[MS-FPSE]:
FrontPage Server Extensions Remote Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
12/18/2006	0.1		MCPP Milestone 2 Initial Availability
03/02/2007	1.0		MCPP Milestone 2
04/03/2007	1.1		Monthly release
05/11/2007	1.2		Monthly release
06/01/2007	1.2.1	Editorial	Revised and edited the technical content.
07/03/2007	2.0	Major	Section 2.3 Added information about how and when an extra 401 response can be sent from the server.
07/20/2007	2.0.1	Editorial	Revised and edited the technical content.
08/10/2007	2.0.2	Editorial	Revised and edited the technical content.
09/28/2007	3.0	Major	Updated and revised the technical content.
10/23/2007	3.0.1	Editorial	Revised and edited the technical content.
11/30/2007	4.0	Major	Added and revised sections.
01/25/2008	5.0	Major	Updated and revised the technical content.
03/14/2008	6.0	Major	Updated and revised the technical content.
05/16/2008	6.0.1	Editorial	Revised and edited the technical content.
06/20/2008	7.0	Major	Updated and revised the technical content.
07/25/2008	8.0	Major	Updated and revised the technical content.
08/29/2008	8.0.1	Editorial	Revised and edited the technical content.
10/24/2008	8.0.2	Editorial	Revised and edited the technical content.
12/05/2008	8.0.3	Editorial	Revised and edited the technical content.
01/16/2009	8.1	Minor	Updated the technical content.
02/27/2009	8.1.1	Editorial	Revised and edited the technical content.
04/10/2009	9.0	Major	Updated and revised the technical content.
05/22/2009	10.0	Major	Updated and revised the technical content.
07/02/2009	11.0	Major	Updated and revised the technical content.
08/14/2009	11.1	Minor	Updated the technical content.
09/25/2009	11.2	Minor	Updated the technical content.

Date	Revision History	Revision Class	Comments
11/06/2009	11.2.1	Editorial	Revised and edited the technical content.
12/18/2009	11.3	Minor	Updated the technical content.
01/29/2010	11.4	Minor	Updated the technical content.
03/12/2010	11.4.1	Editorial	Revised and edited the technical content.
04/23/2010	11.4.2	Editorial	Revised and edited the technical content.
06/04/2010	11.5	Minor	Updated the technical content.
07/16/2010	12.0	Major	Significantly changed the technical content.
08/27/2010	12.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	13.0	Major	Significantly changed the technical content.
11/19/2010	14.0	Major	Significantly changed the technical content.
01/07/2011	14.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	14.0	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

1 Introduction

The FrontPage Server Extensions Remote Protocol specifies a set of server extensions that can be used to augment a basic HTTP server. These extensions provide file server functionality similar to WebDAV, allowing a Web site to be presented as a file share. The use of WebDAV is recommended over the FrontPage Server Extensions Remote Protocol. For more information about WebDAV, see [MS-WDV].

The FrontPage Server Extensions Remote Protocol uses HTTP version 1.1 ([RFC2616]) as a transport. Requests are specialized form posts, and responses are in HTML ([RFC2854]). Despite the use of HTTP, the protocol is intended to be used by a client program, not by the user directly through a Web browser.

The FrontPage Server Extensions Remote Protocol is a subset of a larger protocol known as FrontPage Server Remote Protocol Extensions. The FrontPage Server Extensions Remote Protocol is a protocol that can be used when communicating between client operating systems and HTTP servers. The larger protocol is used to perform a wider array of Web site administration tasks. Because implementations of the FrontPage Server Extensions Remote Protocol on some operating systems also implement FrontPage Server Remote Protocol Extensions, be aware that some servers might not ignore arguments and methods noted in this document as messages that a client cannot send.

The Windows SharePoint Services dialogview is an application of the FrontPage Server Extensions Remote Protocol that is addressed in this document because it has certain behaviors apart from the normal FrontPage Server Extensions Remote Protocol communications. The purpose of the dialogview is to allow a client to display a server-rendered HTML-based rendering of the files located on a particular Web site.

1.1 Glossary

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

domain name (2)
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)
Unicode
Uniform Resource Locator (URL)

The following terms are specific to this document:

content database: A database which holds the backing store for the Windows SharePoint Services server, including its **site collections**, and the contents of the site collection.

derived documents folder: Server implementations might need to generate temporary intermediate files or files not directly uploaded by the client. These files are commonly stored in a folder whose **service-relative** URL is _derived. On Microsoft Windows implementations, it includes *.htx files created by the FrontPage Search component and composite text on .gif images such as those used for **theme elements**. The contents of the folder are considered a server implementation detail, but the list documents method, as specified in section 3.1.5.3.8, allows the client to ask the server to filter out the contents of this folder.

dictionary: A collection of pairs of items. Each pair consists of a key, which is a string and a value that can be of any type. Items in the **dictionary** are retrieved by providing a key for which the **dictionary** returns the associated value.

document checkout: A method used to lock a document to prevent users from concurrently editing the document. For details on **document checkout**, see section 3.

document library: A list created as a container for documents. A document library stores documents and folders. A document library can support publishing status values for documents (for example, draft, checked-in, checked-out, and published).

executable folder: Web servers commonly support the notion of a server-side executable in which pages are rendered by running a routine rather than by returning static contents from a file. These servers generally allow the administrator to enable and disable this feature on a folder-by-folder basis. A folder is called an **executable folder** if this feature is enabled for files within it.

folder: A container within a list that acts like a file system directory. A folder can contain other folders, documents, or list items.

form: A page that allows the creation, viewing or editing of list items in a list.

hidden documents: Files and folders whose URLs contain a path component that begins with an underscore (_).

link fixup: Some document formats support the notion of a link where one document references another document. The server can discover these links to referenced documents and rewrite the links as documents are moved or copied, so the links do not go to stale references. As the server rewrites these links, it is said to be doing **link fixup**.

long-term checkout: A **document checkout** that rejects edits against the file by other clients but, unlike a **short-term checkout**, does not expire unless the client application sends the uncheckout document request, as specified in section 3.1.5.3.15. FrontPage Server Extensions Remote Protocol–compliant clients never use **long-term checkout**, and servers can choose to ignore this value.

metadata: Data that describes objects such as **site collections**, **sites**, **folders**, documents, and other objects used by the client and server.

metadictionary: A **dictionary** for **metadata** with strongly typed values for storing additional properties for a file, folder, or **site**. A metadict contains a collection of properties.

metakey: The string used to look up a value in a **metadict**.

nesting level: A count used during the formatting of messages. It is the number of times an open-bracket (OBRACKET) is sent minus the number of times a close-bracket (CBRACKET) is sent. For details, see section 2.2.1.1.3.

page: A document consisting of HTML that could contain dynamic content such as Web parts that are interpreted before display to a client application.

server-relative: A **server-relative** URL defines the location of an item in relation to the root of the server. A server's URL can be determined by using the URL-to-Web-URL method, as specified in section 3.1.5.3.16.

service-relative: A **service-relative** URL defines the location of an item in relation to the root of the **site**. For example, if a page is located in the root folder of a **site**, the **service-relative** URL consists simply of the page name. The FrontPage Server Extensions Remote Protocol often returns **service-relative** URLs that the client interprets. A service's URL can be determined by using the URL-to-Web-URL method, as specified in 3.1.5.3.16.

short-term checkout: A **document checkout** that automatically expires after a set period of time. While a client application has a **short-term checkout** of a file, edits against the file by other clients are rejected. If the client application does not renew the **short-term checkout**, edits by other clients are allowed after the **short-term checkout** expires.

short file name: A file name that consists of up to eight characters, a period, and up to three characters.

site: An autonomous Web service with a contiguous URL namespace. Each site is a member of a**site collection**, and has its own entry points, **metadata**, and independent administration, authoring, and browsing permissions. A site is a container that can contain documents, **document libraries**, lists, and child sites known as **subsites**. The structure and content of sites, when created, are based on implementation-specific site templates such as but not limited to Team Sites, Document Workspaces, and Meeting Workspaces. Also referred to as a **Web site**.

site collection: A collection of one or more hierarchically nested **sites** within a single **content database** that are managed as a single unit. A site collection can be identified by a unique GUID value or by the URL of the root site of the Site collection.

subsite: A **site** whose URL is composed in part by the URL of a parent site within the same **site collection**. A subsite parent can be either the root site of the **site collection** or another subsite. Each subsite can have independent administration, authoring, and browsing permissions from the root site and other subsites.

task-list files: The FrontPage client has a task list feature that stores information in two files on the server. The **task-list files** are always stored in _vti_pvt/_x_todo.htm and _vti_pvt/_x_todoh.htm.

theme elements: A set of coordinated graphic elements applied to a document or Web page or across all pages in a **Web site**. Themes can consist of designs and color schemes for fonts, link bars, and other page elements.

thicket: A group of supporting files and folders on the server that together store the contents of one logical document, for example, a page or Microsoft Office Word document that includes pictures.

Web site: See **site**.

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

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

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

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989, http://www.ietf.org/rfc/rfc1123.txt

[RFC1341] Borenstein, N., and Freed, N., "MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies", RFC 1341, June 1992, http://www.ietf.org/rfc/rfc1341.txt

[RFC1866] Berners-Lee, T., and Connolly, D., "Hypertext Markup Language - 2.0", RFC 1866, November 1995, http://www.ietf.org/rfc/rfc1866.txt

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.ietf.org/rfc/rfc2119.txt

[RFC2279] Yergeau, F., "UTF-8, A Transformation Format of ISO10646", RFC 2279, January 1998, http://www.ietf.org/rfc/rfc2279.txt

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

[RFC2822] Resnick, P., Ed., "Internet Message Format", STD 11, RFC 2822, April 2001, http://www.ietf.org/rfc/rfc2822.txt

[RFC2854] Connolly, D., and Masinter, L., "The 'text/html' Media Type", RFC 2854, June 2000, http://www.ietf.org/rfc/rfc2854.txt

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, http://www.ietf.org/rfc/rfc3986.txt

[RFC4234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005, http://www.ietf.org/rfc/rfc4234.txt

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-WDV] Microsoft Corporation, "Web Distributed Authoring and Versioning (WebDAV) Protocol: Client Extensions", August 2007.

1.3 Overview

The FrontPage Server Extensions Remote Protocol is used by client applications to display the contents of a **site** as a file system. The FrontPage Server Extensions Remote Protocol provides file uploading and downloading, directory creation and listing, basic file locking, and file movement on a Web server by using a set of methods.

Each message from the client is in the **form** of an **HTTP** POST or GET ([RFC2616] sections 9.5 and 9.3) that includes a set of parameters, and each reply from the server returns a set of values as an HTML response ([RFC2854]). The method parameter defines what operation the server will perform in addition to the meanings of the other parameters and return values.

The client sends method call requests to the server, and the server sends return values to the client via HTML. The server never initiates any communication with the client. All communication is transported over HTTP or secure HTTP (**HTTPS**), as specified in [RFC2616] section 9.1. Method calls are sent as HTTP POSTs, [RFC2616] section 4.2, with the method name and arguments as the message entity, and server responses are sent as a list in the message body, [RFC2616] section 4.2, of an HTTP response. All posts are made to one of several well-defined **URLs** on the server, which can be discovered by clients.

A protocol client initiates its communication with the server by requesting well-defined URLs for further communication, as specified in section 3.1.3.2.1, and determining the version of the server as specified in section 1.7.1. This protocol specifies no additional requirements on the HTTP protocol layer as specified in [RFC2616] for maintaining state across request/responses. See [RFC2616] section 4.4 for more information on the determination of message length, and [RFC2616] section 8 for information on layering HTTP on top of the TCP protocol.

The following sequence diagram depicts a generic FrontPage Server Extensions conversation. A brief explanation of each message follows, and details are defined in sections 2 and 3.

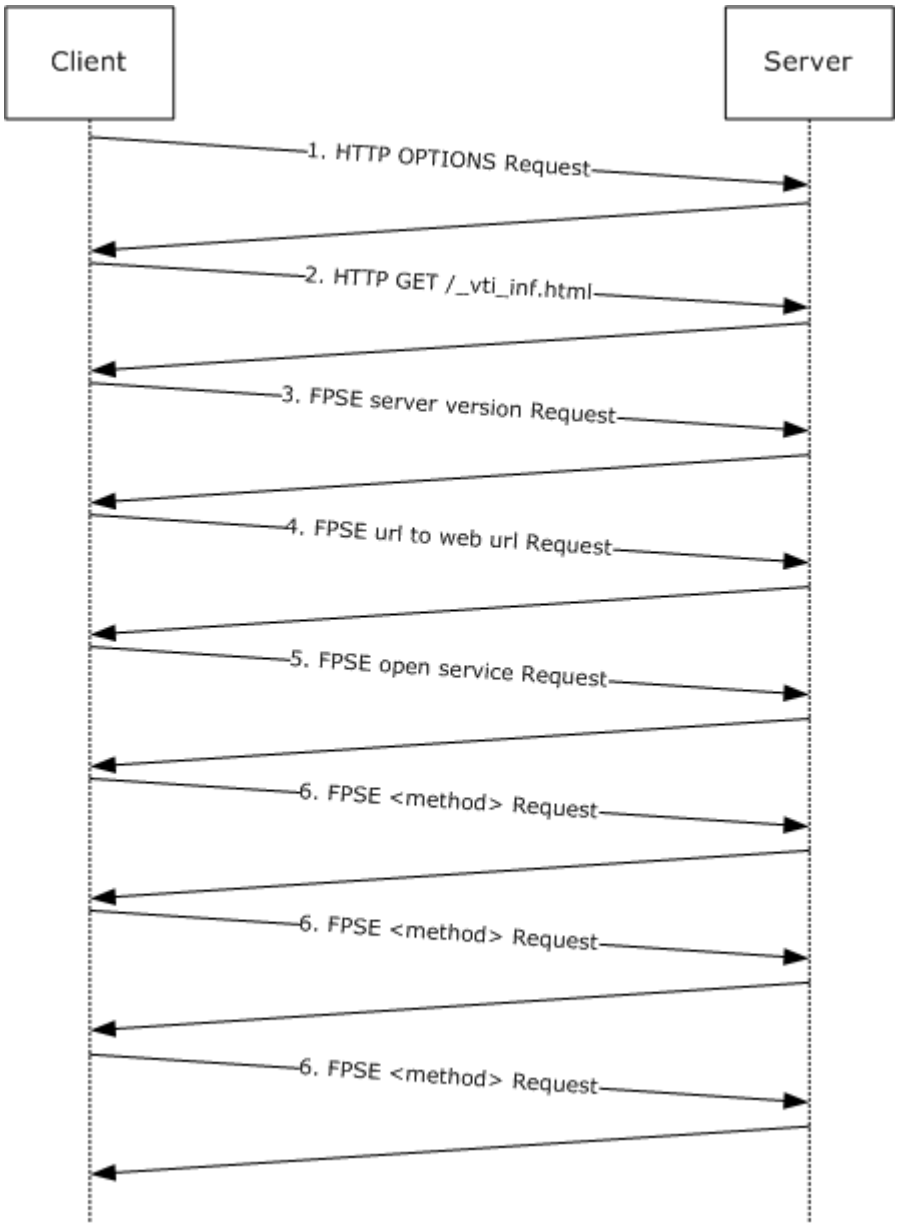

Figure 1: A generic Microsoft FrontPage Server Extension message sequence

1. The HTTP OPTIONS request is sent to determine if the server supports the FrontPage Server Extensions Remote Protocol. If the response contains the MS-Author-Via header, as specified in section 3.1.3.1, the server supports the protocol. This value is often cached by clients.

2. The HTTP GET on _vti_inf.html will return information specifying the well-defined URLs to which the client can POST further method calls.

3. At this point, the client is prepared to start making method calls against the server. The first call is a server version request (see section 3.1.5.3.14) whereby the client negotiates a protocol version with the server.

4. The client can then call the url to web url (see section 3.1.5.3.16) if the site is a **subsite**, that is, one not located at the root of the server's namespace.

5. Next, the client can make an open service request (see section 3.1.5.3.10) on the site that it wants to open. This request is optional, but it will return information about the site's capabilities, such as support for version control.

6. The client can make any method calls against the server. The nature of any further client/server communication is determined by the specific needs of the client at the time.

To give an example of how the protocol is used in Windows, a user might type an HTTP URL into the address bar of a file browser. The file browser contacts the server to determine if it supports the FrontPage Server Extensions Remote Protocol. If so, the file browser uses the FrontPage Server Extensions Remote Protocol to list the contents of the directory that the user entered and to display the contents just as it would display files on the local hard disk.

The Windows SharePoint Services dialogview aspect of the FrontPage Server Extensions Remote Protocol is designed to allow a client to ask the server to provide an HTML rendering of its file structure. The client can then display this HTML rendering and navigate through it.

When using Windows SharePoint Services dialogview (see section 3.1.5.3.17.1), the sequence of messages is similar to that shown in the following figure. For a brief explanation of each message, see sections 2 and 3.

