[MS-INFODCF]:

InfoPath Data Connection File Download Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	New	Initial Availability
6/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
7/13/2009	1.02	Major	Revised and edited the technical content
8/28/2009	1.03	Editorial	Revised and edited the technical content
11/6/2009	1.04	Editorial	Revised and edited the technical content
2/19/2010	2.0	Minor	Updated the technical content
3/31/2010	2.01	Editorial	Revised and edited the technical content
4/30/2010	2.02	Minor	Updated the technical content
6/7/2010	2.03	Editorial	Revised and edited the technical content
6/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.04	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.5	Minor	Clarified the meaning of the technical content.
4/11/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.5	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.5.1	Editorial	Changed language and formatting in the technical content.
2/11/2013	2.5.1	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.6	Minor	Clarified the meaning of the technical content.
11/18/2013	2.6	None	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	2.6	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
4/30/2014	2.6	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	3.0	Major	Significantly changed the technical content.
10/30/2014	3.0	None	No changes to the meaning, language, or formatting of the technical content.
2/26/2016	4.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The InfoPath Data Connection File Download Protocol enables a protocol client to download information defining the connection parameters for a specific remote data store.

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

1.1 Glossary

This document uses the following terms:

200 OK: A response to indicate that the request has succeeded.

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

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

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

authorization: The secure computation of roles and accesses granted to an identity.

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

failure response: An **HTTP** response where the value of the **Status-Code** element is 4xx or 5xx, as described in [RFC2616].

form definition (.xsf) file: An XML file with an .xsf file name extension. The file contains information about the files and components that are used within a form, including user interface customizations, XML schemas, views, business logic (1), events (2), and deployment settings.

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

HTTP method: In an HTTP message, a token that specifies the method to be performed on the resource that is identified by the **Request-URI**, as described in [RFC2616].

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

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

message body: The content within an HTTP message, as described in [RFC2616] section 4.3.

path component: Data that identifies a resource within the scope of a scheme and authority in a **URI**, as described in [RFC3986].

path segment: A portion of a **URI**, as described in [RFC3986]. See also **path component**.

query component: A portion of a **URL** that follows a question mark (?), as described in [RFC3986].

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

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

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

Status-Line: The first line of an HTTP response message, as described in [RFC2616].

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

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

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

UTF-8: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

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

1.2 References

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

1.2.1 Normative References

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

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

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

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

[RFC1945] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996, http://www.ietf.org/rfc/rfc1945.txt

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

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

[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

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

1.2.2 Informative References

[MSDN-UDCV2] Microsoft Corporation, "Universal Data Connection v2.0 Reference and Schema", http://msdn.microsoft.com/en-us/library/ms772017.aspx

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

1.3 Overview

This protocol provides a method for protocol clients to request a **Universal Data Connection (.udc, .udcx) file** in the format described by [MS-UDCX]. This method requires the protocol client to have a **form template (.xsn) file** that uses information in the requested .udc file to connect to a **data source**. The protocol client provides query parameters in the **query component** submitted to the protocol server to identify the requested .udc file and the form template (.xsn) file that contains a reference to this .udc file.

This protocol uses the **Hypertext Transfer Protocol (HTTP)** for transport. The protocol client can use the **GET HTTP method** to download the file, or the **HEAD** HTTP method to determine whether the file exists without downloading it.

A typical scenario for using this protocol would be to access a .udc file that is used by several different form template (.xsn) files that are located on different **sites**. In this scenario, the protocol server can restrict access to the .udc file by verifying the protocol client has **authorization** to use a form template (.xsn) file that contains a reference to this .udc file.

For more information about using the .udc file format, see [MSDN-UDCV2].

1.4 Relationship to Other Protocols

For message transport, this protocol uses the HTTP/1.0 protocol, as described in [RFC1945], the HTTP/1.1 protocol, as described in [RFC2616], or the **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)** protocol, as described [RFC2818].

