[MS-SPDIAG]:

SharePoint Diagnostics Web Service Protocol

Revision Summary

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

Date	Revision History	Revision Class	Comments
			technical content.
7/31/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.1	None	No changes to the meaning, language, or formatting of the technical content.
2/26/2016	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The SharePoint Diagnostics Web Service Protocol enables a protocol client to submit diagnostic reports describing application errors that occur on the protocol client.

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:

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.

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

SOAP action: The HTTP request header field used to indicate the intent of the **SOAP** request, using a URI value. See [SOAP1.1] section 6.1.1 for more information.

SOAP body: A container for the payload data being delivered by a **SOAP message** to its recipient. See [SOAP1.2-1/2007] section 5.3 for more information.

SOAP message: An XML document consisting of a mandatory SOAP envelope, an optional SOAP header, and a mandatory **SOAP body**. See [SOAP1.2-1/2007] section 5 for more information.

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].

Web Services Description Language (WSDL): An XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are described abstractly and are bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints, which describe a network service. WSDL is extensible, which allows the description of endpoints and their messages regardless of the message formats or network protocols that are used.

WSDL message: An abstract, typed definition of the data that is communicated during a **WSDL operation** [WSDL]. Also, an element that describes the data being exchanged between web service providers and clients.

WSDL operation: A single action or function of a web service. The execution of a WSDL operation typically requires the exchange of messages between the service requestor and the service provider.

XML fragment: Lines of text that adhere to XML tag rules, as described in [XML], but do not have a Document Type Definition (DTD) or schema, processing instructions, or any other header information.

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and

local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML namespace prefix: An abbreviated form of an **XML namespace**, as described in [XML].

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

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

1.2 References

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

1.2.1 Normative References

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

[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

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

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

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

[XPATH] Clark, J., and DeRose, S., "XML Path Language (XPath), Version 1.0", W3C Recommendation, November 1999, http://www.w3.org/TR/xpath/

1.2.2 Informative References

None.

1.3 Overview

In many modern web pages, there is a large amount of code (for example, JavaScript) running in client web browser. To help diagnose common errors encountered with the web pages mentioned, it is desirable that the developers of the pages can get detailed information regarding these errors.

This protocol defines an operation that allows a protocol client to submit details about an error report (for example, call stack, error message, or operating environment). The developers can use the submitted error reports to discover and fix errors encountered by the users.

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 Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS), as described in [RFC2818].

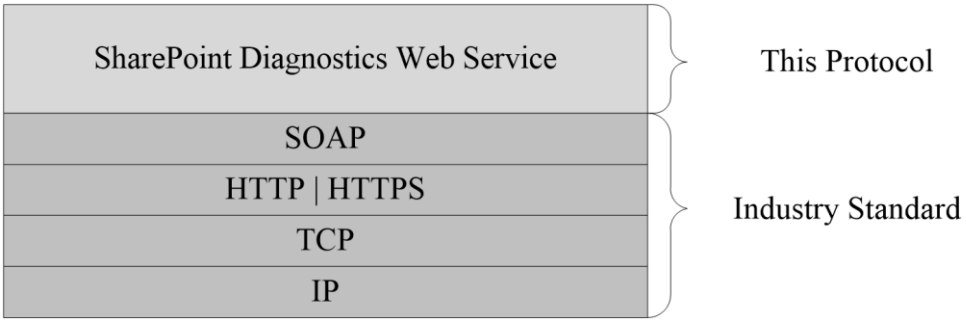

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol operates against a protocol server that exposes one or more endpoint (4) URIs that are known by protocol clients. The endpoint (4) URI of the protocol server and the transport that is used by the protocol server are either known by the protocol client or obtained by using the discovery mechanism that is described in [MS-SPTWS].

The protocol client obtains the requisite ApplicationClassId and ApplicationVersion values and the endpoint (4) URI of the protocol server that provides the discovery mechanism, as described in [MS-SPTWS], by means that are independent of either protocol.

This protocol requires the protocol client to have permission to call the methods on the protocol server.

The protocol client implements the token-based security mechanisms that are required by the protocol server and related security protocols, as described in [MS-SPSTWS].

