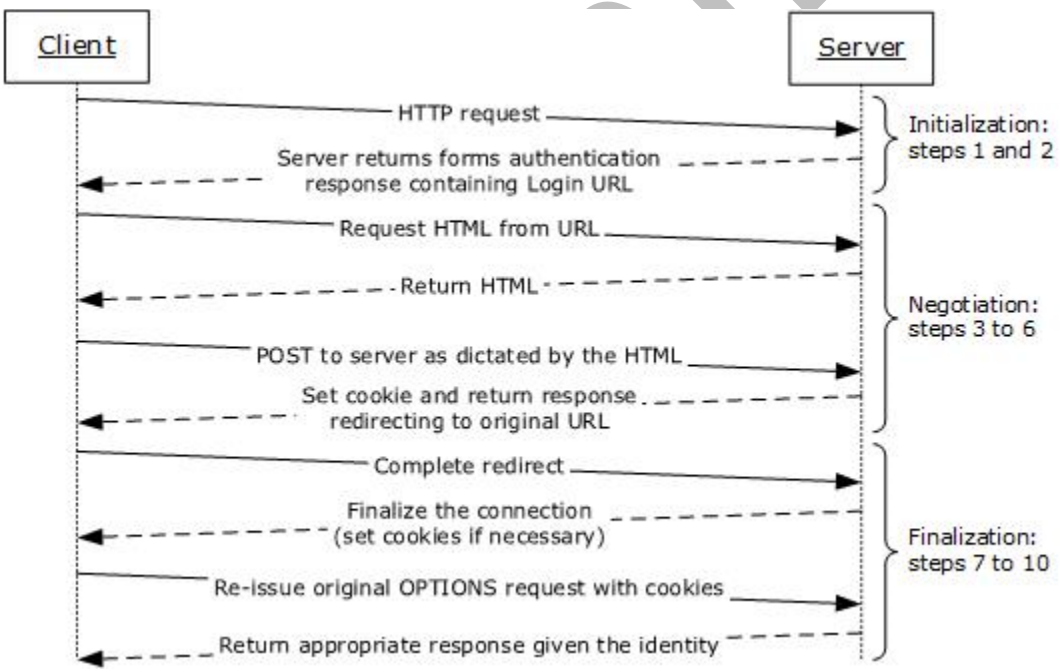

Figure 15: Sequence for opening a document from a Web server gated by forms authentication