The following diagram shows the transport stack that this protocol uses:

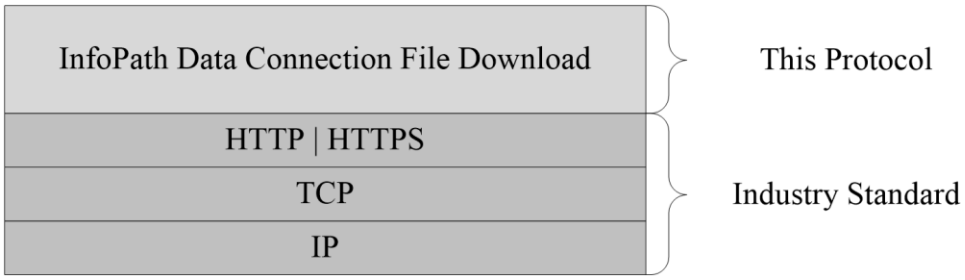

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a site that is identified by a **Uniform Resource Locator (URL)** that is known by protocol clients. The protocol server endpoint is formed by appending "/_layouts/GetDataConnectionFile.aspx " to the URL of the site. For example, a URL could be "http://www.contoso.com/Repository/_layouts/GetDataConnectionFile.aspx".

This protocol assumes that **authentication** has been performed by the underlying protocols.

This protocol assumes that both the protocol client and protocol server have copies of a form template (.xsn) file resource. This protocol does not specify how the protocol client and protocol server obtain their respective copies of this resource.

1.6 Applicability Statement

This protocol is appropriate for providing protocol clients access over HTTP or HTTPS to a .udc file when both the following conditions apply:

- the protocol client is requesting the .udc file because it is used by a form template (.xsn) file that the protocol server also has a copy of.
- the .udc file is referenced by a **connectoid** element in the form template (.xsn) file, as described in [MS-IPFF] section 2.2.147.30 and [MS-IPFF2] section 2.2.2.2.23, with a **connectionLinkType** attribute equal to "store".
- the .udc file is in the format described by [MS-UDCX].

1.7 Versioning and Capability Negotiation

This protocol uses multiple transports with HTTP, as described in section 2.1.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers MUST support HTTP. Protocol servers SHOULD additionally support HTTPS for securing communication with clients.

2.2 Message Syntax

All messages in this protocol MUST be valid HTTP requests and responses, as specified in [RFC1945] or [RFC2616]. The HTTP version, as specified in [RFC1945] or [RFC2616] section 3.1, MUST be either "HTTP/1.0" or "HTTP/1.1". The following subsections detail the relevant portions of HTTP request and response messages.

2.2.1 Request Syntax

2.2.1.1 Request HTTP Method

The protocol client MUST use the **GET** or **HEAD** HTTP methods specified in [RFC1945], section 8 or [RFC2616], section 9. The protocol server determines whether to return a **message body** in the response based on the HTTP method, as specified section 2.2.2.

2.2.1.2 Request-URI Syntax

The **Request-URI** sent in the HTTP request MUST be a valid **Uniform Resource Identifier (URI)**, as specified in [RFC3986]. The **path component** of the Request-URI MUST end with "/_layouts/GetDataConnectionFile.aspx". The following **Augmented Backus-Naur Form (ABNF)** specifies the syntax that the path component MUST adhere to, using the notation specified in [RFC5234]. The ABNF rules **path-absolute** and **path-rootless** are defined in [RFC3986] section 3.3.

```
path    = [base]"/_layouts/GetDataConnectionFile.aspx"
base    = path-absolute / path-rootless
```

The value of the **base** ABNF rule identifies the site the request operates on, and MUST be negotiated prior to initiating this protocol, as described section 1.5.

The query component of the Request-URI MUST be present, and MUST contain 3 query parameters with the following case-insensitive **ASCII** names:

- "Udc"
- "Urn"
- "Version"

The following ABNF specifies the syntax for the query component. The ABNF for the **pct-encoded** rule is specified in [RFC3986].

```
query-component        = <query-parameter>"&"<query-parameter>"&"
   <query-parameter>
query-parameter        = name"="value
name                   = "Udc" / "Urn" / "Version"
```

```
value                 = 1*(allowed-char | optional-allowed-char)
allowed-char          = ALPHA / DIGIT / pct-encoded / "-" / "_" / "." /
   "!" / "@" / "$" / "," / "="
optional-allowed-char = "+" / "'" ; plus and apostrophe symbols
```

The **value** for all query parameters MUST be a non-empty ASCII **string**, and MUST NOT contain "&". The protocol server MUST support values that use only characters matching the **allowed-char** rule. The protocol server SHOULD<1> also support characters matching the **optional-allowed-char** rule. The escaped encoding specified in [RFC3986] section 2 SHOULD<2> be used for other punctuation, space, and non- ASCII characters in the query parameters.