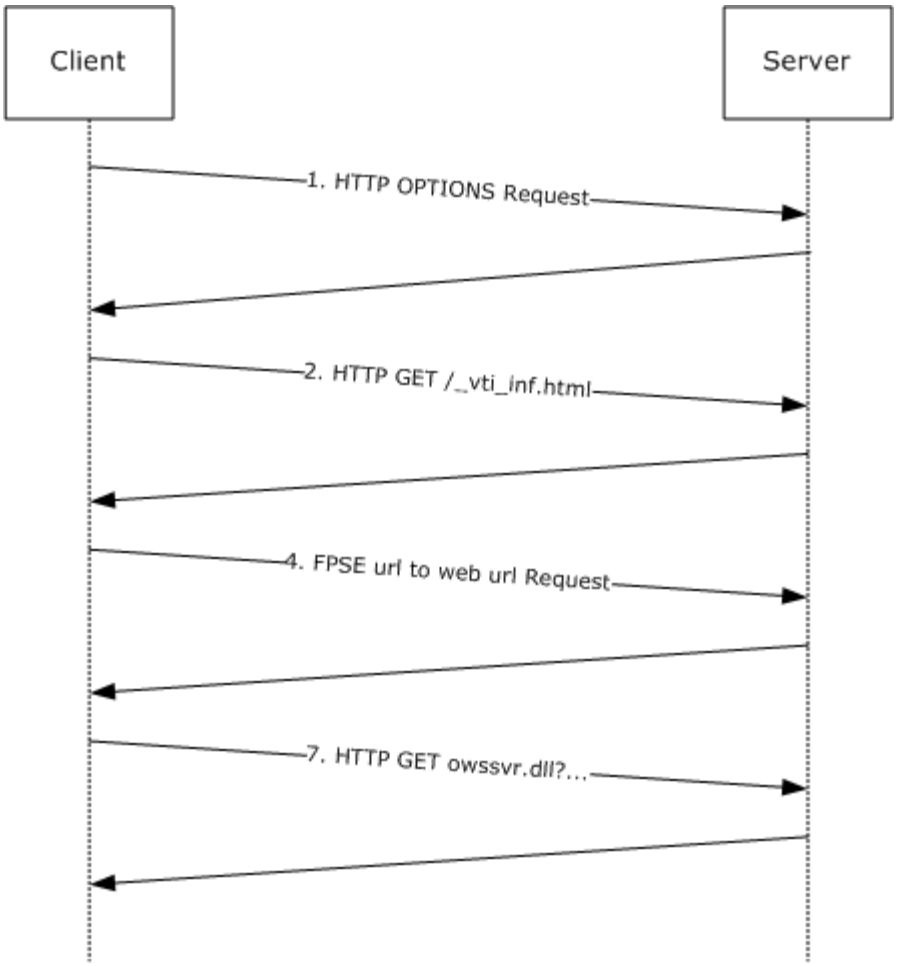

Figure 2: Windows SharePoint Services dialogview message sequence

The first three messages are used identically to those shown in the first figure in this section. The final HTTP GET returns the HTML that the client can use to display the site's file system. This GET is performed against a specific, well-known URL on the server (see section 3.1.3.2).

1.4 Relationship to Other Protocols

The FrontPage Server Extensions Remote Protocol is transported via HTTP version 1.1 GETs, POSTs and responses, as specified in [RFC2616] sections 9.3, 9.5, and 6.

Windows SharePoint Services dialogview (section 3.1.5.3.17.1) relies on the FrontPage Server Extensions Remote Protocol to discover the name and location of the Microsoft Office Web Services Server executable by using the information returned in _vti_inf.html. Windows extensions to WebDAV use its error codes, as specified in [MS-WDV].

1.5 Prerequisites/Preconditions

The client needs to know the URL of the server it wants to communicate with, which is usually passed by the user as the prompt for beginning the FrontPage Server Extensions Remote Protocol

conversation. If required by the server, the client authenticates using the underlying HTTP mechanisms, as specified in [RFC2616] section 14.8.

1.6 Applicability Statement

The FrontPage Server Extensions Remote Protocol is a precursor to the WebDAV protocol and can be used in similar situations. Because it is an earlier technology, most developers will find WebDAV, as specified in [MS-WDV], a more appealing option.

The Windows SharePoint Services dialogview aspect of the protocol, as specified in section 3.1.5.3.17.1, can be used by a client application to allow the server to control the appearance of a file store to a user. For instance, if the server has richer metadata semantics than the typical client can display or uses a nonstandard file and folder hierarchy, this view can be used to offload rendering of the file navigation to the server.

1.7 Versioning and Capability Negotiation

1.7.1 Protocol Versions

Version negotiation is performed by using the server version method (see section 3.1.5.3.14). The client sends its own protocol version in the method name section of the request. The server compares that to the server protocol version and replies to the client. The protocol version the server uses is given in the response header in the form of (Min(ServerVersion, ClientVersion)). The client is expected to use this version for any remaining communications. If the version of the client or server is not supported, the one with the newer protocol version discontinues the conversation.

The structure of a FrontPage Server Extensions Remote Protocol version (see section 2.2.2.2.9), as defined, has four parts: a major version, minor version, phase number, and build number. Thus, a version might look like 1.0.0.3214. Versions grow over time, so 3.0 is considered earlier, or older, than 4.0. To compare versions, begin with the major version, then the minor version, then the phase number, and then the build number to see which one is the latest version.

In the FrontPage Server Extensions Remote Protocol, the client, server, and protocol each have their own version, although all of them follow the same format. The client and server version are used in the negotiation to determine the protocol version. For details, see section 3.1.5.3.14.

All servers reject any client with a version that is earlier than 4.0.2.2611, and clients reject any server with a version that is earlier than 3.0.2.1002. The server returns the error code (0x0004000C) if an incompatible client is encountered. If the version of the server is not supported, the client simply ignores the server, and no further communication with the server is attempted.

The FrontPage Server Extensions Remote Protocol components were shipped as part of Windows and as separate Web downloads. Only the major and minor version numbers are listed. Except as noted previously, any numbers that meet the requirements (see section 2.2.2.2.9) can be used for the phase and build numbers.

For the remainder of this document, the protocol version will be referred to instead of the Windows version to describe support for protocol functionality.

1.7.2 Capability Negotiation

The FrontPage Server Extensions clients and servers perform capability negotiation because some operations are only supported by newer servers. This negotiation is performed by using the site **metadata** that is returned in the server version method, as specified in section 3.1.5.3.14. The metadata contains a set of keys and values. Clients can determine server capabilities by looking for

certain keys in the metadata. Information regarding **metakeys** that denote specific behaviors the server does or does not support are detailed in section 2.

1.8 Vendor-Extensible Fields

There are no vendor-extensible fields in the FrontPage Server Extensions Remote Protocol.

1.9 Standards Assignments

The FrontPage Server Extensions Remote Protocol does not use any standards assignments other than those of HTTP 1.1, as specified in [RFC2616].

2 Messages

The following sections specify the message transport, message syntax, data types, and messages that are used in the FrontPage Server Extensions Remote Protocol.

2.1 Transport

The FrontPage Server Extensions Remote Protocol uses HTTP version 1.1, as specified in [RFC2616], as transport for the GET and POST methods.

2.1.1 Client Requests

Client requests to the server MUST be transmitted as POST or GET methods appended to a URL, hereafter referred to as the URL Mode. For details about the syntax, see section 2.2.1.

If the client request does not conform to the message definitions that follow, the server MUST return an error to the client and stop parsing the request. See section 3.1.5.2 for details on the form of server error responses. See section 2.2.2.3 for details regarding irrecoverable error responses.

2.1.2 Server Responses

Server responses to client requests MUST be transmitted as HTML ([RFC2854]) and are hereafter referred to as HTML Mode. Exceptions are if the server responses are otherwise specified. For details about the syntax, see section 2.2.1.

If the server response does not conform to the message definitions that follow, the client MUST ignore the server response and stop communication with the server.

2.2 Message Syntax

This section specifies the FrontPage Server Extensions syntax and the data types that are used when a client posts FrontPage Server Extensions Remote Protocol requests to a server. It also specifies the syntax that is used by the server to respond to client requests. The syntax and data types are defined using Augmented Backus-Naur Form (ABNF), as specified in [RFC4234].

2.2.1 Syntax

The FrontPage Server Extensions Remote Protocol is used in URL Mode and HTML Mode in client request and server responses, respectively. These two modes differ with respect to encoding rules and the values of certain tokens in the stream. Implementations MUST use the following syntax rules that define these encoding schemes.

All FrontPage Server Extensions Remote Protocol communications are case-sensitive. The reader needs to assume that all strings are case-sensitive unless otherwise noted.

2.2.1.1 Syntax Delimiters

The following two sections specify primitives that are used as punctuation within strings in the full syntax for both URL Mode and HTML Mode, respectively. They are defined for both URL Mode and HTML Mode, so that in the remainder of the document a single definition can be given for higher-level constructs.

2.2.1.1.1 URL Mode

The listed primitives are used as punctuation within a string in URL Mode.

```
PARGSEP = "&"
SARGSEP = ";"
VALSEP = "="
LISTSEP = ";"
OBRACKET = "["
CBRACKET = "]"
STARTLIST = ""
```

2.2.1.1.2 HTML Mode

The listed primitives are used as punctuation within a string in HTML Mode.

```
PARGSEP = LF "<p>"
SARGSEP = LF "<li>"
VALSEP = "="
LISTSEP = LF "<li>"
OBRACKET = LF "<ul>"
CBRACKET = LF "</ul>"
STARTLIST = LF "<li>"
```

2.2.1.1.3 Nesting Level Dependent Elements

An implementation of the FrontPage Server Extensions Remote Protocol MUST keep track of the number of times an OBRACKET is sent minus the number of times a CBRACKET is sent in the current request. Hereafter, the value will be referred to as the **nesting level**. This value affects which delimiters MUST be used.

If the nesting level is zero:

```
ARGSEP = PARGSEP
```

Otherwise, if the nesting level is not zero,

```
ARGSEP = SARGSEP
```

2.2.1.2 Character Escaping

The FrontPage Server Extensions Remote Protocol uses UTF-8, as specified in [RFC2279], as its character encoding. In every instance that follows in this document in which a string is referred to as a literal, it can be assumed that the character is UTF-8 encoded. Depending on the mode, URL Mode or HTML Mode, various character escaping is used, as shown in the following two sections.

2.2.1.2.1 URL Mode

In URL Mode, characters are escaped as follows.

```
ESCAPED-BYTE = ALPHA / DIGIT ; literal meaning
/ "+" ; encoded space
/ "%5c%5c" ; encoded backslash
/ "%5c%3d" ; encoded equal sign
/ "%5c%5b" ; encoded open bracket
/ "%5c%5d" ; encoded close bracket
/ "%5c%3b" ; encoded semicolon
/ "%5c%22" ; encoded double quote
/ "%" 2HEXDIG ; anything not mentioned above
```

A sender SHOULD encode in this order (for example, a space needs to be encoded as "+" rather than "%20"; an A needs to be encoded as "A" rather than "%41"). A receiver MUST decode "%" 2HEXDIG and + to a space. A "%5c" sequence MUST be ignored except:

1. If it is followed by another "%5c" sequence, in which case it MUST be treated as a backslash.
2. If it is followed by “%3d”, “%5b”,”%5d”, “%3b”, or "%22" the "%5c" is ignored, but the character that comes after MUST NOT be treated as a delimiter.

2.2.1.2.2 HTML Mode

In HTML Mode, characters are escaped as follows:

```
ESCAPED-BYTE =
   %d32-33 / %d35-58 / %d63-91 ; literal meaning
   / %d93-122 / %d124 / %d126-127 ; literal meaning
   / "\t" ; encoded tab (%d8)
   / "\b" ; encoded backspace (%d9)
   / "\n" ; encoded newline (%d10)
   / "\f" ; encoded formfeed (%d12)
   / "\r" ; encoded carriage return (%d13)
   / "&#" 2DIGIT ";" ;encoded non-printing characters that are not
                      specially handled (%d0-7 / %d11 / %d14-31) or
                      special printing characters (%d34 / %d59-62 / %d92)
   / "&#" 3DIGIT ";" ;special printing characters (%d123 / %d125) or
                      non-printing 3 digit characters (%d128-255)
```

That is, send "\t" (third expansion) rather than "&#08;" (eighth expansion), and send "<" (eighth expansion) rather than "<" (ninth expansion). However, a receiver MUST accept any of these forms.

2.2.2 Data Types

This section describes the data types that are used when the client posts FrontPage Server Extensions Remote Protocol requests to the server, and the server responds to the client.

2.2.2.1 Primitive Data Types

This section specifies the primitive data types that are used in the FrontPage Server Extensions Remote Protocol, using ABNF as specified in [RFC4234].

2.2.2.1.1 UNSIGNED-INT

The UNSIGNED-INT data type is an unsigned decimal integer that can be represented in 32 bits.

```
UNSIGNED-INT = 1*DIGIT ; default value = "0"
```

2.2.2.1.2 INT

The INT data type is a signed decimal integer that can be represented in 32 bits.

```
INT = [ "-" ] UNSIGNED-INT
```

2.2.2.1.3 BOOLEAN

The BOOLEAN data type represents a value that can be true or false.

```
TRUE = "true"
FALSE = "false"
BOOLEAN = "TRUE" / "FALSE" ; default value = "FALSE"
```

2.2.2.1.4 DOUBLE

The DOUBLE data type is a signed floating point number that can be represented in 64 bits, as specified in [IEEE754].

```
DOUBLE = INT [ "." UNSIGNED INT ] / [ "-" ] "." UNSIGNED-INT
```

2.2.2.1.5 STRING

The STRING data type is an encoded text string of arbitrary length.

```
STRING = *ESCAPED-BYTE
```

Note STRING represents a **Unicode** string with each BYTE corresponding to a byte in a UTF-8 sequence. For instance, the "æ" character (a combined "ae") is "U+00e6", which has a UTF-8 representation of ".". Thus, the string "Cæsar" can be represented as "C%c3%a6sar" in URL Mode and as "CÃ¦sar" in HTML Mode.

2.2.2.1.6 TIME

The TIME data type is a string containing a date and time.

```
TIME = STRING
```

Note TIME values MUST conform to the format specified in [RFC2822] section 3.3.

2.2.2.2 Complex Data Types

This section specifies the complex data types that are used in method requests and responses. These values, in addition to the primitive data types, will be used throughout section 3.1.5.3 to define the data types for arguments and return values.

2.2.2.2.1 Vector

A vector is a typed array of elements whose default value is empty.

```
VECTOR-UNSIGNED-INT = OBRACKET STARTLIST UNSIGNED-INT  *(LISTSEP UNSIGNED-INT) CBRACKET
VECTOR-INT = OBRACKET STARTLIST INT *(LISTSEP INT) CBRACKET
VECTOR-BOOLEAN = OBRACKET STARTLIST BOOLEAN *(LISTSEP BOOLEAN) CBRACKET
VECTOR-DOUBLE = OBRACKET STARTLIST DOUBLE *(LISTSEP DOUBLE) CBRACKET
VECTOR-STRING = OBRACKET STARTLIST STRING *(LISTSEP STRING) CBRACKET
VECTOR-TIME = OBRACKET STARTLIST TIME *(LISTSEP TIME) CBRACKET

VECTOR-X = OBRACKET STARTLIST X *(LISTSEP X) CBRACKET
```

All data types can have a vector type associated with them where X, as in the preceding example, represents the vector data type. For instance, VECTOR-STRING = OBRACKET STARTLIST STRING *(LISTSEP STRING) CBRACKET. X can be a simple type, such as STRING, or a complex type, such as DOCINFO.

2.2.2.2.2 Protocol-Version-String

A PROTOCOL-VERSION-STRING is an identifier for a specific protocol version, used for version negotiation between clients and servers.

```
PROTOCOL-VERSION-STRING = UNSIGNED-INT "." UNSIGNED-INT "."
   UNSIGNED-INT "." UNSIGNED-INT
```

2.2.2.2.3 URL-String

A URL-STRING is a URL in the form of a URI-reference, as specified in [RFC3986].

```
URL-STRING = URI-reference
VECTOR-URL-STRING = OBRACKET STARTLIST URL-STRING *(LISTSEP URL-STRING) CBRACKET
```

The URL-STRING can be further qualified as server-relative or service-relative for specific uses.

2.2.2.2.4 Request-Name-String

A REQUEST-NAME-STRING is an identifier for a method.

```
REQUEST-NAME-STRING = STRING
```

The REQUEST-NAME-STRING MUST be an encoded string containing one of the method name values defined in section 3.1.5.3.

2.2.2.2.5 RPCKEY and RPCVALUE

An RPCKEY and RPCVALUE pair are used to specify methods, parameters, and results.

```
RPCKEY-KEY-STRING = STRING
RPCKEY = [ARGSEP] RPCKEY-KEY-STRING VALSEP
```

The leading ARGSEP MUST be present in an RPCKEY in HTML mode. The leading ARGSEP MUST be present in an RPCKEY in URL Mode except when the RPCKEY is the first key after an OBRACKET or at the start of a response, in which case the leading ARGSEP MUST NOT be present.

```
RPCVALUE = UNSIGNED-INT / INT / BOOLEAN / DOUBLE / STRING
   / TIME / VERSION / URL-STRING
   / METHOD-VALUE / DICT / METADICT / DOCINFO
   / DOCUMENT-LIST-RETURN-TYPE / SERVICE-RETURN-TYPE
   / DOC-INFO-REQUEST / URL-DIRECTORY / STATUS
   / PUT-OPTION / RENAME-OPTION
   / VECTOR-UNSIGNED-INT / VECTOR-INT
   / VECTOR-BOOLEAN / VECTOR-DOUBLE / VECTOR-STRING
   / VECTOR-URL-STRING / VECTOR-URL-DIRECTORY
   / VECTOR-DOCINFO / VECTOR-METADICT
   / VECTOR-X
```

2.2.2.2.6 Method-Key-Value

The METHOD-KEY-VALUE is an RPCKEY RPCVALUE pair that specifies the method used by the server.

```
METHOD-KEY = RPCKEY
METHOD-VALUE = REQUEST-NAME-STRING [":" PROTOCOL-VERSION-STRING]
METHOD-KEY-VALUE = METHOD-KEY METHOD-VALUE
```

The RPC-KEY-STRING in the RPCKEY of a METHOD-KEY MUST be "method".

