Lightweight Message Exchange Profile (LIME)

Version 1.0.0

WP8 2009-12-22

Contents

1 Document information

1.1 Document history

Date	Version	Initials	Changes
2009-01-03	0.1.0	GS	Created template
2009-02-06	0.1.1	PZF	Initial definition and thoughts
2009-02-17	0.3	PZF	Updates based on feedback from GS Significantly changed to be completely WSTransfer based Renamed store to channel
2009-02-17	0.5	GS	Minor editorial changes
2009-03-29	0.6	PZF	Updates based on the 0.5 feedback
2009-04-01	0.7	GS	Minor editorial updates
2009-04-30	0.8	PZF	Restructuring and significant updates based on 0.7 feedback meeting
2009-08-30	0.9	PZF	Updates based on F2F meeting in Copenhagen plus feedback
2009-09-01	0.9	GS	Updated schema, naming + namespaces, references, example XML
2009-09-08	0.9.1	GS	Changed attribute "numOfEntries" to "numberOfEntries" Changed sect. 3.4 "Inbound Message Channel" to include 'EntryList'
2009-09-09	0.9.1	GS	Changed 'type' attribute of most identifiers to 'scheme'
2009-10-22	0.9.5	GS	Updated example XML to valid instance, added example of 'NextPageIdentifier' element, changed size of @size to 'long', added minOccurs to 'NextPageIdentifier' element, removed whitespace from TNS in examples, wsa:EndpointReference is now child element of 'NextPageIdentifier'.
2009-11-17	1.0.0	PZF	Final updates for 1.0.0
2009-11-27	1.0.0	GS	Added synchronous delivery fault, moved section on message headers to the 'common definitions' document, described how documents that are not part of a process are handled, and minor editorial changes.
2009-12-22	1.0.0	PZF	Removed MessageUndelivered (into Common Definitions) and added fixes based on Philip Helger's feedback

1.2 Editor

Paul Fremantle, WSO2

1.3 Contributors (alphabetically)

Jens Jakob Andersen, NITA

Mikkel Hippe Brun, NITA

Mike Edwards, IBM

Paul Fremantle, WSO2

Thomas Gundel, IT Crew

Philip Helger, Bundesrechenzentrum

Hans Guldager Knudsen, Lenio

Bergthór Skúlason, NITA

Gert Sylvest, Avanade

2 Introduction

The Lightweight Message Exchange Profile (LIME) provides a simple low-cost approach for Small and Medium Enterprises (SMEs) to access Business Document Exchange Network (BUSDOX) infrastructure. The "low costs" that this profile is designed to address includes:

- No requirement to host online endpoints, hence no firewall crossing, no server infrastructure.
- No requirement to support "advanced" WS-* standards such as WS-Trust, WS-ReliableMessaging. Only minimal requirement to support WS-Addressing and WS-Transfer only. Since WS-Transfer is a simple WSDL-based specification, the only requirement on the SOAP stack is to support WS-Addressing.

This is achieved through the use of a Business Document Exchange Network (BUSDOX) Access Point [BDEN-CDEF] that supports this profile and manages messages on behalf of the client. It both handles messages destined for the client by storing them in a Message Channel awaiting retrieval and also the Relay Service provides a simple way that the client may send messages to other organizations without requiring to navigate the service metadata. A simple analogy is the POP3/SMTP-Relay services that ISPs provide that enables email access from intermittently connected computers.

2.1 Goals and non-goals

2.1.1 Goals

- Provide an interface to a message channel and relay service that supports intermittently connected systems.
- Provide access over a simple HTTPS-protected channel
- Utilize existing standards where appropriate
- Support the same message format as other BUSDOX Transport Profiles
- Lower the cost of entry for SME's and individuals.

2.1.2 Non-Goals

- This profile does not support end-to-end security or identity. The BUSDOX Lightweight Message Exchange Profile Access Point (LIME-AP) must validate the credentials of customers using the LIME profile and map those credentials into a valid identity to be used for outbound communications.
- This specification is expected to be used in the context of a particular usage of the BUSDOX profiles: for example, the types and formats of participant identifiers are not specified as part of this profile, but in a real deployment would be specified as part of a governance model.

2.2 Terminology

Please see Common Definitions [BDEN-CDEF] section 2.2

2.3 Notational conventions

Notational conventions have been adopted from [WSDL-2.0], see "Common Definitions" [BDEN-CDEF] section 2.2.

