[MS-OXOUM]:
Voice Mail and Fax Objects Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

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

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

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

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1	Major	Initial Availability
04/25/2008	0.2		Revised and updated property names and other technical content.
06/27/2008	1.0		Initial Release.
08/06/2008	1.01		Updated references to reflect date of initial release.
09/03/2008	1.02		Revised and edited technical content.
12/03/2008	1.03		Minor editorial fixes.
03/04/2009	1.04		Revised and edited technical content.
04/10/2009	2.0		Updated technical content and applicable product releases.
07/15/2009	3.0	Major	Revised and edited for technical content.
11/04/2009	3.0.1	Editorial	Revised and edited the technical content.
02/10/2010	3.1.1	Minor	Updated the technical content.
05/05/2010	3.2.0	Minor	Updated the technical content.
08/04/2010	3.2.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/03/2010	3.2.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	4.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Voice Mail and Fax Objects Protocol provides the methods that servers use to manipulate **Unified Messaging** objects.

Sections 1.8, 2, and 3 of this specification are normative and contain RFC 2119 language. Section 1.5 and 1.9 are also normative but cannot contain RFC 2119 language. All other sections and examples in this specification are informative.

1.1 Glossary

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

binary large object (BLOB)
codec
Contact object
mailbox
message class
Message object
recipient
rights-managed e-mail message
Simple Mail Transfer Protocol (SMTP)
special folder
Unified Messaging

The following terms are specific to this document:

fax message: A Message object that contains a digital representation of content received from a fax machine.

missed call notification: A Message object that is intended to convey information about a call that was missed. The Message object contains information about the calling party and the time of the call, but does not contain audio content.

voice message: A Message object that contains audio content recorded by a calling party.

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.

[ASF] Microsoft Corporation, "Advanced Systems Format Specification", December 2004, http://download.microsoft.com/download/7/9/0/790fecaa-f64a-4a5e-a430-0bccdab3f1b4/ASF_Specification.doc

If you have any trouble finding [ASF], please check here.

[G711] ITU-T, "Pulse code modulation (PCM) of voice frequencies", Recommendation G.711, June, 1990, http://www.itu.int/rec/T-REC-G.711-198811-I/en

[GSM610] ETSI, "European digital cellular telecommunications system (Phase 1); Full rate speech; Transcoding (GSM 06.10)", February 1992, http://pda.etsi.org/pda/home.asp?wki_id=v9jLO9Nb7wSVbbYKNyW

[MS-OXCMSG] Microsoft Corporation, "Message and Attachment Object Protocol Specification", June 2008.

[MS-OXOCFG] Microsoft Corporation, "Configuration Information Protocol Specification", June 2008.

[MS-OXOMSG] Microsoft Corporation, "E-Mail Object Protocol Specification", June 2008.

[MS-OXORMMS] Microsoft Corporation, "Rights-Managed E-Mail Object Protocol Specification", June 2008.

[MS-OXOSFLD] Microsoft Corporation, "Special Folders Protocol Specification", June 2008.

[MS-OXPROPS] Microsoft Corporation, "Exchange Server Protocols Master Property List", April 2008.

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

[WAVE] IBM Corporation and Microsoft Corporation, "Multimedia Programming Interface and Data Specifications 1.0", August 1991, http://www-mmsp.ece.mcgill.ca/Documents/AudioFormats/WAVE/Docs/riffmci.pdf

1.2.2 Informative References

[MS-OXGLOS] Microsoft Corporation, "Exchange Server Protocols Master Glossary", April 2008.

1.3 Overview

Unified Messaging objects are items created on behalf of telephone callers or fax senders by the server. These objects are stored in the called party's **mailbox** on the server.

The server creates three types of Unified Messaging objects: **voice messages**, **fax messages**, and **missed call notifications**.

1.4 Relationship to Other Protocols

The Voice Mail and Fax Objects Protocol relies on an understanding of how to work with properties, folders, messages, and attachments (for more information, see [MS-OXPROPS], [MS-OXOSFLD], and [MS-OXCMSG], respectively).