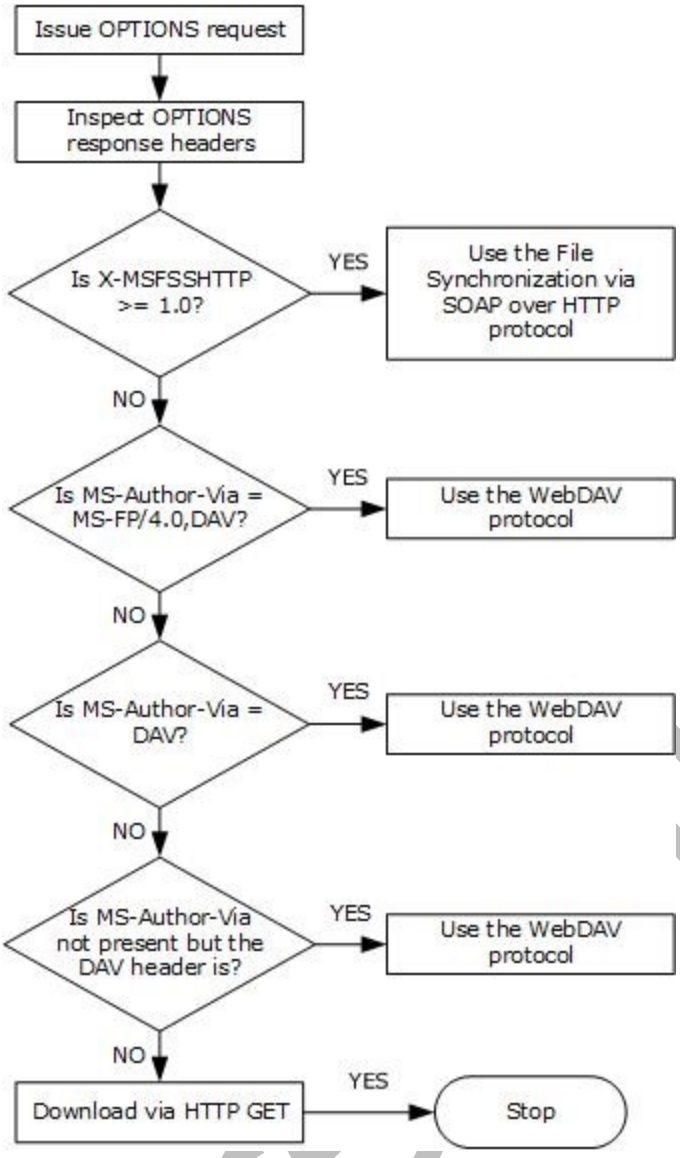

Figure 16: Process for selecting the file access protocol

Preconditions

- The protocol client can connect to the server that hosts the document.
- The user has permissions to access the document on the server.
- The Web server is configured such that the user's identity is established by using forms authentication. The user's identity is transferred between the protocol client and the server by using HTTP state management as described in HTTP State Management Mechanism [RFC2109].
- The protocol client is configured to store and transmit **cookies** as described in HTTP State Management Mechanism [RFC2109].

Main Flow

1. The user attempts to download an access-protected document from a Web server that is gated by forms authentication.

2. Prior to downloading the document, the client application attempts to determine the capabilities of the Web server by making an **HTTP OPTIONS** request.

3. The server issues a response indicating that it is capable of forms authentication in a format described by the Office Forms Based Authentication Protocol [MS-OFBA].

4. The protocol client follows the Office Forms Based Authentication Protocol and issues an HTTP request for the HTML **form** that the user can use to establish identity.

5. On receipt of the HTML from the Web server, the protocol client instantiates a dialog box and renders the form.

6. The user then interacts with the form to establish identity.

7. The Web server then responds with any finalization logic required.

8. Once this process is finished and the user's identity has been established, the protocol client issues an HTTP OPTIONS request to the server again to determine the capabilities of the server.

9. The Web server issues a response containing the headers **X-MSFSSHTTP=1.0** and **MS-Author-Via=MS-FP/4.0,DAV**. The **MS-Author-Via** header is described in Web Distributed Authoring and Versioning (WebDAV) Protocol: Server Extensions [MS-WDVSE].

10. The client application then downloads the file from the Web server by using the File Synchronization via SOAP over HTTP Protocol [MS-FSSHTTP].

11. The client application uses the FrontPage Server Extensions Remote Protocol [MS-FPSE] for document management functionality such as version history, **check in** and **check out** features, and the property panel, which are described in section 2.1.3.

3.2 Example 2: Check Out and Check In a Document

This example demonstrates the process for checking in and checking out a document from a Web server that supports the **check in** and **check out** features. This example builds on the use cases covered in "Download a Document from a Web Server" (section 2.5.2) and "Open a Historical Version of a File from a Web Server" (section 2.5.3).

