

[MS-FORMS]: Forms Service 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.msp>). 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
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Revised and edited the technical content
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Editorial	Revised and edited the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Minor	Clarified the meaning of the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction	5
1.1 Glossary	5
1.2 References	5
1.2.1 Normative References	5
1.2.2 Informative References	6
1.3 Protocol Overview (Synopsis)	6
1.4 Relationship to Other Protocols	6
1.5 Prerequisites/Preconditions	7
1.6 Applicability Statement	7
1.7 Versioning and Capability Negotiation	7
1.8 Vendor-Extensible Fields	7
1.9 Standards Assignments	7
2 Messages	8
2.1 Transport	8
2.2 Common Message Syntax	8
2.2.1 Namespaces	8
2.2.2 Messages	8
2.2.2.1 SOAP Fault Message	8
2.2.3 Elements	9
2.2.4 Complex Types	9
2.2.4.1 SOAPFaultDetails	9
2.2.5 Simple Types	9
2.2.6 Attributes	9
2.2.7 Groups	9
2.2.8 Attribute Groups	9
3 Protocol Details	10
3.1 Forms Service Protocol Server Details	10
3.1.1 Abstract Data Model	10
3.1.2 Timers	11
3.1.3 Initialization	11
3.1.4 Message Processing Events and Sequencing Rules	11
3.1.4.1 GetForm	11
3.1.4.1.1 Messages	11
3.1.4.1.1.1 GetFormSoapIn	11
3.1.4.1.1.2 GetFormSoapOut	12
3.1.4.1.2 Elements	12
3.1.4.1.2.1 GetForm	12
3.1.4.1.2.2 GetFormResponse	12
3.1.4.2 GetFormCollection	13
3.1.4.2.1 Messages	14
3.1.4.2.1.1 GetFormCollectionSoapIn	14
3.1.4.2.1.2 GetFormCollectionSoapOut	14
3.1.4.2.2 Elements	14
3.1.4.2.2.1 GetFormCollection	14
3.1.4.2.2.2 GetFormCollectionResponse	14
3.1.5 Timer Events	15
3.1.6 Other Local Events	15

4 Protocol Examples	16
4.1 GetFormCollection	16
4.2 GetForm	16
5 Security	18
5.1 Security Considerations for Implementers	18
5.2 Index of Security Parameters	18
6 Appendix A: Full WSDL	19
7 Appendix B: Product Behavior	23
8 Change Tracking	24
9 Index	25

1 Introduction

This document specifies the Forms Service Protocol. This protocol enables a client to get a list of forms from a protocol server and to get individual forms from that list. A form is an XML structure with elements describing user interface elements that are displayed on a client-side Web page. The definition, display, and use of a form is outside the scope of this document.

1.1 Glossary

The following terms are defined in [\[MS-GLOS\]](#):

globally unique identifier (GUID)
Hypertext Transfer Protocol (HTTP)
Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)

The following terms are defined in [\[MS-OFCGLOS\]](#):

absolute URL
back-end database server
content database
document
form
item
item identifier
list
list identifier
list item
page
path component
site
SOAP (Simple Object Access Protocol)
SOAP action
SOAP body
SOAP fault
store-relative form
URL (Uniform Resource Locator)
Web Part
Web Part zone
WSDL (Web Services Description Language)
WSDL operation
XML namespace
XML Schema

The following terms are specific to this document:

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.

[MS-WSSCAML] Microsoft Corporation, "[Collaborative Application Markup Language \(CAML\) Structure Specification](#)", June 2008.

[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>

[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>

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

[SOAP1.2/1] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part1-20030624>

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, <http://www.w3.org/TR/2003/REC-soap12-part2-20030624>

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

[XMLNS] World Wide Web Consortium, "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation 8 December 2009, <http://www.w3.org/TR/REC-xml-names/>

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

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

1.2.2 Informative References

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

[MS-OFCGLOS] Microsoft Corporation, "[Microsoft Office Master Glossary](#)", June 2008.