The Voice Mail and Fax Objects Protocol uses the Message and Attachment Object Protocol, as described in [MS-OXCMSG], as a transport protocol between the client and the server.

1.5 Prerequisites/Preconditions

None.

1.6 Applicability Statement

This protocol can be used to show the electronic equivalent of telephony-based messages, such as voice messages, fax messages, and missed call notifications.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

This protocol does not provide any extensibility beyond that which is described in [MS-OXCMSG].

1.9 Standards Assignments

None.

2 Messages

There are three types of Unified Messaging objects: voice messages, missed call notifications, and fax messages.

2.1 Transport

The Voice Mail and Fax Objects Protocol uses the Message and Attachment Object Protocol, as specified in [MS-OXCMSG], to create and store the three types of Unified Messaging objects.

2.2 Message Syntax

2.2.1 Voice Message

Unlike with many other client-server objects, the server creates Unified Messaging objects. The server MUST include the general properties, as specified in [MS-OXCMSG] section 2.2.1.1. The server SHOULD also set the submission properties, as specified in [MS-OXOMSG] section 2.2.3.

2.2.1.1 Message Classes

For voice messages, the server MUST set the value of the **PidTagMessageClass** property ([MS-OXOMSG] section 2.2.2.1) to the following:

- **IPM.Note.Microsoft.Voicemail.UM.CA** for original messages taken with audio content by telephone.
- **IPM.Note.Microsoft.Voicemail.UM** for original messages taken with audio content by telephone, but not as a result of call answering (for example, if the **recipient's** phone did not ring).
- The value of the original **PidTagMessageClass** property suffixed with .Microsoft.Voicemail for messages with audio content that was created in response to other messages. For example, a voice reply to a message of type **IPM.Note** has the type **IPM.Note.Microsoft.Voicemail**.

For fax messages, the server MUST set the value of the **PidTagMessageClass** property to the following:

- **IPM.Note.Microsoft.FAX.CA**

For missed call notifications, the server MUST set the value of the **PidTagMessageClass** property to the following:

- **IPM.Note.Microsoft.Missed.Voice**

2.2.1.2 Attachments

For messages with audio content, the server MUST add the audio content as a file attachment on the message, in accordance with the procedures for attachment handling, as specified in [MS-OXCMSG] section 3.1.4.12.

The attachment file MUST be in either the WAV file format (as specified in [WAVE]), the ASF file format (as specified in [ASF]), or the MP3 file format<1>.

If in the WAV format, the audio **codec** MUST be either G.711 a-law, G.711 m-law, or GSM 6.10 (for details, see [G711] and [GSM610]). If in the ASF file format, the codec MUST be the Windows Media Audio 9 Voice or WMA 2 codec.

In addition to the common properties on the attachment, the server MUST set two other properties on the attachment, as follows:

- **PidTagAttachLongFilename** ([MS-OXPROPS] section 2.659): Set to a unique name in the attachment collection of the message. To function properly, the file name MUST be unique for the attachment order logic specified in section 2.2.1.3. The file extension MUST be ".WAV" for files in the WAV format, MUST be ".wma" for files in the ASF format, and MUST be ".mp3" for files in the MP3 format.
- **PidTagAttachMimeTag** ([MS-OXPROPS] section 2.666): Set to reflect the audio content type of the message.
- For WMA 9 Voice encoded messages, this value MUST be "audio/wma".
- For GSM 6.10 encoded messages, this value MUST be "audio/gsm".
- For G.711 encoded messages, this value MUST be "audio/WAV".
- For MP3 encoded messages, this value MUST be "audio/mp3".

In some situations, a client or server can add more than one audio attachment to a particular message. For example, a voice reply to a voice message can include the original voice content for reference. In such situations, the server SHOULD add an attachment for each voice segment and follow the order that specifies the procedure, as specified in section 2.2.1.3.

