[MS-SPLCHK]:

SpellCheck Web Service 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	Editorial	Revised and edited 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	2.05	Editorial	Changed language and formatting in the technical content.
12/17/2010	2.05	None	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	2.05	None	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	2.05	None	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.6	Minor	Clarified the meaning of the technical content.
4/11/2012	2.6	None	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.6	None	No changes to the meaning, language, or formatting of the technical content.
9/12/2012	2.6	None	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.6.1	Editorial	Changed language and formatting in the technical content.
2/11/2013	2.6.1	None	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.7	Minor	Clarified the meaning of the technical content.
11/18/2013	2.7	None	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
2/10/2014	2.7	None	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.7	None	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	2.7	None	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	2.7	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 SpellCheck Web Service Protocol enables a protocol client to verify the spelling of text content.

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:

chunk: A sequence of words that are treated as a single unit by a module that checks spelling.

language code identifier (LCID): A 32-bit number that identifies the user interface human language dialect or variation that is supported by an application or a client computer.

language-detection module: A module that determines the language code identifier (LCID) of text.

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.

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 fault: A container for error and status information within a SOAP message. See [SOAP1.2-1/2007] section 5.4 for more information.

spell-check module: A module that identifies the individual words within a sequence of words, determines whether each word is spelled correctly, and provides a list of alternative spellings for each misspelled word.

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].

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

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 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.

[MS-LCID] Microsoft Corporation, "Windows Language Code Identifier (LCID) Reference".

[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/

1.2.2 Informative References

None.

1.3 Overview

This protocol enables a protocol client to verify the spelling of a set of words. The protocol allows the protocol client to pass a set of **chunks** to the protocol server and to receive back from the protocol server a set of words identified as containing errors and suggestions for correcting those errors.

A typical scenario for using this protocol is a content editing application that allows users to enter text. Such an application could use this protocol to provide users with a way to check the spelling of the text they have entered.

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].

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

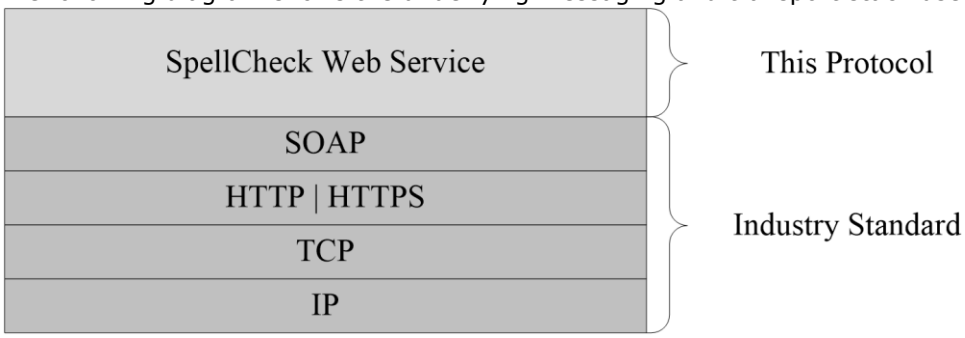

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

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

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

1.6 Applicability Statement

This protocol is intended for submitting less than 5 megabytes of text, divided into no more than 20 chunks, to the protocol server in any one invocation.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

1. **Supported transports:** This protocol uses multiple transports with SOAP as specified in Section 2.1.

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

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses **XML schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and WSDL, as specified in [WSDL].

2.2.1 Namespaces

This protocol specifies and references **XML namespaces** using the mechanisms specified in [XMLNS]. Although this document associates an **XML namespace prefix** for each XML namespace that is used, the choice of a specific XML namespace prefix is implementation-specific and not significant for interoperability. These namespaces are described in the following table.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
tns	http://schemas.microsoft.com/sharepoint/publishing/spelling/	
s	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1] [XMLSCHEMA2]
soap12	http://schemas.xmlsoap.org/wsdl/soap12/	[SOAP1.2/1] [SOAP1.2/2]
(none)	http://schemas.microsoft.com/sharepoint/publishing/spelling/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

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.

2.2.9 Common Data Structures

This specification does not define any common XML schema data structures.

3 Protocol Details

The protocol 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 that are 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, the protocol client SHOULD interpret HTTP status codes returned by the protocol server as specified in [RFC2616] (Section 10, Status Code Definitions).

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

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

3.1 Server Details

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 server MUST maintain a mapping of **LCIDs** to **spell-check modules** such that, for any LCID, at most one spell-check module is found.