2.2.2.2.7 Request Syntax

This section specifies the syntax for a FrontPage Server Extensions Remote Protocol request. A REQUEST consists of a method identifier, which can be followed by parameter names with arguments. For details about which arguments are sent for each method, refer to section 3.1.5.3.

```
REQUEST = METHOD-KEY-VALUE *(ARG-NAME ARG-VALUE) LF
```

The parameter names and arguments for the request are the set of ARG-NAME ARG-VALUE elements that appear after the METHOD-KEY-VALUE.

```
ARG-NAME = RPCKEY
ARG-VALUE = RPCVALUE
```

2.2.2.2.8 Response Syntax

This section specifies the syntax for a FrontPage Server Extensions Remote Protocol response. The specifics of which return values are sent for each method are as specified in section 3.1.5.3.

```
RESPONSE = "<html><head><title>vermeer RPC packet</title></head>" LF
"<body>" METHOD-KEY-VALUE *(RET-NAME RET-VALUE) "</body>" LF "</html>"
LF
```

The return values are the set of RET-NAME RET-VALUE elements that appear after the REQUEST-NAME-STRING.

```
RET-NAME = RPCKEY
RET-VALUE = RPCVALUE
```

2.2.2.2.9 Version

This datatype is used to communicate a version number. The default value is "0.0.0.0".

```
VERSION = OBRACKET "major ver" VALSEP INT ARGSEP "minor ver" VALSEP
INT ARGSEP "phase ver" VALSEP INT ARGSEP "ver incr" VALSEP INT
CBRACKET
```

Major and minor versions are 1.0, 1.1, 2.0, 3.0, 4.0, 5.0, 6.0, and 12.0. In phase version, the values are 0, 1, 2, or 3. The number 0 represents an alpha release or earlier; 1 represents a beta release; 2 represents an official release; 3 represents an updated version increment that is used to differentiate, for example, SP1 from SP2, or internal builds before release.

Note Version numbers are ordered numerically, not lexicographically. For example, 12.9 is earlier than 12.10.

2.2.2.2.10 DICT

The FrontPage Server Extensions Remote Protocol form of a **dictionary** is a DICT.

```
KEY-STRING = STRING ;
```

The key that is used to look up the value is as follows:

```
VALUE-STRING = STRING ;
```

The value that is found with the key is as follows:

```
DICT = OBRACKET [STARTLIST KEY-STRING LISTSEP VALUE-STRING *(LISTSEP
KEY-STRING LISTSEP VALUE-STRING)] CBRACKET ; default value = empty
```

2.2.2.2.11 METADICT

A **METADICT** is a **DICT** structure used to store a metadictionary of metadata associated with files, directories, and services. For a METADICT, the VALUE-STRING, when decoded, MUST be in a special form specified as a METADICT-VALUE.

```
METADICT-VALUE = "T" METADICT-CONSTRAINT-CHAR "|" TIME
/ "V" METADICT-CONSTRAINT-CHAR "|" METADICT-STRING-VECTOR
/ "B" METADICT-CONSTRAINT-CHAR "|" BOOLEAN
 / "I" METADICT-CONSTRAINT-CHAR "|" INT
/ "U" METADICT-CONSTRAINT-CHAR "|" METADICT-INT-VECTOR
/ "D" METADICT-CONSTRAINT-CHAR "|" DOUBLE
/ "S" METADICT-CONSTRAINT-CHAR "|" STRING
```

The METADICT-CONSTRAINT-CHAR is no longer significant but is still present for backward compatibility with existing metadata. It can be considered a hint for the client to adopt the following behavior.

```
X: The client MUST ignore the value.
R: The client MAY read the value but MUST NOT write the value.
W: The client MAY read or write the value.
```

Constraints on modification of metadata are now the responsibility of the server and are described in section 2.2.4.

```
METADICT-INT-VECTOR = "" / INT *(SP INT)
METADICT-STRING-VECTOR = "" / METADICT-STRING-ITEM *(SP METADICT-STRING-ITEM)
METADICT-STRING-ITEM = *METADICT-STRING-ITEM-CHAR
METADICT-STRING-ITEM-CHAR = %x1-1F / %x21-5b / %x5d-ff ; unescaped
/ %x5c SP; escaped space
/ %x5c %x5c; escaped backslash
```

2.2.2.2.12 DOCINFO

The DOCINFO contains a document name and its metadata.

```
DOCINFO = OBRACKET ARGSEP "document_name" VALSEP URL-STRING ARGSEP "meta_info" VALSEP
METADICT CBRACKET
VECTOR-DOCINFO = OBRACKET 1*DOCINFO CBRACKET
```

A DOCINFO assumes that the URL specified by the *document_name* parameter is **service-relative**.

Example (encoded as sent over the wire):

```
%5bdocument%5fname%3dfolder1%2ffolder2%2fsmall%2etxt%3bmeta%5finfo%3d
%5bvti%5ftimelastmodified%3bSW%7c08+June+2006+21%3a40%3a07+%2d0000%5d
%5d
```

Example (decoded for readability):

```
&document=
[document_name=folder1/folder2/small.txt;
meta_info=[vti_modifiedby;SW|user_name;
vti_author;SW|user_name]]
```

2.2.2.2.13 Document-List-Return-Type

The Document-List-Return-Type is used by the server to return a list of documents and their metadata.

```
DOCUMENT-LIST-RETURN-TYPE = OBRACKET *(OBRACKET "document_name" VALSEP
URL-STRING ARGSEP "meta_info" VALSEP METADICT CBRACKET) CBRACKET
```

2.2.2.2.14 Service-Return-Type

The Service-Return-Type is used to return information about a site.

```
SERVER-RELATIVE-URL-STRING = URL-STRING
```

The URL MUST be server-relative.

```
SERVICE-RETURN-TYPE = OBRACKET "service_name" VALSEP
SERVER-RELATIVE-URL-STRING ARGSEP "meta_info" VALSEP METADICT CBRACKET
```

2.2.2.2.15 DOC-INFO-Request

The DOC-INFO-Request is used to return information about a document name and its metadata.

```
DOC-INFO-REQUEST = ARG-NAME DOCINFO
```

The RPC-KEY-STRING in ARG-NAME MUST be "document".

2.2.2.2.16 Url-Directory

The Url-directory provides the name and metadata associated with a given URL.

```
URL-DIRECTORY = OBRACKET "url" VALSEP URL-STRING ARGSEP "meta_info"
VALSEP METADICT CBRACKET
VECTOR-URL-DIRECTORY = OBRACKET STARTLIST URL-DIRECTORY *(LISTSEP URL-DIRECTORY) CBRACKET
```

2.2.2.2.17 Status

The STATUS-CODE datatype is used to send back a status error code.

```
STATUS-CODE = UNSIGNED-INT
```

```
STATUS = OBRACKET "status" VALSEP STATUS-CODE ARGSEP "osstatus" VALSEP
STATUS-CODE ARGSEP "msg" VALSEP STRING ARGSEP "osmsg" VALSEP STRING
CBRACKET
```

2.2.2.2.17.1 Error Codes

This section specifies a selection of common errors that might require client action, and gives details of the client responses that are suggested. The server expects no specific responses from the client in reply to any specific error; however, there are no restrictions on the protocol server as to which errors can be generated at what time, or on the total set of errors which can be used. Therefore, the client is required to be prepared to handle any errors that are generated. If the client encounters an error it does not recognize or cannot interpret, preferably the client can handle this by outputting the error string (message) to the user. All errors from the server are paired with message strings.

Errors from the table below which are generated by few specific methods, or which suggest client action only in specific cases, will be referenced again in the Methods section of the document (3.1.5.3), as notes associated with the method that generates each error.

Error code	Message
0x0002000C	Write error on file 'value'. This error can be generated in any case in which the protocol server fails to write a file. Clients can provide users with the option to retry the operation or to retry with alternate choices in the case that a file cannot be written. If the file in question was being written to the client machine or to a path of the user's choosing, the client can present the user with a more specific response.<1>
0x00020019	Cannot rename 'value' to 'value': destination already exists. This error can be generated in response to any attempt by the server to rename an object when an object with the chosen name already exists. The client can provide an error message to the user, and can opt to re-attempt the operation specifying a different value for the destination name. Methods: move document
0x00040006	RPC syntax error: expected 'value' after 'value' but saw 'value' instead. This represents a general syntax error; it can occur whenever the client makes an invalid request. Direct client action is not necessary, other than returning the error to the user; however, this error is noteworthy because it indicates an issue with the client request. Client implementations need to determine why incorrect syntax was provided.
0x0004000B	Client closed connection. This error is generated in response to the server-client connection being closed unexpectedly on the client side. The client can opt to re-establish and retry whatever transaction was in progress. This error is also noteworthy for client implementations to determine why the connection was closed prematurely.
0x0004000C	The version of Windows SharePoint Services running on the server is more recent than the version of 'value' you are using. A more recent version of 'value' is needed. This error is returned in the case of unsuccessful version negotiation. (This topic is outlined in more detail in section 1.7.) The client can opt to respond by blocking the attempted action that would follow version negotiation and/or by returning the error to the user directly. If proper version negotiation occurs, this error is most commonly generated in the context

Error code	Message
	of the server version method (section 3.1.5.3.14), but is not exclusive to that context.
0x00090002	A file with the name 'value' already exists. It was last modified by 'value' on 'value'. This error is generated when the server encounters an existing file that was not expected to be present. This error can also occur when "modified by" dates between files sharing the same name were expected to match. The client can opt to provide this message to the user, or retry the operation with a different filename. Methods: (move document, remove document, put document, and put documents)
0x00090005	The URL 'value' is invalid. It can refer to a nonexistent file or folder, or refer to a valid file or folder that is not in the current Web. This error can be generated when the client requests an invalid URL. The client can return this message to the user or retry the operation with another URL.<2>
0x00090006	There is no file with URL 'value' in this Web. This error is generated when the client requests a URL that references no known file. The client can return the error message to the user or retry with another URL. In some situations the client might decide to ignore the error.<3>
0x00090007	The folder that would hold URL 'value' does not exist on the server. This error can occur either when a directory that the server expected to be present was not found, or when the client requested a folder that was not present. The client can return the error to the user or retry with another URL. In some situations, the client might decide to ignore the error.<4>
0x0009000D	A folder with the name 'value' already exists. This error can be generated when an attempt is made to create or rename a folder, and the target folder name already exists. The client can return this error to the user, or retry the operation again with another folder name. Methods: create url-directories, create url-directory
0x0009000E	The file 'value' is checked out or locked for editing by 'value'. This error is generated when a change (or another lock) is attempted on a file that is already locked. The client can opt to provide the user with the full error in sharing scenarios. Method: put document, put documents, checkout document
0x0009000F	The file 'value' is not checked out. This error is generated when an upload or checkin/unlock is attempted for an existing document that was not checked out. The client can return the error message to the user. Methods: put document, put documents, uncheckout document
0x0009001E	Some files have been automatically checked out from the source control repository. This error can be generated in rare cases during document checkin with specific server configuration in source control scenarios. The client can return the error to the user.<5>
0x00090023	The folder 'value' does not exist. Create the folder and then retry the operation.

Error code	Message
	This error can be generated when an operation specifies a nonexistent folder name. The client can return this error to the user, or can opt to prompt the user to retry with another value.
0x000E0002	The method is not recognized. This error is generated when the server has received an invalid request from the client for a method that does not exist on the server. No immediate client action has to be taken; this error suggests troubleshooting on the part of client implementations to determine which invalid request the client was generating.
0x0009003E	Only exclusive checkout for document 'value' is supported with SharePoint Designer–based locking. This error is generated when the client requests shared locking, in the case that the lightweight source control model is enabled. The client can opt to display an error message to the user. Methods: checkout document and get document
0x001E0002	Access denied. This is a generic access denied error. The server can generate this error, as opposed to returning a 401 NOT FOUND error, in cases where the server deliberately chooses not to allow the client to reauthenticate. The client can show the user this error, but cannot attempt to reauthenticate, and cannot treat this as a 401 NOT FOUND error.

2.2.2.2.18 Put-Option

The Put-Option is used to define the behavior of file upload operations.

```
PUT-OPTION-VAL = "atomic"
```

If this flag is specified, the server does all the needed checking to ensure that all the files can be updated before changing the first one. The server MAY ignore this.<6>

```
PUT-OPTION-VAL =/ "checkin"
```

The document is checked in after it is saved. This flag is only used to support **long-term checkout** operations. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this. Servers MAY ignore this parameter if they choose not to support long-term checkout.

```
PUT-OPTION-VAL =/ "checkout"
```

Valid only if checkin is specified. Notifies the source control of the new content (checkin), but keeps the document checked out. (This is the equivalent to checking the document in, and then checking it out again.) Clients conforming to the FrontPage Server Extensions Remote Protocol defined in this document MUST NOT send this option. Servers MAY ignore this parameter.

```
PUT-OPTION-VAL =/ "createdir"
```

The parent directory is created if it does not exist. When this option is not sent by the client, the server MUST require that the parent directory of a file or folder exists; if the client sends this option, the server SHOULD create the immediate parent of the file being created if needed and if possible. For example, if this option is sent and folder1/folder2/file.txt is being created, the protocol requires the server to create folder2 if needed and possible, but does not require it to create folder1 if it does not already exist.

```
PUT-OPTION-VAL =/ "edit"
```

Uses the date and time the document was last modified to determine whether the item has been concurrently modified by another user. This flag is used to prevent race conditions where two users could edit the same data. If this flag is specified and the inbound modification time does not match the value on the server, the server MUST reject the upload. The client SHOULD send this flag unless a higher level has indicated it needs to overwrite changes. The client MUST send either this flag or the "overwrite" flag, but not both.

```
PUT-OPTION-VAL =/ "forceversions"
```

Not used by the FrontPage Server Extensions Remote Protocol. Acts as though versioning is enabled, even if it is not. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this option. Servers MAY ignore this parameter.<7>

```
PUT-OPTION-VAL =/ "listthickets"
```

Requests that metadata be returned for **thicket** supporting files. The server MUST act as though this parameter was sent if the effective protocol version is less than 5.0.

```
PUT-OPTION-VAL =/ "migrationsemantics"
```

Not used by the FrontPage Server Extensions Remote Protocol. Preserves information about who created the file and when. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this option. The server MAY ignore this option. If the server wants to honor this option, it SHOULD do additional authorization and ignore the option if the authorization fails.<8>

```
PUT-OPTION-VAL =/ "noadd"
```

Does not add the document to source control. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this option. The server SHOULD ignore this option.<9>

```
PUT-OPTION-VAL =/ "overwrite"
```

Uses the date and time the document was last modified, as specified in the inbound metadata, rather than the time on the server. The client MUST send either this flag or the "edit" flag, but not both.

```
PUT-OPTION-VAL =/ "thicket"
```

Specifies that the associated file is a thicket supporting file. The server SHOULD detect that the upload includes a thicket supporting file and infer this flag.

```
PUT-OPTION = *(PUT-OPTION-VAL ",") PUT-OPTION-VAL
```

The PUT-OPTION data type MUST contain at least one PUT-OPTION-VAL.

2.2.2.2.19 Rename-Option

The Rename-Option datatype is used to define the behaviors of a rename operation.

```
RENAME-OPTION-VAL = "createdir"
```

Creates the parent directory if it does not already exist. This flag is analogous to the "createdir" PUT-OPTION-VAL (as specified in Put-Option) and has the same semantics.

```
RENAME-OPTION-VAL =/ "findbacklinks"
```

Requests that servers, implementing **link fixup**, fix the linked files other than those moved. The server MAY ignore this flag.

```
RENAME-OPTION-VAL =/ "nochangeall"
```

Do not perform link fixup on links in moved documents. This parameter is used in publishing scenarios. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this option. The server MAY ignore this.<10>

```
RENAME-OPTION-VAL =/ "patchprefix"
```

Simulates the move of a directory rather than a file. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this option; the server SHOULD ignore this flag for the usage defined in this document.

```
RENAME-OPTION = "none"
/ RENAME-OPTION-VAL *("," RENAME-OPTION-VAL)
```

Note The client MUST send "none" if it does not want to specify any of the options given by a RENAME-OPTION-VAL.

2.2.2.3 Irrecoverable Error Responses

When a request is not syntactically valid (for example, if a string other than "true" or "false" is given when a BOOLEAN value is expected), when the server is required to allocate more memory than it can, or if the server is in some other irrecoverable failure situation, the server MUST abandon further processing of the client request. If the server encounters an error and cannot proceed with the request, it MUST return a STATUS in HTML Mode. The client SHOULD recognize a STATUS returned by the server as an indication that the server processing of the request failed in some way and SHOULD abandon any parsing context it was in.

2.2.3 Entry Points

Each method call is an HTTP POST by the client to a URL on the server. There are four entry points on any server, which can be discovered by the clients (see section 4.1.1). Each method entry denotes which entry point MUST be used to call that method. The four entry points are detailed in the following table:

Name	Description
FPShtmlScriptUrl	Used to retrieve the server version method, as specified in section 3.1.5.3.14; also for the URL-to-Web-URL method, as specified in 3.1.5.3.16.
FPAuthorScriptURL	Used for all methods to deal with document manipulation.
FPAdminScriptURL	Not used in the FrontPage Server Extensions Remote Protocol.
TPScriptURL	Used by Windows SharePoint Services dialogview 3.1.5.3.17.1.

2.2.4 Metadata

Files, folders, and sites in servers have an associated metadictionary, which contains strings (called keys or metakeys) that are mapped to strongly typed values. These metakey-value pairs are called metadata. Server implementations use the metadictionary to store details about entities for later use by the server. Clients store values in metadictionaries for later use by the same client or other clients. A limited number of well-known metakeys are used for client/server communication. These shared metakeys are specified in this document.

The metabase (consisting of a series of metakeys, or FrontPage Server Extension settings and properties) can be found in the _VTI_CNF folder beneath each folder on the server in the default file.

The metabase uses the colon character to separate the metakey name from the datatype. Next, the pipe ("|") character is used to separate the datatype from the metakey value. A metakey consists of a single line within the default file.

2.2.4.1 Type

Each metakey listed has an associated value Type, which is one of the METADICT-VALUE types defined in section 2.2.2.2.11. These are generic types, which can be further specified in the individual metakey's description.