2.2.1.3 Attachment Order

The server MUST create the **PidTagVoiceMessageAttachmentOrder** property ([MS-OXPROPS] section 2.1176) on any message that contains audio attachments. Both clients and servers SHOULD consult this property to determine the order in which to play any audio attachments, including cases in which there is only one attachment.

The content of the property is a list of PidTagAttachLongFilename values ([MS-OXPROPS] section 2.659) for audio file attachments that are to be played as part of the message. The order MUST be the reverse of the order in which the attachments were added; that is, the most recently added message first, the next most recently added message second, and so on.

The file names MUST be separated by semicolons. Each file name can be prefixed or suffixed with spaces. The first file name in the list can be preceded by a semicolon, and the last file name in the list can be suffixed with a semicolon.

For example, for a message that contains only one voice file attachment, for which the **PidTagAttachLongFilename** property's value is "vm.wav", acceptable values for **PidTagVoiceMessageAttachmentOrder** property include but are not limited to the following:

- vm.wav
- ;vm.wav
- ; vm.wav

Or, for example, a message contains three attachments, the **PidTagAttachLongFilename** values for which are "vm1.wav", "vm2.wav", and "vm3.wav". The files were added in the order "vm1.wav",

then "vm2.wav", and then "vm3.wav". Acceptable values for the **PidTagVoiceMessageAttachmentOrder** property include but not limited to the following:

- vm3.wav;vm2.wav;vm1.wav
- vm3.wav; vm2.wav; vm1.wav
- ;vm3.wav;vm2.wav;vm1.wav
- Vm3.wav;vm2.wav;vm1.wav
- ;vm3.wav;vm2.wav;vm1.wav

2.2.1.4 Audio Notes

The client can enable a user to annotate a voice message with textual information after it has been delivered to the user's mailbox. For example, some users might find it useful to take note of a telephone number or name that was included in the audio content of the message.

If the client saves that textual information on the message, it MUST set the **PidNameAudioNotes** property ([MS-OXPROPS] section 2.372), to the value of that textual information.

2.2.1.5 ASR Data

Automatic speech recognition (ASR) data<2> refers to the text transcription of an audio attachment. In an unprotected voice message, this data is stored in the **PidNameAutomaticSpeechRecognitionData** property ([MS-OXPROPS] section 2.378). In a protected voice message, it is handled as an attachment instead. As with other attachments in a **rights-managed e-mail message**, the attachment is stored in the "Attachment List" storage of the encrypted **binary large object (BLOB)**, as specified in [MS-OXORMMS] section 3.1.4.1.1.

2.2.1.5.1 ASR XML

2.2.1.5.1.1 Simple Types

2.2.1.5.1.1.1 evm:breakWeightType

The **evm:breakWeightType** simple type represents a coarse classification of the magnitude of a break in the speech data that was processed to obtain a transcript.

```
<xs:simpleType name="breakWeightType">
    <xs:restriction base="xs:NCName">
        <xs:enumeration value="low"/>
        <xs:enumeration value="medium"/>
        <xs:enumeration value="high"/>
    </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.1.2 evm:confidenceBandType

The **evm:confidenceBandType** simple type represents a coarse classification of a confidence result (that is itself represented as an **evm:zeroToUnityDoubleType** simple type). A value of "low" indicates that the transcript is probably significantly inaccurate. The heuristics for classification are not described here.

```
<xs:simpleType name="confidenceBandType">
     <xs:restriction base="xs:NCName">
          <xs:enumeration value="low"/>
          <xs:enumeration value="medium"/>
          <xs:enumeration value="high"/>
     </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.1.3 evm:recoErrorType

The **evm:recoErrorType** simple type represents success, or the types of errors, for voice recognition.

```
<xs:simpleType name="recoErrorType">
     <xs:restriction base="xs:NCName">
          <xs:enumeration value="success"/>
          <xs:enumeration value="audioQualityPoor"/>
          <xs:enumeration value="languageNotSupported"/>
          <xs:enumeration value="rejected"/>
          <xs:enumeration value="badRequest"/>
          <xs:enumeration value="systemError"/>
          <xs:enumeration value="other"/>
     </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.1.4 evm:recoResultType

The **evm:recoResultType** simple type represents the result types for voice recognition.

