[MS-PUBWS]:
Publishing Web Service Protocol Specification

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
10/06/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.02	Editorial	Revised and edited the technical content
07/13/2009	1.03	Major	Revised and edited the technical content
08/28/2009	1.04	Editorial	Revised and edited the technical content
11/06/2009	1.05	Editorial	Revised and edited the technical content
02/19/2010	2.0	Minor	Updated 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	Minor	Updated the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in 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

This document specifies the Publishing Web Service Protocol, which enables a protocol client to perform tasks related to template-based published content on a protocol server. The protocol client receives information about one or more objects from the protocol server, and the protocol client submits to the protocol server changes to the states of one or more objects.

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)
XML

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

content type
content type identifier
deployment package
descendant content type
displayed version
document
document library
document stream
field
file
folder
list
list item
major version
minor version
moderated object
moderation status
page layout
site
site collection
site-relative URL
SOAP (Simple Object Access Protocol)
SOAP action
SOAP body
SOAP fault
SOAP message
URL (Uniform Resource Locator)
WSDL (Web Services Description Language)
WSDL operation
XML namespace
XML namespace prefix
XML Schema

The following terms are specific to this document:

content placeholder: A region within a page layout that is populated dynamically with the value of the publishing page field to which it is bound.

publishing object: Any file, document, or list item that is versioned or moderated, or has a publishing schedule.

publishing page: A document that binds to a page layout to generate an HTML page for display to a reader. Publishing pages have specific fields that contain the content that is displayed in an HTML page.

source variation site: A Web site (2) that contains a collection of publishing pages to be copied to other sites, which are referred to as target variation sites. After the publishing pages are copied to a target variation site, they can be translated into another language. See also target variation site.

target variation site: A Web site (2) to which a collection of publishing pages were copied from another site, which is referred to as a source variation site. See also source variation site.

variations: An application that facilitates translation and related management processes for Web sites (2) and publishing pages. It can be used to copy content from one site, which is referred to as the source variation site, to one or more other sites, which are referred to as target variation sites. After the content is copied, it can be translated into different languages for those target variation sites.

XMLHTTPRequest (XHR): A software component that is used by browser-based scripts to transfer data between a Web browser and a Web server.

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-PRIMEPF] Microsoft Corporation, "Deployment Package Format Specification", June 2008.

[MS-WSSFO] Microsoft Corporation, "Windows SharePoint Services (WSS): File Operations Database Communications Protocol Specification", April 2008.

[MS-WSSTS] Microsoft Corporation, "Windows SharePoint Services Technical Specification", June 2008.

[RFC1766] Alvestrand, H., "Tags for the Identification of Languages", RFC 1766, March 1995, http://www.ietf.org/rfc/rfc1766.txt

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

[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 provides publishing and content management-related functionality. Operations are exposed as **WSDL operations** that are initiated by the client. The protocol client sends a **SOAP message** to the protocol server to execute a specific operation using the supplied parameters. When the operation finishes execution, the protocol server sends a SOAP message that contains the optional output parameters to the protocol client. The Publishing Web Service protocol defines nine operations which are divided into four specific classes.

1.3.1 Page Layout Operations

This protocol exposes three operations associated with **page layout**. The protocol client can create page layouts, disconnect **publishing pages** from page layouts and reconnect publishing pages to page layouts.

1.3.2 Status Operations

This protocol exposes three operations which retrieve specific properties from a **publishing object** or set of publishing objects that are stored on the protocol server.

1.3.3 Translation Operations

This protocol exposes two operations that export and import multilingual content. These operations are associated with a process or processes that export, translate, and then import publishing pages associated with a **source variation site**. Content is exported and imported as a **deployment package (1)**.

1.3.4 Wait Operation

This operation causes the protocol server to wait for a specific amount of time before sending an empty response to the protocol client.

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:

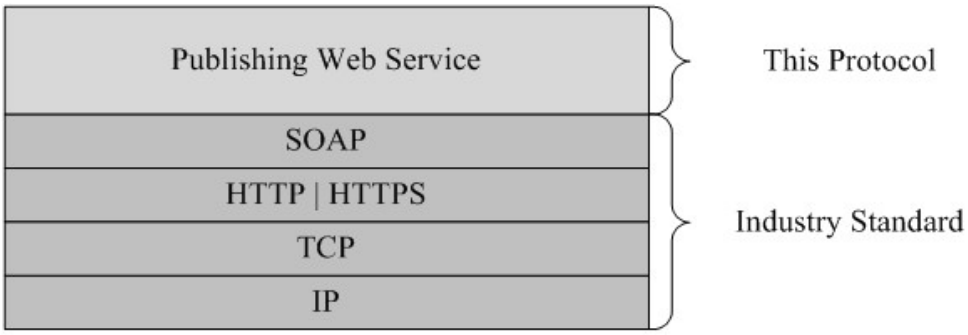

Figure 1: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

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

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

1.6 Applicability Statement

This protocol is used in several independent scenarios associated with the classes of operations described in section 1.3.

The **Page Layout** operations are used by a remote management suite to create new page layouts and to disconnect publishing pages from page layouts and reconnect publishing pages to page layouts.

The **Status** operations provide remote access to versioning, moderation, and scheduling information to drive external workflows and for views that summarize information about publishing objects.

The **Translation** operations are used with an automated translation workflow.

The **Wait** operation is used in browser scripts by the synchronous **XMLHTTPRequest (XHR)** object to block the current thread of execution for a specific amount of time. Any compliant protocol client that adheres to this specification can use the operation.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- **Supported Transports:** This protocol uses multiple transports with SOAP as described in section 2.1.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

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

Protocol messages MUST be formatted as specified in [SOAP1.1] section 4 SOAP Envelope, or in [SOAP1.2/1] section 5 SOAP Message Construct. Protocol server faults MUST be sent to the protocol client using either HTTP status codes as specified in [RFC2616] section 10 Status Code Definitions, or using **SOAP faults** as specified 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 structures used by this protocol. The syntax of the structures uses **XML Schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and **WSDL**, as specified in [WSDL].

2.2.1 Namespaces

This section specifies **XML namespaces** using the mechanisms specified in [XMLNS]. Although each specific **XML namespace prefix** is uniquely associated with each XML namespace, the specific XML namespace prefix choice is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
soap	http://schemas.xmlsoap.org/wsdl/soap/	[SOAP1.1]
tns	http://schemas.microsoft.com/sharepoint/soap/	
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/soap/	
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]

2.2.2 Messages

None.

2.2.3 Elements

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

2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions specified by this protocol. XML schema complex type definitions which are used by a single operation are specified with the operation.

Complex type	Description
PublishingObjectStatus	Encapsulates several properties related to the state and status of a publishing object.
ArrayOfString	Encapsulates an array of strings.
ArrayOfPublishingObjectStatus	Encapsulates an array of **PublishingObjectStatus** elements.

2.2.4.1 PublishingObjectStatus

Represents properties of a publishing object.

```
<s:complexType name="PublishingObjectStatus">
  <s:sequence>
    <s:element name="ObjectType">
      <s:simpleType>
        <s:restriction base="s:string">
          <s:enumeration value="File"/>
          <s:enumeration value="AccessDenied"/>
          <s:enumeration value="FileNotFound"/>
          <s:enumeration value="UnrecoverableFailure"/>
          <s:enumeration value="Undefined"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
    <s:element name="Url" type="s:string" minOccurs="0"/>
    <s:element name="PublishingUrl" type="s:string" minOccurs="0"/>
    <s:element name="Description" type="s:string" minOccurs="0"/>
    <s:element name="LastMajorVersion" type="s:decimal"/>
    <s:element name="LastMajorModifiedTime" type="s:dateTime"/>
    <s:element name="LastMinorVersion" type="s:decimal"/>
    <s:element name="LastMinorModifiedTime" type="s:dateTime"/>
    <s:element name="ScheduledStartTime" type="s:dateTime"/>
    <s:element name="ScheduledEndTime" type="s:dateTime"/>
    <s:element name="ModerationStatus">
      <s:simpleType>
        <s:restriction base="s:string">
          <s:enumeration value="Approved"/>
          <s:enumeration value="Denied"/>
          <s:enumeration value="Pending"/>
          <s:enumeration value="Draft"/>
          <s:enumeration value="Scheduled"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
  </s:sequence>
</s:complexType>
```

ObjectType: An enumeration that identifies the success or failure of the operation. MUST be one of the following values:

Value	Meaning
File	The publishing object status properties were retrieved successfully, and the accompanying element values are valid.