2.2.4.2 Client Access

The Client Access heading refers to whether the client is able to set this metadata on the server.

- Read-only: Some communication from server to client is based on configuration information and site settings or document information that is parsed and returned to the client; the client cannot change this information. These metakeys are identified as read-only.
- Read/write: Metadata that the client is able to set on the server is identified as read/write.

2.2.4.3 Applies To

Metadata is associated with various entities on the server, which are identified in the Applies To heading.

- Service: The Service value refers to metadata associated with the Server or a particular site.

- Folder: **Folder** refers to metadata associated with a folder, directory, or list.
- File: File refers to metadata associated with a file or document.

2.2.4.4 vti_casesensitiveurls

Type: INT

Client Access: Read-only

Applies to: Service

Description:

Contains an INT flag indicating whether the server is case-insensitive with respect to URLs. If the value is 0, the default, the server is case-insensitive with respect to URLs. If the value is 1, the server is case-sensitive with respect to URLs.

The server SHOULD include this key as an INT in the metadata returned by the open service method.

The server MUST return 0 or 1 for this metakey.

It MUST only be 0 if URLs that differ only by case are considered equivalent.

The client SHOULD assume that the value is 0 if this key is not present.

```
Examples:

vti_casesensitiveurls;IX|0
vti_casesensitiveurls;IX|1
```

2.2.4.5 vti_dirlateststamp

Type: TIME

Client Access: Read-only

Applies to: Folder

Description:

The vti_dirlateststamp metakey is a time stamp that records the approximate time of the first client call to the list documents or move documents methods that included this folder, following a change to the folder contents or metadata.

The server SHOULD include this key as a date in folder metadata, but not child web folder metadata, that it returns to the client.<11>

If the client caches the response of the list documents method requests, it SHOULD cache this time stamp and send this value in the folderList parameter in subsequent calls to the list documents method. Servers SHOULD use the value during processing of a call to the list documents method for optimization, to only return data for folders that are out of date on the client.

If the client caches the response of the list documents method requests, it SHOULD cache this information as well.

This value SHOULD be used in the *folderList* parameter when the list documents method is called again to refresh the cached response.

```
Example:

vti_dirlateststamp;TX|08+Jan+2000+19:09:27+-0000
```

2.2.4.6 vti_filesize

Type: INT

Client Access: Read-only

Applies to: File

Description:

The size of the document in bytes.

The server MUST determine the size of the file in bytes and return this value on request by the client.

The server MUST return this key as an integer in the file metadata it returns to the client. It MUST be the size of the file in bytes.

```
Example:

vti_filesize;IX|1120
```

2.2.4.7 vti_hassubdirs

Type: BOOLEAN

Client Access: Read-only

Applies to: Folder

Description:

The folder has subdirectories.

The server MAY return this key and set it to TRUE only if the folder has subdirectories. The client can use this key to decide whether to display user interface elements to expand a node in a rendered directory hierarchy. Clients MUST NOT rely on a server response to this parameter.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_hassubdirs;BR|false
vti_hassubdirs;BR|true
```

2.2.4.8 vti_isbrowsable

Type: BOOLEAN

Client Access: Read-only

Applies to: Folder

Description:

The server SHOULD return this key and set its value to TRUE only if the folder contents are accessible through normal HTTP requests.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Example:

vti_isbrowsable;BR|false
vti_isbrowsable;BR|true
```

2.2.4.9 vti_ischildweb

Type: BOOLEAN

Client Access: Read-only

Applies to: Folder

Description:

The folder is the root of another site (a subsite) within this site.

The server SHOULD include this key on folder metadata it enumerates when the folder is the root of another service. The client can use this information to avoid further calls to the url to web url method when traversing a folder hierarchy that might span services. The client also MAY use this key to indicate to the user that the folder represents a service boundary.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_ischildweb;BR|false
vti_ischildweb;BR|true
```

2.2.4.10 vti_isexecutable

Type: BOOLEAN

Client Access: Read-only

Applies to: Folder

Description:

The server SHOULD include this BOOLEAN key on folder metadata, but not child web folder metadata. A value of TRUE indicates that the server permits execution of programs in the folder.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_isexecutable;BR|false
vti_isexecutable;BR|true
```

2.2.4.11 vti_isscriptable

Type: BOOLEAN

Client Access: Read-only

Applies to: Folder

Description:

The server SHOULD include this BOOLEAN key on folder metadata, but not child web folders. A value of TRUE indicates that the file or the contents of the folder can be executed if they are script files or static content. A value of FALSE only allows static files to be served.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_isscriptable;BR|false
vti_isscriptable;BR|true
```

2.2.4.12 vti_longfilenames

Type: INT

Client Access: Read-only

Applies to: Service

Description:

A flag indicating whether the server supports long file names of up to 255 characters. If the value of this metakey is 1, the server supports long file names. If the value of this metakey is 0, the server does not support long file names.

The server MUST set this value to 0 if it only supports **short file names**; otherwise, it SHOULD set this value to 1.<12>

The client MUST send only short file names when dealing with a server reporting 0 for this value.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_longfilenames;IX|0
vti_longfilenames;IX|1
```

2.2.4.13 vti_metatags

Type: METADICT-STRING-VECTOR

Client Access: Read-only

Applies to: File

Description:

A list of the META element tag settings for the current document, if any.

For HTML files, the server SHOULD maintain a list of META tags in the file. If the server maintains this list, this key MUST be present and MUST have a pair of entries for each META element tag. For META element tags that have the HTTP-EQUIV attribute, the first string in the pair MUST be "HTTP-EQUIV=" followed by the value of the HTTP-EQUIV attribute; the second string in the pair MUST be the value of the CONTENT attribute. See [RFC1866] section 5.2.5 for details on the CONTENT attribute.

For META element tags that have NAME and CONTENT attributes, the first string in the pair MUST be the value of the NAME attribute, and the second string MUST be the value of the CONTENT attribute.

A client MAY alter the way it displays files based on this value.<13>

The server MUST parse the document for this value and MAY cache the value for return to the client on request. The client cannot set this value directly but MAY change it by updating the document.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Examples:

vti_metatags;VR|HTTP-EQUIV=Content-Type text/html;\ charset=utf-16
vti_metatags;VR|HTTP-EQUIV =Content-Language en-us
```

2.2.4.14 vti_showhiddenpages

Type: INT

Client Access: Read-write

Applies to: Service

Description:

A flag indicating whether the server is configured to return information for **hidden documents**.

The server SHOULD<14> return this key following a call to the open service method. If the value is 0, the server is indicating that it is not configured to support returning information hidden documents. If the value is 1, server support for hidden documents is available, and the client MUST send a value of true for the listHiddenDocs parameter to the list documents method to have hidden documents included in the results.

```
Examples:

vti_showhiddenpages;IW|0
```

```
vti_showhiddenpages;IW|1
```

2.2.4.15 vti_sourcecontrolcheckedoutby

Type: STRING

Client Access: Read-only

Applies to: File

Description:

The login name of the user who has checked out the document under source control.

The server MUST include this value only if a document is checked out either short-term or long-term. The server MUST record the authenticated login username of the client when a document is checked out and store it in this metakey for return to the client on request.

The value stored in this metakey MUST be the same as the vti_username value of the user who has the document checked out.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Example:

vti_sourcecontrolcheckedoutby;SX|CORPDOMAIN\\johnsmith
```

2.2.4.16 vti_sourcecontroltimecheckedout

Type: TIME

Client Access: Read-only

Applies to: File

Description:

The time the document was checked out on the server.

A server SHOULD include this value only if a file is checked out either short-term or long-term. When it is set, it MUST provide a time stamp indicating when the file was checked out.<15>

The client cannot set this value directly but sets it as a side-effect of a check-out method or get document method with a check-out parameter set.

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Example:

vti_sourcecontroltimecheckedout;TX|15 Apr 2000 12:52:03 -0000
```

2.2.4.17 vti_thicketdir

Type: STRING, BOOLEAN

Client Access: Read-only

Applies to: File, Folder

Description:

This metakey contains different content depending on whether it appears in the metadata for a document or a folder.

Folder

When applied to a folder, this metakey contains a BOOLEAN flag specifying whether the folder contains the supporting files for a thicket.<16>

The server SHOULD include this metakey and set its value to TRUE for a folder that contains files supporting an HTML thicket. For folders that do not contain supporting files, the server SHOULD omit this but MAY send the key as FALSE. The client SHOULD adapt its rendering so that it does not show folders for which this key is TRUE.

Document

When applied to a document, this metakey contains a STRING with the name of the folder being used for supporting thicket files.

The server SHOULD include this metakey for a document that is the main file of an HTML thicket.

The server SHOULD update document and folder metakeys to indicate the presence and relationship of an HTML thicket document and its thicket supporting folder when handling the put document method with a put_option value of "thicket".

```
Example:

vti_thicketdir;SW|jobs/index_files
```

2.2.4.18 vti_thicketsupportingfile

Type: BOOLEAN

Client Access: Read-only

Applies to: File

Description:

Indicates whether or not the document is a supporting file in an HTML thicket.

The server SHOULD include this key and set its value to TRUE for a file that is a supporting file in an HTML thicket. For files that are not supporting files, the server SHOULD omit this but MAY send the key as false. The client SHOULD adapt its rendering so that it does not show files for which this key is TRUE.

The server SHOULD use and maintain its own value rather than use a client-supplied value for this key. It MAY consider values passed by the client or the put document PUT-OPT-VAL of "thicket".

```
Examples:
```

```
vti_thicketsupportingfile;BW|false
vti_thicketsupportingfile;BW|true
```

2.2.4.19 vti_timecreated

Type: TIME

Applies to: File, Folder

Description:

The time the document or folder was created.

The server SHOULD<17> include this key for files and folders, but not child web folders. It SHOULD reflect the date and time the file or folder was created. The client can use this value to render details about files or folders.<18>

The server MUST use and maintain its own value rather than use a client-supplied value for this key.

```
Example:

vti_timecreated;TR|24 Apr 2000 15:54:33 -0000
```

2.2.4.20 vti_timelastmodified

Type: TIME

Client Access: Read-only

Applies to: File, Folder

Description:

The time the document or folder was most recently modified.

The server SHOULD include this key for files and folders, but not child web folders.<19> It SHOULD approximate the date and time the file or folder was last modified. As specified in Put-Option (see section 2.2.2.2.18), under the edit and overwrite values, the server MUST use this for concurrency control and thus might not be able to make this reflect the time that the file was last modified. The client MAY use this value to render details about files or folders; however, vti_timelastwritten will often be more appropriate when it is available. The client SHOULD send this value in accordance with its use, as specified in section 2.2.2.2.18, under the edit and overwrite values.<20>

```
Example:

vti_timelastmodified;TR|24 Apr 2000 15:54:33 -0000
```

2.2.4.21 vti_timelastwritten

Type: TIME

Client Access: Read-only

Applies to: File, Folder

Description:

The time that the document or folder was last saved to local storage on the server.

The server SHOULD include this metakey for files and folders, but not child Web folders. The server SHOULD record the date and time that the file or folder content was modified in this metakey. The client SHOULD use this value to render details about files or folders but MAY use vti_timelastmodified for that purpose.

The server SHOULD use and maintain its own value rather than use a client-supplied value for this key.<21>

```
Example:

vti_timelastwritten;TR|24 Apr 2000 15:54:33 -0000
```

2.2.4.22 vti_title

Type: STRING

Client Access: Read-write

Applies to: File, Service

Description:

The user-readable title of a document or site.

The server SHOULD maintain this key for a site as a user-readable description of the site. The client MAY use this string to refer to the site when presenting information to a user.

The server SHOULD<22> maintain this key for documents. If the document is an HTML document on the server, the server SHOULD parse the document for the content of a TITLE element tag and MAY cache this value for return to the client. The client MAY update this metakey for HTML documents, and the server SHOULD rewrite the document with an updated TITLE element.

For file streams that the server is unable to parse, the server SHOULD accept a client-supplied metakey value and return it on client request.

The client SHOULD use this metakey value where the title of the document is to be displayed to the user.

```
Example:

vti_title;SR|Trey Research
```

2.2.4.23 vti_username

Type: STRING

Client Access: Read-only

Applies to: Service

Description:

The current authenticated login user name associated with the client.

The server SHOULD include this metakey in the site metadata. When the key is present, the client SHOULD use this key for comparisons with vti_sourcecontrolcheckedoutby.

The server SHOULD include this key in the service metadata. Its value is a string that uniquely identifies the user to the server. When the key is present, the client SHOULD use this key rather than any information that it passed to the HTTP layer for authentication for comparison with vti_sourcecontrolcheckedoutby.

The server SHOULD use and maintain its own value rather than use a client-supplied value for this key.<23>

```
Example:

vti_username;SX|CORPDOMAIN\\johnsmith
```

2.2.4.24 mf-file-status

Type: INT

Client Access: Read-only

Applies to: Service

Description:

The current file transfer status, sent as part of the get documents method server response. A value of 0 indicates success. Any nonzero value indicates a failure.

3 Protocol Details

3.1 Server Details

A FrontPage Server Extensions client SHOULD initialize a connection with the server. The client can then make as many method calls against the server as needed. The FrontPage Server Extensions Remote Protocol, like HTTP 1.1, as specified in [RFC2616], is a stateless protocol. As such, connections do not need to be closed.

The Windows SharePoint Services dialogview client of the FrontPage Server Extensions Remote Protocol is simply used for viewing information on a server. After the dialogview connection is initialized, the client MAY make any supported calls against the server. The FrontPage Server Extensions Remote Protocol connections do not need to be closed.

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

3.1.1 Abstract Data Model

The FrontPage Server Extensions Remote Protocol is used to communicate with remote file systems for administrative purposes. For those reasons, a data model that looks like a file system is useful.

The FrontPage Server Extensions Remote Protocol has three concepts in its file system: files, directories, and services. Each of these structures MUST have a metadictionary of metadata associated with it. This metadata can be used by clients or servers to store whatever information is relevant to the object. The metadata metadictionary is also often used as a way for the server to communicate information about a file to the client, such as its check-out state.

In addition to the metadata metadictionary, the relevant properties of each file system concept are as follows:

1. Files need to have a content stream, that is, an array of sequenced bytes, that represent the contents of the file.

2. Directories do not have a content stream, but are able to contain files or other directories.

3. Services model the concept of a site. Like directories, they do not have a content stream and are able to contain files or directories. Services are also able to answer questions about their own capabilities (for example, whether they support source control). The capabilities of a service are communicated to clients by using the metadictionary of the service.

3.1.1.1 Source Control

Servers MUST support **short-term checkout** and can implement a source control sandbox. The server MAY offer users the option to turn the source control sandbox off.<24>

For a server that has the source control sandbox turned off: When a document is checked out, the server MUST refuse to perform any operations on the document that are not sent by the user who has the document checked out, including another user requesting to read the document.

For a server that has the source control sandbox turned on: When a document is checked out, the server MUST refuse any requests to modify the document that are not sent by the user who has the

document checked out. However, if another user merely requests the contents of the document, the server can respond with the document stream as it appeared when the document was checked out.

A document that is checked out short-term can have the checkout released in either of two ways: the client can send an uncheckout document request to the server or the short-term checkout expires.

3.1.2 Timers

3.1.2.1 Short-Term Checkout Timer

A document checked out short-term has a time-out value associated with the checkout. The length of the short-term checkout timer is determined based on client input, as detailed in the checkout document method (section 3.1.5.3.2).<25>

3.1.3 Initialization

Microsoft FrontPage Server Extensions initialization requires the following operations to take place.

3.1.3.1 Determining Server Capabilities

A prospective client MUST perform an HTTP OPTIONS request, as specified in [RFC2616] section 9.2, against a server to determine if it is a FrontPage Server Extensions Remote Protocol server.

When receiving an OPTIONS request, a FrontPage Server Extensions Remote Protocol server MUST return the header 'MS-Author-Via:' with a value that includes 'MS-FP/4.0' to indicate that the server supports the FrontPage Server Extensions Remote Protocol. The server SHOULD list the protocols in the MS-Author-Via header in order of preference from highest to lowest. The client SHOULD use the first client-supported protocols listed in the MS-Author-Via header.

When receiving an OPTIONS request, a server that supports SharePoint Team Services dialogview MUST return the header '50_Collab' to indicate that the server supports the dialogview for any client with the user agent "Microsoft Data Access Internet Publishing Provider". Results of the HTTP OPTIONS request that are returned from the server SHOULD be cached by clients as a performance optimization.

3.1.3.2 Determining Entry Points

3.1.3.2.1 Client Request for Entry Point HTML Page

If the server supports the FrontPage Server Extensions Remote Protocol, the client SHOULD perform an HTTP GET of _vti_inf.html at the server root by using the following to determine the entry point for the FrontPage Server Extensions:<26>

```
http://fpseserver/_vti_inf.html
```

where fspeserver is the root site of the server.

3.1.3.2.2 Server Entry Point HTML Page Response

The server MUST reply to the HTTP GET of _vti_inf.html with an HTML **page** containing an HTML comment that is an ENTRY-POINT-COMMENT, defined as follows.

```
ENTRY-POINT-COMMENT = COMMENT-BEGIN + SHTML-ENTRY-POINT +
AUTHOR-ENTRY-POINT + ADMIN-ENTRY-POINT + TPSCRIPT-ENTRY-POINT +
COMMENT-CLOSE

COMMENT-BEGIN = "<!-- FrontPage Configuration Information FPVersion="
+ DQUOTE + VERSION + DQUOTE + LF

SHTML-ENTRY-POINT = "FPShtmlScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

AUTHOR-ENTRY-POINT = "FPAuthorScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

ADMIN-ENTRY-POINT = "FPAdminScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

TPSCRIPT-ENTRY-POINT = "TPScriptUrl=" + DQUOTE +
SERVICE-RELATIVE-URL + DQUOTE + LF

COMMENT-CLOSE = "-->"
```

Servers SHOULD return a comment that defines the entry points as follows, as clients MAY assume these values.