```
<xs:simpleType name="recoResultType">
     <xs:restriction base="xs:NCName">
          <xs:enumeration value="disabled"/>
          <xs:enumeration value="skipped"/>
          <xs:enumeration value="attempted"/>
          <xs:enumeration value="partial"/>
     </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.1.5 evm:versionNumberType

The **evm:versionNumberType** simple type represents the server version number format.

```
<xs:simpleType name="versionNumberType">
     <xs:restriction base="xs:token">
          <xs:pattern value="\d+\.\d+\.\d+\.\d+"/>
     </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.1.6 evm:zeroToUnityDoubleType

The **evm:zeroToUnityDoubleType** simple type represents probabilistic information.

```
<xs:simpleType name="zeroToUnityDoubleType">
     <xs:restriction base="xs:double">
          <xs:minInclusive value="0.0"/>
          <xs:maxInclusive value="1.0"/>
     </xs:restriction>
</xs:simpleType>
```

2.2.1.5.1.2 Complex Types

2.2.1.5.1.2.1 evm:recoObjectType

The **evm:recoObjectType** complex type represents information for a section of a voice recognition transcript.

```
<xs:complexType name="recoObjectType">
   <xs:simpleContent>
      <xs:extension base="xs:string">
         <xs:attribute name="ts" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:time"/>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="te" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:time"/>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="c" use="required">
            <xs:simpleType>
               <xs:restriction base="evm:zeroToUnityDoubleType"/>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="nx" use="optional">
            <xs:simpleType>
               <xs:restriction base="xs:token"/>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="id" use="required">
            <xs:simpleType>
               <xs:restriction base="xs:ID"/>
            </xs:simpleType>
         </xs:attribute>
         <xs:attribute name="be" use="optional" default="1">
            <xs:simpleType>
               <xs:restriction base="xs:boolean"/>
            </xs:simpleType>
         </xs:attribute>
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

Element	Type	Definition
be	**xs:Boolean**	Optional. Indicates whether the element is calculated to be on the most probable (1-best) path through the transcript (if "1" or "true"), or not (if "0" or "false").
c	**evm:zeroToUnityDoubleType** (section 2.2.1.5.1.1.6)	Required. Indicates the speech recognition system's confidence in this suggestion.
id	**xs:ID**	Required. Uniquely identifies the element within the transcript.
nx	**xs: token**	Optional. If this is not the final element of the transcript, the value of the attribute contains the identifier (ID) of the following element — that is, the next in time order.
te	**xs:time**	Required. Indicates the time (measured from the start of the audio) at which the corresponding utterance ends.
ts	**xs:time**	Required. Indicates the time (measured from the start of the audio) at which the corresponding utterance begins.
wb	**xs:breakWeightType** (section 2.2.1.5.1.1.1)	Optional. Indicates the relative weight of the break in the speech.

2.2.1.5.1.3 Elements

2.2.1.5.1.3.1 ASR

The **ASR** element is the root element of a transcript. Its attributes refer to the transcript as a whole. It contains elements that describe individual recognition objects (words, numbers, pauses, and so on) and possibly also describe associated features (names, telephone numbers, and so on).

```
      <xs:element name="ASR">
           <xs:complexType>
                <xs:choice minOccurs="0" maxOccurs="unbounded">
                     <xs:element ref="evm:ErrorInformation" minOccurs="0"/>
                     <xs:element ref="evm:Text"/>
                     <xs:element ref="evm:Break"/>
                     <xs:element ref="evm:Feature"/>
                </xs:choice>
                <xs:attribute name="productID" type="xs:unsignedInt" use="optional"
default="0"/>
                <xs:attribute name="confidence" type="evm:zeroToUnityDoubleType"
use="required"/>
                <xs:attribute name="confidenceBand" type="evm:confidenceBandType"
use="optional" default="medium"/>
                <xs:attribute name="lang" type="xs:language" use="required"/>
                <xs:attribute name="productVersion" type="evm:versionNumberType"
use="optional"/>
                <xs:attribute name="recognitionError" type="evm:recoErrorType" use="optional"
default="success"/>
                <xs:attribute name="recognitionResult" type="evm:recoResultType"
use="required"/>
                <xs:attribute name="schemaVersion" type="evm:versionNumberType"
use="required"/>
```

```
        </xs:complexType>
    </xs:element>
```