Value	Meaning
FileNotFound	The specified publishing object does not exist or the current user is not authorized to see it.
AccessDenied	Access was denied while retrieving the status of the specified publishing object.
Undefined	The publishing object was specified by a **URL** with an unsupported scheme, the object specified is not a publishing object, or the publishing object was explicitly excluded from the result set. See section 3.1.4.8.2.1.
UnrecoverableFailure	Unrecoverable failure while attempting to fetch the status of the specified publishing object.

Url: The URL of the publishing object. If the **ObjectType** element is set to File, this element MUST be present. If the **ObjectType** element is not set to File and this element is present, the protocol client MUST ignore it.

PublishingUrl: A URL used to retrieve a human-readable HTML representation of the publishing object. If the **ObjectType** element is set to File, this element MUST be present. If the publishing object is a publishing page, the value MUST be the URL of the page itself. Otherwise, the value MUST be any URL that can be used to obtain an HTML page that provides a human-readable view of the publishing object. If the **ObjectType** element is not set to File and this element is present, the protocol client MUST ignore it.

Description: A brief message explaining the failure in the case that the **ObjectType** element is set to UnrecoverableFailure. If the **ObjectType** element is not set to UnrecoverableFailure this element MUST not be present.

LastMajorVersion: The **displayed version** of the latest **major version** of the publishing object. The integral part of the decimal contains the major version number and the fractional part contains the minor version number. In the case of a major version, the major version number MUST be an integer between 1 and 8388608 and the minor version number MUST be 0. If the publishing object does not have a major version, this element MUST be set to 0. If the **ObjectType** element is not set to File, the protocol client MUST ignore this element.

LastMajorModifiedTime: The date and time at which the most recent major version of the publishing object was modified. If the publishing object has no major version, this element MUST be set to 0001-01-01T00:00:00. If the **ObjectType** element is not set to File, the protocol client MUST ignore this element.

LastMinorVersion: The displayed version of the latest **minor version** of the publishing object. The integral part of the decimal contains the major version number and the fractional part contains the minor version number. In the case of a minor version, the major version number MUST be an integer between 0 and 8388607 and the minor version number MUST be an integer between 1 and 511. If the publishing object has no minor version, this element MUST be set to 0. If the **ObjectType** element is not set to File, the protocol client MUST ignore this element.

LastMinorModifiedTime: The date and time at which the latest minor version of the publishing object was modified. If the publishing object has no minor version, this element MUST be set to 0001-01-01T00:00:00. If the **ObjectType** element is not set to File, this element MUST be ignored by the protocol client.

ScheduledStartTime: The date and time at which the publishing object was or will be made available for viewing. If the container (the **list (1)**, **document library**, or **folder**) of the publishing object does not support scheduling, this element MUST be set to 0001-01-01T00:00:00. If the container supports scheduling but the publishing object is set to be published immediately, the

element MUST be set to 1900-01-01T00:00:00Z. If the **ObjectType** element is not set to File, this element MUST be ignored by the protocol client.

ScheduledEndTime: The date and time at which the publishing object was or will be made unavailable for viewing. If the container (the list (1), document library, or folder) of the publishing object does not support scheduling, this element MUST be set to 9999-12-31T23:59:59.9999999. If the container supports scheduling but the publishing object is set to never be un-published, this element MUST be set to 2050-01-01T00:00:00Z. If the **ObjectType** element is not set to File, this element MUST be ignored by the protocol client.

ModerationStatus: The current **moderation status** of the publishing object. MUST be one of the following values, each of which corresponds to one of the moderation status values specified in [MS-WSSFO] section 2.2.3.13.

Value	Meaning
Approved	The publishing object is available for display in public views. Corresponds to a moderation status value of 0 as specified in [MS-WSSFO] section 2.2.3.13.
Denied	The latest request for approval on the publishing object was denied. Corresponds to a moderation status value of 1 as specified in [MS-WSSFO] section 2.2.3.13.
Pending	The latest request for approval on the publishing object is pending approval. Corresponds to a moderation status value of 2 as specified in [MS-WSSFO] section 2.2.3.13.
Draft	The publishing object is currently being edited; no approval process is currently active on this item. Corresponds to a moderation status value of 3 as specified in [MS-WSSFO] section 2.2.3.13.
Scheduled	The publishing object has been approved for display and is scheduled to be automatically published at a specific time. Corresponds to a moderation status value of 4 as specified in [MS-WSSFO] section 2.2.3.13.

If the container (the list (1), document library, or folder) of the publishing object does not support **moderated objects**, this element MUST be set to Approved. If the **ObjectType** element is not set to File, this element MUST be ignored by the protocol client.

2.2.4.2 ArrayOfString

The **ArrayOfString** complex type represents a sequence of strings.

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

string: A single element in the string array.

2.2.4.3 ArrayOfPublishingObjectStatus

The **ArrayOfPublishingObjectStatus** complex type represents a sequence of **PublishingObjectStatus** elements.

```
<s:complexType name="ArrayOfPublishingObjectStatus">
```

```
  <s:sequence>
    <s:element name="PublishingObjectStatus" type="tns:PublishingObjectStatus"
nillable="true" minOccurs="0" maxOccurs="unbounded"/>
  </s:sequence>
</s:complexType>
```

PublishingObjectStatus: A single element in the PublishingObjectStatus array.

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 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 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 using either HTTP status codes or SOAP faults, as specified previously in this section.

3.1 PublishingServiceSoap Server Details

The following high-level sequence diagram specifies the operation of the protocol. All operations consist of a basic request-response pair, and the protocol server treats each request as an independent transaction that is unrelated to any previous request.

Figure 2: Sequence diagram showing the typical message pattern

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.

See [MS-WSSTS] section 2.1 for more information about the underlying abstract data model.

3.1.1.1 Page Layouts

A page layout is a **document** that contains static markup along with dynamic script and controls, including **content placeholders**. A publishing page can be associated with a page layout that serves as the default template for the page. When the page is rendered for display to a user, the content placeholders are filled with the values of specific **fields (2)** from the publishing page. Combining a page layout and a publishing page produces a single HTML page for display. Each page layout has title and description properties along with an associated **content type** that specifies the fields to which the page layout connects. Only a publishing page that has the same content type, or

a **descendant content type**, applied can be bound to the page layout. The relationships are specified in the following diagram:

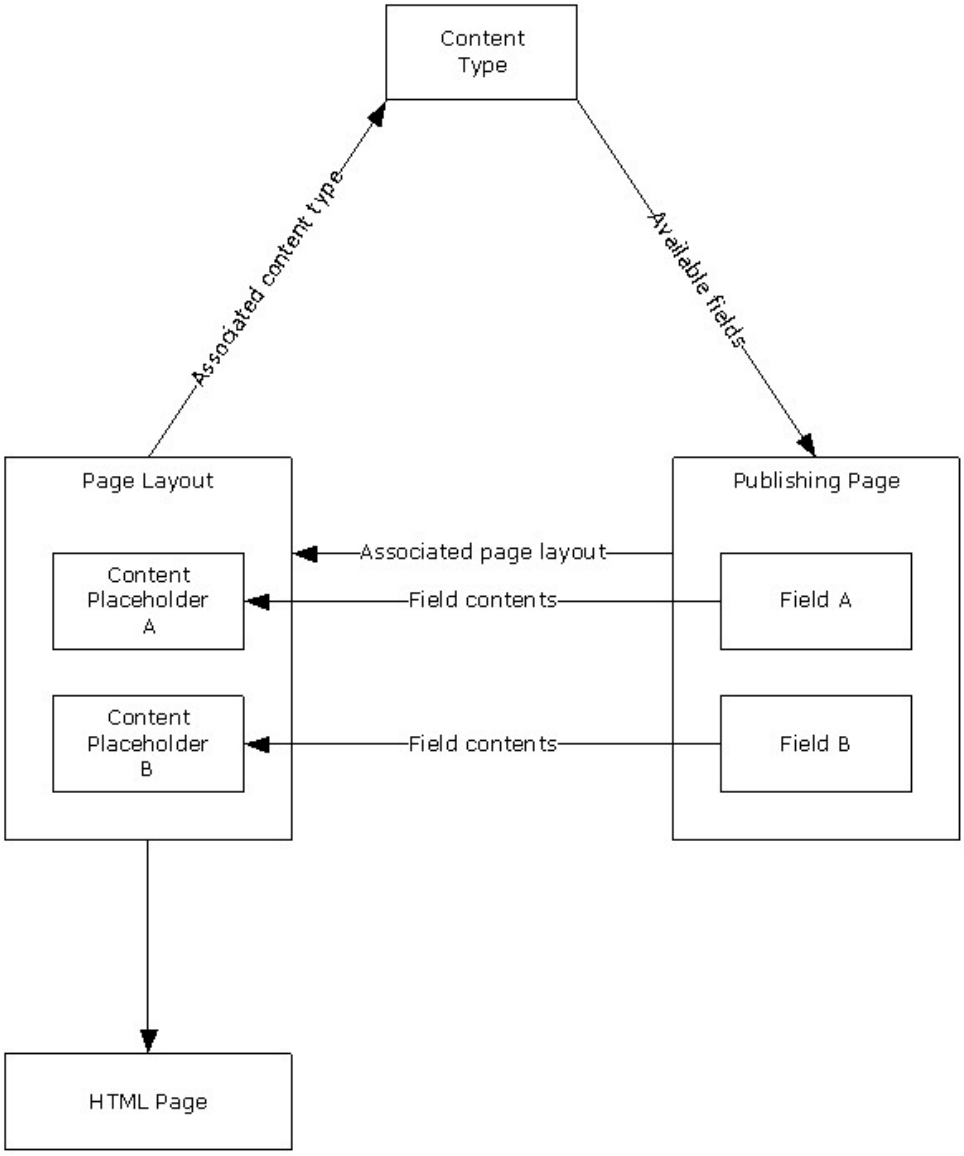

Figure 3: Relationship between page layout, content type, and publishing page

A page layout is often shared by many pages and modifying it will affect all associated pages. A publishing page can be disconnected from its page layout to support the ability to customize the page markup without modifying the page layout. When a publishing page is disconnected from its page layout, the page layout **document stream** is stored in the publishing page so that it can be edited independently of the original page layout. A publishing page can be reconnected to its page layout, which causes the customizations to be discarded. At that point, the publishing page uses the shared page layout to render the page.

3.1.1.2 Scheduling

Scheduling is a term applied to a feature that automates the process of making a **list item** or document available for viewing at a specific time, and removing the list item or document from view at a specific time. The protocol client can use the **Status Operations** to retrieve the scheduled start and end times of publishing objects.

3.1.1.3 Variations

Variations is a term applied to a feature that helps manage multilingual content. A **variations** feature automates the process of copying site (2) hierarchies and publishing pages for display in different languages. Editors modify the source variation site and publishing pages it contains, and then the variations feature automatically generates one or more **target variation sites** and publishing pages for one or more languages. The variations feature also keeps track of which publishing page fields (2) are translatable. The protocol client can use the **ExportObjects** and **ImportObjects** operations to extract, and update multilingual content created by such a feature.

3.1.2 Timers

The protocol server requires a timer to implement the functionality specified by the **Wait** operation in section 3.1.4.9. The Wait timer measures the length of time specified in the operation's input message.

3.1.3 Initialization

None.

3.1.4 Message Processing Events and Sequencing Rules

Operation	Description
CreatePageLayout	Creates a new page layout within the specified **site collection**.
DisconnectPageLayout	Disconnects the specified publishing page from its page layout.
ExportObjects	Exports publishing page content from the specified variant site (2) to a deployment package (1).
GetObjectStatus	Retrieves the status of the specified publishing object.
GetObjectStatusCollection	Retrieves the status of a number of publishing objects at the same time.
GetObjectStatusCollectionWithExclusions	Retrieves the status of a range of publishing objects while excluding a specified publishing object.
ImportObjects	Imports translated content from a deployment package (1) into the specified variant site (2).
ReconnectPageLayout	Reconnects the specified publishing object with its page layout.
Wait	Causes the protocol server to wait for the specified period of time before sending an empty response to the protocol client.

3.1.4.1 CreatePageLayout

This operation is used to create a new page layout within the specified **site collection**.

```
<wsdl:operation name="CreatePageLayout">
    <wsdl:input message="tns:CreatePageLayoutSoapIn" />
    <wsdl:output message="tns:CreatePageLayoutSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **CreatePageLayoutSoapIn** request message to the protocol server, as specified in section 3.1.4.1.1.1.

The protocol server MUST attempt to create a new page layout with the specified file name within the specified site collection, and associate it with the specified associated content type identifier.

- The protocol server MUST respect the specified file name but MAY<1> override the file name extension if one was specified.
- If title or description values are present in the input, the protocol server MUST set the corresponding page layout properties accordingly.
- If the specified site collection does not exist, the specified page layout name is already in use by a page layout in the site collection, or the specified **content type identifier** does not match the identifier of an existing content type, the protocol server MUST return a SOAP fault.

When the operation finishes execution, the protocol server MUST send a **CreatePageLayoutSoapOut** message that contains a **CreatePageLayoutResult** element set to the absolute URL of the newly created page layout, as specified in section 3.1.4.1.1.2 and section 3.1.4.1.2.2.

3.1.4.1.1 Messages

3.1.4.1.1.1 CreatePageLayoutSoapIn

The protocol client sends this message to the protocol server to initiate a **CreatePageLayout** operation.

The **SOAP action** value of the message is specified as:

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

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

3.1.4.1.1.2 CreatePageLayoutSoapOut

The protocol server sends this message to the protocol client when the **CreatePageLayout** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.1.2 Elements

3.1.4.1.2.1 CreatePageLayout

This element represents the body of the **CreatePageLayoutSoapIn** message and contains input parameters associated with the **CreatePageLayout** operation.

```
<s:element name="CreatePageLayout">
  <s:complexType>
    <s:sequence>
      <s:element name="pageLayoutName">
        <s:simpleType>
          <s:restriction base="s:string">
            <s:minLength value="1"/>
            <s:maxLength value="128"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element name="associatedContentTypeId">
        <s:simpleType>
          <s:restriction base="s:string">
            <s:pattern value="0x([0-9A-Fa-f][1-9A-Fa-f]|[1-9A-Fa-f][0-9A-Fa-f]|00[0-9A-Fa-
f]{32})*" />
            <s:minLength value="2"/>
            <s:maxLength value="1026"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element name="title" minOccurs="0">
        <s:simpleType>
          <s:restriction base="s:string">
            <s:maxLength value="255"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element name="description" type="s:string" minOccurs="0"/>
      <s:element name="siteUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

pageLayoutName: The file name of the new page layout. Additional constraints on the value of this element are specified in [MS-WSSTS] section 2.2.1.

If the file name does not specify an extension, the value MUST be 123 or fewer characters long.

associatedContentTypeId: The content type identifier of the content type to associate with the page layout. Additional constraints on the value of the element are specified in [MS-WSSTS] section 2.1.2.8.1.

title: The title of the new page layout. Additional constraints on the value of this element are specified in [MS-WSSTS] section 2.3.1, Text.

description: A description of the new page layout. Additional constraints on the value of this element are specified in [MS-WSSTS] section 2.3.1, Note.

siteUrl: The absolute URL of the site collection in which to create the page layout.