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

1.3 Protocol Overview (Synopsis)

This protocol enables clients to obtain a list of **forms (2)** contained in a **list(1)** and to obtain information about individual forms (2). This protocol follows a straightforward request-response pattern.

1.4 Relationship to Other Protocols

This protocol uses the **SOAP** message protocol for formatting request and response messages, as described in [\[SOAP1.1\]](#), [\[SOAP1.2/1\]](#) and [\[SOAP1.2/2\]](#). It transmits those messages by using **HTTP**, as described in [\[RFC2616\]](#), or **HTTPS**, as described in [\[RFC2818\]](#).

The following diagram shows the underlying messaging and transport stack used by the protocol:

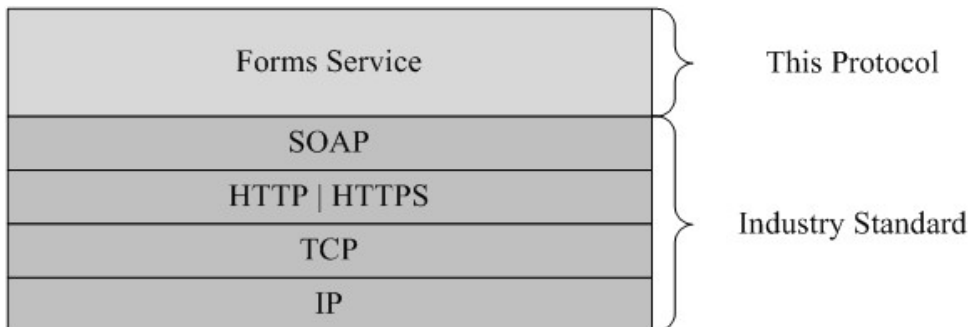


Figure 1: This protocol in relation to other protocols

This protocol operates against a **site (2)** that is identified by a **URL** that is known by protocol clients. The protocol server endpoint is formed by appending `"/_vti_bin/Forms.asmx"` to the URL of the site, for example, `http://www.contoso.com/Repository/_vti_bin/Forms.asmx`.

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

1.5 Prerequisites/Preconditions

This protocol enables protocol clients to obtain a list of forms (2) contained in a list (1) and to obtain information about individual forms (2).

1.6 Applicability Statement

This document covers versioning issues in the following areas:

Supported Transports: This protocol uses multiple transports with SOAP as described in **Transport** (section [2.1](#)).

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers **MUST** support SOAP over HTTP. Protocol servers **SHOULD** additionally support SOAP over HTTPS for securing communication with protocol clients.

Protocol messages **MUST** be formatted as specified either in [\[SOAP1.1\]](#) section 4, SOAP Envelope, or in [\[SOAP1.2/1\]](#) section 5, SOAP Message Construct. Protocol server faults **MUST** be returned either using HTTP status codes as specified in [\[RFC2616\]](#) section 10, Status Code Definitions, or using **SOAP faults** as specified either in [\[SOAP1.1\]](#) section 4.4, SOAP Fault, or in [\[SOAP1.2/1\]](#) section 5.4, SOAP Fault.

2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses the **XML Schema** as defined in [\[XMLSCHEMA1\]](#) and [\[XMLSCHEMA2\]](#), and **WSDL** as defined in [\[WSDL\]](#).

2.2.1 Namespaces

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

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
tns	http://schemas.microsoft.com/sharepoint/soap/	[SOAP1.2/1] and [SOAP1.2/2]
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] and [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] and [SOAP1.2/2]
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
mime	http://schemas.xmlsoap.org/wsdl/mime/	[WSDL]
tm	http://microsoft.com/wsdl/mime/textMatching/	
soapenc	http://schemas.xmlsoap.org/soap/encoding/	[SOAP1.2/1] [SOAP1.2/2]
http	http://schemas.xmlsoap.org/wsdl/http/	[RFC2616]

2.2.2 Messages

2.2.2.1 SOAP Fault Message

This protocol enables a protocol server to notify a protocol client with application-level faults by using a SOAP fault response. In a SOAP fault, the detail element contains application-specific error information.