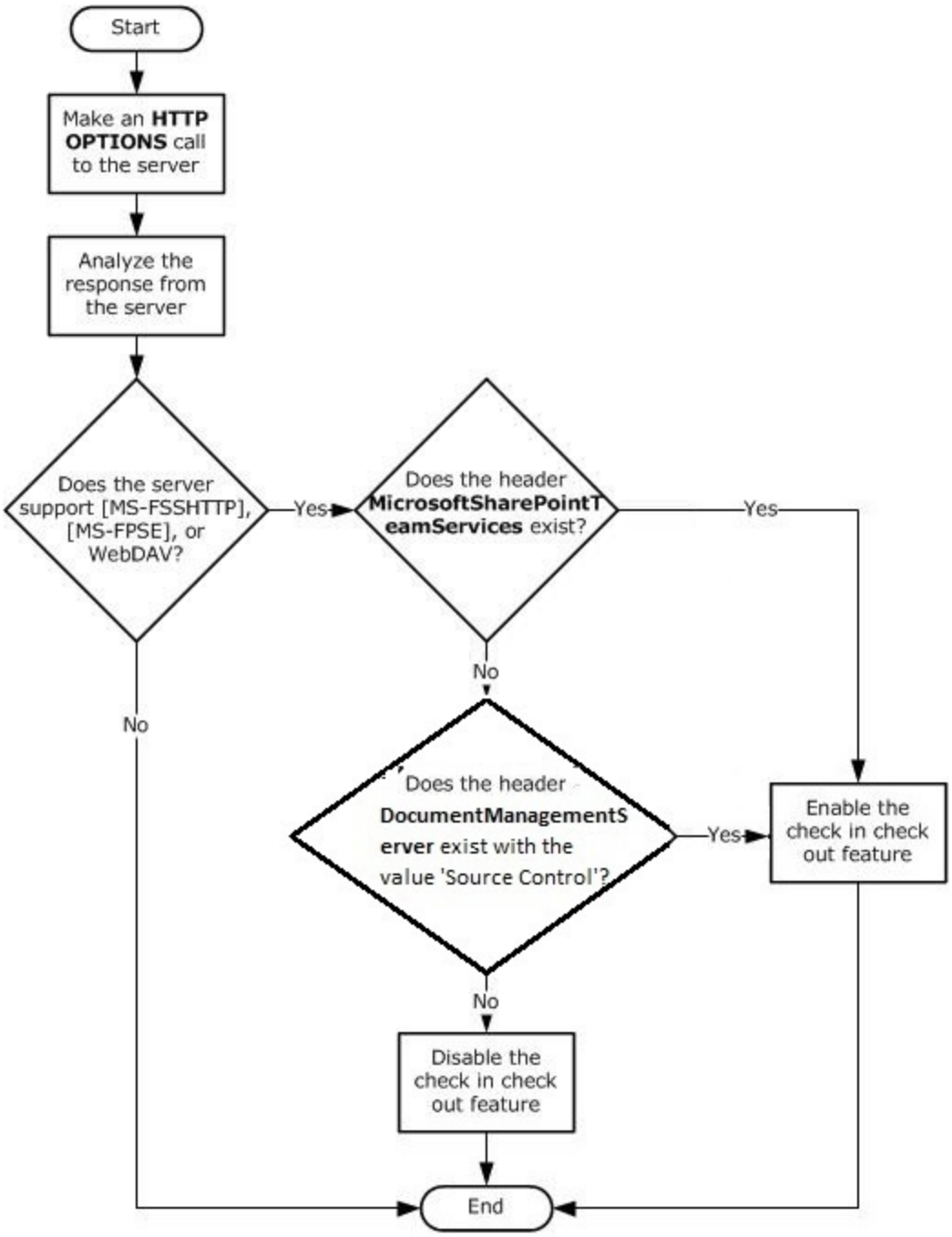

Figure 17: Process for enabling the check in and check out features in client applications

Preconditions

- The user has a document open from a Web server that supports the check in and check out features.
- The table in section 2.1.3.1.3 describes the conditions in which the client applications enable the check in and check out features.
- The user has permission to check out and edit the document on the Web server.

Main Flow

1. The user navigates to the option in the client application to check out the document.
2. The document is checked out from the Web server by using the Lists Web Service Protocol [MS-LISTSWS].
3. The user edits the document.
4. The user chooses the option in the client application for document to be checked in after upload.
5. The document is uploaded to the server by an HTTP PUT request. The Web authoring protocol, selected as described in example section 3.1, is used to lock and unlock the specific file on the server.
6. After the document has been uploaded to the server, the client application checks the document in by using the Lists Web Service Protocol [MS-LISTSWS].

3.3 Example 3: Synchronize a SharePoint List with Outlook

This example illustrates the process for synchronizing a SharePoint **list (1)** with a client application implementing the Lists Client Sync Protocol [MS-OUTSPS], such as Microsoft Outlook. It corresponds to the use case described in "Synchronize a SharePoint List with Outlook" (section 2.5.6).