Attribute	Type	Definition
confidence	**evm:zeroToUnityDoubleType** (section 2.2.1.5.1.1.6)	Required. Indicates the overall confidence in the recognition results. This is calculated by the speech recognition system as a weighted average over the individual recognition elements.
confidenceBand	**evm:confidenceBandType** (section 2.2.1.5.1.1.2)	Optional. Provides a general indication of the system's overall confidence in the recognition results.
lang	**xs:language**	Required. Indicates the language in which the attempt at automatic speech recognition was made.
productID	**xs:unsignedInt**	Optional. If present, this attribute identifies the product or service that was used to produce the transcript. Values will be assigned to partner products and services by Microsoft. Partners MUST provide their ID when sending the transcript.<3>
productVersion	**evm:versionNumberType** (section 2.2.1.5.1.1.5)	Optional. If present, indicates the version of the software that was used to produce the transcript.<4>
recognitionError	**evm:recoErrorType** (section 2.2.1.5.1.1.3)	Optional. If present, provides for a more specific indication of the success or failure of the recognition than does the **recognitionResult** attribute.
recognitionResult	**evm:recoResultType** (section 2.2.1.5.1.1.4)	Required. Indicates whether an attempt at recognition was made and, if so, whether the recognition was completed.
schemaVersion	**evm:versionNumberType**	Required. Indicates the version of the schema description. This SHOULD always be "1.0.0.0".

2.2.1.5.1.3.2 Break

The **Break** element represents a discontinuity in the semantic content of a recording. For example, the speech might have paused for significantly longer than the typical amount of time between words. There is no expected value; all relevant information is contained in the attributes.

```
    <xs:element name="Break">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="evm:recoObjectType">
                    <xs:attribute name="wt" type="evm:breakWeightType" use="optional"
default="medium"/>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
```

```
</xs:element>
```

2.2.1.5.1.3.3 ErrorInformation

The **ErrorInformation** element provides a mechanism for the partner to return more detailed information when the **recognitionError** attribute of the **ASR** element is set to a value other than "success". The content of the element is expected to contain some diagnostic information that can help recipients of the document to understand why the transcript was not produced as expected. This element is required and expected only when the **recognitionResult** attribute of the **ASR** element has a value of either "skipped" or "partial". It can also be omitted unless the **recognitionError** attribute of the **ASR** element has a value of "other".

```
<xs:element name="ErrorInformation">
     <xs:complexType>
          <xs:simpleContent>
               <xs:extension base="xs:token">
                    <xs:attribute name="lang" type="xs:language" use="required"/>
               </xs:extension>
          </xs:simpleContent>
     </xs:complexType>
</xs:element>
```

Attribute	Type	Definition
lang	**xs:language**	Required. Indicates the language in which the error description is written. This is not required to be the same as the language in which the attempt at speech recognition was made.

2.2.1.5.1.3.4 Feature

The **Feature** element represents an assignment of special meaning to one or more **Text** elements in the transcript. The **Text** elements are contained within the **Feature** element.

```
<xs:element name="Feature">
     <xs:complexType>
          <xs:sequence maxOccurs="unbounded">
               <xs:element ref="evm:Text"/>
          </xs:sequence>
          <xs:attribute name="class" use="required">
               <xs:simpleType>
                    <xs:restriction base="xs:token"/>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="reference" type="xs:token" use="optional"/>
     </xs:complexType>
</xs:element>
```

Attribute	Type	Definition
class	**xs:token**	Required. Indicates the type of feature that has been identified.