The following specifies the structure of the SOAP fault used by this protocol.


```

<s:element name="detail">
  <s:complexType>
    <s:sequence>
      <s:element name="errorString" type="s:string"/>
      <s:element name="errorCode" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:element>

```

errorString: The description of the error.

errorCode: Operation-specific error codes are defined in the messages sub-section for the operations that return SOAP faults.

2.2.3 Elements

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

2.2.4 Complex Types

2.2.4.1 SOAPFaultDetails

This complex type defines a SOAP fault as follows:

```

<s:schema xmlns:s="http://www.w3.org/2001/XMLSchema" targetNamespace="
http://schemas.microsoft.com/sharepoint/soap">
  <s:complexType name="SOAPFaultDetails">
    <s:sequence>
      <s:element name="errorstring" type="s:string"/>
      <s:element name="errorcode" type="s:string" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
</s:schema>

```

errorstring: A human-readable text explaining the application-level fault.

errorcode: The hexadecimal representation of a 4-byte result code. The format of the string MUST be 0xAAAAAAAA.

2.2.5 Simple Types

This specification does not define any common XML Schema simple type definitions.

2.2.6 Attributes

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

2.2.7 Groups

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

2.2.8 Attribute Groups

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

3 Protocol Details

The client side of this protocol is 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.

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [\[RFC2616\]](#) section 10, Status Code Definitions.

This protocol enables protocol servers to notify protocol clients of application-level faults by using SOAP faults. This protocol enables protocol servers to provide additional details for SOAP faults by including a **detail** element as specified in either [\[SOAP1.1\]](#) section 4.4, SOAP Fault, or [\[SOAP1.2/1\]](#) section 5.4, SOAP Fault, that conforms to the XML schema of the **SOAPFaultDetails** complex type specified in **SOAPFaultDetails** (section [2.2.4.1](#)). Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

This protocol enables protocol servers to perform implementation-specific authorization checks and notify protocol clients of authorization faults, using either HTTP status codes or SOAP faults as specified previously in this section.

3.1 Forms Service Protocol Server Details

The following diagram describes the communication between the protocol client and the protocol server.

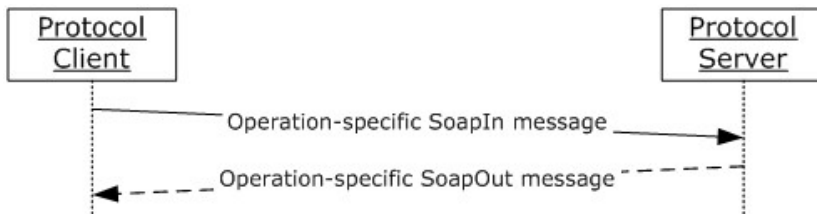


Figure 2: Sequence of methods for this protocol

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.

The protocol provides access to form (2) data stored in the **back-end database server**. The protocol does not create or modify forms (2), but enables listing and retrieval of form (2) data. The service uses the following types of data.

- **Lists:** A set of information about all lists (1) in a **content database**. Each entry has a **list identifier** and is represented by a **store-relative form** URL.
- **Items:** A set of information about all **items** in a content database. These entries have **item identifiers**.

- **Page:** An HTML **document** that may contain dynamic content such as a **Web Part** that is interpreted before being displayed in a client application.
- **Form:** A **page** that enables the creation, viewing, or editing of **list items**.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the **WSDL operations** defined in this protocol.

Operation	Description
GetForm	This operation is used to get information about a single form (2).
GetFormCollection	This operation is used to get information about all forms (2) on a particular list (1).