Pseudo-schemas are provided for each component, before the description of the component. They use BNF-style conventions for attributes and elements: "?" denotes optionality (i.e. zero or one occurrences), "*" denotes zero or more occurrences, "+" one or more occurrences, "[" and "]" are used to form groups, and "|" represents choice. Attributes are conventionally assigned a value which corresponds to their type, as defined in the normative schema. Elements with simple content are conventionally assigned a value which corresponds to the type of their content, as defined in the normative schema. Pseudo schemas do not include extension points for brevity.

```
<!-- sample pseudo-schema -->
<defined_element
    required_attribute_of_type_string="xs:string"
    optional_attribute_of_type_int="xs:int"? >
 <required_element />
 <optional_element />?
 <one_or_more_of_these_elements />+
 [ <choice_1 /> | <choice_2 /> ]*
</defined_element>
```

2.3.1 Normative references

[BDEN-CDEF] Business Document Exchange Network - Common Definitions, CommonDefinitions.pdf

[WS-T] "Web Services Transfer (WS-Transfer)", W3C Working Draft 24 September 2009, http://www.w3.org/TR/2009/WD-ws-transfer-20090924/

[WSA-1.0] "Web Services Addressing 1.0 - Core" (http://www.w3.org/TR/2005/CR-ws-addr-core-20050817/) and "Web Services Addressing 1.0 - SOAP Binding", http://www.w3.org/TR/ws-addr-soap/

[XML-DSIG] "XML Signature Syntax and Processing (Second Edition)", http://www.w3.org/TR/xmldsig-core/

[RFC-2119] "Key words for use in RFCs to Indicate Requirement Levels", http://www.ietf.org/rfc/rfc2119.txt

[SOAP-1.1] "Simple Object Access Protocol (SOAP) 1.1", http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

2.3.2 Non-normative references

[WSDL-2.0] "Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language", http://www.w3.org/TR/wsdl20/

2.4 Namespaces

The following table lists XML namespaces that are used in this specification. The choice of any namespace prefix is arbitrary and not semantically significant.

Namespace Prefix	Namespace
wsa	http://www.w3.org/2005/08/addressing
s	http://schemas.xmlsoap.org/soap/envelope/
lime	http://busdox.org/transport/lime/1.0/
ids	http://busdox.org/transport/identifiers/1.0/
xs	http://www.w3.org/2001/XMLSchema

3 Introduction and overview

3.1 Example flows

The Lightweight Message Exchange Profile is designed to allow systems to participate in the BUSDOX infrastructure without needing to access service metadata directly or host an Access Point. Instead, they may choose to use the Lightweight Message Exchange Profile (LIME) to communicate with a service provider. A simple analogy is Internet email: Large companies may run their own Simple Mail Transport Protocol (SMTP) server and proprietary email clients to create and read messages, but individuals or small companies rely on an ISP to provide an SMTP Relay and POP3 or IMAP server.