The query component (1) MUST NOT contain extra query parameters beyond the 3 required parameters. The protocol server MUST ignore the order of these parameters. For example, the following two query components (1) are identical for purposes of this protocol:

```
Udc=sample.udcx&Urn=urn:sample&Version=1.0.0.0
```

and

```
Version=1.0.0.0&Udc=sample.udcx&Urn=urn:sample
```

A complete example of Request-URIs are shown in section 4.

2.2.1.3 Request Headers Syntax

The following request header is relevant to this protocol:

- **Accept:** Specified in [RFC1945], section D.2 or [RFC2616], section 14.1. The protocol client SHOULD specify this header with the value "*/*". The protocol server MAY<3> ignore the value of this header.

2.2.2 Response Syntax

2.2.2.1 Response Status-Line

The response **Status-Line** MUST be valid according to [RFC1945] or [RFC2616] section 6.1. The protocol server MUST return a **200 OK** for successful requests.

The protocol server MUST return a 4*xx* or 5*xx* **Status-Code**, as specified in [RFC1945] or [RFC2616] section 6.1.1 to indicate that a request failed. The protocol server MUST return the Status-Code 403 if the query component of the Request-URI does not match the syntax specified in section 2.2.1.2. The protocol server SHOULD<4> use the Status-Code 401 to indicate that the protocol client can retry the request using a different authentication protocol or different properties.

For information about the message body to return for different Status-Codes, see section 2.2.2.3.

2.2.2.2 Response Headers Syntax

The following response headers are relevant to this protocol:

Content-Type: MUST be present and set to "text/xml; charset=utf-8" on 200 OK responses.

2.2.2.3 Response Message Body Syntax

The following table specifies the required message body content based on the HTTP method and response Status-Code.

HTTP Method	Response Status-Code	Response Message-Body
GET	200	MUST be a valid .udc file in the format specified by [MS-UDCX] and MUST use **UTF-8** encoding.
GET	4*xx* or 5*xx*	MUST NOT be a .udc file. SHOULD<5> be a message describing the failure reason, but MAY<6> be omitted by the protocol server.
HEAD	Any valid Status-Code	MUST be omitted by the protocol server.

3 Protocol Details

3.1 Common Details

This section specifies details common to both protocol server and protocol client behavior.

Except where specified, protocol clients SHOULD interpret HTTP Status-Codes returned by the protocol server as specified in [RFC1945] section 9 or [RFC2616] section 10.

This protocol allows protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults using HTTP Status-Codes.

3.1.1 Abstract Data Model

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

Requested .udc file: The .udc file that the protocol client is attempting to download using this protocol. The requested .udc file is identified by the *Udc* query parameter in the query component submitted to the protocol server, as specified in the following table.

Request Site: The site that this request is running on. This is determined from the **path segment** of the Request-URI that matches the **base** ABNF rule, as specified in section 2.2.1.2.

Referring Form Template: A form template (.xsn) file that references the requested .udc file. The protocol client and protocol server are both expected to have a copy of this file, as described in section 1.5. The referring form template is uniquely identified by the *Urn* and *Version* query parameters, as specified in the following table.

Referring Element: An element in the **form definition (.xsf) file** of the referring form template that refers to the requested .udc file. The reason that this protocol requires *Urn* and *Version* query parameters is that the protocol server can then verify that a referring element exists. A process for finding a referring element is specified in section 3.2.5.

Query Parameters: The *Udc*, *Urn*, and *Version* query parameters specified in section 2.2.1.2. The values of these parameters are **strings** matching the **value** rule in the ABNF specified in section 2.2.1.2. The protocol client and protocol server interpret values of these parameters, as specified in the following table.

Parameter	Description
Udc	This parameter identifies the file name of the .udc file to be returned.
Urn	This parameter partially identifies the referring form template. The value of this parameter MUST be the value of the **name** attribute on the **xDocumentClass** element in the referring form template, as specified in [MS-IPFF] section 2.2.20 or [MS-IPFF2] section 2.2.1.2.1.
Version	This parameter partially identifies the referring form template. This parameter MUST be a **string** valid according to the **xdSolutionVersion** type specified in [MS-IPFF] section 2.2.10 or [MS-IPFF2] section 2.2.1.1.10. This value MUST be the value of the **solutionVersion** attribute on the **xDocumentClass** element specified in [MS-IPFF] section 2.2.20 or [MS-IPFF2] section 2.2.1.2.1 in the referring form template.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Protocol Server Details

3.2.1 Abstract Data Model

Requested .udc file: Described in section 3.1.1.

Request Site: Described in section 3.1.1.

Referring Form Template: Described in section 3.1.1.