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 WSDL operations as defined by this specification:

Operation	Description
SpellCheck	Runs a spell-check module on the text data contained in the **SpellCheckSoapIn** message.

3.1.4.1 SpellCheck

This operation is used to run a spell-check module on the text data contained in the **SpellCheckSoapIn** message.

```
<wsdl:operation name="SpellCheck">
    <wsdl:input message="tns:SpellCheckSoapIn" />
    <wsdl:output message="tns:SpellCheckSoapOut" />
</wsdl:operation>
```

The protocol client sends **a SpellCheckSoapIn** request message, and the protocol server MUST respond with a **SpellCheckSoapOut** response message, as follows:

1. If the user is not authenticated, the protocol server MUST send a **SpellCheckSoapOut** response with **errorCode** set to "UserNotAuthenticated" and stop further processing.
2. If the **chunksToSpell** list is empty, the protocol server MUST send a **SpellCheckSoapOut** response with **errorCode** set to "Ok" and **detectedLanguage** set to 0 and stop further processing.
3. If the **declaredLanguage** is set to -1, the protocol server MUST determine an LCID for the text in the chunks. The protocol server MUST set **detectedLanguage** to the determined value for all subsequent logic and in the **SpellCheckSoapOut** response that it will eventually send.
4. If **declaredLanguage** is not set to -1, the protocol server SHOULD<1> set **detectedLanguage** to the value of **declaredLanguage** for all subsequent logic and in the **SpellCheckSoapOut** response that it will eventually send.
5. The protocol server MUST identify the correct spell-check module by consulting its mapping of LCIDs to spell-check modules. If the protocol server's mapping of LCIDs to spell-check modules does not contain an entry for the value of **detectedLanguage**, the protocol server MUST send a **SpellCheckSoapOut** response with **errorCode** set to "SpellCheckerNotInstalled" and stop further processing.
6. The protocol server MUST run the identified spell-check module on the chunks from the **SpellCheckSoapIn** message. A successful run of the spell-check module MUST generate a list of 0 or more words identified as either **UnknownWord** or **RepeatWord**<2>, and a list of suggestions for each word identified as **UnknownWord**. If the spell-check module is successful, the protocol server MUST send a **SpellCheckSoapOut** response containing the results from the spell-check module and with **errorCode** set to Ok. If the spell-check module fails, the protocol server MUST send a **SpellCheckSoapOut** response with **errorCode** set to "UnexpectedError".

3.1.4.1.1 Messages

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

Message	Description
SpellCheckSoapIn	Request for a **SpellCheck** operation.
SpellCheckSoapOut	Response to a **SpellCheck** operation.

3.1.4.1.1.1 SpellCheckSoapIn

The request **WSDL message** for a **SpellCheck** **WSDL operation**.

The **SOAP action** value is:

```
http://schemas.microsoft.com/sharepoint/publishing/spelling/SpellCheck
```

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

3.1.4.1.1.2 SpellCheckSoapOut

The response WSDL message for a **SpellCheck** WSDL operation.

The SOAP action value is:

```
http://schemas.microsoft.com/sharepoint/publishing/spelling/SpellCheck
```

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

3.1.4.1.2 Elements

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

Element	Description
SpellCheck	The input data for a **SpellCheck** operation.
SpellCheckResponse	The result data for a **SpellCheck** operation.

3.1.4.1.2.1 SpellCheck

The input data for a **SpellCheck** WSDL operation.

```
<s:element name="SpellCheck">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="chunksToSpell" type="tns:ArrayOfString"/>
      <s:element minOccurs="1" maxOccurs="1" name="declaredLanguage" type="s:int"/>
      <s:element minOccurs="1" maxOccurs="1" name="useLad" type="s:boolean"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

chunksToSpell: The set of chunks that the protocol client requests to have checked.

declaredLanguage: The LCID of the chunks. This MUST be set to one of the following:

- A valid LCID as defined in [MS-LCID].
- -1, if the protocol client cannot determine the correct LCID.

If **declaredLanguage** is set to -1, the protocol server SHOULD<3> determine an LCID to retrieve the spell-check module.

useLad: The protocol client SHOULD<4> set **useLad** to **false**.

3.1.4.1.2.2 SpellCheckResponse

The result data for a **SpellCheck** WSDL operation.