1.6 Applicability Statement

This protocol is intended to transfer small amounts of data (less than 6 kilobytes) from a protocol client to a protocol server. Therefore, the protocol client is expected to gather and format relevant information (such as the call stack) in an **XML fragment**.

This protocol is not intended to transfer large regions of memory or other comprehensive error data collection from a protocol client.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports**: This protocol can be implemented by using transports that support sending Simple Object Access Protocol (SOAP) messages, as described in section 2.1.
- **Protocol Versions**: This protocol is not versioned.

Capability Negotiation: This protocol does not support version negotiation.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Protocol servers MUST support SOAP over HTTP or HTTPS.

All protocol messages MUST be transported by using HTTP bindings at the transport level.

Protocol messages MUST be formatted as specified in either [SOAP1.1] section 4 or [SOAP1.2/1] section 5. Protocol server faults MUST be returned by using either HTTP status codes, as specified in [RFC2616] section 10, or SOAP faults, as specified in [SOAP1.1] section 4.4 or [SOAP1.2/1] section 5.4.

If the HTTPS transport is used, a server certificate MUST be deployed.

This protocol MAY transmit an additional SOAP header, the **ServiceContex**t header, as specified in [MS-SPSTWS].

This protocol does not define any means for activating a protocol server or protocol client. The protocol server MUST be configured and begin listening in an implementation-specific way. In addition, the protocol client MUST know the format and transport that is used by the protocol server, for example, the SOAP format over an HTTP transport.

2.2 Common Message Syntax

This section contains common definitions used by this protocol. The syntax of the definitions uses an **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 not significant for interoperability.

Prefix	Namespace URI	Reference
http	http://schemas.xmlsoap.org/wsdl/http/	[RFC2616]
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
tns	http://schemas.microsoft.com/sharepoint/diagnostics/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]

2.2.2 Messages

This specification does not define any common **WSDL message** definitions.

2.2.3 Elements

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

2.2.4 Complex Types

This specification does not define any common XML schema complex type definitions.

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 group definitions.

2.2.8 Attribute Groups

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

3 Protocol Details

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

Except where specified, protocol clients SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC2616] section 10.

This protocol allows protocol servers to notify protocol clients of application-level faults using SOAP faults. Except where specified, these SOAP faults are not significant for interoperability, and protocol clients can interpret them in an implementation-specific manner.

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

3.1 Server Details

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

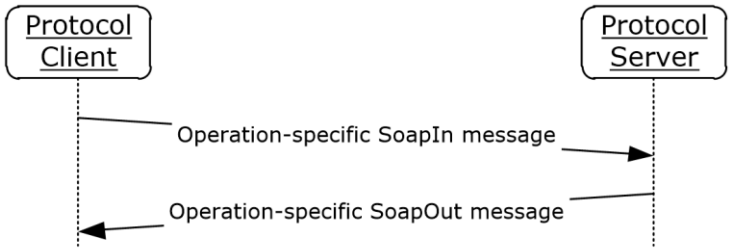

Figure 2: Message exchange between client and server

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.

This protocol does not dictate any specific information required in the error report. If available, the error report data includes information about the client operating environment (such as web browser name, browser version, and protocol client language). The error report data includes information about the error (message, **URL**, line number, and call stack). The error report includes information about the origination of the error (application name, file name). The error report is specified in section 3.1.4.1.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

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

Operation	Description
SendClientScriptErrorReport	This operation is used to submit error reports originating from the protocol client to the protocol server.

3.1.4.1 SendClientScriptErrorReport

This operation is used to submit error reports originating from the protocol client to the protocol server.

The following is the WSDL port type specification of the **SendClientScriptErrorReport** **WSDL operation**.

```
<wsdl:operation name="SendClientScriptErrorReport"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:input message="tns:SendClientScriptErrorReportSoapIn"/>
  <wsdl:output message="tns:SendClientScriptErrorReportSoapOut"/>
</wsdl:operation>
```

The protocol client sends a **SendClientScriptErrorReportSoapIn** request WSDL message, and the protocol server responds with a **SendClientScriptErrorReportSoapOut** response WSDL message.