3.1.4.1.2.2 CreatePageLayoutResponse

This element represents the body of the **CreatePageLayoutSoapOut** message and contains output parameters associated with the **CreatePageLayout** operation.

```
<s:element name="CreatePageLayoutResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="CreatePageLayoutResult" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

CreatePageLayoutResult : The absolute URL of the new page layout.

3.1.4.2 DisconnectPageLayout

This operation disconnects the specified publishing page from its page layout.

```
<wsdl:operation name="DisconnectPageLayout">
    <wsdl:input message="tns:DisconnectPageLayoutSoapIn" />
    <wsdl:output message="tns:DisconnectPageLayoutSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **DisconnectPageLayoutSoapIn** request message to the protocol server, as specified in section 3.1.4.2.1.1.

The protocol server MUST attempt to disconnect the specified publishing page from its page layout.

- If the specified publishing page does not exist, if the specified publishing page is not checked out, if the specified publishing page is already disconnected from its page layout, or if the page layout associated with the publishing page does not exist, the protocol server MUST return a SOAP fault.

When the operation finishes execution, the protocol server MUST send a **DisconnectPageLayoutSoapOut** message, as specified in section 3.1.4.2.1.2.

3.1.4.2.1 Messages

3.1.4.2.1.1 DisconnectPageLayoutSoapIn

The protocol client sends this message to the protocol server to initiate a **DisconnectPageLayout** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.2.1.2 DisconnectPageLayoutSoapOut

The protocol server sends this message to the protocol client when the **DisconnectPageLayout** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.2.2 Elements

3.1.4.2.2.1 DisconnectPageLayout

This element represents the body of the **DisconnectPageLayoutSoapIn** message and contains input parameters associated with the **DisconnectPageLayout** operation.

```
<s:element name="DisconnectPageLayout">
  <s:complexType>
    <s:sequence>
      <s:element name="pageUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

pageUrl: The absolute URL of the publishing page which is to be disconnected from its page layout.

3.1.4.2.2.2 DisconnectPageLayoutResponse

This element represents the body of the **DisconnectPageLayoutSoapOut** message and contains output parameters associated with the **DisconnectPageLayout** operation.

```
<s:element name="DisconnectPageLayoutResponse">
  <s:complexType/>
</s:element>
```

3.1.4.3 ReconnectPageLayout

This operation is used to reconnect the specified publishing page with its page layout.

```
<wsdl:operation name="ReconnectPageLayout">
    <wsdl:input message="tns:ReconnectPageLayoutSoapIn" />
    <wsdl:output message="tns:ReconnectPageLayoutSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **ReconnectPageLayoutSoapIn** request message to the protocol server, as specified in section 3.1.4.3.1.1.

The protocol server MUST attempt to reconnect the publishing page to its page layout, as follows:

- If the specified publishing page does not exist, if it is not checked out, or if it is already connected to its associated page layout, the protocol server MUST return a SOAP fault.
- If the associated page layout is no longer available, the publishing page MUST be connected to any one of the available page layouts that have an associated content type matching the content type of the publishing page.

- If no such page layout can be found, the protocol server MUST return a SOAP fault.

When the operation finishes execution, the protocol server MUST send a **ReconnectPageLayoutSoapOut** message containing a **ReconnectPageLayoutResult** element, as specified in section 3.1.4.3.1.2 and section 3.1.4.3.2.2.

See section 3.1.1.1 for more information about the relationship between publishing pages, page layouts, and content types.

3.1.4.3.1 Messages

3.1.4.3.1.1 ReconnectPageLayoutSoapIn

The protocol client sends this message to the protocol server to initiate a **ReconnectPageLayout** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.3.1.2 ReconnectPageLayoutSoapOut

The protocol server sends this message to the protocol client when the **ReconnectPageLayout** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.3.2 Elements

3.1.4.3.2.1 ReconnectPageLayout

This element represents the body of the **ReconnectPageLayoutSoapIn** message and contains input parameters associated with the **ReconnectPageLayout** operation.

```
<s:element name="ReconnectPageLayout">
  <s:complexType>
    <s:sequence>
      <s:element name="pageUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

pageUrl: The absolute URL of the publishing page which is to be reconnected to its page layout.

3.1.4.3.2.2 ReconnectPageLayoutResponse

This element represents the body of the **ReconnectPageLayoutSoapOut** message and contains output parameters associated with the **ReconnectPageLayout** operation.

```
<s:element name="ReconnectPageLayoutResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="ReconnectPageLayoutResult" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

ReconnectPageLayoutResult: Specifies the page layout that the publishing page was connected to. The value MUST be composed of a condition code, followed by a comma, followed by the **site-relative URL** of the page layout.

The condition code MUST be one of the following values:

Value	Meaning
1	The page was reconnected to the same page layout to which it was connected previously.
2	The page was reconnected to a page layout different from the page layout to which it was connected previously; the original page layout was unavailable.

3.1.4.4 ExportObjects

This operation exports translatable publishing page content from a variant site (2) to a deployment package (1) for subsequent translation and importation back into the variant site. See section 3.1.1.3 for details on variations and multilingual content.

```
<wsdl:operation name="ExportObjects">
    <wsdl:input message="tns:ExportObjectsSoapIn" />
    <wsdl:output message="tns:ExportObjectsSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending an **ExportObjectsSoapIn** message to the protocol server, as specified in section 3.1.4.4.1.1.

The protocol server MUST attempt to export translatable publishing pages within the variant site to a deployment package (1). Required configuration and extension of the nominal deployment package format is specified in section 3.1.4.4.2.2.

If the specified site does not exist, the site is not a variant site, or the operation fails for any other reason, the protocol server MUST either return a SOAP fault or send an **ExportObjectSoapOut** message, as specified in section 3.1.4.4.1.2, with an empty **ExportObjectsResponse** element. The protocol client MUST NOT distinguish between the two cases.

If the operation completes successfully, the protocol server MUST send an **ExportObjectsSoapOut** message to the protocol client. The **ExportObjectsResponse** element, specified in section 3.1.4.4.2.2, MUST contain the binary data of the created deployment package (1).

3.1.4.4.1 Messages

3.1.4.4.1.1 ExportObjectsSoapIn

The protocol client sends this message to the protocol server to initiate an **ExportObjects** operation.

The SOAP action value of the message is specified as:

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

The SOAP body contains an **ExportObjects** element.

3.1.4.4.1.2 ExportObjectsSoapOut

The protocol server sends this message to the protocol client when the **ExportObjects** operation finishes execution.

The SOAP action value of the message is specified as:

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

The SOAP body contains an **ExportObjectsResponse** element.

3.1.4.4.2 Elements

3.1.4.4.2.1 ExportObjects

This element represents the message body of the **ExportObjectsSoapIn** message and contains input parameters associated with the **ExportObjects** operation.

```
<s:element name="ExportObjects">
  <s:complexType>
    <s:sequence>
      <s:element name="webUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

webUrl: The absolute URL of the variant site (2) which is to have its content exported.

3.1.4.4.2.2 ExportObjectsResponse

This element represents the body of the **ExportObjectsSoapOut** message and contains output parameters associated with the **ExportObjects** operation.

```
<s:element name="ExportObjectsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="ExportObjectsResult" type="s:base64Binary" minOccurs="0"/>
    </s:sequence>
  </s:complexType>
```

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

ExportObjectsResult: This element represents exported publishing page content. The binary value MUST be a valid deployment package (1) as specified in [MS-PRIMEPF]. Required deployment package configuration is as follows:

- The deployment package MUST contain representations of each publishing page in the site (2).
- The deployment package MUST be compressed. See [MS-PRIMEPF] section 1.3.
- The deployment package MUST contain all publishing page content, not just incremental changes. See [MS-PRIMEPF] section 2.2.4.2.
- The deployment package MUST contain all descendant objects such as content type definitions required by the publishing pages. See [MS-PRIMEPF] section 2.2.4.3.
- The deployment package MUST NOT contain any security attributes. See [MS-PRIMEPF] section 2.2.4.4.
- The deployment package MUST contain the current version of each publishing page in the site as specified in [MS-PRIMEPF] section 2.2.4.5.
- The deployment package MUST contain an additional **XML** file named VariationsLanguageSettings.xml containing content that MUST adhere to the definition of the **LanguageSettings** element as specified in section 3.1.4.4.2.3.
- If the content contains any translatable fields (2), the package MUST contain an additional XML file named TranslatableFieldSettings.xml containing content that MUST adhere to the definition of the **Fields** element as specified in section 3.1.4.4.2.4.