```
<!-- FrontPage Configuration Information FPVersion="12.0.0.000"
FPShtmlScriptUrl="_vti_bin/shtml.dll/_vti_rpc"
FPAuthorScriptUrl="_vti_bin/_vti_aut/author.dll"
FPAdminScriptUrl="_vti_bin/_vti_adm/admin.dll"
TPScriptUrl="_vti_bin/owssvr.dll"
-->
```

For descriptions of the fields used in the HTML comment, see section 2.2.3. Each method description contains a section that defines which entry point that method will use. Clients MUST post to the correct entry point, or the server SHOULD ignore their request.

The client SHOULD then call the server version method on the root of the server to determine the latest (highest) server version of the protocol that the server supports. The client SHOULD use its own version number for the PROTOCOL-VERSION-STRING of the METHOD-VALUE, unless and until it knows the server version number, in which case it SHOULD use whichever of the two version numbers is lower.

On completing this initialization process, and thereafter the client SHOULD send that lower version number and the server MUST respond with either the same version number it received or with its own version number.

If the client is opening a site that is not at the root of the server, the client MUST call the url to web url request. This method accepts a **server-relative** URL, such as "/subsite/folder/document.txt". It would return the server-relative URL of the site, "/subsite", and the service-relative URL of the item, "folder/document.txt".

The client MUST then post all further methods to the site that it wants to communicate with. The *service_name* parameter MUST NOT be used by the client to denote what site it is communicating with, as this parameter is ignored by the server. For example, if the client wants to check out "/subsite/folder/document.txt", it needs to post to the SHTML-ENTRY-POINT of the subsite. Assuming the default value of the SHTML-ENTRY-POINT, that would be:

```
http://fpseserver/subsite/_vti_bin/shtml.dll/_vti_rpc
```

The client then refers to the file it wants to check out by its service-relative URL within the FrontPage Server Extensions Remote Protocol request. Finally, the client SHOULD call the open service method on the appropriate site to begin the conversation. The client can then make whatever method calls it needs against the server.

The SharePoint Team Services dialogview client SHOULD perform an HTTP HEAD request, as specified in [RFC2616], section 9.4 against owssvr.dll (for details, see section 3.1.3). If the return code from the server is 200, the client SHOULD perform an HTTP GET, as specified in [RFC2616], section 9.3 to retrieve the full page body. If the return code from the server is 410, the client MUST NOT perform an HTTP GET on the server.

3.1.4 Higher-Layer Triggered Events

There are no higher-layer triggered events for the FrontPage Server Extensions Remote Protocol server. Each client request is triggered by the client application's needs.

3.1.5 Processing Events and Sequencing Rules

Aside from initialization, methods can be called in any order, as determined by the client application's needs. The only methods forming a strong logical pair are the checkout document and the uncheckout document methods. Clients that call the former SHOULD call the latter, because a document that is checked out cannot be edited by any other users until the checkout is revoked.

Protocol Examples, section 4, provides examples that show common operations being performed against the server, giving some guidance about common method sequences.

3.1.5.1 HTTP Headers

The client SHOULD send an X-Vermeer-Content-Type header, as specified in [RFC2616], section 14.17, with the same value as the standard HTTP Content-Type header to safeguard against one-click attacks (see section 5.1). The server MUST use this header, if present, to determine the Content-Type of the request. If this header is not present, the server SHOULD fail the request.<27>

Clients MUST also include the string "FrontPage" (case-sensitive) in its User-Agent header, as specified in [RFC2616], section 14.43. The server MAY alter its responses when the client does not do this.<28>

Except as specified in the get documents method, server responses MUST have the HTTP Content-Type "application/x-vermeer-rpc".

3.1.5.2 Method Formatting

A set of formatting conventions are used for each FrontPage Server Extensions method specified in the Methods section later in this document.

The Methods section begins with a brief description of the method's purpose. The Tokens section follows the description and corresponds to the REQUEST element that specifies the set of arguments for each method. Clients can pass in any subset of the given arguments, although some arguments are required. If an argument is not required, unless otherwise specified, the server SHOULD use a default value of FALSE for BOOLEAN arguments, 0 for INT, UNSIGNED-INT, and DOUBLE arguments, an empty string for STRING and URL-STRING arguments, an empty METADICT for METADICT arguments, and an empty VECTOR-X for VECTOR-X type arguments. Clients MAY pass the

arguments in any order, and servers MUST accept them in any order. Clients SHOULD NOT pass arguments unless they are mentioned in this document. When a server is passed an argument that it does not recognize, the server SHOULD treat it as a syntax error. It MAY choose to ignore the parameter instead.<29>

Each argument definition starts with an argument name in bold type. The argument name corresponds to the ARG-NAME element. The data type is listed in the argument description, and it defines the required format for the ARG-VALUE element.

The URL section defines the URL as provided by _vti_inf.html to which clients MUST post when using this method.

The Return Values section specifies the set of return values, and its formatting is similar to the Fields section. The RET-NAME corresponds to the return value name in bold type. The format of the RET-VALUE is defined by the return value type. Servers can return arguments in any order, and clients MUST accept the parameters in any order.

In error conditions at the FrontPage Server Extensions Remote Protocol level, the server SHOULD return a RET-NAME RET-VALUE pair where the RET-NAME is "status" and the RET-VALUE is a STATUS object (see section 2.2.2.2.17). A client SHOULD ignore all other return values if a status is present. Even though this is an error, it SHOULD be encapsulated in an HTTP 200 response, as specified in [RFC2616] section 10.4.2.

Lower layers (such as HTTP or IP) can also return errors. For example, the FrontPage Server Extensions Remote Protocol layer on the server SHOULD indicate to the HTTP layer that it requires authentication, which in turn SHOULD cause the HTTP layer to send a 401 message in response, as specified in [RFC2616] section 10.4.2, to unauthenticated requests.

If the lower layers pass on an unauthenticated request from the client to the FrontPage Server Extensions Remote Protocol server, the server SHOULD respond with an HTTP 401 as specified in [RFC2616] section 10.4.2. It MAY include an entity body containing a descriptive error message in the response.<30>

3.1.5.3 Methods

Each request by a client that uses the FrontPage Server Extensions Remote Protocol MUST begin with a field that contains the method name. For example, the get document request has the string "get document" in the method field. Following the method name field is the list of arguments that MAY be specified in any order.

The client POSTs the following FrontPage Server Extensions Remote Protocol requests to the server. These requests are specified in the sections that follow.

3.1.5.3.1 Common Method Arguments

The following argument values are common to many of the methods.

Tokens

validateWelcomeNames: This parameter MUST NOT be passed by a client that conforms to the FrontPage Server Extensions Remote Protocol. The server MAY ignore this value; clients MUST NOT rely on a server response to this parameter.

listLinkInfo: Clients conforming to the FrontPage Server Extensions Remote Protocol MUST send false for this parameter. The server MAY ignore this parameter; clients MUST NOT rely on a server response to this parameter.

service_name: This parameter is obsolete, but the client MAY send this URL-STRING defining the server-relative URL of the site that the request is addressed to. However, the server MUST ignore it. The URL to which the request is posted MUST be used by the server to determine what site the request is issued against.

return_stats: The client MUST NOT send this BOOLEAN value. The server SHOULD ignore this value; clients MUST NOT rely on a server response to this parameter. <31>

3.1.5.3.2 checkout document

The checkout document request is used by the client to enable the currently authenticated user to make changes to a document under source control.

Tokens

service_name: This parameter is deprecated; see service_name in section 3.1.5.3.1.

document_name: The client MUST send and the server MUST interpret this URL-STRING as the service-relative path of the document to check out.

force: This parameter is an INT bitmask; bits 0 and 1 are defined as follows. The client MUST set all unused bits to zero; the server MUST ignore unused bits.

Mask	Meaning
Bit 0 0x00000001	Force checkout. This feature is currently unimplemented and reserved for future use. Clients MUST NOT set this bit, and servers MUST ignore it.
Bit 1 0x00000002	Refresh short-term checkout. The client MUST set this bit if, and only if, it already has a short-term checkout on the file and wants to extend the time-out on it. The server MUST attempt to create a new short-term checkout if this bit is cleared, and it MUST attempt to extend an existing short-term checkout if this bit is set. It SHOULD return an error (0x0009000E) if this bit is not set and a short-term checkout exists.<32>

The client MUST set all unused bits to zero, and the server MUST ignore all unused bits. Range: -2147483648 to 2147483647; only 0 and 2 are currently allowed.

timeout: An INT that defines the number of minutes that a short-term checkout is requested. To retain the lock, the client MUST renew its short-term checkout within this interval. The client MUST NOT send negative values, and the server SHOULD ignore them by not specifying a timeout for the short-term checkout. The value zero is reserved for server implementations that understand long-term checkout and thus MUST NOT be sent by clients. Servers MAY ignore requests in which this parameter is set to zero by not specifying a time for the short-term checkout. Range: 1 to 2147483647.<33>

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error codes 0x0009000E and 0x0009003E 2.2.2.2.17.1.

meta_info: A METADICT containing information about the document that has been checked out.

3.1.5.3.3 create url-directories

The create url-directories request allows the client to create one or more directories (folders) on the site. This operation is not atomic. If the bulk operation fails, some of the earlier directories might have been created. In the case of a failure, the client SHOULD query the server with list documents or if it needs to determine what folders were created.Clients SHOULD use this method rather than create url-directory.

Tokens

service_name: This parameter is deprecated: For semantics, see service_name in section 3.1.5.3.1.

urldirs: A VECTOR-URL-DIRECTORY specifying the names and metadata of folders to create. The client MUST specify one URL-DIRECTORY for each folder it wants to create. The server SHOULD create the directories in the order specified. The client MUST send this parameter.

URL

FPAuthorScriptURL

Return Values

The server can return the error code 0x0009000D 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

3.1.5.3.4 create url-directory

The create url-directory request is used by the client to create a folder for the current site. Clients SHOULD use create url-directories instead of using this deprecated method. However, to maintain compatibility with all client versions, servers MUST support this method.

Tokens

service_name: This parameter is deprecated: For semantics, see service_name in section 3.1.5.3.1.

url: A URL-STRING specifying the URL of the directory to be created. The client MUST send this parameter.

executable: A BOOLEAN indicating if the security settings for the newly created directory can allow execution of entities within it. If TRUE, the request is to create an **executable folder**. Clients that conform to the FrontPage Server Extensions Remote Protocol MUST send FALSE. Servers MUST ignore this for architectures that do not support the notion of an executable directory. The server SHOULD ignore this for security reasons.<34>

URL

FPAuthorScriptURL

Return Values

The server can return the error code 0x0009000D 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

urldir: A URL-DIRECTORY containing the metadata for the directory that was created.

3.1.5.3.5 getDocsMetaInfo

The getDocsMetaInfo request is used by the client to retrieve metadata for the files and directories in the current site. The getDocsMetaInfo request is similar in function to the list documents request except that it only retrieves metadata for the files or folders specified through the *url_list* parameter. The list documents requests information about all of the files and folders under a given directory; getDocsMetaInfo requests information only about the URLs requested.

Tokens

service_name: This parameter is deprecated: See service_name in section 3.1.5.3.1.

listHiddenDocs: A BOOLEAN value that specifies whether the client requests the hidden documents in a site be included in the documents listed. If TRUE, the server MAY list hidden documents; if FALSE, the server MAY leave hidden documents out of the list returned.<35>

listLinkInfo: For semantics, see section 3.1.5.3.1.

url_list: A VECTOR-STRING list of service-relative URLs about what client wants information. If url_list is empty, it is treated as though it is a single vector with "" (the root of the site).

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptURL

Return Values

document_list: A DOCUMENT-LIST-RETURN-TYPE specifying the name and metadata for the set of documents.

urldirs: A VECTOR-URL-DIRECTORY containing information about folders and subsites in the current site.

3.1.5.3.6 get document

The get document request can be used by the client to retrieve a document and its metadata for viewing or editing on a client.

Tokens

service_name: This parameter is deprecated. For semantics, see service_name in section 3.1.5.3.1.

document_name: A URL-STRING that the client MUST send, and the server MUST interpret this value as the service-relative path of the document to retrieve. The client MUST send this parameter.

effective_protocol_version: A VERSION provides a way for the client to supersede the version sent as the PROTOCOL-VERSION-STRING in the METHOD-KEY-VALUE of the request. For details, see section 2.2.2.2.6. A client conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this parameter. A server SHOULD ignore this parameter.

old_theme_html: A BOOLEAN value that determines how **theme elements** are rendered on a page. If TRUE, the page is returned using HTML that is compatible with effective_protocol_version. If FALSE, the effective_protocol_version corresponds to Microsoft FrontPage 97 or an earlier version of FrontPage that does not support themes. In this case, the theme information disappears, and the theme elements are rendered in the HTML used on the page. A client conforming to the FrontPage Server Extensions Remote Protocol MUST send FALSE (either by omitting the parameter and taking the default or by explicitly sending FALSE). A server MAY assume this parameter to be FALSE.

expandWebPartPages: A BOOLEAN value reserved for future use. This parameter MUST be ignored by the server regardless of a TRUE or FALSE value. The client MUST send false for this parameter, either explicitly or by omitting the parameter.

force: A BOOLEAN value reserved for future use. This parameter MUST be ignored by the server, regardless of a TRUE or FALSE value. The client MUST send FALSE for this parameter, either explicitly or by omitting the parameter.

doc_version: A STRING containing the source control version number of the document being retrieved. An empty string (the default value) is a request for the current version of the document. A client conforming to the FrontPage Server Extensions Remote Protocol MUST omit this parameter or explicitly send the empty string. A server MAY ignore this parameter and always send the most recent version of the document.

get_option: A STRING value that determines how documents are checked out of source control. Passing any string not listed in the following table is considered the same as "none".

Value	Meaning
none	Do not check out the file.
chkoutExclusive	Check out the file exclusively.
chkoutNonExclusive	Check out the file nonexclusively, if the source control system is configured to allow nonexclusive checkouts. If it is not, the server SHOULD treat this as chkoutExclusive instead.<36>

The checkout MUST logically occur before the server begins sending the document to the client. If the checkout cannot occur, the server MUST return an error message with a Status set to one of the error values listed in section 2.2.2.2.17.1 and not return the document.

timeout: An UNSIGNED-INT that specifies the number of minutes the server MUST retain the short-term checkout. To retain the lock longer, the client MUST renew its short-term checkout within this interval. For details, see checkout document, section 3.1.5.3.2. The client MUST NOT send negative values, and the server SHOULD ignore them by not specifying a timeout. The value zero is reserved for server implementations that understand long-term checkout and thus MUST NOT be sent by clients. Servers MAY ignore requests in which this parameter is set to zero by not specifying a timeout. Range from 1 through 4294967296.<37>

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error code 0x0009003E 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

document: The DOCINFO that contains the document name and metadata for the document that has been retrieved.

3.1.5.3.7 get documents

The get documents request is used by the client to retrieve a set of documents for viewing on a client computer.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

url_list: A VECTOR-STRING list of service-relative URLs specifying what documents will be retrieved by the current request.

effective_protocol_version: This parameter has the same semantics as effective_protocol_version in get document.

old_theme_html: This parameter has the same semantics as old_theme_html version in get document.

expandWebPartPages: A BOOLEAN reserved for future use. This parameter MUST be ignored by the server regardless of a TRUE or FALSE value. The client MUST send false for this parameter either explicitly or by omitting the parameter.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server MUST return a multipart/mixed MIME document ([RFC1341]), and the responses MUST be in URL Mode rather than HTML Mode.

Each part of the response MUST be one of the following three types: UrlArgs part, DocInfo part, or DocData part. These types are defined and scoped to this section; they exist merely to provide convenient names for the concepts.

The response MUST begin with a UrlArgs part. After the first part, each part determines the type of the next part as shown in the following illustration.

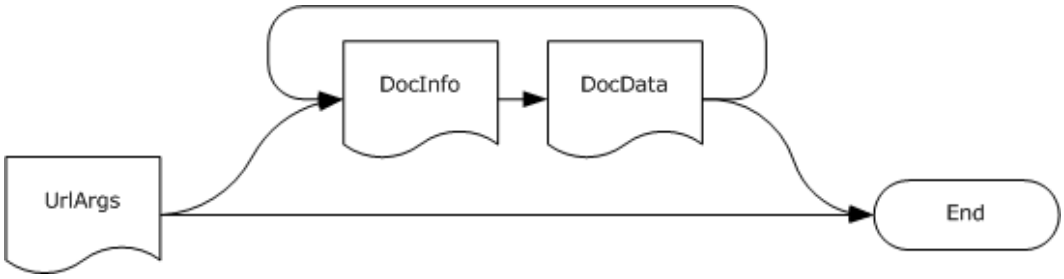

Figure 3: Type encoding sequence for POST

UrlArgs part: The UrlArgs part of the response MUST have a content-type of application/x-www-form-urlencoded and MUST have a METHOD-KEY-VALUE whose REQUEST-NAME-STRING is "get documents" followed by a RET-NAME/RET-VAL pair whose RPC-KEY-STRING is "current_time" and whose RPCVALUE is a TIME.