Attribute	Type	Definition
reference	**xs:token**	Optional. If data relevant to the **Feature** markup exists outside the transcript, this attribute has to contain a pointer that will enable an application to locate and (with sufficient permission) access the data.

The following table lists the supported values of the **Feature class** attribute.

Feature class name	Reference?	Description
Contact	Yes	A personal contact of the Unified Messaging-enabled user to whom the voice message was sent. The reference is the Item ID of the **Contact object**, as returned by the server.
Date	Yes	A date. The reference represents a canonical version of the date. This can be in either an **xs:date** format or a regional format deduced from the recognition language that is being used.
Mailbox	Yes	A mailbox-enabled user. The reference is the primary **Simple Mail Transfer Protocol (SMTP)** address of the user.
PersonName	Yes	A person's name. The reference has the same value as the contained text.
PhoneNumber	No	A series of digits (and possibly other characters), probably representing a telephone number. The value can be expanded to a canonical form in line with regional conventions that are deduced from the recognition language that is being used.

2.2.1.5.1.3.5 Text

The **Text** element represents a portion of a transcript that can be a single word or number. This is contained as the value of the element.

```
<xs:element name="Text">
     <xs:complexType>
          <xs:simpleContent>
               <xs:extension base="evm:recoObjectType"/>
          </xs:simpleContent>
     </xs:complexType>
</xs:element>
```

2.2.2 Protected Voice Message

A protected voice message<5> is similar to a rights-managed e-mail message, as specified in [MS-OXORMMS] section 2.2.1. However, the client application needs to be aware of subtle differences between a rights-managed e-mail message and a protected voice message when rendering protected voice messages.

2.2.2.1 Messages

2.2.2.1.1 Message Classes

A protected voice message is represented by the following **message classes**:

- **IPM.NOTE.rpmsg.Microsoft.VoiceMail.UM.CA**, for original messages taken with audio content by telephone as a result of call answering.
- **IPM.NOTE.rpmsg.Microsoft.VoiceMail.UM**, for original messages taken with audio content by telephone as a result of any scenario other than call answering.

2.2.2.1.2 Message Content

As specified in [MS-OXORMMS], a rights-managed e-mail message consists of a wrapper message with the original e-mail content encrypted as a BLOB in an attachment. The attachment has the following requirements:

- **PidNameContentClass** ([MS-OXPROPS] section 2.432) MUST be set to "rpmsg.message".
- **PidTagAttachLongFilename** ([MS-OXPROPS] section 2.659) MUST be set to "message.rpmsg".
- **PidTagAttachMimeTag** ([MS-OXPROPS] section 2.666) MUST be set to "application/x-microsoft-rpmsg-message".

A protected voice message follows this convention. A nonprotected voice message contains one or more audio attachments and voice message preview data in the PidNameAutomaticSpeechRecognitionData property ([MS-OXPROPS] section 2.378). In the case of a protected voice message, the audio attachment(s) and voice message preview data are treated as attachments and are stored within the encrypted BLOB. These attachments MUST be stored in the "Attachment List" storage, as specified in [MS-OXORMMS] section 3.1.4.1.3.2.

2.2.2.2 Audio Attachments

Audio attachments carry the audio content of a voice message. This section describes how audio attachments are handled with protected voice messages.

2.2.2.2.1 Encrypted Audio Attachment

When an audio attachment is added to the "Attachment list" storage in the encrypted BLOB, it is encrypted. Depending on the original codec that is used to encode the audio attachment, the encrypted audio attachment carries the file name extension "umrmwav", "umrmwma", or "umrmmp3".

The content of the **PidTagVoiceMessageAttachmentOrder** property ([MS-OXPROPS] section 2.1176) in an unprotected voice message contains the list of the file names of the audio attachments. This is true for protected voice messages, except that all of the attachment file names have the ".umrmwav", ".umrmwma", or ".umrmmp3" extension.