```
<s:element name="SpellCheckResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="SpellCheckResult"
type="tns:SpellCheckResults"/>
    </s:sequence>
```

```
  </s:complexType>
</s:element>
```

SpellCheckResult: The result data for a **SpellCheck** WSDL operation.

3.1.4.1.3 Complex Types

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

Complex type	Description
SpellCheckResults	The result data for a **SpellCheck** operation.
ArrayOfSpellingErrors	A list of errors, where each **SpellingErrors** element represents a chunk and its associated spelling errors.
SpellingErrorsType	A list of spelling errors found in a specific chunk.
ArrayOfFlaggedWord	A list of **FlaggedWord** elements, where each **FlaggedWord** element represents a word that has been identified as having a spelling error.
FlaggedWordType	A word that contains a spelling error.
ArrayOfSuggestions	A list of **SpellingSuggestions** elements, where each **SpellingSuggestions** element represents a word and a set of suggested corrections for that word.
Suggestions	A list of suggestions for a specific word.
ArrayOfString	An array of **strings**.

3.1.4.1.3.1 SpellCheckResults

The result data for a **SpellCheck** WSDL operation.

```
<s:complexType name="SpellCheckResults">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="errorCode" type="s:string"/>
    <s:element minOccurs="1" maxOccurs="1" name="detectedLanguage" type="s:int"/>
    <s:element minOccurs="0" maxOccurs="1" name="spellingErrors"
type="tns:ArrayOfSpellingErrors" />
    <s:element minOccurs="0" maxOccurs="1" name="spellingSuggestions"
type="tns:ArrayOfSuggestions" />
  </s:sequence>
</s:complexType>
```

errorCode : The return value, which MUST be one of the following values:

Value	Meaning
Ok	The operation finished successfully.
SpellCheckerNotInstalled	There is no entry for the LCID of the chunks in the mapping of LCIDs to spell-check modules.

Value	Meaning
UnexpectedError	An unexpected error occurred during the operation.
UserNotAuthenticated	The user is not authenticated.

detectedLanguage: The LCID that was used to select a spell-check module.

The **detectedLanguage** MUST be set to one of the following:

- The LCID that was determined by the protocol server.
- The LCID that is specified by the protocol client under the **declaredLanguage** element.
- 0, if the protocol server sends the **SpellCheckSoapOut** response before applying either of the two previous rules.

If **detectedLanguage** is set to 0, it MUST be ignored by the protocol client.

spellingErrors: The list of errors found in the chunks. If the **chunksToSpell** is not empty and **errorCode** is set to "Ok", this element MUST be set to a list that contains values, or set to an empty list. Otherwise, the element MUST NOT be present.

spellingSuggestions: The list of suggestions for the **UnknownWord** errors found in chunks. If the **chunksToSpell** is not empty and **errorCode** is set to "Ok", this element MUST be set to a list that contains values, or set to an empty list. Otherwise, this element MUST NOT be present.

3.1.4.1.3.2 ArrayOfSpellingErrors

A list of errors, where each **SpellingErrors** element represents a chunk and its associated spelling errors.

```
<s:complexType name="ArrayOfSpellingErrors">
  <s:sequence>
    <s:element name="SpellingErrors" type="tns:SpellingErrorsType" minOccurs="0"
maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

SpellingErrors: Each **SpellingErrors** element represents one chunk and its associated spelling errors. The value of the **chunkIndex** element within the **SpellingErrors** element MUST be unique across all **SpellingErrors** elements within an **ArrayOfSpellingErrors**.

3.1.4.1.3.3 SpellingErrorsType

A list of spelling errors found in a specific chunk.

```
<s:complexType name="SpellingErrorsType">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="chunkIndex" type="s:int"/>
    <s:element minOccurs="1" maxOccurs="1" name="flaggedWords"
type="tns:ArrayOfFlaggedWord"/>
  </s:sequence>
</s:complexType>
```

chunkIndex: The index of the chunk. It MUST be greater than or equal to 0 and less than the number of chunks in the **chunksToSpell** element.

flaggedWords: The list of words from the chunk identified as containing spelling errors. Each word in this list MUST exist in the chunk.

3.1.4.1.3.4 ArrayOfFlaggedWord

A list of **FlaggedWord** elements, where each **FlaggedWord** element represents a word that has been identified as having a spelling error.

```
<s:complexType name="ArrayOfFlaggedWord">
  <s:sequence>
    <s:element name="FlaggedWord" type="tns:FlaggedWordType" minOccurs="0"
maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

FlaggedWord: A word that contains a spelling error.

3.1.4.1.3.5 FlaggedWordType

Represents a word that contains a spelling error.