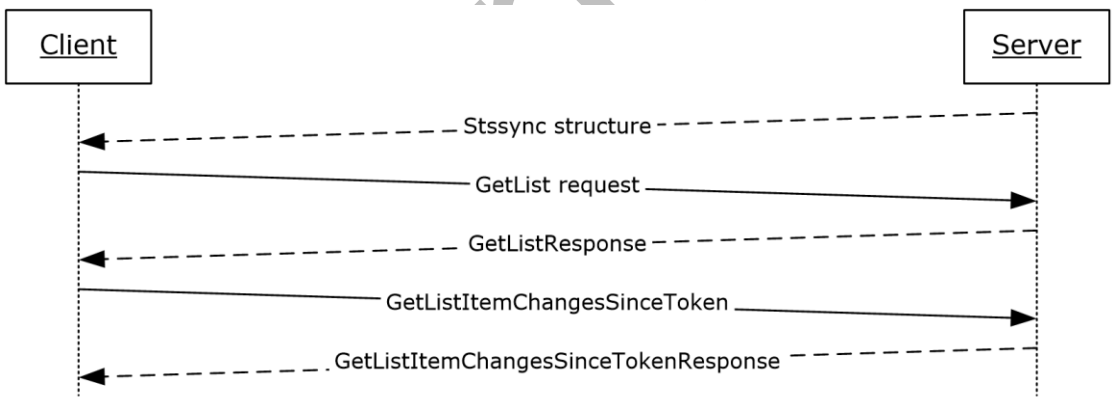

Figure 18: Sequence for synchronizing a SharePoint list (1) with Outlook

Preconditions

- The client application is an implementer of the Lists Client Sync Protocol [MS-OUTSPS].
- The client application can connect to the server that hosts the list (1).
- The user has permissions to access the list (1) on the server.

Main Flow

1. The user navigates to the Web server and connects the list (1) to the client application.

2. The browser generates the protocol URL, which it hands off to the operating system by using the StsSync Data Structure [MS-STSSYN].

3. The client application receives the protocol URL and parses it according to the StsSync Data Structure [MS-STSSYN].

4. Using the information obtained from the StsSync structure, the client application sends a **GetList** request to the server, as described in the Lists Web Service Protocol [MS-LISTSWS].

5. Using the Lists Web Service Protocol [MS-LISTSWS], the server responds with a **GetListResponse** call containing the **ListDefinitionSchema**.

6. The client application processes the **ListDefinitionSchema**, determining which fields are relevant.

7. Using the Lists Web Service Protocol [MS-LISTSWS], the client application sends a **GetListItemChangesSinceToken** request to the server that includes the relevant fields from the **ListDefinitionSchema** and the **GetListItemChangesSinceToken.viewFields.ViewFields** structure.

8. The server responds with the Lists Web Service Protocol [MS-LISTSWS] **GetListItemChangesSinceTokenResponse** call containing items in the list (1).

9. The client application processes the items in the **GetListItemChangesSinceTokenResponse**, as described in [MS-OUTSPS].

10. The client application processes the attributes of the **GetListItemChangesSinceTokenResponse** and determines whether more items are present. If so the client proceeds to step 7.

11. The client application adds the items to its local storage by sending an **HTTP GET** request.

3.4 Example 4: View the First Slide of a Presentation in a Web Browser

This example illustrates the process for viewing the first **slide** of a **presentation** in a Web Browser. It corresponds to the use case described in "View the First Slide of a Broadcast Presentation in a Web Browser" (section 2.5.10).