2.2.2.3 Protected Voice Message Property

The **PidNameXRequireProtectedPlayOnPhone** property ([MS-OXPROPS] section 2.557) is set on the outer message of the protected voice message. When this property is set to "TRUE", the client that renders this message MUST NOT allow users to listen to the voice attachment by means of the e-mail client. The client MUST only offer user the Play-On-Phone feature as an option for listening to the voice message.

2.2.3 UI Configuration

A client application can surface an enhanced user interface (UI) for **Message objects** with the message classes specified in section 2.2.1.1 for some users and not for others. In addition, the client can show UI configuration information related to a user's telephony experience for some users and not for others. The server SHOULD store settings for these options on a per-user basis, and the client MUST consult these settings before it attempts to implement the aforementioned UI segmentation.

This could be useful in a scenario in which a certain group of users are not provisioned by their administrator to receive the message classes specified in section 2.2.1.1 and/or are not provisioned to have telephony access to their messages.

If the client or server sets or uses this configuration information, it MUST treat this information as a Dictionary stream by using the Configuration Information Protocol, as specified in [MS-OXOCFG].

The Dictionary Stream object MUST be stored in the Inbox **special folder**, as specified in [MS-OXOSFLD] section 2.2.4.

The Dictionary Stream object MUST have the **PidTagMessageClass** string property ([MS-OXPROPS] section 2.884) set on it. The value of this property MUST be **IPM.Configuration.UMOLK.UserOptions**.

The Dictionary SHOULD include the following setting. If the setting does not appear in the Dictionary, or the Dictionary does not exist, the following default value SHOULD be assumed:

- outlookFlags
- Name: (string) "outlookFlags"
- Value: (32-bit integer). The least significant bit MUST correspond to whether the client surfaces special UI information for message classes that are specified in section 2.2.1.1, with 1 corresponding to "do surface" and 0 (zero) corresponding to "do not surface". The second-least significant bit MUST correspond to whether the client surfaces telephony configuration UI to the user, with 1 corresponding to "do surface" and 0 (zero) corresponding to "do not surface". Therefore, the value MUST take one of the following four values:

1."0x00000000:" Neither the surface special UI for the message classes described in section 2.2.1.1 nor the one for telephony configuration purposes.

2."0x00000001": The surface special UI for the message classes specified in section 2.2.1.1, but not for telephony configuration purposes.

3."0x00000002": Do not show the surface special UI for the message classes specified in section 2.2.1.1, but do show special UI for telephony configuration purposes.

4."0x00000003": The surface special UI for the message classes specified in section 2.2.1.1 and also for telephony configuration purposes.

- Default: (32-bit integer) "0x00000000"

3 Protocol Details

3.1 Client Details

The client role is to display the Unified Messaging objects specified in section 2.2.1.1. There are two possible levels of client experience: down-level and up-level.

A "down-level" experience does nothing apart from the basic client role specified in [MS-OXCMSG] for Message objects. For an example of this experience, see section 4.1.1.

Alternatively, the client can provide an "up-level" experience for displaying Unified Messaging objects, including the ability to edit audio notes (section 2.2.1.4) and/or providing a means to automatically play back the audio content of a message by using the attachments (section 2.2.1.2) and the attachment order information (section 2.2.1.3). For an example of this experience, see section 4.1.2.

3.1.1 Abstract Data Model

See [MS-OXOMSG].

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

The server role in this protocol is to create the message types, as specified in section 2, in addition to the core server behavior as specified in [MS-OXCMSG].

When the server receives a message of one of the types specified in this document, the following additional properties MAY be set (these properties are specified in [MS-OXPROPS]).