```
<s:complexType name="FlaggedWordType">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="word" type="s:string"/>
    <s:element minOccurs="1" maxOccurs="1" name="type" type="tns:SpellingErrorType"/>
    <s:element minOccurs="1" maxOccurs="1" name="offset" type="s:int"/>
  </s:sequence>
</s:complexType>
```

word: The word that contains the error.

type: The type of error found.

offset: The character index into the chunk at which the word appears. This element MUST be set to the character index of the first character of the word within the chunk where 0 is the index of the first character of the chunk. This element MUST be greater than or equal to 0 and less than the number of characters in the chunk.

3.1.4.1.3.6 ArrayOfSuggestions

A list of **SpellingSuggestions** elements, where each **SpellingSuggestions** element represents a word and a set of suggested corrections for that word.

```
<s:complexType name="ArrayOfSuggestions">
  <s:sequence>
    <s:element name="SpellingSuggestions" type="tns:Suggestions" minOccurs="0"
maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

SpellingSuggestions: An unrecognized word and a set of suggested correct spellings for that word. The value of each **word** element within a **SpellingSuggestions** element MUST be unique across all **SpellingSuggestions** elements in the enclosing **ArrayOfSuggestions**.

3.1.4.1.3.7 Suggestions

A list of suggestions for a specific word.

```
<s:complexType name="Suggestions">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="word" type="s:string"/>
    <s:element minOccurs="1" maxOccurs="1" name="sug" type="tns:ArrayOfString"/>
  </s:sequence>
</s:complexType>
```

word: The word to which the suggestions apply.

sug: The list of suggestions for the word.

3.1.4.1.3.8 ArrayOfString

An array of strings.

```
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element name="string" type="s:string" minOccurs="0" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

string: The string that represents one element of the array.

3.1.4.1.4 Simple Types

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

Simple type	Description
SpellingErrorType	An enumeration of spelling error types.

3.1.4.1.4.1 SpellingErrorType

An enumeration of spelling error types.

```
<s:simpleType name="SpellingErrorType">
  <s:restriction base="s:string">
    <s:enumeration value="RepeatWord"/>
    <s:enumeration value="UnknownWord"/>
  </s:restriction>
</s:simpleType>
```

The following table specifies the allowed values of the **SpellingErrorType** enumeration.

Value	Meaning
RepeatWord	Two instances of the word were found juxtaposed within the chunk. If there are more than two instances juxtaposed, then each consecutive two instances are evaluated separately.
UnknownWord	The spell-check module identified the word as a misspelled word.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

4 Protocol Examples

A protocol client constructs the following WSDL message to detect spelling errors in a chunk of text:

```
<?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>
    <SpellCheck xmlns="http://schemas.microsoft.com/sharepoint/publishing/spelling/">
      <chunksToSpell>
        <string>Internet Explorer Enhanced Security Configuration is
an option that is provided in Windows Server 2003 operating systems.
You can use it to quickly enhance Internet Explorer security settings
for all users.
When you enable Internet Explorer Enhanced Security Configuration, it
sets Internet Explorer security settings to limit how users browse
Internet and intranet Web sites. This reduces the exposure of your
server to Web sites that might pose a security risk. For more
information, including the complete list of changes that are
implemented by Internet Explorer Enhanced Security Configuration, see Internet Explorer
Enhanced Security Configuration overview.
You might want to check for mispalled errors and duplicate duplicate words.</string>
        <string>A second mispalled word</string>
      </chunksToSpell>
      <declaredLanguage>1033</declaredLanguage>
      <useLad>false</useLad>
    </SpellCheck>
  </soap:Body>
</soap:Envelope>
```

The protocol server would then respond 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>
    <SpellCheckResponse xmlns="http://schemas.microsoft.com/sharepoint/publishing/spelling/">
      <SpellCheckResult>
        <errorCode>Ok</errorCode>
        <detectedLanguage>1033</detectedLanguage>
        <spellingErrors>
          <SpellingErrors>
            <chunkIndex>0</chunkIndex>
            <flaggedWords>
              <FlaggedWord>
                <word>mispalled</word>
                <type>UnknownWord</type>
                <offset>692</offset>
              </FlaggedWord>
              <FlaggedWord>
                <word>duplicate</word>
                <type>RepeatWord</type>
                <offset>733</offset>
              </FlaggedWord>
            </flaggedWords>
          </SpellingErrors>
          <SpellingErrors>
            <chunkIndex>1</chunkIndex>
            <flaggedWords>
              <FlaggedWord>
                <word>mispalled</word>
                <type>UnknownWord</type>
                <offset>9</offset>
```

```
              </FlaggedWord>
            </flaggedWords>
          </SpellingErrors>
        </spellingErrors>
        <spellingSuggestions>
          <SpellingSuggestions>
            <word>mispalled</word>
            <sug>
              <string>misspelled</string>
              <string>miscalled</string>
              <string>misplaced</string>
              <string>misplayed</string>
            </sug>
          </SpellingSuggestions>
        </spellingSuggestions>
      </SpellCheckResult>
    </SpellCheckResponse>
  </soap:Body>
</soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

This protocol introduces no additional security considerations beyond those applicable to its underlying protocols.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

For ease of implementation, the full WSDL and schema are provided in this appendix.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/publishing/spelling/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
targetNamespace="http://schemas.microsoft.com/sharepoint/publishing/spelling/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:documentation xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">This web service
identifies spelling mistakes and recommends suggestions for correction.</wsdl:documentation>
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/sharepoint/publishing/spelling/">
      <s:element name="SpellCheck">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="chunksToSpell"
type="tns:ArrayOfString" />
            <s:element minOccurs="1" maxOccurs="1" name="declaredLanguage" type="s:int" />
            <s:element minOccurs="1" maxOccurs="1" name="useLad" type="s:boolean" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfString">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="string" type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="SpellCheckResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="SpellCheckResult"
type="tns:SpellCheckResults" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="SpellCheckResults">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="errorCode" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="detectedLanguage" type="s:int" />
          <s:element minOccurs="0" maxOccurs="1" name="spellingErrors"
type="tns:ArrayOfSpellingErrors" />
          <s:element minOccurs="0" maxOccurs="1" name="spellingSuggestions"
type="tns:ArrayOfSuggestions" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfSpellingErrors">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="SpellingErrors"
type="tns:SpellingErrorsType" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="SpellingErrorsType">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="chunkIndex" type="s:int" />
          <s:element minOccurs="1" maxOccurs="1" name="flaggedWords"
type="tns:ArrayOfFlaggedWord" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfFlaggedWord">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="FlaggedWord"
type="tns:FlaggedWordType" />
        </s:sequence>
```

```
      </s:complexType>
      <s:complexType name="FlaggedWordType">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="word" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="type" type="tns:SpellingErrorType" />
          <s:element minOccurs="1" maxOccurs="1" name="offset" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="SpellingErrorType">
        <s:restriction base="s:string">
          <s:enumeration value="RepeatWord" />
          <s:enumeration value="UnknownWord" />
        </s:restriction>
      </s:simpleType>
      <s:complexType name="ArrayOfSuggestions">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="SpellingSuggestions"
type="tns:Suggestions" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="Suggestions">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="word" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="sug" type="tns:ArrayOfString" />
        </s:sequence>
      </s:complexType>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="SpellCheckSoapIn">
    <wsdl:part name="parameters" element="tns:SpellCheck" />
  </wsdl:message>
  <wsdl:message name="SpellCheckSoapOut">
    <wsdl:part name="parameters" element="tns:SpellCheckResponse" />
  </wsdl:message>
  <wsdl:portType name="Spelling_x0020_ServiceSoap">
    <wsdl:operation name="SpellCheck">
      <wsdl:input message="tns:SpellCheckSoapIn" />
      <wsdl:output message="tns:SpellCheckSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="Spelling_x0020_ServiceSoap" type="tns:Spelling_x0020_ServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="SpellCheck">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/publishing/spelling/SpellCheck"
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="Spelling_x0020_ServiceSoap12" type="tns:Spelling_x0020_ServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="SpellCheck">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/publishing/spelling/SpellCheck"
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 SharePoint Server 2007
- Microsoft SharePoint Server 2010
- Microsoft SharePoint Server 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: In Office SharePoint Server 2007, if **useLad** is set to **true** or **declaredLanguage** is set to -1, the protocol server calls its **language-detection module** and uses the detected LCID to retrieve the spell-check module. If language-detection module fails to determine an LCID, the **declaredLanguage** is used, provided that it is not -1. If it is -1, the language of the current **website** is used as the LCID.

<2> Section 3.1.4.1: The Office SharePoint Server 2007 spell-check modules do not identify any words as being **RepeatWord**.

<3> Section 3.1.4.1.2.1: In Office SharePoint Server 2007, if **useLad** is set to true or **declaredLanguage** is set to -1, the protocol server calls its language-detection module and uses the detected LCID to retrieve the spell-check module. If language-detection module fails to determine an LCID, the **declaredLanguage** is used, provided that it is not -1. If it is -1, the language of the current website is used as the LCID.

<4> Section 3.1.4.1.2.1: In Office SharePoint Server 2007, the protocol client sets **useLad** to true if the protocol client requests that the protocol server attempt to detect the LCID of the chunks by using a language-detection module.

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