Referring Element: Described in section 3.1.1.

Query Parameters: Described in section 3.1.1.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

The protocol server MUST process request messages received from the protocol client as follows:

- First the protocol server MUST identify the request site from the Request-URI. The protocol server MUST return a **failure response** if there is no site associated with the Request-URI or if the protocol client does not have authorization to access the path in the request URI.
- Then the protocol server MUST find the referring form template identified by the *Urn* and *Version* query parameters using the interpretation of those parameters specified in section 3.1.1. The protocol server MUST return a failure response if no referring form template can be found on the request site (2), or if any implementation-specific authorization for this file fails.
- Then the protocol server MUST find the requested .udc file identified by the *Udc* query parameter. The protocol server MUST return a failure response if it cannot find the requested .udc file or if any implementation-specific authorization fails.
- The protocol server MUST find the referring element in the referring form template as follows, or return a failure response if no referring element can be found. If a **connectoid** element is found that passes all of these checks, it is a valid referring element.
  - The referring element MUST be a **connectoid** element, as specified in [MS-IPFF] section 2.2.147.30 or [MS-IPFF2] section 2.2.2.2.23.
  - This **connectoid** element MUST have a **source** attribute where the rightmost path segment equals the value of the *Udc* query parameter. The protocol server MUST use a case-insensitive ASCII **string** for this comparison.
  - This **connectoid** element MUST have a **connectionLinkType** attribute with the value "store".
- After performing these validation steps and any other implementation-specific validation<7>, the protocol server MUST return an HTTP response message, as specified in section 2.2.2.
  - If all validation steps succeed and a .udc file and referring element are found, the protocol server MUST return a 200 OK response. If the request HTTP method was **GET**, the protocol server MUST return a UTF-8 encoded message body containing the .udc file that is identified by the *Udc* query parameter. If the HTTP method was **HEAD**, the protocol server MUST NOT return a message body (1).
  - Otherwise a failure response MUST be returned. For the response syntax for failure responses, see section 2.2.2.1 and section 2.2.2.3.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Protocol Client Details

3.3.1 Abstract Data Model

Requested .udc file: Described in section 3.1.1**.**

Request Site: Described in section 3.1.1**.**

Referring Form Template: Described in section 3.1.1**.**

Referring Element: Described in section 3.1.1**.**

Query Parameters: Described in section 3.1.1**.**

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

Request messages sent by the protocol client MUST be in the syntax specified in section 2.2.1.

The protocol client MUST interpret the protocol server response based on the Status-Code as follows:

- **200:** The request was successful. If the HTTP method was **GET**, the protocol client can assume the message body MUST be a valid UTF-8 encoded .udc file in the format specified by [MS-UDCX].
- **401:** The request failed because of an authentication or authorization failure.
- ***4xx*/*5xx*:** The request failed and, if present, the message body MUST NOT be interpreted as a .udc file.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

4 Protocol Examples

The following examples show sample interactions between the protocol client and the protocol server.

4.1 Making a Successful GET Request

This example shows the messages exchanged when the protocol client makes a successful **GET** request to a protocol server using this protocol.

The following example is a protocol client request.

```
GET /_layouts/GetDataConnectionFile.aspx?Udc=sample.udcx&Urn=urn:sample&Version=1.0.0.0
HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0
Host: www.contoso.com
```

The following example is a protocol server response.

```
HTTP/1.1 200 OK
Connection: Keep-Alive
Cache-Control: Private
Date: Tue, 22 Jan 2008 01:13:30 GMT
Server: Microsoft-IIS/6.0
Content-Type: text/xml; charset=utf-8
Content-Length: 1234

<?xml version="1.0" encoding="UTF-8"?>
<?MicrosoftWindowsSharePointServices ContentTypeID="0x102030FF"?>
<udc:DataSource MajorVersion="2" MinorVersion="0"
xmlns:udc=http://schemas.microsoft.com/office/infopath/2006/udc>
...
</udc:DataSource>
```

For complete examples and specification of the **DataSource** element in a .udc file, see [MS-UDCX].

4.2 Making a Successful HEAD Request

This example shows the messages exchanged when the protocol client makes a successful **HEAD** request to a protocol server using this protocol.