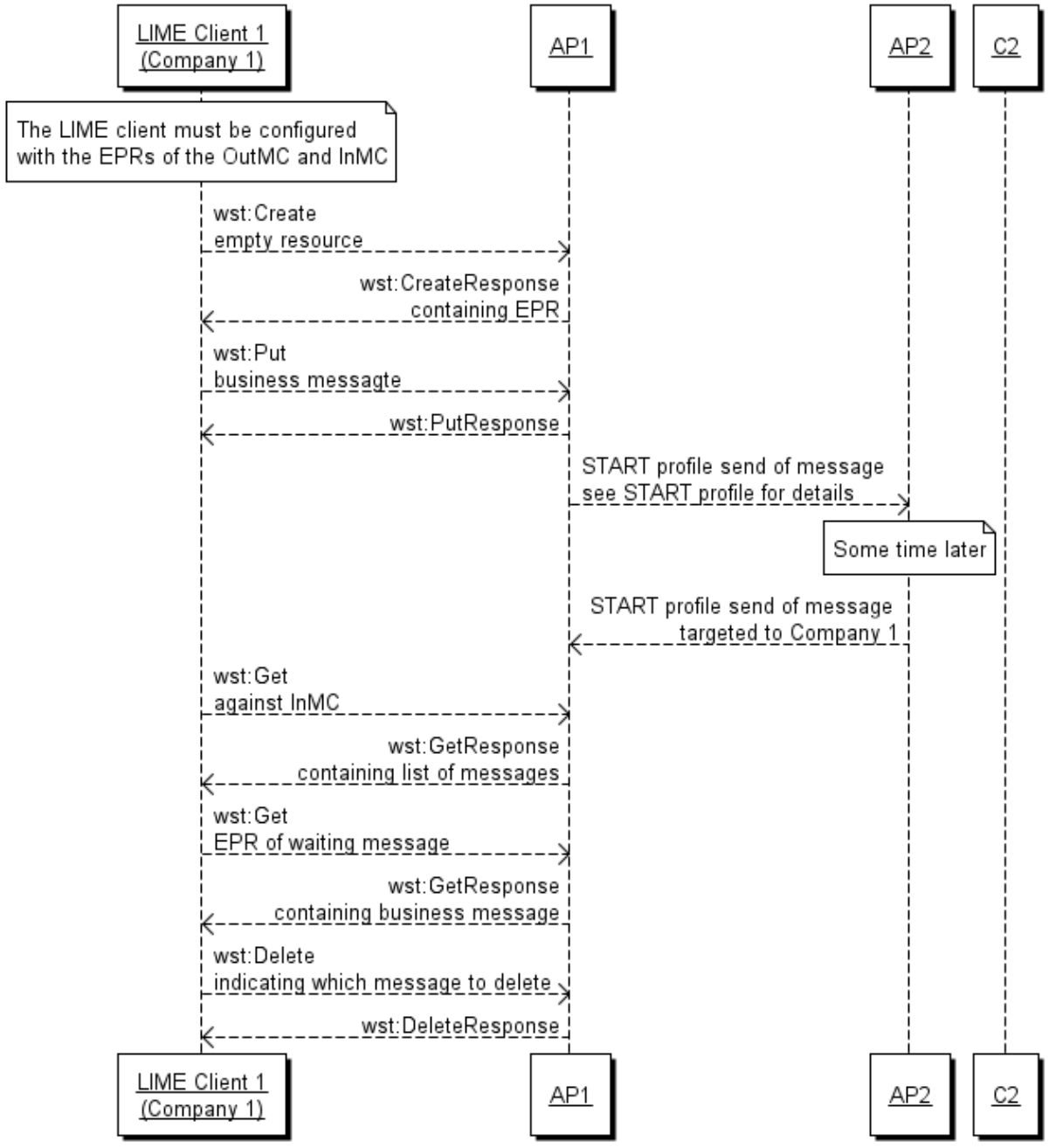
This profile describes the approach that a LIME Client (LC) can use to send and receive messages from an LIME-enabled Access Point.

The diagram below shows a simple example flow. The LC needs to send a message to a company which uses an Access Point we will call AP2. However, the LC only needs to be configured to talk to a single local access point (AP). Initially the business user creates a business message using the software (out of scope for this profile). The requirements are that the business message complies with the BUSDOX specifications and that the correct participant identifiers (see section 4.2) are made available to the LC.

The LC sends a Create message to the AP, which initiates the message flow and causes the AP to create a fixed Endpoint Reference (EPR) Resource. The message is then put into this resource by the LC. This model ensures that messages delivered exactly once to the AP. Once the message is delivered to the access point, it looks up the recipient's AP and transfers the message.

105 The LC also polls the AP for any incoming messages. This is done by “Get”-ting a list of available messages
106 from the AP, and then individually retrieving each available message (if any) using another Get.

107



108

109 Figure 1 - Sequence diagram showing a typical LIME profile usage

110 ## 3.2 Technical Overview of the Profile

111 The profile defines a set of technologies that are used together:

112 - HTTPS and Basic Authentication for security

- SOAP 1.1 for the base communications
- WS-Transfer as a standard approach to accessing the message channels
- BUSDOX specific headers to define standard metadata
- BUSDOX specific XML Schema to define the message list XML format

Together these different technologies are used together to define a simple protocol that can allow an intermittently connected computer to fully participate in a BUSDOX infrastructure so long as they have a Lightweight Message Exchange Profile Access Point (LIME-AP) available.