3.1.4.4.2.3 Language Settings

This element represents the body of the VariationsLanguageSettings.xml document and MUST adhere to the following XML schema definition (XSD).

```
<s:element name="LanguageSettings">
  <s:complexType>
    <s:sequence>
      <s:element name="SourceLanguage" type="s:string"/>
      <s:element name="TargetLanguage" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

SourceLanguage: The language of the source variation site. The value MUST be a valid language tag as specified in [RFC1766] section 2.

TargetLanguage: The language of the target variation site, and therefore the desired language of the content. The value MUST be a valid language tag as specified in [RFC1766] section 2.

3.1.4.4.2.4 Fields

This element represents the body of the TranslatableFieldSettings.xml document and MUST adhere to the following XSD.

```
<s:element name="Fields">
  <s:complexType>
    <s:sequence>
      <s:element name="Field" maxOccurs="unbounded">
        <s:complexType>
          <s:simpleContent>
            <s:extension base="s:string">              <s:attribute name="Id" use="required">
<xs:simpleType>                  <xs:restriction base="xs:string">
<xs:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-
F]{12}"/>                  </xs:restriction>                </xs:simpleType>
</s:attribute>
            </s:extension>
          </s:simpleContent>
        </s:complexType>
      </s:element>
    </s:sequence>
  </s:complexType>
</s:element>
```

Field: Each element represents a field (2) which contains a string value to translate. The element value represents the display name of the field.

Field.Id: The **GUID** of the field represented by the associated **Field** element.

3.1.4.5 ImportObjects

This operation is used to import translated content back into the variant site collection from which it was exported.

```
<wsdl:operation name="ImportObjects">
    <wsdl:input message="tns:ImportObjectsSoapIn" />
    <wsdl:output message="tns:ImportObjectsSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending an **ImportObjectsSoapIn** message to the protocol server, as specified in section 3.1.4.5.1.1.

The protocol server MUST attempt to import the specified content into the specified site collection. If an imported **list item** or **file** collides with an object that already exists in the store, any existing checked-out versions of the item MUST be checked in and the imported object MUST be stored as a new minor version of the existing document.

If the specified site collection does not exist, the format of the content is not valid, or the **ImportObjects** operation fails for any other reason, the protocol server MUST either return a SOAP fault or send an **ImportObjectsSoapOut** message, as specified in section 3.1.4.5.1.2, with the **ImportObjectsResult** element set to False. The protocol client MUST NOT distinguish between the two cases.

If the operation completes successfully, the protocol server MUST send an **ImportObjectsSoapOut** message to the protocol server. The **ImportObjectsResult** element, specified in section 3.1.4.5.2.2, MUST be set to true.

3.1.4.5.1 Messages

3.1.4.5.1.1 ImportObjectsSoapIn

The protocol client sends this message to the protocol server to initiate an **ImportObjects** operation.

The SOAP action value of the message is specified as:

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

The SOAP body contains an **ImportObjects** element.

3.1.4.5.1.2 ImportObjectsSoapOut

The protocol server sends this message to the protocol client when the **ImportObjects** operation finishes execution.

The SOAP action value of the message is specified as:

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

The SOAP body contains an **ImportObjectsResponse** element.

3.1.4.5.2 Elements

3.1.4.5.2.1 ImportObjects

This element represents the message body of the **ImportObjectsSoapIn** message and contains input parameters associated with the **ImportObjects** operation.

```
<s:element name="ImportObjects">
  <s:complexType>
    <s:sequence>
      <s:element name="siteUrl" type="s:string"/>
      <s:element name="fileContent" type="s:base64Binary"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

siteUrl: Represents the absolute URL of the site collection into which to import the content.

fileContent: Represents the packaged content to import. The binary value MUST be a valid deployment package (1) that was created using the same deployment settings as specified for the **ExportObjectsResponse** element in section 3.1.4.4.2.2.

3.1.4.5.2.2 ImportObjectsResponse

This element represents the body of the **ImportObjectsSoapOut** message and contains output parameters associated with the **ImportObjects** operation.

```
<s:element name="ImportObjectsResponse">
```

```
    <s:complexType>
      <s:sequence>
        <s:element name="ImportObjectsResult" type="s:boolean"/>
      </s:sequence>
    </s:complexType>
  </s:element>
```

ImportObjectsResult: This element indicates whether the **ImportObjects** operation was successful.

3.1.4.6 GetObjectStatus

This operation obtains status information for the specified publishing object.

```
<wsdl:operation name="GetObjectStatus">
    <wsdl:input message="tns:GetObjectStatusSoapIn" />
    <wsdl:output message="tns:GetObjectStatusSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **GetObjectStatusSoapIn** message to the protocol server, as specified in section 3.1.4.6.1.1.

The protocol server MUST attempt to build a **PublishingObjectStatus** element, as specified in section 2.2.4.1, to send to the protocol client.

- If the specified URL uses either the javascript://, mailto:// or news:// scheme, the **ObjectType** element MUST be set to "Undefined".
- If the specified URL does not refer to an object within a site collection on the protocol server, or if the protocol client is unauthorized to access it, the **ObjectType** element MUST be set to either "FileNotFound" or "AccessDenied". The protocol client MUST NOT distinguish between the two cases.
- If the specified object is found but it is not a list item, document, or file, the **ObjectType** element MUST be set to "Undefined".
- If the specified publishing object is found and all properties are retrieved successfully, the **ObjectType** element MUST be set to "File".
- If any other failures are encountered while processing the operation, the protocol server MUST return a SOAP fault.

When the operation finishes execution, the protocol server MUST send a **GetObjectStatusSoapOut** message that includes the **PublishingObjectStatus** element, as specified in section 3.1.4.6.1.2 and section 3.1.4.6.2.2.

3.1.4.6.1 Messages

3.1.4.6.1.1 GetObjectStatusSoapIn

The protocol client sends this message to the protocol server to initiate a **GetObjectStatus** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.6.1.2 GetObjectStatusSoapOut

The protocol server sends this message to the protocol client when the **GetObjectStatus** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.6.2 Elements

3.1.4.6.2.1 GetObjectStatus

This element represents the body of the **GetObjectStatusSoapIn** message and contains input parameters associated with the **GetObjectStatus** operation.

```
<s:element name="GetObjectStatus">
  <s:complexType>
    <s:sequence>
      <s:element name="objectUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

objectUrl: The absolute URL of the publishing object for which status is to be fetched.

3.1.4.6.2.2 GetObjectStatusResponse

This element represents the body of the **GetObjectStatusSoapOut** message and contains output parameters associated with the **GetObjectStatus** operation.

```
<s:element name="GetObjectStatusResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetObjectStatusResult" type="tns:PublishingObjectStatus"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetObjectStatusResult: Contains publishing object properties specified in section 2.2.4.1, **PublishingObjectStatus**.

3.1.4.7 GetObjectStatusCollection

This operation is intended to obtain status information for a collection of specified publishing objects, performing the **GetObjectStatus** operation in bulk. The operation is deprecated because it

is of no practical use as specified and there are no existing protocol clients that depend on it. A protocol server is still considered compliant if it does not support this operation.

```
<wsdl:operation name="GetObjectStatusCollection">
    <wsdl:input message="tns:GetObjectStatusCollectionSoapIn" />
    <wsdl:output message="tns:GetObjectStatusCollectionSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **GetObjectStatusCollectionSoapIn** message to the protocol server as specified in section 3.1.4.7.1.1.

Regardless of the input, the protocol server MUST return a SOAP fault.

3.1.4.7.1 Messages

3.1.4.7.1.1 GetObjectStatusCollectionSoapIn

The protocol client sends this message to the protocol server to initiate a **GetObjectStatusCollection** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.7.1.2 GetObjectStatusCollectionSoapOut

The protocol server sends this message to the protocol client when the **GetObjectStatusCollection** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.7.2 Elements

3.1.4.7.2.1 GetObjectStatusCollection

This element represents the body of the **GetObjectStatusCollectionSoapIn** message, and contains input parameters associated with the **GetObjectStatusCollection** operation.