3.1.4.1 GetForm

This operation obtains information about a form (2). The request requires the name of the form (2) and the URL of a list (1). This operation is defined as follows:

```
<wsdl:operation name="GetForm">
  <wsdl:input message="GetFormSoapIn" />
  <wsdl:output message="GetFormSoapOut" />
</wsdl:operation>
```

The protocol client sends a **GetFormSoapIn** request message, and the protocol server responds with a **GetFormSoapOut** response message, as follows:

The protocol server **MUST** return a SOAP exception if the **listName** specified in the **GetForm** element does not exist. The detail error string **SHOULD** contain an error message to present to the user.

The server **MUST** return a SOAP exception if the **formUrl** specified in the **GetForm** element does not exist.

3.1.4.1.1 Messages

3.1.4.1.1.1 GetFormSoapIn

This structure contains the **GetForm** operation request with the parameters provided by the protocol client.

The **SOAP action** value of the message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/GetForm
```

The **SOAP body** contains a **GetForm** element.

3.1.4.1.1.2 GetFormSoapOut

The SOAP action value of the message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/GetForm
```

The SOAP body contains a **GetFormResponse** element.

3.1.4.1.2 Elements

3.1.4.1.2.1 GetForm

The **GetForm** structure contains the parameters for the **GetForm** operation. This structure is defined as follows:

```
<s:element name="GetForm">
  <s:complexType>
    <s:sequence>
      <s:element name="listName" type="s:string"/>
      <s:element name="formUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

listName: A **GUID** encoded in a string or the title of a list (1) on the site (2).

formUrl: A URL that identifies the particular form (2) returned. This MUST be an **absolute URL** that SHOULD end with the hierarchical **path component** of the URL with no input component.

3.1.4.1.2.2 GetFormResponse

This structure contains the return value for the **GetForm** operation. This structure is defined as follows:

```
<s:element name="GetFormResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetFormResult" minOccurs="0">
        <s:complexType>
          <s:sequence>
            <s:element name="Form">
              <s:complexType>
                <s:attribute name="Type"
                  type="s:string" use="required">
                  <xs:simpleType>
                    <xs:restriction base="xs:string">
                      <xs:enumeration value="DisplayForm" />
                      <xs:enumeration value="EditForm" />
                      <xs:enumeration value="NewForm" />
                      <xs:enumeration value="NewFormDialog" />
                      <xs:enumeration value="SolutionForm" />
                      <xs:enumeration value="" />
                    </xs:restriction>
                  </xs:simpleType>
                </s:attribute>
                <s:attribute name="Name"
                  type="s:string"/>
              </s:complexType>
            </s:element>
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

```

        <s:attribute name="Url"
            type="s:string" use="required"/>
        <s:attribute name="Default"
            type="s:TRUEFALSE"/>
        <s:attribute name="FormID"
            type="s:string"/>
        <s:attribute name="Template"
            type="s:string"/>
        <s:attribute name="WebPartZoneID"
            type="s:string"/>
        <s:attribute name="SetupPath"
            type="s:string"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType></s:element>

```

GetFormResult: A structure that holds the data returned from the WSDL operation.

Type: A string indicating the general type of form (2) as specified in [\[MS-WSSCAML\]](#) section 2.3.1.5.

Name: The title of the form (2).

Url: The URL of the page hosting the form (2).

Default: A Boolean value as specified in [\[MS-WSSCAML\]](#) section 2.1.12. Specifies whether the form (2) is the default form (2) of the list (1). It is an optional attribute that appears only if previously set in the XML schema of the form (2) definition.

FormId: A string that contains a positive integer. This MUST be unique for each form (2) in the back-end database server. It is an optional attribute that appears only if previously set in the XML schema of the form (2) definition.

Template: The form (2) template name.

WebPartZoneID: Identifier of the **Web Part zone** that contains the form (2).

SetupPath: Source path to the uncustomized document of the form (2).