```
"method" VALSEP "get documents" ":" PROTOCOL-VERSION-STRING
ARGSEP "current_time" TIME LF
```

The UrlArgs part MUST either be followed by a DocInfo part or the response MUST end.

DocInfo part: The DocInfo part MUST be application/x-www-form-urlencoded and be a DOC-INFO-REQUEST. Each DocInfo part MUST be followed by a DocData part.

DocData part: The DocData part MUST have a Content-Type of application/octet-stream. It MUST contain the stream of the document corresponding to the previous DocInfo part. Each DocData part MUST be followed by a DocInfo part or the response MUST terminate.

Note The mf-file-status metadata MUST be added to the METADICT returned in the response. If it is nonzero, the remainder of the response SHOULD be discarded by the client. This key is not considered metadata about the file but rather metadata about the transport, which the client SHOULD examine and SHOULD remove before passing on the METADICT to higher layers.

3.1.5.3.8 list documents

The list documents request is used by the client to request a list of files, folders, and sites contained in a given folder.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

folderList: The folderList parameter is a METADICT containing the names of folders as the keys and the corresponding time stamps of type TIME for the client's cached metadata for those folders as the values. The client SHOULD send the time stamps for the folders' metadata if it has cached this information. If a folder's name and time stamp is missing from the folderList parameter, the server MUST return that folder's contents with all available metadata. Otherwise, the server MUST return all available metadata for only those files, folders, and subsites that have changed since the time stamp recorded in the folder list. For files, folders, and subsites that have not changed, the server SHOULD return an empty METADICT to indicate that the client's cache is still valid. The client MUST interpret an empty METADICT as an indication of validity.

The following example of a value for the *folderList* parameter requests metadata for all files in the root folder of the **Web site** and in the images folder that have a time stamp later than the specified dates and times. For other files in these folders, only the file name and an empty METADICT are included in the response. For all other folders within the initialUrl parameter, all available metadata MUST be returned.

```
[;TX|06 Apr 2006 20:03:02 -0000;images;TX|05 Apr 2006
20:03:02 -0000]
```

initialUrl: A URL-STRING containing the URL of the folder from which documents are listed. This MUST be a service-relative URL.

listBorders: A BOOLEAN value that specifies whether the contents of the shared borders directory that contains shared border pages is to be listed. If TRUE, the contents are listed; if FALSE, they SHOULD NOT be listed. The server SHOULD honor this parameter, but the client MUST NOT rely on it.<38>

listChildWebs: A BOOLEAN value that specifies whether the server response includes the names of the child site folders in the urldirs return value. If TRUE, and the listFolders parameter is also sent as TRUE, the names SHOULD be included; if FALSE, they SHOULD NOT be included. If the client sends this parameter as TRUE, the client SHOULD also send the listFolders parameter as TRUE. The server SHOULD ignore a TRUE value for this parameter if the listFolders parameter is not also TRUE. The server SHOULD honor this parameter, but the client MUST NOT rely on it.<39>

listDerived: A BOOLEAN value that specifies whether the list of files in the **derived documents folder** is included in the response. If TRUE, the list of files SHOULD be included as part of the response; if FALSE, they SHOULD NOT be included. The server SHOULD honor this parameter, but the client MUST NOT rely on it.<40>

listExplorerDocs: A BOOLEAN value that specifies if **task-list files** (_vti_pvt/_x_todo.htm and _vti_pvt/_x_todoh.htm) are listed. If TRUE, the files MAY be listed. If FALSE, they MUST NOT be listed. FrontPage Server Extensions Remote Protocol clients MUST NOT send this parameter; the server SHOULD ignore it.<41>

listFiles: A BOOLEAN value that specifies whether the client requests information about the files in each directory that appears in the response. If TRUE, the server MUST include the document_list return value; if FALSE, the server SHOULD exclude the document_list return value. The server SHOULD use a default value of TRUE for this parameter unless the client specifies otherwise<42>

listFolders: A BOOLEAN value that specifies whether the server response includes the names and metadata of folders under the URL specified by the *initialUrl* parameter. If TRUE, the directory names and metainformation SHOULD be included; if FALSE, they MUST NOT be included. The server SHOULD use a default value of TRUE for this parameter unless the client specifies otherwise.

The server SHOULD honor this parameter, but the client MUST NOT rely on it.

listHiddenDocs: A BOOLEAN value that specifies whether hidden documents in a site are listed. If true, the documents SHOULD be listed; if false, they SHOULD NOT be listed. The server SHOULD honor this parameter, but the client MUST NOT rely on it.

listIncludeParent: A BOOLEAN value that specifies whether an entry for the initialUrl field is included in the server response. If TRUE, the entry SHOULD be included; if FALSE, it SHOULD NOT be included. The server SHOULD honor this parameter, but the client MUST NOT rely on it.

listLinkInfo: For semantics, see section 3.1.5.3.1. The server SHOULD use a default value of TRUE for this parameter unless the client specifies otherwise.

listRecurse: A BOOLEAN value that specifies whether the client requests the server to recursively list the subfolders of folders under the URL specified by the *initialUrl* parameter. If TRUE, the subfolders SHOULD be listed; if FALSE, they SHOULD NOT be listed. The server SHOULD use a default value of TRUE for this parameter unless the client specifies otherwise. If listRecurse is TRUE, all children of that folder are listed; otherwise, only the immediate children are listed. The server SHOULD honor this parameter, but the client MUST NOT rely on it.

listThickets: A BOOLEAN value that specifies whether thicket supporting files and folders are included in the server response. If TRUE, the supporting files and folders SHOULD be included; if FALSE, they SHOULD NOT be included. The server SHOULD use a default value of TRUE for this parameter unless the client specifies otherwise. The server SHOULD honor this parameter, but the client MUST NOT rely on it.

platform: A STRING specifying the operating system of the client. This parameter MUST NOT be passed by clients conforming to the FrontPage Server Extensions Remote Protocol.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

document_list: A DOCUMENT-LIST-RETURN-TYPE that the server SHOULD omit if the *listFiles* parameter was sent as FALSE. If this value is returned, the server MUST list the names and metadata for the requested set of documents specified by the parameters of the request.<43>

bot_list: A DOCUMENT-LIST-RETURN-TYPE return value that MUST NOT be included when the *platform* parameter is not sent. Because clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this parameter, the server need not support this return value.<44>

urldirs: A VECTOR-URL-DIRECTORY containing the names and metadata about folders and root directories of subsites. The server MUST omit this return value if the *listFolders* parameter was sent as FALSE. If this value is returned, the server MUST enumerate the folders and Web sites specified by parameters of the request.

3.1.5.3.9 move document

The move document request is used by the client to change the URL of a selected document or folder in the site. Note that moving a document will change its URL.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

oldUrl: A URL-STRING that specifies the original service-relative URL for a document or folder whose URL is to be changed. The client MUST send this parameter.

newUrl: A URL-STRING that specifies the new service-relative URL for a document or folder whose URL is to be changed. The client MUST send this parameter.

url_list: A VECTOR-URL-STRING list of service-relative URLs of documents whose links ought to be considered for link fixup purposes. This is a hint passed from the client to the server. Servers that implement link fixup SHOULD NOT rely on the client sending the correct list. Clients that conform to the FrontPage Server Extensions Remote Protocol MUST send an empty list (either explicitly or by omitting the parameter and taking the default). Servers MAY ignore this parameter.

rename_option: This tells the server to change certain behaviors during the rename or copy operation, as defined in Rename-Option.<45>

put_option: A set of flags describing how the operation ought to behave as detailed in Put-Option. In particular, the server MUST overwrite an existing file or folder if, and only if, the "overwrite" flag is added.<46>

docopy: A BOOLEAN value specifying whether the move document request copies or moves a file or folder to the destination. If TRUE, the file or folder SHOULD be copied; if FALSE, the file or folder SHOULD be moved. The default value is FALSE.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error codes 0x00020019 and 0x00090002 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

oldUrl: A URL-STRING specifying the former URL for a document or folder whose name or directory has changed.

newUrl: A URL-STRING specifying the new URL for a document or folder whose name or directory has changed.

document_list: A DOCUMENT-LIST-RETURN-TYPE specifying the set of documents whose metadata has changed as a byproduct of the move due to fixing links.

moved_docs: A DOCUMENT-LIST-RETURN-TYPE of URLs and metadata for moved or copied files. Clients MAY assume and servers MUST ensure that this return value is equivalent to the document_list return value of the list documents method (section 3.1.5.3.8) when newUrl is the URL of a folder, assuming that the following three parameter values were passed into list documents (section 3.1.5.3.8).

Parameter values passed into list document:

- *initialUrl*: The value that the client passed to the *newUrl* parameter of the current call to move the documents.
- *listRecurse*: TRUE.
- *includeParent*: TRUE

moved_dirs: A VECTOR-URL-DIRECTORY of URLs and metadata for moved or copied folders. Clients MAY assume and servers MUST ensure that this return value is equivalent to the urldirs return value of the list documents method (section 3.1.5.3.8) when newUrl is the URL of a folder, assuming the following parameter values were passed into list documents (section 3.1.5.3.8).

Parameter values passed into list documents:

- *initialUrl*: The value that the client passed to the *newUrl* parameter of the current call to move the documents.
- *listRecurse*: TRUE.

- *includeParent*: TRUE

As with the preceding moved_docs return value, the client SHOULD use this result to update any metadata cache it is maintaining.

3.1.5.3.10 open service

The open service request is used to provide site metadata to the client computer.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

effective_protocol_version: This parameter has similar semantics to effective_protocol_version in get document. It is used in publishing scenarios in which the client passes the version of the intended target server. A client conforming to this subset of the protocol MUST NOT send this parameter. A server SHOULD ignore this parameter.<47>

URL

FPAuthorScriptUrl

Return Values

service: A SERVICE_RETURN_TYPE specifying the service name and site metadata.

3.1.5.3.11 put document

The put document request is used by the client to write a single file to a directory in an existing site.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

document: A DOCINFO specifying the name and metadata of the file to write to the directory. The client MUST send this parameter.

put_option: A set of flags describing how the operation behaves; see Put-Option for specific semantics for the options.

comment: A STRING that provides a checkin comment for the file being uploaded. The server MUST ignore this parameter unless the "checkin" PUT-OPTION-VAL is specified in the *put_option* parameter. Because clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send the "checkin" PUT_OPTION_VAL, as specified in Put-Option, servers MAY ignore this parameter.

keep_checked_out: A BOOLEAN value used to determine a specified document's behavior in source control. If TRUE, the document SHOULD be checked in to source control and immediately checked back out; if FALSE, the document SHOULD be checked in. The server MUST treat this as equivalent to the "checkout" PUT-OPTION-VAL, as specified in Put-Option. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST send FALSE, either explicitly or by omitting this parameter.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error codes 0x00090002, 0x0009000E, and 0x0009000F 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

document: A DOCINFO containing the name and metadata of the document as it was saved. Note that even though the return value is called "document", it does not contain the document stream. The server MAY update the metadata when the document is saved. The server MUST return the updated metadata in the document return value.

Note The document contents MUST be sent right after the parameters in the client request. Because arguments always end with an LF in URL Mode, there is no need for further delimiters. The document stream cannot be encoded in any way, as whatever is sent over the wire will be directly stored as the file contents, without further translation.

3.1.5.3.12 put documents

The put documents request is used by the client to write multiple files to a site. This request SHOULD be used when a higher level wants to save a document that contains other files (such as graphics) from an application directly to a site directory.

This request is different from other FrontPage Server Extensions Remote Protocol requests because it receives multiple streams. The following details show how the server MUST parse the method.

The HTTP POST MUST be a multipart/mixed MIME document. Each part of the POST MUST be one of the following four types: UrlArgs, DocInfo, DocData, or End. These types are defined and scoped to this section; they exist merely to provide convenient names for the concepts.

The request MUST begin with a UrlArgs part. After the first part, each part determines the type of the next part, as shown in the following illustration.

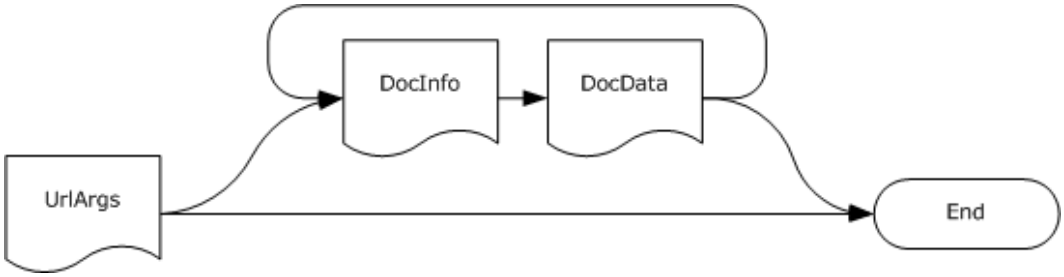

Figure 4: Type encoding sequence for POST

The UrlArgs part MUST have a content-type of application/x-www-form-urlencoded, as specified in [RFC2616] section 14.17, and MUST be parsed using the same rules as a normal method request. The REQUEST-NAME-STRING, as specified in [RFC4234], MUST be put documents and it accepts the list of parameters given in the Tokens section that follows. The server SHOULD fail with error code 0x0004000C if the client is an earlier version than the server supports, just like any other method.

If the next part exists, it MUST be DocInfo part with a content-type of application/x-www-form-urlencoded, as specified in [RFC2616] section 14.17, or it MUST be End with a content-type of text/html.

DocInfo part: The server SHOULD parse the DocInfo part as a DOC-INFO-Request (for details, see DOC-INFO-Request (section 2.2.2.2.15)). The next MIME part MUST be a DocData part.

DocData part: A DocData part can have any content-type and MUST be the stream corresponding to the DOCINFO sent in the prior DocInfo MIME part. If the next part exists, it SHOULD be DocInfo if its content-type is application/x-www-form-urlencoded, as specified in [RFC2616] section 14.17, or SHOULD be End if its content-type is text/html, also specified in [RFC2616].

End part: An End part MUST be ignored by the server. Any subsequent part SHOULD also be considered to be an End part.

Note The server SHOULD accept and ignore a DocInfo/DocData pair if the URL in the DOCINFO is empty. Clients SHOULD avoid doing this because it wastes bandwidth.

If the "atomic" PUT-OPTION-VAL is specified as defined in Put-Option, the server either succeeds in storing all the documents or does not store any of them. If the client does not request the "atomic" option, or if the server does not honor the option, and if the server is unable to store a document, then documents in the request before the one that failed MUST be stored, and documents after the one that failed MUST NOT be stored.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

put_option: A set of flags describing the client's requested behavior. For details, see Put-Option.

time_tokens: This parameter uses a VECTOR-TIME. Each time the value that is specified in the *time_tokens* parameter is checked, in addition to values specified through the vti_timelastmodified metakey and putOption = edit check. If this parameter is sent, it MUST either be empty (which is equivalent to not sending it) or it MUST contain a TIME entry for each document that will be sent in the remainder of the request. In that case, the server SHOULD check this value in addition to the vti_timelastmodified metakey sent in each request.<48>

listFiles: A BOOLEAN value that specifies if the docs return value will be included in the response. The server MUST include the docs return value if, and only if, this parameter is TRUE.

listLinkInfo: See section 3.1.5.3.1.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error codes 0x00090002, 0x0009000E, and 0x0009000F 2.2.2.2.17.1.

docs: A VECTOR-DOCINFO containing information about the saved documents. This value MUST NOT be included in the response if listFiles is passed as FALSE.

error-index: This INT is only returned in error conditions for a single document rather than the transfer in general (for example, because of a time stamp mismatch, a document exists, or a document is checked out). If present, it MUST be the zero-based index of the document that the server was unable to store.

document: This DOC-INFO MUST be returned if, and only if, error-index is also returned. This indicates the document URL and metadata for the document that could not be uploaded.

3.1.5.3.13 remove documents

The remove documents request is used by the client to delete specific documents or folders from the site.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

url_list: A VECTOR-URL-STRING list of service-relative URLs that the client wants to be deleted. The server MUST delete the URLs listed here, subject to authorization checks and the time tokens parameter value.

time_tokens: If present and nonempty, the value lists the vti_timelastmodified for the corresponding documents as known by the client. If the VECTOR-TIME is empty or the parameter is not present, the server MUST ignore this parameter; otherwise, it MAY refuse to delete documents in which the date does not match the actual vti_timelastmodified as known on the server.<49>

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error code 0x00090002 2.2.2.2.17.1.

message: A STRING description of the action taken by the server. This is intended for debugging and SHOULD be ignored by the client.

removed_docs: A VECTOR-DOCINFO containing the name and metadata for the documents that were removed. The server SHOULD send empty METADICTs in this value.<50>

removed_dirs: A VECTOR-DOCINFO containing the name and metadata for the folders that were removed. The server SHOULD send empty METADICTs in this return value.<51>

failed_docs: A VECTOR-DOCINFO specifying the name and metadata for the documents that failed to be removed. The server MUST respond with a (potentially empty) list of documents that could not be removed.

failed_dirs: A VECTOR-URL-DIRECTORY specifying the name and metadata for the folders that failed to be removed. The server MUST respond with a (potentially empty) list of folders that could not be removed.

3.1.5.3.14 server version

The server version request is to be used by the client to request the version of the server extensions in use on the site server<52>.

Tokens

None: The argument list for this request SHOULD be empty. The server MUST ignore any parameters sent.

URL

FPShtmlScriptUrl

Return Values

server version: A VERSION specifying the current version of the server (not the effective protocol version). The server MUST respond with its actual version, which might be larger than the effective protocol version in the PROTOCOL-VERSION-STRING built in to all the FrontPage Server Extensions Remote Protocol responses, as described in section 2.2.2.2.9.

source control: An INT indicates whether the server supports the checkout document and uncheckout document requests. This value MUST equal 0 if the server does not support these requests; otherwise, this value MUST equal 1. Nonzero values other than 1 are reserved, but the client MUST interpret any nonzero value as if it were the value 1.

As with other methods, the effective protocol version negotiated by using the mechanism defined in section 3.1.3.2 SHOULD be returned in the METHOD-KEY-VALUE element of the response.

3.1.5.3.15 uncheckout document

The uncheckout document request is used by the client to reverse a long-term checkout of a file from source control. If the file has changed since it was checked out, those changes are reverted. This request is also used to release a short-term checkout, in which case changes are not reverted. Because clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT assume the server supports long-term checkouts, only the latter use is pertinent.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

document_name: A URL-STRING specifying the service-relative path of the current document.

force: A BOOLEAN value that specifies whether the server reverses the checkout of a file by another user. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT send this parameter as TRUE. The server MAY ignore this value. If the server implements this functionality, it SHOULD do client authorization checks for permissions to perform this action and ignore the parameter if those checks fail.

time_checked_out: A TIME indicating the client's record of the time and date at which the file was last checked out. The server MAY refuse to revert a checkout if the time does not match the server's record of the time the file was checked out.

rlsshortterm: A BOOLEAN value indicates if the client wants to release a short-term checkout or a long-term checkout. If TRUE, the server MUST release the lock; otherwise, the server SHOULD release any long-term checkout the client has acquired. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT assume the server supports long-term checkout and thus MUST send TRUE for this parameter. The server can return an appropriate error if the client does not have the kind of checkout it is trying to undo.

validateWelcomeNames: For semantics, see section 3.1.5.3.1.

URL

FPAuthorScriptUrl

Return Values

The server can return the error code 0x0009000F 2.2.2.2.17.1.

meta_info: The METADICT information about the document that has been unchecked out.

3.1.5.3.16 url to web url

The url to web url request is used by the client to decompose the URL into the server-relative URL of the site that contains the URL and the server-relative URL for the file within the site.

Tokens

service_name: This parameter is deprecated. See service_name in section 3.1.5.3.1.

url: A URL-STRING specifying the server-relative URL of the object that the client wants to decompose.

flags: The flags parameter. An INT MUST be ignored by the server but MAY be sent by the client and SHOULD equal zero.

URL

FPShtmlScriptUrl

Return Values

webUrl: A URL-STRING specifying the server-relative URL of the site.

fileUrl: A URL-STRING specifying the service-relative URL of the file.

3.1.5.3.17 SharePoint Team Services

SharePoint Team Services enables file sharing and discussions among members of a group. SharePoint Team Services also adds collaboration and sharing capabilities to a site. These capabilities allow teams to share and maintain information without the need for central administration of a site. SharePoint Team Services authoring and administration of sites complements the FrontPage Server Extensions Remote Protocol.

As with the FrontPage Server Extensions Remote Protocol, SharePoint methods are remote procedures that are transported in an HTTP request to a server. However, rather than a transport with a POST request, this SHOULD be sent as a GET or a HEAD command with the standard application/x-www-form-urlencoded content type as described in HTML 2.0, [RFC1866] sections 8.2.1 and 8.2.2.

3.1.5.3.17.1 dialogview

The dialogview request is used by a client to obtain an HTML-rendered view of the document libraries within a site, a specific **document library**, or a folder within a document library, which is used in a dialog box for opening or saving files; or opens the property form that is used when saving a file.

Tokens

dialogview: Specifies the view to display. Possible values include the following. <53>

Mask	Meaning
FileOpen	The Open dialog box. The server MUST return an HTML document formatted to render a file open dialog; it SHOULD NOT encourage users to select documents that do not exist.
FileSave	The Save dialog box. The server MUST return an HTML document formatted to render a file save dialog; it SHOULD allow for picking an existing document or creating a new one.
SaveForm	The Property form. To view the property form used when saving a file, the *location* parameter specifies the file in the document library. Clients conforming to the FrontPage Server Extensions Remote Protocol MUST NOT use this value for the dialogview parameter. The server MAY treat this as any other nonsupported value.
All other values	The server MUST return an HTTP 410 (GONE) response.

location: Specifies the site-relative URL of a document library or of a folder or file within a document library. If SaveForm is specified in dialogview, the URL for the *location* parameter MUST point to the file being saved. If the *location* parameter is passed without specifying a value, this method SHOULD display a view of all the document libraries in the site.

FileDialogFilterValue: Specifies the file type extension by which to filter the view in the file dialog box (for example, *.doc, *.txt, or *.htm).

URL

TPScriptURL

Return Values

Success: Displays the view of a document library or folder that is used in a file dialog box, in all document libraries in the site, or in the form used to specify document properties when saving a file.

Error: If the server requires further authentication or authorization, it MUST trigger the HTTP layer to return a 401 message, access denied. The HTTP layer on the client and server then manages authenticating the user. Otherwise, if the request has the *dialogview* parameter but has some other error (for instance, if the *location* or *dialogview* parameters are not valid), the server MUST send a 410 GONE response. The client conforming to the subset of the protocol described here MUST send the *dialogview* parameter to any request to TPScriptURL.

If the result is not an error, the server's response MUST contain an HTML table whose ID is "FileDialogView". Each file that the server wants to show the client MUST be represented as a TR element in the table that has an ID containing the complete URL of the file to be represented and has an expando property fileattribute="file". The server MUST repeat this for each folder, except that it should have the fileattribute="folder" expando property.

If the result is an error, the HTTP return value MUST be 410 (GONE), and the server MAY add contents that, if present, SHOULD be an HTML-formatted error message for display to an end user.

The client SHOULD render the HTML returned by the server, detect mouse clicks and other selection gestures that it wants to respond to, and use the HTML document object model to determine the TR on which the selection occurred. If the user selects a folder, the client SHOULD make a new dialogview request with the location specified by the folder. If the user selects a file, the client SHOULD treat that as the file the user wants to open or save over.

If the client gets an error, it MAY fall back to an alternate dialog method (potentially populated by the list documents method) or present the error to the user.

3.1.6 Timer Events

3.1.6.1 Short-Term Checkout Timer Expiry

When the short-term checkout timer on a document expires, the server MUST clear the short-term checkout on the document. This will leave the document open for editing by any user. If the client wants to prevent short-term checkout from expiring, the client MUST send another checkout document request for the same document before the checkout has expired.

3.1.7 Other Local Events

None.

4 Protocol Examples

The following sections specify protocol examples.

4.1 Example Entry Point for FrontPage Server Extensions

4.1.1 First Determining the Entry Point

Each method specification gives an entry point that corresponds to one of four URLs that are returned when a client performs an HTTP GET on _vti_inf.html. This section details how to determine the URL to POST given the known entry point.

4.1.1.1 First Entry Point Example

If the client wants to call the server version method (section 3.1.5.3.14), it needs to use the FPShtmlScriptUrl entry point. For details, see section 3.1.3. If it is making this call against the root of the server, the URL is as follows.