- PidTagVoiceMessageSenderName property ([MS-OXPROPS] section 2.1178): The name of the sender if it was known at the time the message was taken, obtained by resolving the line ID against a directory.
- PidTagSenderTelephoneNumber property ([MS-OXPROPS] section 2.1112): The calling line ID number of the sender.
- PidTagVoiceMessageDuration property ([MS-OXPROPS] section 2.1177): The number of seconds of audio that was recorded.
- PidTagCallId property ([MS-OXPROPS] section 2.693): A unique identifier that is associated with the phone call.

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

None.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4 Protocol Examples

4.1 Playing a Voice Message

Both examples in the following sections assume that a voice message has been stored by the server, as specified in section 2.

4.1.1 Down-Level Experience

A client consults the configuration information specified in section 2.2.3 and sees that the *outlookFlags* parameter setting indicates that the client provides a down-level experience for the voice message object that it is about to display.

To provide the down-level experience, the client renders the voice message with all the functionality it would give to a typical Message object, as specified in [MS-OXOMSG]. In particular, it enables the user to access the audio attachment that is included in the message by using the standard mechanism provided by the client for accessing attachments.

Having accessed the content of the audio attachment file, the user uses an audio player application on his or her local computer that supports the attachment codec to play the audio content.

4.1.2 Up-Level Experience

A client consults the configuration information specified in section 2.2.3 and sees that the *outlookFlags* parameter setting indicates that the client provides an up-level experience.

The up-level experience of the client includes the ability to click a single "Play" button and hear all audio attachments on the message played in the reverse order in which the attachments were added. The user presses this button, and the client consults the attachment order information on the message (section 2.2.1.3) and sees that the value is "vm2.wma;vm1.wma". From this value, the client knows that there are two attachments on the voice message object with the **PidTagAttachLongFilename** property ([MS-OXPROPS] section 2.659) values "vm2.wma" and "vm1.wma", respectively.

The client downloads the attachment named "vm2.wma" and uses an audio player on the user's local computer to play the WMA 9 Voice audio content; it recognizes that the attachment is encoded with WMA 9 Voice because the **PidTagAttachMimeTag** property ([MS-OXPROPS] section 2.666) value of the attachment is "audio/wma". After the audio finishes playing, the client downloads "vm1.wma" and plays it in a similar way.

The client up-level experience of the client application also includes the ability to read and edit audio notes directly on the voice message, and the user decides whether or not to use this feature. The client provides an editable area on the screen into which the user can type a set of notes. When the user is finished, the client persists the entered text in the **PidNameAudioNotes** property ([MS-OXPROPS] section 2.372) of the Voice Message object. The next time the user views this particular Voice Message object, he sees the notes he typed because the client displays the content of the **PidNameAudioNotes** property of the Voice Message object.

5 Security

5.1 Security Considerations for Implementers

There are no special security considerations that are specific to the Voice Mail and Fax Objects Protocol. Note, however, that general security considerations that pertain to the underlying transport do apply to this protocol (for more information, see [MS-OXCMSG]).

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- Microsoft® Exchange Server 2003
- Microsoft® Exchange Server 2007
- Microsoft® Exchange Server 2010
- Microsoft® Office Outlook® 2003
- Microsoft® Office Outlook® 2007
- Microsoft® Outlook® 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 2.2.1.2: Exchange 2003 and Exchange 2007 do not support the MP3 format.

<2> Section 2.2.1.5: ASR data is not available in Exchange 2003 and Exchange 2007.

<3> Section 2.2.1.5.1.3.1: Exchange 2010 inserts a value of "925712" in transcripts that it generates.

<4> Section 2.2.1.5.1.3.1: Transcripts that are generated by Unified Messaging in Exchange 2010 take the form "14.nn.nnnn.nnn", with n representing digits.

<5> Section 2.2.2: Protected voice mail is not available in Exchange 2003 and Exchange 2007.

7 Change Tracking

This section identifies changes that were made to the [MS-OXOUM] protocol document between the November 2010 and March 2011 releases. 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.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 language and 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 or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- 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 protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1 Introduction	Added information about which sections of the specification are normative and can contain RFC 2119 language.	Y	New content added for template compliance.

8 Index

U

V