3.1.4.2 GetFormCollection

This operation lists all of the forms (2) on a list (1). This operation is defined as follows:

```

<wsdl:operation name="GetFormCollection">
    <wsdl:input message="GetFormCollectionSoapIn" />
    <wsdl:output message="GetFormCollectionSoapOut" />
</wsdl:operation>

```

The protocol client sends a **GetFormCollectionSoapIn** request message, and the protocol server responds with a **GetFormCollectionSoapOut** response message, as follows:

If the list (1) specified by the **listName** in the **GetFormCollection** element is not found or the **GetFormCollection** element is empty, the protocol server MUST respond with a SOAP exception.

This structure holds a list of **Form** elements, each of which describes a single form (2).

3.1.4.2.1 Messages

3.1.4.2.1.1 GetFormCollectionSoapIn

The SOAP action value of the message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/GetFormCollection
```

The SOAP body contains a **GetFormCollection** element.

3.1.4.2.1.2 GetFormCollectionSoapOut

The SOAP action value of the message is defined as follows:

```
http://schemas.microsoft.com/sharepoint/soap/GetFormCollection
```

The SOAP body contains a **GetFormCollectionResponse** element.

3.1.4.2.2 Elements

3.1.4.2.2.1 GetFormCollection

This structure is used to pass the parameter for the **GetFormCollection** WSDL operation. This operation is defined as follows:

```
<s:element name="GetFormCollection">
  <s:complexType>
    <s:sequence>
      <s:element name="listName" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

listName: A GUID encoded in a string or the title of a list (1) from which to fetch forms (2).

3.1.4.2.2.2 GetFormCollectionResponse

This structure holds a list of **Form** elements, each of which describes a single form (2). This structure is defined as follows:

```
<s:element name="GetFormCollectionResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetFormCollectionResult">
        <s:complexType>
          <s:sequence>
            <s:element name="Forms">

```

```
<s:complexType>
  <s:sequence>
    <s:element name="Form" minOccurs="0"
      maxOccurs="unbounded">
      <s:complexType>
        <s:attribute name="type" type="s:string"
          use="required"/>
        <s:attribute name="url" type="s:string"
          use="required"/>
      </s:complexType>
    </s:element>
  </s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
```

GetFormCollectionResult: This structure holds the list of form (2) information.

Type: A string with the form (2) type. It MUST be the same type as the one defined in **GetFormResponse** (section [3.1.4.1.2.2](#)).

Url: A URL identifying the form (2). The contents of the referenced object MUST contain a **FormDefinition** as specified in [\[MS-WSSCAML\]](#) section 2.3.2.5.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

This section provides some example client-server exchanges. White space has been added to improve readability.

4.1 GetFormCollection

A client-server exchange for the **GetFormCollection** operation resembles the following example.

Client to server:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetFormCollection
      xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <listName>Custom Test</listName>
    </GetFormCollection>
  </soap:Body>
</soap:Envelope>
```

Server response:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetFormCollectionResponse
      xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <GetFormCollectionResult>
        <Forms>
          <Form Url="Lists/Custom Test/DispForm.aspx"
            Type="DisplayForm" />
          <Form Url="Lists/Custom Test/EditForm.aspx"
            Type="EditForm" />
          <Form Url="Lists/Custom Test/NewForm.aspx"
            Type="NewForm" />
        </Forms>
      </GetFormCollectionResult>
    </GetFormCollectionResponse>
  </soap:Body>
</soap:Envelope>
```

4.2 GetForm

A client-server exchange for the **GetForm** operation resembles the following example.

Client request:

```
<?xml version="1.0" encoding="utf-8"?>
```



```
<soap:Envelope
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetForm xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <listName>{FDA9C196-8655-45C4-91BB-2B42682CD278}</listName>
      <formUrl>http://office/Lists/Custom Test/NewForm.aspx</formUrl>
    </GetForm>
  </soap:Body>
</soap:Envelope>
```

Server response:

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetFormResponse xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <GetFormResult>
        <Form Type="NewForm"
          Url="NewForm.aspx"
          SetupPath="pages\form.aspx"
          WebPartZoneID="Main" />
      </GetFormResult>
    </GetFormResponse>
  </soap:Body>
</soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL (Web Services Description Language) is provided here.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/soap/">
      <s:import namespace="http://www.w3.org/2001/XMLSchema" />
<s:element name="GetForm">
  <s:complexType>
    <s:sequence>
      <s:element name="listName" type="s:string"/>
      <s:element name="formUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:element name="GetFormResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetFormResult" minOccurs="0">
        <s:complexType>
          <s:sequence>
            <s:element name="Form">
              <s:complexType>
                <s:attribute name="Type"
type="s:string" use="required">
                  <xs:simpleType>
                    <xs:restriction base="xs:string">
                      <xs:enumeration value="DisplayForm" />
                      <xs:enumeration value="EditForm" />
                      <xs:enumeration value="NewForm" />
                      <xs:enumeration value="NewFormDialog" />
                      <xs:enumeration value="SolutionForm" />
                      <xs:enumeration value="" />
                    </xs:restriction>
                  </xs:simpleType>
                </s:attribute>
                <s:attribute name="Name"
type="s:string"/>
                <s:attribute name="Url"
type="s:string" use="required"/>
                <s:attribute name="Default"
type="s:boolean"/>
                <s:attribute name="FormID"
type="s:string"/>
                <s:attribute name="Template"
type="s:string"/>
                <s:attribute name="WebPartZoneID"
```

```

        type="s:string"/>
        <s:attribute name="SetupPath"
            type="s:string"/>
    </s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
</s:sequence>
</s:complexType>
</s:element>
<s:element name="GetFormCollection">
    <s:complexType>
        <s:sequence>
            <s:element name="listName" type="s:string" />
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="GetFormCollectionResponse">
    <s:complexType>
        <s:sequence>
            <s:element name="GetFormCollectionResult">
                <s:complexType>
                    <s:sequence>
                        <s:element name="Forms">
                            <s:complexType>
                                <s:sequence>
                                    <s:element name="Form" minOccurs="0"
                                        maxOccurs="unbounded">
                                        <s:complexType>
                                            <s:attribute name="type" type="s:string"
                                                use="required"/>
                                            <s:attribute name="url" type="s:string"
                                                use="required"/>
                                        </s:complexType>
                                    </s:element>
                                </s:sequence>
                            </s:complexType>
                        </s:element>
                    </s:sequence>
                </s:complexType>
            </s:element>
        </s:sequence>
    </s:complexType>
</s:element>
<s:element name="detail">
    <s:complexType>
        <s:sequence>
            <s:element name="errorString" type="s:string"/>
            <s:element name="errorCode" type="s:string" minOccurs="0"/>
        </s:sequence>
    </s:complexType>
</s:element>
</s:schema>
<s:schema xmlns:s="http://www.w3.org/2001/XMLSchema" targetNamespace="
http://schemas.microsoft.com/sharepoint/soap">
    <s:complexType name="SOAPFaultDetails">
        <s:sequence>
            <s:element name="errorstring" type="s:string"/>

```

```

        <s:element name="errorCode" type="s:string" minOccurs="0"/>
    </s:sequence>
</s:complexType>
</s:schema>
</wsdl:types>
<wsdl:message name="GetFormCollectionSoapIn">
    <wsdl:part name="parameters" element="tns:GetFormCollection" />
</wsdl:message>
<wsdl:message name="GetFormCollectionSoapOut">
    <wsdl:part name="parameters" element="tns:GetFormCollectionResponse" />
</wsdl:message>
<wsdl:message name="GetFormSoapIn">
    <wsdl:part name="parameters" element="tns:GetForm" />
</wsdl:message>
<wsdl:message name="GetFormSoapOut">
    <wsdl:part name="parameters" element="tns:GetFormResponse" />
</wsdl:message>
<wsdl:portType name="FormsSoap">
    <wsdl:operation name="GetFormCollection">
        <wsdl:input message="tns:GetFormCollectionSoapIn" />
        <wsdl:output message="tns:GetFormCollectionSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetForm">
        <wsdl:input message="tns:GetFormSoapIn" />
        <wsdl:output message="tns:GetFormSoapOut" />
    </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="FormsSoap" type="tns:FormsSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetFormCollection">
        <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetFormCollection" style="document"
/>
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetForm">
        <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/GetForm"
style="document" />
        <wsdl:input>
            <soap:body use="literal" />
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal" />
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="FormsSoap12" type="tns:FormsSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="GetFormCollection">
        <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetFormCollection" style="document"
/>
        <wsdl:input>
            <soap12:body use="literal" />

```