4 Definition of the Message Channel

4.1 Concepts

A message channel is a WS-Transfer endpoint that either accepts or retrieves messages from an LC. A single channel may handle both incoming and outgoing messages or there may be independent channels. The profile assumes that there may be independent channels and therefore that the LC is provided with addressing information for both channels. For the remainder of this specification all references assume that there are two independent channels. However, the specification is written in such a way that there may be a single channel - in either case the specification can operate correctly.

The Outbound message channel (OutMC) accepts outbound messages (messages from LC to AP) using the WS-Transfer Create and Put operations, and the inbound message channel (InMC) offers inbound messages (messages from AP to LC) using the WS-Transfer Get method, and allows these to be deleted using the WS-Transfer Delete method.

4.2 BUSDOX defined headers

Every BUSDOX message has associated metadata included so that Access Points can route messages without needing to look inside the business message. Therefore this profile defines the following mandatory header blocks.

The Common Definitions document [BDEN-CDEF] defines the following identifiers in section 3.7:

- RecipientIdentifier
- SenderIdentifier
- DocumentIdentifier
- ProcessIdentifier
- MessageIdentifier
- ChannelIdentifier

For an XML Schema for these elements, see 'Common Definitions' [BDEN-CDEF].

4.2.1 MessageWaiting Header

The LIME-enabled Access Point MAY indicate to a LIME client that there are messages waiting in the Outbound Message Channel. This header MAY be added into any response message flowing to the LIME Client. The header element is:

```
<lime:MessageWaiting/>
```

The header MUST NOT have any attributes.

4.2.2 About ProcessIdentifier

If a document is not part of a well defined business process, the ProcessIdentifier header must still be present. It then MUST hold the value 'busdox:noprocess' and scheme 'busdox-procid-transport', see the section on process identifiers in [BDEN-CDEF].

4.2.3 About Message Identifier

Because BUSDOX Messages may pass between several parties (for example in the "four-corner" model, from LC to AP to AP to LC), it is necessary to have a constant message identifier that uniquely identifies the message across multiple hops. This message identifier is contained in the:

```
<ids:MessageIdentifier>
```

element.

During message resending in the PUT phase, the MessageIdentifier header MUST be the same for each resend of the same message. The ids:MessageIdentifier is created by the AP (as a reference parameter), and is then sent along with the business message as it passes on to other APs.

4.3 Use of WS-Transfer

For access to the Message Channels in this profile the LC uses the WS-Transfer specification [BDEN-CDEF] Business Document Exchange Network - Common Definitions, CommonDefinitions.pdf

[WS-T][1]. WS-Transfer is used to send messages from the LC to the AP as well as retrieve waiting messages.

In order to ensure the reliable sending of messages we use a pattern that we call CreatePut[2].

Receiving messages is done with two or more Gets – the first lists a page of available messages, further requests may retrieve individual messages or further pages of message listing. Messages SHOULD be deleted (using WS-T Delete) once successfully retrieved. WS-Transfer does not define a resource listing model, so this profile defines a simple XML Schema for lists.

4.3.1 Securing channels

The LIME-AP MAY secure the two message channels in the following fashion:

- The LC can list messages using the Get interface on the InMC. The LC can Get and Delete messages in the Inbound Message Channel. It cannot Create or send (Put) messages in the Inbound Message Channel.

- The LC may Create and send (Put) messages in the Outbound Message Channel. It cannot list messages, Get, or Delete messages in the OutMC.

Of course, this model assumes that there are two independent channels. In the case where there is a single channel operating as both InMC and OutMC, the associated security must be determined by the AP.

[1] It is expected that future versions of LIME or errata will update to the Final version of WS-Transfer as and when this becomes available.

[2] This is based on a common pattern used with REST services.

In this model it is possible to have a single outbound message channel shared by many companies. This is not normative. Another possible alternative is that the same channel identifier is used to both send and receive messages.

The LIME-AP MUST support the listing interface and WS-Transfer GET/DELETE on the InMC.

The LIME-AP MUST support CREATE/PUT on the OutMC.

4.3.2 Use of WS-Addressing Reference Parameters

WS-Transfer supports the use of any WS-Addressing Reference Parameters to define resources that are transferred. However, for the purpose of this profile, we define specific SOAP headers/reference parameters to be used. These headers MUST be used. The profile authors understand the W3C guidance that EPRs are designed to be opaque. However, the authors believe there are two significant benefits to specifying the reference parameters:

- A clear basis for comparing endpoint references, since EPRs are clearly defined.
- The configuration of the LC is simpler, because channels can always be configured with a combination of URL and Channel Identifier.