```
<s:element name="GetObjectStatusCollection">
  <s:complexType>
    <s:sequence>
      <s:element name="objectUrls" type="tns:ArrayOfString"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

objectUrls: Each element in the collection specifies the absolute URL of a publishing object for which status is to be retrieved.

3.1.4.7.2.2 GetObjectStatusCollectionResponse

This element represents the body of the **GetObjectStatusCollectionSoapOut** message and contains output parameters associated with the **GetObjectStatusCollection** operation.

```
<s:element name="GetObjectStatusCollectionResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetObjectStatusCollectionResult"
type="tns:ArrayOfPublishingObjectStatus"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetObjectStatusCollectionResult: Each element in the collection contains content that represents publishing object properties as specified section 2.2.4.3, **ArrayOfPublishingObjectStatus**.

3.1.4.8 GetObjectStatusCollectionWithExclusions

This operation obtains status information for a collection of specified publishing objects, excluding a specified publishing object.

```
<wsdl:operation name="GetObjectStatusCollectionWithExclusions">
    <wsdl:input message="tns:GetObjectStatusCollectionWithExclusionsSoapIn" />
    <wsdl:output message="tns:GetObjectStatusCollectionWithExclusionsSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **GetObjectStatusCollectionWithExclusionsSoapIn** message to the protocol server, formatted as specified in section 3.1.4.8.1.1.

If the **thisPageUrl** element is set to a URL with a domain not equivalent to the domain of the protocol server, the protocol server MUST return a SOAP fault. Otherwise, the protocol server MUST perform the **GetObjectStatus** operation on each URL in the specified collection, as specified in section 3.1.4.6, with two modifications:

- If the **thisPageUrl** element has the same value as the current URL from the input collection, as in the URL that indentifies the object for which status is to be retrieved, the protocol server MUST set the corresponding **ObjectType** element to Undefined.
- In the case of an unexpected failure, rather than returning a SOAP fault the protocol server MUST set the corresponding **ObjectType** element to UnrecoverableFailure and set the **Description** element to a brief failure notification message.

The resulting **PublishingObjectStatus** elements MUST be added to an **ArrayOfPublishingObjectStatus** element in the same order that the corresponding input URLs were found. When the operation finishes execution, the protocol server MUST send a **GetObjectStatusCollectionWithExclusionsSoapOut** message containing the result values as specified in section 3.1.4.8.1.2 and section 3.1.4.8.2.2.

3.1.4.8.1 Messages

3.1.4.8.1.1 GetObjectStatusCollectionWithExclusionsSoapIn

The protocol client sends this message to the protocol server to initiate a **GetObjectStatusCollectionWithExclusions** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.8.1.2 GetObjectStatusCollectionWithExclusionsSoapOut

The protocol server sends this message to the protocol client when the **GetObjectStatusCollectionWithExclusions** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.8.2 Elements

3.1.4.8.2.1 GetObjectStatusCollectionWithExclusions

This element represents the body of the **GetObjectStatusCollectionWithExclusionsSoapIn** message and contains input parameters associated with the **GetObjectStatusCollectionWithExclusions** operation.

```
<s:element name="GetObjectStatusCollectionWithExclusions">
  <s:complexType>
    <s:sequence>
      <s:element name="objectUrls" type="tns:ArrayOfString"/>
      <s:element name="thisPageUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

objectUrls: Each element in the collection specifies the absolute URL of a publishing object for which status is retrieved.

thisPageUrl: The absolute URL of a publishing object that has properties which are not retrieved even if the URL is found in the **objectUrls** collection. The domain specified by this URL MUST match the domain of the protocol server.

3.1.4.8.2.2 GetObjectStatusCollectionWithExclusionsResponse

This element represents the body of the **GetObjectStatusCollectionWithExclusionsSoapOut** message and contains output parameters associated with the **GetObjectStatusCollectionWithExclusions** operation.

```
<s:element name="GetObjectStatusCollectionWithExclusionsResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetObjectStatusCollectionWithExclusionsResult"
type="tns:ArrayOfPublishingObjectStatus"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

GetObjectStatusCollectionWithExclusionsResult: Each element in the collection contains content that represents various publishing object properties as specified in section 2.2.4.3, **ArrayOfPublishingObjectStatus**, subject to the restrictions specified in section 3.1.4.8.

3.1.4.9 Wait

This operation forces the protocol server to wait for the specified amount of time before sending a response to the protocol client.

```
<wsdl:operation name="Wait">
    <wsdl:input message="tns:WaitSoapIn" />
    <wsdl:output message="tns:WaitSoapOut" />
</wsdl:operation>
```

The protocol client initiates the operation by sending a **WaitSoapIn** message to the protocol server, as specified in section 3.1.4.9.1.1.

The protocol server MUST initiate a millisecond timer that expires after the period designated by the **millisecondsToWait** element. When the timer expires, the protocol server MUST send a **WaitSoapOut** message as specified in section 3.1.4.9.1.2. If the operation fails for any reason, the protocol server MUST return a SOAP fault.

3.1.4.9.1 Messages

3.1.4.9.1.1 WaitSoapIn

The protocol client sends this message to the protocol server to initiate a **Wait** operation.

The SOAP action value of the message is specified as:

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

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

3.1.4.9.1.2 WaitSoapOut

The protocol server sends this message to the protocol client when the **Wait** operation finishes execution.

The SOAP action value of the message is specified as:

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

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

3.1.4.9.2 Elements

3.1.4.9.2.1 Wait

This element represents the body of the **WaitSoapIn** message and contains input parameters associated the **Wait** operation.

```
<s:element name="Wait">
  <s:complexType>
    <s:sequence>
      <s:element name="millisecondsToWait" type="s:int"/>
    </s:sequence>
  </s:complexType>
</s:element>
```

millisecondsToWait: The number of milliseconds to wait before responding. The value MUST be non-negative.

3.1.4.9.2.2 WaitResponse

This element represents the body of the **WaitSoapOut** message and contains output parameters associated with the **Wait** operation.

```
<s:element name="WaitResponse">
  <s:complexType/>
</s:element>
```

3.1.5 Timer Events

When the Wait timer expires, the protocol server MUST respond to the protocol client with a **WaitSoapOut** message, as specified in section 3.1.4.9.

3.1.6 Other Local Events

None.

4 Protocol Examples

The following examples describe SOAP message pairings in hypothetical scenarios. As described in section 3.1, all operations are stateless and can be used independently.

4.1 Page and Page Layout Editing Suite

Client applications can leverage the functionality of publishing pages and page layouts to create new page layouts, disconnect publishing pages from page layouts to make customizations to a specific page, and reconnect pages to layouts to discard those customizations. Using the **CreatePageLayout**, **DisconnectPageLayout**, and **ReconnectPageLayout** operations in association with document editing functionality, the user can create a client-server publishing page editing suite.