The following example is a protocol client request.

```
HEAD /_layouts/GetDataConnectionFile.aspx?Udc=sample.udcx&Urn=urn:sample&Version=1.0.0.0
HTTP/1.1
Host: www.contoso.com
Content-Length: 0
Pragma: no-cache
```

The following example is a protocol server response.

```
HTTP/1.1 200 OK
Cache-Control: private
Date: Tue, 22 Jan 2008 01:13:30 GMT
```

```
Server: Microsoft-IIS/6.0
Content-Type: text/xml; charset=utf-8
Content-Length: 1252
```

4.3 Receiving a Failure Response From the Protocol Server

This example shows the messages exchanged when the protocol server returns a failure response.

The following example is a protocol client request.

```
GET / layouts/GetDataConnectionFile.aspx?Udc=NotFound.udcx&Urn=urn:sample&Version=1.0.0.1
HTTP/1.1
Accept: */*
User-Agent: Mozilla/4.0
Host: www.contoso.com
```

The following example is a protocol server response.

```
HTTP/1.1 403 Forbidden
Connection: Keep-Alive
Cache-Control: Private
Date: Tue, 22 Jan 2008 01:13:30 GMT
Server: Microsoft-IIS/6.0
Content-Length: 0
```

5 Security

5.1 Security Considerations for Implementers

The information in a .udc file is likely to be security-sensitive. For example, the file could contain server names, account names, and passwords. Using a method of authentication is recommended to establish the protocol client's identity and validate authorization for the requested resource.

The .udc file returned by this protocol could describe a data connection that is not located on the protocol server. Sending data to or receiving data from such a data connection could pose security risks if that data connection is not trusted. The protocol client implementation can mitigate this risk by validating data that is not transferred to a location that is not trusted. The protocol server implementation can mitigate this risk by only allowing highly trusted users to add or modify the .udc file or files to be returned by this protocol.

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 Office Forms Server 2007
- Microsoft Office InfoPath 2007
- Microsoft InfoPath 2010
- Microsoft InfoPath 2013
- Microsoft Office SharePoint Server 2007
- Microsoft SharePoint Server 2010
- Microsoft SharePoint Server 2013
- Microsoft SharePoint Server 2010 Enterprise
- Microsoft SharePoint Server 2016

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

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

<1> Section 2.2.1.2: Characters that match the **optional-allowed-char** rule are supported in query parameter values by SharePoint Server 2010 Enterprise and SharePoint Server 2013. Office SharePoint Server 2007 fails the request with a 403 Status-Code if either of the **optional-allowed-char** characters is present in a query parameter value.

<2> Section 2.2.1.2: The Office InfoPath 2007 client will send non-ASCII characters in the *udc* parameter if the **source** attribute on the **connectoid** element specified in [MS-IPFF] 2.2.147.30 contains characters not in the ASCII character set. Office SharePoint Server 2007, SharePoint Server 2010 Enterprise and SharePoint Server 2013 respond to these requests with a 403 failure Status-Code.

<3> Section 2.2.1.3: Office SharePoint Server 2007, SharePoint Server 2010 Enterprise and SharePoint Server 2013 ignore the **Accept**, **Accept-Charset**, **Accept-Encoding**, and **Accept-Language** headers when returning a .udc file. A message body of content-type "text/xml; charset=utf-8" can be returned even if that conflicts with the request headers sent by the protocol client.

<4> Section 2.2.2.1: Office SharePoint Server 2007, SharePoint Server 2010 Enterprise and SharePoint Server 2013 return a 401 Unauthorized Status-Code if the client is not authorized to access the path in the Request-URI, and returns a 403 Forbidden Status-Code if the client is authorized to access the path in the Request-URI but is not authorized to access a resource identified by the query parameters in the Request-URI.

<5> Section 2.2.2.3: The Office InfoPath 2007, InfoPath 2010 and InfoPath 2013 protocol clients ignore the message body of any failure response.

<6> Section 2.2.2.3: Office SharePoint Server 2007, SharePoint Server 2010 Enterprise and SharePoint Server 2013 always return an empty body when returning a 403 Status-Code, and when returning a 401 Status-Code returns either an empty body or a text/html message body, depending on authentication settings configured by the administrator and the credentials provided in the request message.

<7> Section 3.2.5: The Office SharePoint Server 2007, SharePoint Server 2010 Enterprise and SharePoint Server 2013 implementations of this protocol do the following to address security considerations:

1. This implementation requires that both the .udc file returned by this protocol and the referring form template (.xsn) file are uploaded by an administrator.

2. This implementation validates that the protocol client has authorization to view the form template (.xsn) file identified by the query parameters in a Web browser.

7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

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

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

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

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
6 Appendix A: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

8 Index

S

T

V