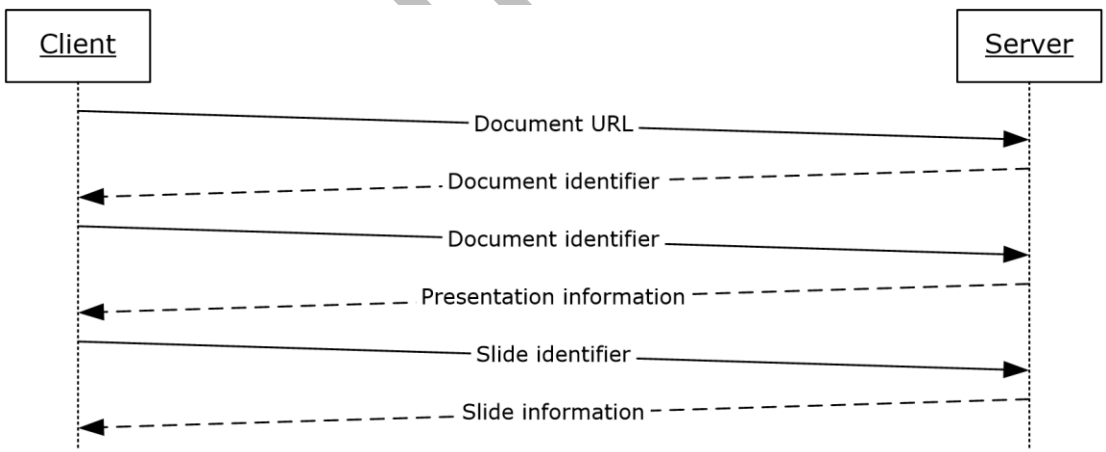

Figure 19: Sequence for viewing the first slide of a presentation in a Web browser

Preconditions

- The client application is a Web page.

- The client application can connect to the server that hosts the document.
- The user has permissions to access the document on the server.
- The client application has obtained the **URL** of the document stored on the server.

Main Flow

1. The user opens the client application in a Web browser.
2. The client application sends a request to the server with the URL of the document by using the PowerPoint Web Editor Data Protocol [MS-PWEDPS].
3. The server finds the document and issues a response that includes an identifier for this document.
4. The client application sends a request to the server for information about the presentation by using the PowerPoint Web Viewer Presentation Data Protocol [MS-PWVPDP].
5. The server loads the requested document and extracts information about the presentation.
6. The server issues a response with information about the presentation including a list of slides in the presentation.
7. The client application sends a request for information about the first slide in the list by using the PowerPoint Web Viewer Presentation Data Protocol [MS-PWVPDP].
8. The server issues a response with information about this slide, including the background appearance, a list of shapes, and a list of image resources.
9. The client application sends a request for each of the image resources.
10. The client application renders the slide by using the slide information and images returned by the server.

3.5 Example 5: Start a Broadcast Slide Show

This example illustrates the process for starting a **broadcast slide show** session. It corresponds to the use case described in "Start a Broadcast Slide Show" (section 2.5.11).