3.1.4.1.1 Messages

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

Message	Description
SendClientScriptErrorReportSoapIn	The request WSDL message for the **SendClientScriptErrorReport** WSDL operation.
SendClientScriptErrorReportSoapOut	The response WSDL message for the **SendClientScriptErrorReport** WSDL operation.

3.1.4.1.1.1 SendClientScriptErrorReportSoapIn

The request WSDL message for the **SendClientScriptErrorReport** WSDL operation.

The **SOAP action** value is:

```
http://schemas.microsoft.com/sharepoint/diagnostics/SendClientScriptErrorReport
```

The **SOAP body** contains the **SendClientScriptErrorReport** element.

3.1.4.1.1.2 SendClientScriptErrorReportSoapOut

The response WSDL message for the **SendClientScriptErrorReport** WSDL operation.

The SOAP body contains the **SendClientScriptErrorReportResponse** element.

3.1.4.1.2 Elements

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

Element	Description
SendClientScriptErrorReport	The input data for the **SendClientScriptErrorReport** WSDL operation.
SendClientScriptErrorReportResponse	The result data for the **SendClientScriptErrorReport** WSDL operation.

3.1.4.1.2.1 SendClientScriptErrorReport

The **SendClientScriptErrorReport** element specifies the input data for the **SendClientScriptErrorReport** WSDL operation.

```
<xs:element name="SendClientScriptErrorReport" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="message" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="1" name="file" type="xs:string"/>
      <xs:element minOccurs="1" maxOccurs="1" name="line" type="xs:int"/>
      <xs:element minOccurs="0" maxOccurs="1" name="client" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="1" name="stack" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="1" name="team" type="xs:string"/>
      <xs:element minOccurs="0" maxOccurs="1" name="originalFile" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

message: A string containing the message associated with the current error.

file: A string containing the URL file name associated with the current error.

line: An integer containing the line number associated with the current error.

client: A string argument representing the protocol client operating environment.<1>

stack: A string argument representing the call stack of the error.<2>

team: A string containing the application associated with the current error.

originalFile: A string containing the physical file name associated with the current error.

3.1.4.1.2.2 SendClientScriptErrorReportResponse

The **SendClientScriptErrorReportResponse** element specifies the result data for the **SendClientScriptErrorReport** WSDL operation.

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

```
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="1" name="SendClientScriptErrorReportResult"
type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

SendClientScriptErrorReportResult: Implementation specific result. The protocol client MUST NOT rely on this data to follow any particular format.

3.1.4.1.3 Complex Types

None.

3.1.4.1.4 Simple Types

None.

3.1.4.1.5 Attributes

None.

3.1.4.1.6 Groups

None.

3.1.4.1.7 Attribute Groups

None.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

To submit an error report to the server, the protocol client constructs the following **SOAP message**:

```
<?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>
    <SendClientScriptErrorReport
xmlns="http://schemas.microsoft.com/sharepoint/diagnostics/">
          <message>'null'%20is%20null%20or%20not%20an%20object</message>
          <file>init.debug.js</file>
          <line>407</line>
          <client>
                   &lt;client&gt;
                    &lt;browser name=&quot;Microsoft Internet Explorer&quot;
version=&quot;7&quot;/&gt;
                    &lt;useragent&gt;Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0;
Trident/4.0; SLCC1; .NET CLR 2.0.50727; .NET CLR 1.1.4322; .NET CLR 3.5.30729; .NET CLR
3.0.30618; MS-RTC LM 8; InfoPath.2)&lt;/useragent&gt;
                    &lt;language&gt;en-us&lt;/language&gt;
&lt;location&gt;http://www.example.com/SitePages/Home.aspx&lt;/location&gt;
                    &lt;/client&gt;
           </client>
           <stack>
                    &lt;stack&gt;
                    &lt;function depth=&quot;0&quot; signature=&quot;CancelEvent(e)&quot;&gt;
                    &lt;![CDATA[function CancelEvent(e) {
                      ULSxSy:;
                      e.cancelBubble=true;
                      if(e.preventDefault)
                        e.preventDefault();
                      if(e.stopPropogation)
                        e.stopPropogation();
                      e.returnValue=false;
                      return false;
                      }]]&gt;
                    &lt;argument name=&quot;e&quot;
type=&quot;object&quot;&gt;{undefined}&lt;/argument&gt;
            </stack>
          <team>Example</team>
          <originalFile>init.debug.js</originalFile>
     </SendClientScriptErrorReport>
  </soap:Body>
</soap:Envelope>
```