Create a new page layout:

Request Body:

```
<?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>
    <CreatePageLayout xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <pageLayoutName>TestPageLayout</pageLayoutName>
      <associatedContentTypeId>0x010100D5C2F139516B419D801AC6C18942554D
</associatedContentTypeId>
      <title>Test Page Layout</title>
      <description>Layout used to demonstrate the functioning of the CreatePageLayout
operation</description>
      <siteUrl>http://www.contoso.com</siteUrl>
    </CreatePageLayout>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <CreatePageLayoutResponse       xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <CreatePageLayoutResult>http://www.contoso.com/PageLayouts/TestPageLayout.aspx
</CreatePageLayoutResult>
    </CreatePageLayoutResponse>
  </soap:Body>
</soap:Envelope>
```

Edit a publishing page and associate it with this page layout independent of the protocol. If customizations are desired, disconnect the publishing page from its page layout:

Request Body:

```
<?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>
    <DisconnectPageLayout       xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <pageUrl>http://www.contoso.com/Pages/TestPage.aspx</pageUrl>
    </DisconnectPageLayout>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <DisconnectPageLayoutResponse
xmlns="http://schemas.microsoft.com/sharepoint/soap/"/>
  </soap:Body>
</soap:Envelope>
```

Edit the publishing page independent of the protocol. If customizations are no longer desired, reconnect the publishing page with its page layout:

Request Body:

```
<?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>
    <ReconnectPageLayout       xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <pageUrl>http://www.contoso.com/Pages/TestPage.aspx</pageUrl>
    </ReconnectPageLayout>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <ReconnectPageLayoutResponse      xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <ReconnectPageLayoutResult>1,PageLayouts/TestPageLayout.aspx
</ReconnectPageLayoutResult>
    </ReconnectPageLayoutResponse>
  </soap:Body>
</soap:Envelope>
```

4.2 Translation Workflow

The processes of collecting and extracting content for translation and importing translated content can be automated by a remote protocol client that uses the **ExportObjects** and **ImportObjects** operations.

Extract translatable content

Request Body:

```
<?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>
    <ExportObjects xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <webUrl>http://office/ja-JP</webUrl>
    </ExportObjects>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <ExportObjectsResponse       xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <ExportObjectsResult>ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuv
wxyz0123456789+/...</ExportObjectsResult>
    </ExportObjectsResponse>
  </soap:Body>
</soap:Envelope>
```

Translate content independent of to the protocol. Import translated content

Request Body:

```
<?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>
    <ImportObjects xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <siteUrl>http://office/ja-JP</siteUrl>
      <fileContent>ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123
456789+/...</fileContent>
    </ImportObjects>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <ImportObjectsResponse       xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <ImportObjectsResult>true</ImportObjectsResult>
    </ImportObjectsResponse>
  </soap:Body>
</soap:Envelope>
```

Publish translated content for viewing independently of external to the protocol.

4.3 Publishing Dashboard

Protocol clients can retrieve scheduling, moderation, and versioning information about publishing objects from many different protocol servers to build current dashboard views.

Request Body:

```
<?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>
    <GetObjectStatus xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <objectUrl>http://www.contoso.com/Pages/TestPage.aspx</objectUrl>
    </GetObjectStatus>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <GetObjectStatusResponse      xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <GetObjectStatusResult>
        <ObjectType>File</ObjectType>
        <Url>http://www.contoso.com/Pages/TestPage.aspx</Url>
        <PublishingUrl>http://office/en-us/TestPage.aspx</PublishingUrl>
        <LastMajorVersion>3.0</LastMajorVersion>
        <LastMajorModifiedTime>2008-02-01T19:54:04</LastMajorModifiedTime>
        <LastMinorVersion>0</LastMinorVersion>
        <LastMinorModifiedTime>0001-01-01T00:00:00</LastMinorModifiedTime>
        <ScheduledStartTime>1900-01-01T00:00:00Z</ScheduledStartTime>
        <ScheduledEndTime>2050-01-01T00:00:00Z</ScheduledEndTime>
        <ModerationStatus>Approved</ModerationStatus>
      </GetObjectStatusResult>
    </GetObjectStatusResponse>
  </soap:Body>
</soap:Envelope>
```

4.4 Script Sleep

The **Wait** operation is most commonly used by browser-based scripts associated with the synchronous XMLHTTPRequest (XHR) object to force the current script thread to stop execution for a specific amount of time.

Request Body:

```
<?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>
    <Wait xmlns="http://schemas.microsoft.com/sharepoint/soap/">
      <millisecondsToWait>5000</millisecondsToWait >
    </Wait>
  </soap:Body>
</soap:Envelope>
```