```
</wsdl:input>
<wsdl:output>
  <soap12:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetForm">
  <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/GetForm"
style="document" />
  <wsdl:input>
    <soap12:body use="literal" />
  </wsdl:input>
  <wsdl:output>
    <soap12:body use="literal" />
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>
</wsdl:definitions>
```

7 Appendix B: Product Behavior

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

- Microsoft® Office 2003
- The 2007 Microsoft® Office system
- Microsoft® Office 2010 suites
- Microsoft® SharePoint® Foundation 2010
- Windows® SharePoint® Services 2.0
- Windows® SharePoint® Services 3.0

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.

8 Change Tracking

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

9 Index

A

Abstract data model
[server](#) 10
[Applicability](#) 7
[Attribute groups](#) 9
[Attributes](#) 9

C

[Capability negotiation](#) 7
[Change tracking](#) 24
Client
[overview](#) 10
Complex types
[SOAPFaultDetails](#) 9

D

Data model - abstract
[server](#) 10

E

Events
[local - server](#) 15
[timer - server](#) 15
Examples
[GetForm](#) 16
[GetFormCollection](#) 16
[overview](#) 16

F

[Fields - vendor-extensible](#) 7
[Forms Service Protocol interface](#) 10
[Full WSDL](#) 19

G

[GetForm example](#) 16
[GetFormCollection example](#) 16
[Glossary](#) 5
[Groups](#) 9

I

[Implementer - security considerations](#) 18
[Index of security parameters](#) 18
[Informative references](#) 6
Initialization
[server](#) 11
[Interfaces – Forms Service Protocol](#) 10
[Introduction](#) 5

L

Local events
[server](#) 15

M

Message processing
[server](#) 11
Messages
[attribute groups](#) 9
[attributes](#) 9
[elements](#) 9
[groups](#) 9
[namespaces](#) 8
[simple types](#) 9
[SOAP Fault Message](#) 8
[SOAP Fault Message message](#) 8
[SOAPFaultDetails complex type](#) 9
[syntax](#) 8
[transport](#) 8

N

[Namespaces](#) 8
[Normative references](#) 5

O

Operations
[GetForm](#) 11
[GetFormCollection](#) 13
[Overview \(synopsis\)](#) 6

P

[Parameters - security index](#) 18
[Preconditions](#) 7
[Prerequisites](#) 7
[Product behavior](#) 23

R

References
[informative](#) 6
[normative](#) 5
[Relationship to other protocols](#) 6

S

Security
[implementer considerations](#) 18
[parameter index](#) 18
Sequencing rules
[server](#) 11
Server
[abstract data model](#) 10
[Forms Service Protocol interface](#) 10
[GetForm operation](#) 11
[GetFormCollection operation](#) 13
[initialization](#) 11
[local events](#) 15
[message processing](#) 11

- [overview](#) 10
- [sequencing rules](#) 11
- [timer events](#) 15
- [timers](#) 11
- [Simple types](#) 9
- [SOAPFaultDetails complex type](#) 9
- [Standards assignments](#) 7
- Syntax
 - [messages - overview](#) 8

T

- Timer events
 - [server](#) 15
- Timers
 - [server](#) 11
- [Tracking changes](#) 24
- [Transport](#) 8
- Types
 - [simple](#) 9

V

- [Vendor-extensible fields](#) 7
- [Versioning](#) 7

W

- [WSDL](#) 19