```
/_vti_bin/shtml.dll/_vti_rpc
```

If the client is making a call against a subsite located at /search/, the URL is as follows.

```
/search/_vti_bin/shtml.dll/_vti_rpc.
```

4.1.1.2 Second Entry Point Example

If the client wants to call the open service method (section 3.1.5.3.10), it needs to use the FPAuthorScriptURL entry point.

```
POST
/site_url/_vti_bin/_vti_aut/author.dll HTTP/1.0
.
.
.
method=open+service:6.0.n.nnnn
```

The first line shows a post to /site_url/_vti_bin/_vti_aut/author.dll, which is the FPAuthorScriptURL entry point for the subsite called 'site_url'.

4.1.2 SharePoint Services Entry Note

The TPScriptUrl field is only present on servers that have SharePoint Services (or SharePoint Team Services) enabled. It is a service-relative URL and refers to the URL to POST for Windows SharePoint Services methods. For details, see section 3.1.5.3.17.

4.2 Example Trace for Posts

The following is an example trace for common operations that are performed on the client. The example shows operations such as opening a Web folder, copying and pasting to (or from) a Web folder, opening a file, saving changes in a file, and closing a file.

Note In the example, WWW-Authenticate headers, as specified in [RFC2616] section 14.47, have been removed, "DOMAIN1" is a placeholder for **domain name**, "testuser" is a placeholder for user name, and "fpseserver" is a placeholder for an actual server name. All the lines, except for any text files that are uploaded, are to be terminated by "\n" rather than the "\r\n", which is standard on Windows operating systems.

Note Many of the examples below use URL encoding as specified in [RFC3986] section 2.1.

4.2.1 Querying for URLs to Post

Any user action requiring the client to interact with the server through FrontPage Server Extensions requires the client to know what URLs to POST. Consequently, any FrontPage Server Extensions Remote Protocol conversation will begin with the client posting an HTTP GET to /_vti_inf.html to determine the URLs of author.dll, shtml.dll (for details, see section 3.1.3.2.1), and so on.

4.2.1.1 Client HTTP GET Request for _vti_inf.html

```
GET /_vti_inf.html HTTP/1.1
Date: Thu, 08 June 2006 21:39:52 GMT
MIME-Version: 1.0
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MS FrontPage 12.0)
Host: fpseserver
Accept: auth/sicily
Content-Length: 0
Connection: Keep-Alive
Cache-Control: no-cache
```

4.2.1.2 Server HTTP Response

```
HTTP/1.1 200 OK
Content-Length: 1754
Content-Type: text/html
Last-Modified: Thu, 08 June 2006 21:04:13 GMT
Accept-Ranges: bytes
ETag: "2f7ad4cbe6fc61:33a"
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Date: Thu, 08 June 2006 21:39:42 GMT

<html>

<head>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
<title> FrontPage Configuration Information </title>
</head>

<body>
<!-- _vti_inf.html version 0.100>
<!--
This file contains important information used by the FrontPage client
(the FrontPage Explorer and FrontPage Editor) to communicate with the
FrontPage server extensions installed on this web server.
```

```
The values below are automatically set by FrontPage at installation.
Normally, you do not need to modify these values, but in case
you do, the parameters are as follows:

'FPShtmlScriptUrl', 'FPAuthorScriptUrl', and 'FPAdminScriptUrl'
specify the relative urls for the scripts that FrontPage uses for
remote authoring.  These values SHOULD NOT be changed.

'FPVersion' identifies the version of the FrontPage Server Extensions
installed, and SHOULD NOT be changed.
--><!-- FrontPage Configuration Information
    FPVersion="5.0.2.6738"
    FPShtmlScriptUrl="_vti_bin/shtml.dll/_vti_rpc"
    FPAuthorScriptUrl="_vti_bin/_vti_aut/author.dll"
    FPAdminScriptUrl="_vti_bin/_vti_adm/admin.dll"
    TPScriptUrl="_vti_bin/owssvr.dll"
-->
<p><!--webbot bot="PurpleText"
preview="This page is placed into the root directory of your FrontPage
web when FrontPage is installed.  It contains information used by the
FrontPage client to communicate with the FrontPage Server Extensions
installed on this web server.  You SHOULD NOT delete this file."
--></p>

<h1>FrontPage Configuration Information </h1>

<p>In the HTML comments, this page contains configuration
information that the FrontPage Explorer and FrontPage Editor need to
communicate with the FrontPage Server Extensions installed on
this web server. Do not delete this page.</p>
</body>
</html>
```

4.2.2 Opening a Web Folder

This example uses the server version request (section 3.1.5.3.14) to query the server's version and uses the list documents request (section 3.1.5.3.8) to enumerate the documents in the root of the server. This part of the example corresponds to opening a folder as a Web folder in a Web browser.

4.2.2.1 Client Calls server version Method

```
POST /_vti_bin/shtml.dll/_vti_rpc HTTP/1.1
Date: Thu, 08 June 2006 21:39:52 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 42
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=server+version%3a12%2e0%2e0%2e3417
```

4.2.2.2 Server Responds to server version Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:39:42 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=server version:5.0.2.6738
<p>server version=
<ul>
<li>major ver=5
<li>minor ver=0
<li>phase ver=2
<li>ver incr=6738
</ul>
<p>source control=1
</body>
</html>
```

4.2.2.3 Client Calls list documents Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:01 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 336
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=list+documents%3a5%2e0%2e2%2e6738&service%5fname=
&listHiddenDocs=false&listExplorerDocs=false&listRecurse=
false&listFiles=true&listFolders=true&listLinkInfo=
false&listIncludeParent=true&listDerived=false&listBorders=
false&listChildWebs=true&listThickets=true&initialUrl=&folderList=
%5b%3bTW%7c08+June+2006+21%3a04%3a14+%2d0000%5d
```

4.2.2.4 Server Responds to list documents Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:39:51 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc
```

```
<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=list documents:5.0.2.6738
<p>document_list=
<ul>
<ul>
<li>document_name=Thicket test.htm
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:28:52 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:27:31 -0000
<li>vti_title
<li>SW|Test
<li>vti_nexttolasttimemodified
<li>TR|08 June 2006 21:28:39 -0000
<li>vti_filesize
<>IR|930
<li>vti_metatags
<li>VR|HTTP-EQUIV&#61;Content-Type text/html&#59;&#92;charset&#61;
windows-1252 Generator Microsoft&#92; Word&#92; 12&#92; (filtered)
<li>vti_charset
<li>SR|windows-1252
<li>vti_generator
<li>SR|Microsoft Word 12 (filtered)
<li>vti_timelastwritten
<li>TX|08 June 2006 21:28:52 -0000
</ul>
</ul>
</ul>
<p>urldirs=
<ul>
<ul>
<li>url=
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|true
<li>vti_dirlateststamp
<li>TW|08 June 2006 21:28:52 -0000
</ulul>
</ul>
<ul>
<li>url=aspnet_client
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
```

```
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|false
<li>vti_hassubdirs
<li>BR|true
</ul>
</ul>
<ul>
<li>url=images
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
<ul>
<li>url=Thicket Test_files
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|true
<li>vti_isscriptable
<li>BR|true
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
<ul>
<li>url=_private
<li>meta_info=
<ul>
<li>vti_isexecutable
<li>BR|false
<li>vti_isbrowsable
<li>BR|false
<li>vti_isscriptable
<li>BR|false
<li>vti_hassubdirs
<li>BR|false
</ul>
</ul>
</ul>
</body>
</html>
```

4.2.3 Copying a File to a Web Folder

This example uses the url to web url request (section 3.1.5.3.16) to discover where a file (in this case, /small.txt) belongs and uses the put document request (section 3.1.5.3.11) to upload it. This part of the example corresponds to a copy/paste operation into the Web folder.

4.2.3.1 Client Calls url to web url Method

```
POST /_vti_bin/shtml.dll/_vti_rpc HTTP/1.1
Date: Thu, 08 June 2006 21:40:17 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 68
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=url+to+web+url%3a5%2e0%2e2%2e6738&url=%2fsmall%2etxt&flags=0
```

4.2.3.2 Server Responds to url to web url Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:07 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=url to web url:5.0.2.6738
<p>webUrl=/
<p>fileUrl=small.txt
</body>
</html>
```

4.2.3.3 Client Calls put document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:17 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 224
Content-Type: application/x-vermeer-urlencoded
X-Vermeer-Content-Type: application/x-vermeer-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=put+document%3a5%2e0%2e2%2e6738&service%5fname=&document=%5b
document%5fname%3dsmall%2etxt%3bmeta%5finfo%3d%5b%5d%5d&put%5foption
```

```
=edit%2catomic%2cthicket&comment=&keep%5fchecked%5fout=false
This is a text file.
```

4.2.3.4 Server Responds to put document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:07 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=put document:5.0.2.6738
<p>message=successfully put document 'small.txt' as 'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti_backlinkinfo
<li>VX|
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
</ul>
</ul>
</body>
</html>
```

4.2.4 Downloading a File from a Web Folder

This example uses the get document request (section 3.1.5.3.6) to download the file. This part of the example corresponds to a copy/paste operation from the Web folder.

4.2.4.1 Client Calls get document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:40:30 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 162
Content-Type: application/x-www-form-urlencoded
```

```
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=get+document%3a5%2e0%2e2%2e6738&service%5fname=&document
%5fname=small%2etxt&old%5ftheme%5fhtml=false&force=true&get
%5foption=none&doc%5fversion=&timeout=0
```

4.2.4.2 Server Responds to get document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:40:20 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=get document:5.0.2.6738
<p>message=successfully retrieved document 'small.txt' from
'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti_backlinkinfo
<li>VX|
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
</ul>
</ul>
</body>
</html>
This is a text file.
```

4.2.5 Opening a File in a Web Folder

When opening a file, as seen in the next part of the example, a client application calls the get document request (section 3.1.5.3.6) with a time-out of a 10-minute short-term checkout, as can be seen at the end of the get document request, in the section that follows. This guarantees that the document cannot be modified by other users while it is open in the client application.

4.2.5.1 Client Calls get document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:45 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 175
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=get+document%3a5%2e0%2e2%2e6738&service%5fname=&document%5f
name=small%2etxt&old%5ftheme%5fhtml=false&force=false&get%5foption=
chkoutExclusive&doc%5fversion=&timeout=10
```

4.2.5.2 Server Responds to get document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:35 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=get document:5.0.2.6738
<p>message=successfully retrieved document 'small.txt' from
'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|28
<li>vti_backlinkinfo
<li>VX|
<li>vti_sourcecontrollockexpires
<li>TR|08 June 2006 21:51:35 -0000
<li>vti_sourcecontrolcheckedoutby
<li>SR|DOMAIN1&#92;testuser
<li>vti_sourcecontrolmultiuserchkoutby
<li>VR|DOMAIN1&#92;&#92;testuser
<li>vti_timelastwritten
<li>TX|08 June 2006 21:40:07 -0000
```

```
</ul>
</ul>
</body>
</html>
This is a small text file.
```

4.2.6 Saving a File to a Web Folder

Changing and saving a file, as seen in the next part of the example, requires calling the put document request (section 3.1.5.3.11).

4.2.6.1 Client Calls put document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:57 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 290
Content-Type: application/x-vermeer-urlencoded
X-Vermeer-Content-Type: application/x-vermeer-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=put+document%3a5%2e0%2e2%2e6738&service%5fname=&document=%5
bdocument%5fname%3dsmall%2etxt%3bmeta%5finfo%3d%5bvti%5ftimelastmodi
fied%3bTW%7c08+June+2006+21%3a40%3a07+%2d0000%5d%5d&put%5foption=edi
t&comment=&keep%5fchecked%5fout=false
This is a small text file. Now, a little bigger.
```

4.2.6.2 Server Responds to put document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:47 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc


<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=put document:5.0.2.6738
<p>message=successfully put document 'small.txt' as 'small.txt'
<p>document=
<ul>
<li>document_name=small.txt
<li>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
```

```
<li>vti_timelastmodified
<li>TR|08 June 2006 21:41:47 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_backlinkinfo
<li>VX|
<li>vti_sourcecontrollockexpires
<li>TR|08 June 2006 21:51:35 -0000
<li>vti_sourcecontrolcheckedoutby
<li>SR|DOMAIN1&#92;testuser
<li>vti_sourcecontrolmultiuserchkoutby
<li>VR|DOMAIN1&#92;&#92;testuser
<li>vti_nexttolasttimemodified
<li>TW|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|51
<li>vti_timelastwritten
<li>TX|08 June 2006 21:41:47 -0000
</ul>
</ul>
</body>
</html>
```

4.2.7 Closing a File

Finally, this example shows what happens when the file is closed in the client application, which requires a call to the uncheckout document request (section 3.1.5.3.15) to release the lock. Note that the example does not illustrate the effects of waiting 10 minutes to cause the client application to renew the short-term checkout, which would have caused a checkout document request (section 3.1.5.3.2) to be sent with a *timeout* parameter.

4.2.7.1 Calls uncheckout document Method

```
POST /_vti_bin/_vti_aut/author.dll HTTP/1.1
Date: Thu, 08 June 2006 21:41:59 GMT
MIME-Version: 1.0
User-Agent: MSFrontPage/12.0
Host: fpseserver
Accept: auth/sicily
Content-Length: 120
Content-Type: application/x-www-form-urlencoded
X-Vermeer-Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Cache-Control: no-cache

method=uncheckout+document%3a5%2e0%2e2%2e6738&service%5fname=
&document%5fname=small%2etxt&force=false&rlsshortterm=true
```

4.2.7.2 Server Responds to uncheckout document Method

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 08 June 2006 21:41:49 GMT
Server: Microsoft-IIS/6.0
```

```
X-Powered-By: ASP.NET
MicrosoftOfficeWebServer: 5.0_Pub
Content-type: application/x-vermeer-rpc

<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=uncheckout document:5.0.2.6738
<p>meta_info=
<ul>
<li>vti_author
<li>SR|DOMAIN1&#92;testuser
<li>vti_modifiedby
<li>SR|DOMAIN1&#92;testuser
<li>vti_timelastmodified
<li>TR|08 June 2006 21:41:47 -0000
<li>vti_timecreated
<li>TR|08 June 2006 21:40:07 -0000
<li>vti_backlinkinfo
<li>VX|
<li>vti_nexttolasttimemodified
<li>TW|08 June 2006 21:40:07 -0000
<li>vti_filesize
<li>IR|51
<li>vti_timelastwritten
<li>TX|08 June 2006 21:41:47 -0000
</ul>
</body>
</html>
```

5 Security

5.1 Security Considerations for Implementers

5.1.1 One-Click Attacks

It is possible for an attacker to lure a user to a malicious page, such as by sending the user a URL in e-mail. When the user visits the malicious page, that page can perform a silent POST to the server. Because the FrontPage Server Extensions Remote Protocol is merely an HTTP POST, this means the attacker can lure the user into performing any FrontPage Server Extensions Remote Protocol operation against any server. This sort of attack is termed a one-click attack.

To prevent this type of attack, servers can require all incoming FrontPage Server Extensions Remote Protocol requests to have the HTTP header X-Vermeer-Content-Type, as specified in [RFC2616] section 14.17. Because normal Web browsers do not send this header, requiring it effectively prevents users from browsing to a page that can execute a silent FrontPage Server Extensions Remote Protocol method call. It is strongly recommended that all implementations of the FrontPage Server Extensions Remote Protocol require this header to prevent one-click attacks.

5.1.2 Permissions for Entry Points

Servers have traditionally restricted access to methods to certain classes of users. Although this restriction is not required by the FrontPage Server Extensions Remote Protocol, it is recommended because some methods, such as remove documents (section 3.1.5.3.13) can be damaging to user data.

The FrontPage Server Extensions Remote Protocol has traditionally determined which users can call which methods based on the method entry points. Methods whose entry point is FPShtmlScriptUrl can usually be called by any user. Methods with the FPAuthorScriptURL entry point are restricted to users who can read or write documents on the server. The reason for this model is that methods such as remove documents (section 3.1.5.3.13) are considered more dangerous than server version (section 3.1.5.3.14). As such, restricting unauthenticated users from even calling the more powerful methods provides an extra layer of security.

Implementers of the FrontPage Server Extensions Remote Protocol are free to restrict method entry point security if they choose to, or they can rely on the object permissions discussed as follows.

5.1.3 Permissions for Objects

Like most file systems, FrontPage Server Extensions Remote Protocol objects can have security. The granularity of this security is up to the server implementers. Windows implementations of the FrontPage Server Extensions Remote Protocol provide the capability to granularly control read access and write access on files, folders, and services. On a secured server, each method call checks the appropriate rights before executing. If the user does not have sufficient rights, the implementation then triggers the HTTP layer to return a 401 message, access denied. The HTTP layer on the client and server then manages authenticating the user, if that user does in fact have permissions.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft Windows NT® operating system
- Microsoft Windows® 2000 operating system
- Windows® XP operating system
- Windows Server® 2003 operating system
- Windows Vista® operating system
- Windows® 7 operating system
- Windows Server® 2008 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

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

<1> Section 2.2.2.2.17.1: Windows can present this error to the user as a disk full error.

<2> Section 2.2.2.2.17.1: Windows clients might interpret this as a "not found" error.

<3> Section 2.2.2.2.17.1: Some Windows clients explicitly ignore this error because it can be expected in certain cases; for example, caching operations or "most recently used" lists.

<4> Section 2.2.2.2.17.1: Windows clients generally ignore this error during bulk operations. This error is also generally ignored when removing files.

<5> Section 2.2.2.2.17.1: Windows will display an informational message rather than an error in this case.

<6> Section 2.2.2.2.18: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 ignore this parameter; versions 4.0 and 5.0 do not.

<7> Section 2.2.2.2.18: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 do not ignore this parameter; all other versions ignore it.

<8> Section 2.2.2.2.18: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 accept this parameter and require the requesting user to be a site administrator.

<9> Section 2.2.2.2.18: The FrontPage Server Extensions Remote Protocol versions 4.0 and 5.0 accept this parameter if source control is enabled.

<10> Section 2.2.2.2.19: The FrontPage Server Extensions:Website Management protocol versions 4.0 and 5.0 accept this parameter.

<11> Section 2.2.4.5: Version 5.0 and 12.0 servers do not include vti_dirlateststamp key for folders in create url-directory and remove documents (section 3.1.5.3.13) methods.

<12> Section 2.2.4.12: FrontPage Server Extensions Remote Protocol Server versions 4.0 and 5.0 set a value here that reflects the underlying file system it stores documents in. Versions 6.0 and 12.0 always send 1 for this value.

<13> Section 2.2.4.13: The Windows client uses this metadata to avoid fetching the content of the file just to discover metatags with NAME="progid" and NAME="generator"; these are used to display icons for HTML files and to select an appropriate editor.

<14> Section 2.2.4.14: The FrontPage Server Extensions Remote Protocol version 5.0 does not return this key.

<15> Section 2.2.4.16: The FrontPage Server Extensions Remote Protocol version 5.0 does not include vti_sourcecontroltimecheckedout (section 2.2.4.16) metadata in the return values, when using checkout document (section 3.1.5.3.2) method or get document (section 3.1.5.3.6) method to perform a short-term checkout.

<16> Section 2.2.4.17: The FrontPage Server Extensions Remote Protocol version 5.0 doesn't return vti_thicketdir as a BOOLEAN flag for a folder if the folder contains the supporting files for a thicket, but the service-relative URLs of the corresponding files.

<17> Section 2.2.4.19: The FrontPage Server Extensions Remote Protocol version 5.0 does not include this key.

<18> Section 2.2.4.19: Version 5.0 and 12.0 servers do not include vti_timecreated key for folders in create url-directory and remove documents (section 3.1.5.3.13) methods.

<19> Section 2.2.4.20: Version 5.0 and 12.0 servers do not include vti_timelastmodified key for folders in create url-directory and remove documents (section 3.1.5.3.13) methods.

<20> Section 2.2.4.20: The FrontPage Server Extensions Remote Protocol client for versions 4.0, 5.0, 6.0, and 12.0 uses this value when rendering a file's or folder's time last modified.

<21> Section 2.2.4.21: The FrontPage Server Extensions Remote Protocol Server version 5.0 does not include this key for folders. The FrontPage Server Extensions Remote Protocol Server versions 4.0, 6.0, and 12.0 do not include this key. The FrontPage Server Extensions Remote Protocol Client versions 4.0, 5.0, and 6.0 are not sensitive to this key.

<22> Section 2.2.4.22: The FrontPage Server Extensions Remote Protocol versions 5.0 and 12.0 do not return this metakey.

<23> Section 2.2.4.23: FrontPage Server Extensions Remote Protocol Server versions 4.0 and 5.0 do not include this key, versions 6.0 and 12.0 do. FrontPage Server Extensions Remote Protocol Client versions 4.0 and 5.0 only consider information passed to the underlying HTTP authentication mechanism.

<24> Section 3.1.1.1: The FrontPage Server Extensions Remote Protocol versions 4.0 and 5.0 allow users to turn the source control sandbox off; versions 6.0 and 12.0 do not.

<25> Section 3.1.2.1: All Windows operating systems request a short-term checkout length of two minutes. The clients will attempt to renew the short-term checkout 10 seconds before it expires.

<26> Section 3.1.3.2.1: Windows Vista does not perform this GET and instead assumes the values shown in the example in section 3.1.3.2.1.

<27> Section 3.1.5.1: The FrontPage Server Extensions Remote Protocol versions 4.0, 5.0, and 12.0 do not fail the request and instead use the normal HTTP Content-Type header. This action has potential security implications (see section 5.1).

<28> Section 3.1.5.1: If the client does not include FrontPage in its User-Agent string, Windows server will respond with the HTTP Content-Type as "text/html" and present more simplistic error strings.

<29> Section 3.1.5.2: Versions 4.0, 5.0, 6.0, and 12.0 of the FrontPage Server Extensions Remote Protocol server will treat unknown arguments as a syntax error if the method takes any parameters. For methods that take no parameters, such as server version (section 3.1.5.3.14), the FrontPage Server Extensions Remote Protocol server will ignore the parameters.

<30> Section 3.1.5.2: The FrontPage Server Extensions Remote Protocol server versions 5.0, 6.0, and 12.0 will erroneously return a badly formed response message body that is not compliant with [RFC2616] for method calls made without authentication that result in an HTTP 401 error response. FrontPage Server Extensions Remote Protocol server version 4.0 does not have this defect and does not emit a response message body with an HTTP 401 error response.

All of the methods listed in section 3.1.5.3 are known to have the defect in FrontPage Server Extensions Remote Protocol server versions 5.0, 6.0, and 12.0 when returning an HTTP 401 response, with the exceptions of get documents, server version, url to web url, and dialogview.

This is an example of the badly formed message body returned by version 5.0 of FrontPage Server Extensions from a typical method call made without authentication, in this case, "open service".

```
<html dir="ltr">
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" name="CharsetDefinition">
</HEAD><body ID=idErr><p><H2>You are not authorized to view this page</H2></p>

<p>You do not have permission to view this page using your current user account.<br>
Please try the following:<br>
<li>If you have another user account with a higher level of permission, click <br>
your browser's Back button to try again using that account.
</li><li>If you believe you should be able to view this page, contact the Web site
administrator.</li></p>

</body></html>
<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=open service:5.0.2.6738
<p>status=
<ul>
<li>status=917505
<li>osstatus=0
<li>msg=The user 'DOMAIN&#92;username' is not authorized to execute the 'Author Pages'
method.
<li>osmsg=
</ul>
</body>
</html>
```

This is an example of the badly formed message body returned by version 12.0 of the FrontPage Server Extensions Remote Protocol; the result from version 6.0 is identical except for the version number reported in the method=open service line.

```
<html dir="ltr">
<HEAD>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" name="CharsetDefinition">
</HEAD><body ID=idErr><p><H2>Access denied.</H2></p>

<p>You do not have permission to perform this action or access this resource.</p>

<!-- commentElt Access denied. --></body></html>
<html><head><title>vermeer RPC packet</title></head>
<body>
<p>method=open service:12.0.0.4518
<p>status=
<ul>
<li>status=917556
<li>osstatus=0
<li>msg=You are not authorized to execute this operation.
<li>osmsg=
</ul>
</body>
</html>
```

The response message body created by the FrontPage Server Extensions Remote Protocol server software exhibiting this defect is badly formed due to the presence of two separate <HTML> sections, which can cause unexpected behavior in an insufficiently robust client that attempts to render or otherwise make use of the body.

All existing FrontPage Server Extensions Remote Protocol clients ignore the message body, if any, returned with an HTTP 401 response. Since an update or future version of FrontPage Server Extensions Remote Protocol server might correct this defect, clients MUST ignore the message body.

<31> Section 3.1.5.3.1: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 servers ignore this parameter. Versions 4.0 and 5.0 respond to the parameters with a rpc_stats return value with information about how much time the command spent running, waiting, and so on. The information returned is only useful for debugging purposes and is not considered part of the protocol.

<32> Section 3.1.5.3.2: The FrontPage Server Extensions Remote Protocol version 5.0 server does not return error (0x0009000E) if the force bit1 is not set to Refresh short-term checkout, and a short-term checkout exists.

<33> Section 3.1.5.3.2: The FrontPage Server Extensions Remote Protocol versions 5.0 will instead throw a status error stating, "Illegal parameter value for parameter 'timeout'".

<34> Section 3.1.5.3.4: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 do not have the notion of an executable directory. Version 4.0 servers create a server directory with execution set to "Scripts and Executables".

<35> Section 3.1.5.3.5: The FrontPage Server Extensions Remote Protocol server versions 5.0, 6.0 and 12.0 ignore the listHiddenDocs parameter value and always return information for hidden documents.

<36> Section 3.1.5.3.6: The FrontPage Server Extensions Remote Protocol version 5.0 returns a status error when get_option is "chkoutNonExclusive" and does not checkout the file exclusively.

<37> Section 3.1.5.3.6: The FrontPage Server Extensions Remote Protocol versions 5.0 will instead throw a status error stating, "Illegal parameter value for parameter 'timeout'".

<38> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol server version 12.0 does not return the contents of the shared borders directory when listBorders (section 3.1.5.3.8) is true.

<39> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol server version 12.0 includes the child site folders in the urldirs return value when listChildWebs (section 3.1.5.3.8) is false.

<40> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol clients send listDerived=false in the request and do not request the contents of a _derived folder. The FrontPage Server Extensions Remote Protocol server version 12.0 does not return the list of files when listDerived (section 3.1.5.3.8) is true.

<41> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol clients do not send this value; Version 6.0 and 12.0 servers will ignore this value if it is received. Version 5.0 servers do not ignore this value and return the task-list files if they exist.

<42> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol version 12.0 always returns the document information and metadata even *listFiles* is FALSE.

<43> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol version 5.0 doesn't omit document_list, but returns an empty DOCUMENT-LIST-RETURN-TYPE when *listFiles* is set as FALSE.

<44> Section 3.1.5.3.8: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 servers return an empty DOCUMENT-LIST-RETURN-TYPE if the platform parameter is sent. The FrontPage Server Extensions Remote Protocol versions 4.0 and 5.0 enumerate a folder (not accessible in the URL namespace) that the platform parameter identifies. The default configuration has folders named "WinI386" and "all". Server administrators can install modules into these folders or create new sibling directories.

<45> Section 3.1.5.3.9: The FrontPage Server Extensions Remote Protocol version 12.0 move document (section 3.1.5.3.9) method does not create a missing parent folder when rename_option sets createdir.

<46> Section 3.1.5.3.9: The FrontPage Server Extensions Remote Protocol versions 5.0 and 12.0 move document (section 3.1.5.3.9) method does not create the folder when put_option sets createdir.

<47> Section 3.1.5.3.10: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 ignore this parameter.

<48> Section 3.1.5.3.12: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 do not support this behavior; however, versions 4.0 and 5.0 do.

<49> Section 3.1.5.3.13: The FrontPage Server Extensions Remote Protocol versions 6.0 and 12.0 do not support this behavior; however, versions 4.0 and 5.0 do.

<50> Section 3.1.5.3.13: The FrontPage Server Extensions Remote Protocol versions 5.0, 6.0, and 12.0 send an empty METADICTs, and version 4.0 send the METADICT that existed before the document was deleted.

<51> Section 3.1.5.3.13: The FrontPage Server Extensions Remote Protocol version 6.0 send an empty METADICTs, and versions 4.0, 5.0, and 12.0 send the METADICT that existed before the document was deleted.

<52> Section 3.1.5.3.14: The following table specifies which version of the FrontPage Server Extensions Remote Protocol is contained in each version of Windows.

Revision summary	In-box FPSE server	Other supported FPSE servers
Windows NT 3.51	None	None
Windows NT 4.0	None	None
Windows XP (x86)	4.0	None
Windows XP 64-Bit Edition	None	None
Windows Server 2003 (x86)	5.0	5.0, 6.0, 12.0
Windows Server 2003 (x64)	None	6.0, 12.0
Windows Server 2003 R2	Microsoft FrontPage Server Extensions 5.0, Windows SharePoint Services 3.0	5.0, 6.0, 12.0
Windows Vista	None	None
Windows Server 2008	12.0	5.0

<53> Section 3.1.5.3.17.1: Windows Vista only supports the Windows SharePoint Services dialogview aspect of the FrontPage Server Extensions Remote Protocol. All other aspects of the protocol are deprecated in favor of WebDAV. For more information about WebDAV, see [MS-WDV].

7 Change Tracking

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

8 Index

U

V