Response Body:

```
<?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>
    <WaitResponse xmlns="http://schemas.microsoft.com/sharepoint/soap/"/>
  </soap:Body>
</soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

Particular care is to be taken with authentication and authorization, or protocol clients using the **Wait** operation, because malicious client computers can use up protocol server resources by repeatedly running an operation with long wait parameters.

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 is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://schemas.microsoft.com/sharepoint/soap/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
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:element name="Wait">
  <s:complexType>
    <s:sequence>
      <s:element name="millisecondsToWait" type="s:int"/>
    </s:sequence>
  </s:complexType>
</s:element>
      <s:element name="WaitResponse">
  <s:complexType/>
</s:element>
      <s:element name="GetObjectStatus">
  <s:complexType>
    <s:sequence>
      <s:element name="objectUrl" type="s:string"/>
    </s:sequence>
  </s:complexType>
</s:element>
      <s:element name="GetObjectStatusResponse">
  <s:complexType>
    <s:sequence>
      <s:element name="GetObjectStatusResult" type="tns:PublishingObjectStatus"/>
    </s:sequence>
  </s:complexType>
</s:element>
      <s:complexType name="PublishingObjectStatus">
  <s:sequence>
    <s:element name="ObjectType">
      <s:simpleType>
        <s:restriction base="s:string">
          <s:enumeration value="File"/>
          <s:enumeration value="AccessDenied"/>
          <s:enumeration value="FileNotFound"/>
          <s:enumeration value="UnrecoverableFailure"/>
          <s:enumeration value="Undefined"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
    <s:element name="Url" type="s:string" minOccurs="0"/>
    <s:element name="PublishingUrl" type="s:string" minOccurs="0"/>
    <s:element name="Description" type="s:string" minOccurs="0"/>
    <s:element name="LastMajorVersion" type="s:decimal"/>
    <s:element name="LastMajorModifiedTime" type="s:dateTime"/>
    <s:element name="LastMinorVersion" type="s:decimal"/>
    <s:element name="LastMinorModifiedTime" type="s:dateTime"/>
    <s:element name="ScheduledStartTime" type="s:dateTime"/>
```

```
      <s:element name="ScheduledEndTime" type="s:dateTime"/>
      <s:element name="ModerationStatus">
        <s:simpleType>
          <s:restriction base="s:string">
            <s:enumeration value="Approved"/>
            <s:enumeration value="Denied"/>
            <s:enumeration value="Pending"/>
            <s:enumeration value="Draft"/>
            <s:enumeration value="Scheduled"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
    </s:sequence>
  </s:complexType>
        <s:element name="GetObjectStatusCollection">
    <s:complexType>
      <s:sequence>
        <s:element name="objectUrls" type="tns:ArrayOfString"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="GetObjectStatusCollectionResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="GetObjectStatusCollectionResult"
  type="tns:ArrayOfPublishingObjectStatus"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:complexType name="ArrayOfString">
    <s:sequence>
      <s:element name="string" type="s:string" nillable="true" minOccurs="0"
  maxOccurs="unbounded"/>
    </s:sequence>
  </s:complexType>
  <s:complexType name="ArrayOfPublishingObjectStatus">
    <s:sequence>
      <s:element name="PublishingObjectStatus" type="tns:PublishingObjectStatus"
  nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    </s:sequence>
  </s:complexType>
  <s:element name="GetObjectStatusCollectionWithExclusions">
    <s:complexType>
      <s:sequence>
        <s:element name="objectUrls" type="tns:ArrayOfString"/>
        <s:element name="thisPageUrl" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="GetObjectStatusCollectionWithExclusionsResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="GetObjectStatusCollectionWithExclusionsResult"
  type="tns:ArrayOfPublishingObjectStatus"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="ExportObjects">
    <s:complexType>
```

```
      <s:sequence>
        <s:element name="webUrl" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="ExportObjectsResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="ExportObjectsResult" type="s:base64Binary" minOccurs="0"/>
      </s:sequence>
    </s:complexType>
  </s:element>
  <s:element name="ImportObjects">
    <s:complexType>
      <s:sequence>
        <s:element name="siteUrl" type="s:string"/>
        <s:element name="fileContent" type="s:base64Binary"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="ImportObjectsResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="ImportObjectsResult" type="s:boolean"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="CreatePageLayout">
    <s:complexType>
      <s:sequence>
        <s:element name="pageLayoutName">
          <s:simpleType>
            <s:restriction base="s:string">
              <s:minLength value="1"/>
              <s:maxLength value="128"/>
            </s:restriction>
          </s:simpleType>
        </s:element>
        <s:element name="associatedContentTypeId">
          <s:simpleType>
            <s:restriction base="s:string">
              <s:pattern value="0x([0-9A-Fa-f][1-9A-Fa-f]|[1-9A-Fa-f][0-9A-Fa-f]|00[0-9A-Fa-
f]{32})*" />
              <s:minLength value="2"/>
              <s:maxLength value="1026"/>
            </s:restriction>
          </s:simpleType>
        </s:element>
        <s:element name="title" minOccurs="0">
          <s:simpleType>
            <s:restriction base="s:string">
              <s:maxLength value="255"/>
            </s:restriction>
          </s:simpleType>
        </s:element>
        <s:element name="description" type="s:string" minOccurs="0"/>
        <s:element name="siteUrl" type="s:string"/>
      </s:sequence>
    </s:complexType>
```

```
  </s:element>
        <s:element name="CreatePageLayoutResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="CreatePageLayoutResult" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="DisconnectPageLayout">
    <s:complexType>
      <s:sequence>
        <s:element name="pageUrl" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="DisconnectPageLayoutResponse">
    <s:complexType/>
  </s:element>
  <s:element name="ReconnectPageLayout">
    <s:complexType>
      <s:sequence>
        <s:element name="pageUrl" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
        <s:element name="ReconnectPageLayoutResponse">
    <s:complexType>
      <s:sequence>
        <s:element name="ReconnectPageLayoutResult" type="s:string"/>
      </s:sequence>
    </s:complexType>
  </s:element>
    </s:schema>
    </wsdl:types>
    <wsdl:message name="WaitSoapIn">
      <wsdl:part name="parameters" element="tns:Wait" />
    </wsdl:message>
    <wsdl:message name="WaitSoapOut">
      <wsdl:part name="parameters" element="tns:WaitResponse" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusSoapIn">
      <wsdl:part name="parameters" element="tns:GetObjectStatus" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusSoapOut">
      <wsdl:part name="parameters" element="tns:GetObjectStatusResponse" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusCollectionSoapIn">
      <wsdl:part name="parameters" element="tns:GetObjectStatusCollection" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusCollectionSoapOut">
      <wsdl:part name="parameters" element="tns:GetObjectStatusCollectionResponse" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusCollectionWithExclusionsSoapIn">
      <wsdl:part name="parameters" element="tns:GetObjectStatusCollectionWithExclusions" />
    </wsdl:message>
    <wsdl:message name="GetObjectStatusCollectionWithExclusionsSoapOut">
      <wsdl:part name="parameters"
  element="tns:GetObjectStatusCollectionWithExclusionsResponse" />
    </wsdl:message>
```

```
<wsdl:message name="ExportObjectsSoapIn">
  <wsdl:part name="parameters" element="tns:ExportObjects" />
</wsdl:message>
<wsdl:message name="ExportObjectsSoapOut">
  <wsdl:part name="parameters" element="tns:ExportObjectsResponse" />
</wsdl:message>
<wsdl:message name="ImportObjectsSoapIn">
  <wsdl:part name="parameters" element="tns:ImportObjects" />
</wsdl:message>
<wsdl:message name="ImportObjectsSoapOut">
  <wsdl:part name="parameters" element="tns:ImportObjectsResponse" />
</wsdl:message>
<wsdl:message name="CreatePageLayoutSoapIn">
  <wsdl:part name="parameters" element="tns:CreatePageLayout" />
</wsdl:message>
<wsdl:message name="CreatePageLayoutSoapOut">
  <wsdl:part name="parameters" element="tns:CreatePageLayoutResponse" />
</wsdl:message>
<wsdl:message name="DisconnectPageLayoutSoapIn">
  <wsdl:part name="parameters" element="tns:DisconnectPageLayout" />
</wsdl:message>
<wsdl:message name="DisconnectPageLayoutSoapOut">
  <wsdl:part name="parameters" element="tns:DisconnectPageLayoutResponse" />
</wsdl:message>
<wsdl:message name="ReconnectPageLayoutSoapIn">
  <wsdl:part name="parameters" element="tns:ReconnectPageLayout" />
</wsdl:message>
<wsdl:message name="ReconnectPageLayoutSoapOut">
  <wsdl:part name="parameters" element="tns:ReconnectPageLayoutResponse" />
</wsdl:message>
<wsdl:portType name="PublishingServiceSoap">
  <wsdl:operation name="Wait">
    <wsdl:input message="tns:WaitSoapIn" />
    <wsdl:output message="tns:WaitSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetObjectStatus">
    <wsdl:input message="tns:GetObjectStatusSoapIn" />
    <wsdl:output message="tns:GetObjectStatusSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetObjectStatusCollection">
    <wsdl:input message="tns:GetObjectStatusCollectionSoapIn" />
    <wsdl:output message="tns:GetObjectStatusCollectionSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="GetObjectStatusCollectionWithExclusions">
    <wsdl:input message="tns:GetObjectStatusCollectionWithExclusionsSoapIn" />
    <wsdl:output message="tns:GetObjectStatusCollectionWithExclusionsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ExportObjects">
    <wsdl:input message="tns:ExportObjectsSoapIn" />
    <wsdl:output message="tns:ExportObjectsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="ImportObjects">
    <wsdl:input message="tns:ImportObjectsSoapIn" />
    <wsdl:output message="tns:ImportObjectsSoapOut" />
  </wsdl:operation>
  <wsdl:operation name="CreatePageLayout">
    <wsdl:input message="tns:CreatePageLayoutSoapIn" />
    <wsdl:output message="tns:CreatePageLayoutSoapOut" />
  </wsdl:operation>
```

```
    <wsdl:operation name="DisconnectPageLayout">
      <wsdl:input message="tns:DisconnectPageLayoutSoapIn" />
      <wsdl:output message="tns:DisconnectPageLayoutSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="ReconnectPageLayout">
      <wsdl:input message="tns:ReconnectPageLayoutSoapIn" />
      <wsdl:output message="tns:ReconnectPageLayoutSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="PublishingServiceSoap" type="tns:PublishingServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="Wait">
      <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Wait"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetObjectStatus">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatus" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetObjectStatusCollection">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatusCollection"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetObjectStatusCollectionWithExclusions">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatusCollectionWithExclusi
ons" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ExportObjects">
      <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ExportObjects"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
```

```
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ImportObjects">
      <soap:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/ImportObjects"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreatePageLayout">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/CreatePageLayout" style="document"
/>
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DisconnectPageLayout">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/DisconnectPageLayout"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ReconnectPageLayout">
      <soap:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ReconnectPageLayout"
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="PublishingServiceSoap12" type="tns:PublishingServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="Wait">
      <soap12:operation soapAction="http://schemas.microsoft.com/sharepoint/soap/Wait"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="GetObjectStatus">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatus" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetObjectStatusCollection">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatusCollection"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetObjectStatusCollectionWithExclusions">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/GetObjectStatusCollectionWithExclusi
ons" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ExportObjects">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ExportObjects" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ImportObjects">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ImportObjects" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="CreatePageLayout">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/CreatePageLayout" style="document"
/>
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
```

```
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="DisconnectPageLayout">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/DisconnectPageLayout"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="ReconnectPageLayout">
      <soap12:operation
soapAction="http://schemas.microsoft.com/sharepoint/soap/ReconnectPageLayout"
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® Designer 2007
- Microsoft® SharePoint® Designer 2010
- Microsoft® Office SharePoint® Server 2007
- Microsoft® SharePoint® Server 2010

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: Office SharePoint Server 2007 and SharePoint Server 2010 respect the specified file name if it ends with ".aspx". Otherwise, ".aspx" is appended to the specified file name.

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