The protocol server then responds with the following:

```
<?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>
  <SendClientScriptErrorReportResponse
xmlns="http://schemas.microsoft.com/sharepoint/diagnostics/">
    <SendClientScriptErrorReportResult>Example
     14.0.4020
     407
     'null' is null or not an object
     Microsoft Internet Explorer
     7
     init.debug.js  CancelEvent
    </SendClientScriptErrorReportResult>
  </SendClientScriptErrorReportResponse>
```

```
  </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 is provided in this appendix.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://schemas.microsoft.com/sharepoint/diagnostics/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
targetNamespace="http://schemas.microsoft.com/sharepoint/diagnostics/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:documentation>SharePoint Diagnostics Web Service</wsdl:documentation>
  <wsdl:types>
    <xs:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/diagnostics/">
      <xs:element name="SendClientScriptErrorReport">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="message" type="xs:string"/>
            <xs:element minOccurs="0" maxOccurs="1" name="file" type="xs:string"/>
            <xs:element minOccurs="1" maxOccurs="1" name="line" type="xs:int"/>
            <xs:element minOccurs="0" maxOccurs="1" name="client" type="xs:string"/>
            <xs:element minOccurs="0" maxOccurs="1" name="stack" type="xs:string"/>
            <xs:element minOccurs="0" maxOccurs="1" name="team" type="xs:string"/>
            <xs:element minOccurs="0" maxOccurs="1" name="originalFile" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="SendClientScriptErrorReportResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" maxOccurs="1" name="SendClientScriptErrorReportResult"
type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:portType name="SharePointDiagnosticsSoap">
    <wsdl:operation name="SendClientScriptErrorReport">
      <wsdl:input message="tns:SendClientScriptErrorReportSoapIn"/>
      <wsdl:output message="tns:SendClientScriptErrorReportSoapOut"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="SharePointDiagnosticsSoap" type="tns:SharePointDiagnosticsSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="SendClientScriptErrorReport">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/diagnostics/SendClientScriptErrorReport"
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="SharePointDiagnosticsSoap12" type="tns:SharePointDiagnosticsSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="SendClientScriptErrorReport">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/diagnostics/SendClientScriptErrorReport"
style="document"/>
      <wsdl:input>
        <soap12:body use="literal"/>
```

```
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:message name="SendClientScriptErrorReportSoapIn">
    <wsdl:part name="parameters" element="tns:SendClientScriptErrorReport"/>
  </wsdl:message>
  <wsdl:message name="SendClientScriptErrorReportSoapOut">
    <wsdl:part name="parameters" element="tns:SendClientScriptErrorReportResponse"/>
  </wsdl:message>
</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 SharePoint Foundation 2010
- Microsoft SharePoint Foundation 2013
- Microsoft SharePoint Server 2016

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

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

<1> Section 3.1.4.1.2.1: The string SHOULD be a valid XML fragment when all the predefined entities are replaced by their character references per XML Specification. SharePoint Foundation 2010 looks specifically for the following nodes (expressed using [XPATH] notation): `client/browser/@name`, `client/browser/@version`, and `client/language`. Other nodes in the XML fragment are ignored.

<2> Section 3.1.4.1.2.1: The string SHOULD be a valid XML fragment when all the predefined entities are replaced by their character references per XML Specification. SharePoint Foundation 2010 looks specifically for the following node (expressed using [XPATH] notation): `stack/function[@depth="0"]/@signature`. Other nodes in the XML fragment are ignored.

8 Change Tracking

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

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

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

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

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

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

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

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

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

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

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

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

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

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

9 Index

V

W