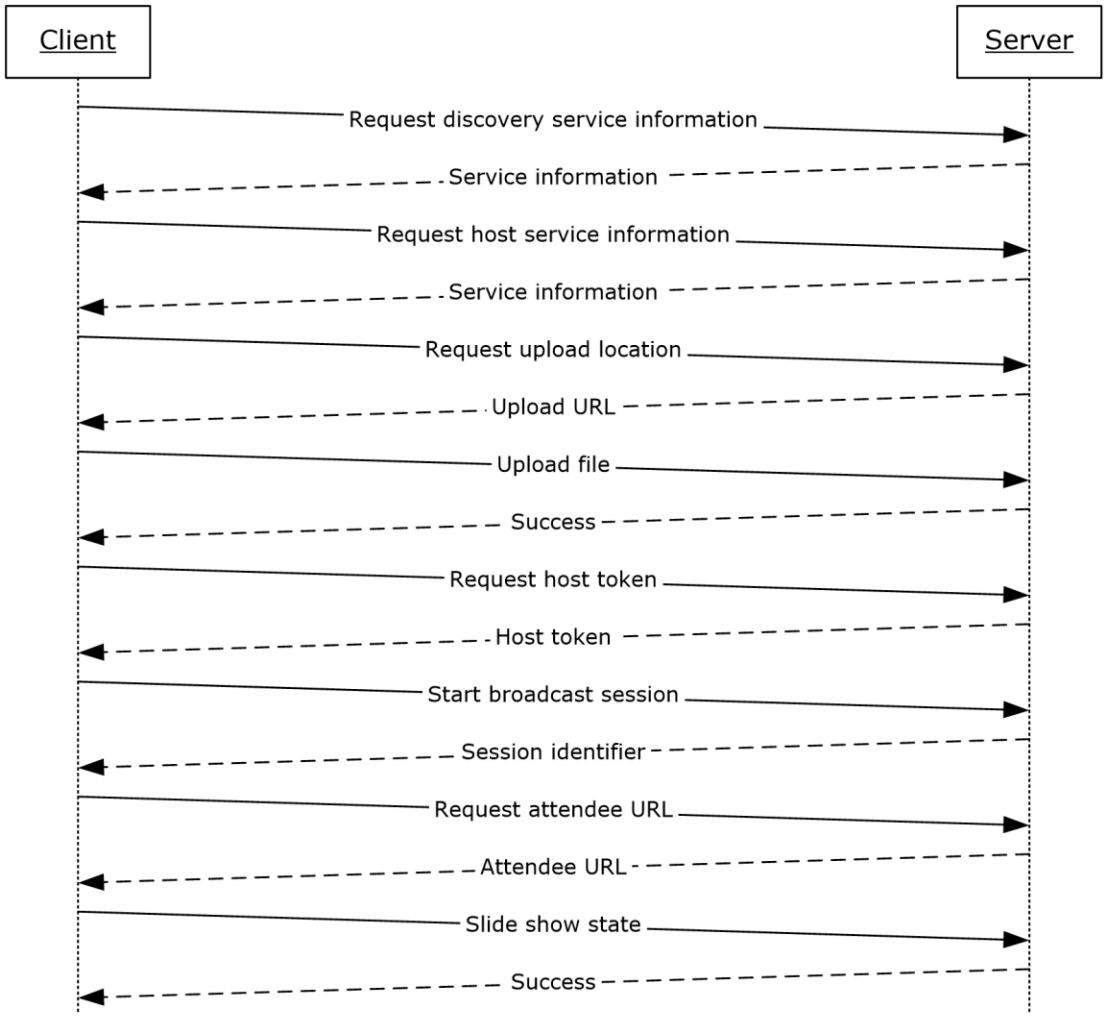

Figure 20: Sequence for starting a broadcast slide show

Preconditions

- The protocol client can connect to the broadcast server.
- The user has permissions to start a broadcast on the server.

Main Flow

1. The user chooses to start a broadcast in the client application.
2. The protocol client sends a request to the server for host information by using the PowerPoint Web Broadcast Discovery Protocol [MS-PWBDPS].
3. The server issues a response with the version number and **URL** of a service that implements the PowerPoint Web Broadcast Host Protocol [MS-PWBHPS].
4. The protocol client sends a request for server information to this URL by using the PowerPoint Web Broadcast Host Protocol [MS-PWBHPS].
5. The server issues a response with information about the service, including the capabilities it supports, the URL of a service that implements the PowerPoint Web Broadcast Protocol [MS-

PWBPS], and the method that the protocol client can use to upload the file: HTTP Extensions for Distributed Authoring WebDAV [RFC2518] or an **HTTP POST** request.

6. The protocol client sends a request for the location to upload the **presentation**.
7. The server replies with the URL of an upload location.
8. The protocol client uploads the presentation to the specified location by using the protocol specified in step 5.
9. The protocol client sends a request for a broadcast host token by using the PowerPoint Web Broadcast Host Protocol [MS-PWBHPS].
10. The server generates a host token that represents the uploaded file and issues a response with this identifier.
11. The protocol client sends a request to start a broadcast session by using the PowerPoint Web Broadcast Protocol [MS-PWBPS].
12. The server generates a unique identifier for the session and stores information about the session.
13. The server issues a response with the session identifier.
14. The protocol client sends this session identifier in a request to generate an attendee URL by using the PowerPoint Web Broadcast Host Protocol [MS-PWBHPS].
15. The server generates a unique attendee URL for the session and issues a response with this URL.
16. The protocol client displays this URL to the user so they can share it with remote attendees.
17. The user starts presenting in the client application.
18. The protocol client sends information about the state of the slide show to the server by using the PowerPoint Web Broadcast Protocol [MS-PWBPS].

3.6 Example 6: Synchronize IME with a Remote List

This example illustrates the process for synchronizing a custom dictionary of **IME** with a remote **list (1)**. It corresponds to the use case described in "Synchronize IME with a Remote List" (section 2.5.12).

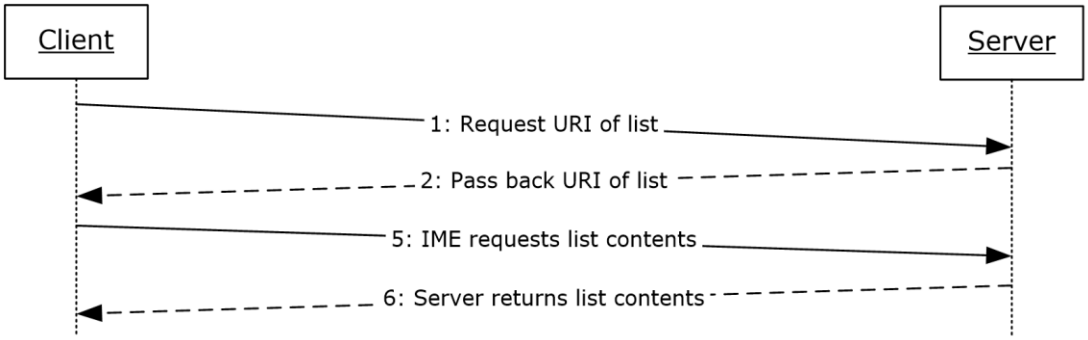

Figure 21: Sequence for synchronizing IME with a remote list

Preconditions

- The protocol client can connect to the remote list (1).

- The user has permissions to download list (1) contents from the remote list (1) by using the Lists Web Service Protocol [MS-LISTSWS].
- The remote list (1) has the item scheme described in the IMESync Syntax Structure [MS-IMESYN].

Main Flow

1. The user clicks a link or button on a Web page to issue the URI protocol string described in the IMESync Syntax Structure [MS-IMESYN].
2. The protocol server passes the URI protocol string back to the protocol client.
3. On the protocol client side an IME component is launched and passed the URI protocol string.
4. The IME component parses the URI protocol string and extracts the location of the remote list (1).
5. The protocol client sends a request to get the list (1) contents to the protocol server located by the URI protocol string by using the Lists Web Service Protocol [MS-LISTSWS].
6. The protocol server returns the list (1) contents by using the Lists Web Service Protocol [MS-LISTSWS].
7. The protocol client receives the list (1) contents and converts it to an IME custom dictionary.
8. IME starts using this newly created custom dictionary with another existing IME dictionary.

4 Microsoft Implementations

There are no variations in the behavior of the Office Client Protocols system in different versions of Microsoft Office system beyond those described in the specifications of the protocols supported by the system, as listed in section 2.2.

The information in this specification is applicable to the following versions of Microsoft Office:

- Microsoft Office 2003
- the 2007 Microsoft Office system
- Microsoft Office 2010 suites
- Microsoft Office 2013
- Microsoft Office 2019 Preview

Exceptions, if any, are noted in the following section.

4.1 Product Behavior

<1> Section 2.1.20: This service is available only in Office 2013.

<2> Section 2.1.21: This interface is available only in Office 2013.

<3> Section 2.2.4: This service is available only in Office 2013.

<4> Section 2.2.10: This service is available only in Office 2013.

<5> Section 2.2.11: This interface is available only in Office 2013.

5 Change Tracking

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

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

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

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

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

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

Section	Description	Revision class
4 Microsoft Implementations	Updated list of supported products.	Major

6 Index

G

H

I

M

O

P

R

S

T

U

V

W