The ChannelIdentifier is a URI which uniquely identifies a channel. Every WS-Transfer request against a channel MUST have the ChannelIdentifier reference parameter present.

```
<ids:ChannelIdentifier>xs:anyURI</ids:ChannelIdentifier>
```

The MessageIdentifier is a URI which uniquely identifies a message. The message identifier is consistent across multiple hops.

```
<ids:MessageIdentifier>xs:anyURI</ids:MessageIdentifier>
```

4.4 Inbound Message Channel

The LC retrieves messages from a specific Inbound Message Channel (InMC), identified by an Endpoint Reference (EPR) provided by the LIME-AP. The EPR contains a unique identifier for the Channel known as the Channel Identifier (ChannelIdentifier). For example the EPR of the Inbound Message Channel may contain a Channel Identifier (ChannelIdentifier) that is based on the company registration number. Please note that the actual ChannelIdentifier is defined by the AP's system and is only relevant when talking to that access point.

Here is an example EPR for an inbound message channel:

```
<wsa:EndpointReference>
   <wsa:Address>
http://LIME-AP.my-van.com:80/services/messagechannel
   </wsa:Address>
   <wsa:ReferenceParameters>
          <ids:ChannelIdentifier>55038353</ids:ChannelIdentifier>
   </wsa:ReferenceParameters>
</wsa:EndpointReference>
```

The manner in which this EPR is provided to the LC is out-of-scope: for example it may be manually entered as part of the user configuration of the LC.

The LC may have many message channels that it can access. The Message Channel may store any number of messages.

In order to allow the LC to find and access these messages a three-step process is used:

1. First the LC uses the WS-Transfer Get operation to retrieve a list of messages that are waiting to in the channel.

2. The LC uses the WS-Transfer interface on the EPR to retrieve (GET) the message. If there is a failure retrieving the message, the LC may repeat this step as needed.

3. Once the message is successfully retrieved, the LC SHOULD use the WS-Transfer Delete operation to delete the message from the channel.

An example flow can be seen in Figure 1.

Each individual message in a channel has an Endpoint Reference which contains both the ChannelIdentifier as well as a unique MessageIdentifier as reference parameters. Here is an example of an Endpoint Reference for a message.

```
<wsa:EndpointReference>
   <wsa:Address>
http://LIME-AP.van.co.uk:80/services/transfer
   </wsa:Address>
   <wsa:ReferenceParameters>
          <ids:ChannelIdentifier>55038353</ids:ChannelIdentifier>
          <ids:MessageIdentifier>
              uuid:f8290-432lkj2349-8aiuyfga0
          <ids:MessageIdentifier>
   </wsa:ReferenceParameters>
</wsa:EndpointReference>
```

4.4.1 Finding available messages

To find available messages, the LC simply does a WS-Transfer Get on the Channel – in other words the Get is targeted against the channel EPR.

If the channel access control permissions allow this, the channel responds with an XML list of available messages. In order to be efficient, the list can be paginated. The Channel decides a maximum number of messages to list in a page, and each page of the list is a resource that can be transferred. Performing a GET on the main channel EPR always returns start of the list. The list is ordered by the time of creation of the messages.

The LC may adopt two strategies for downloading messages:

- The LC may simply retrieve the first page, download all listed messages, delete them, and then relist. The next page will then be listed.

- The LC may retrieve the first page of listed messages. At the end of this list is the EPR of the next page. The LC may then retrieve the next page, and so forth until it has listed all available messages.

The full schema for the paginated list interface is in the Appendix, and the definition of the meaning of the XML document elements is defined as follows.

The operation provides a paginated list of available messages. Each page has up to n messages, where n is a number defined by the channel server system. It is recommended that n=100 as a simple default.

/ lime:PageList

This element contains a page of entries, which may be downloaded individually. It also contains a reference to additional pages of entries, if such exist.

/ lime:PageList / lime:EntryList

This element contains the individual entries; each entry is a document that can be downloaded by the client.

/ lime:PageList/lime:EntryList /@numberOfEntries

This attribute is a long number, containing the number of entries returned in this page. This number will match the number of /lime:PageList/lime:Entry elements that are children of this /lime:Pagelist element.

/ lime:PageList/lime:EntryList /lime:Entry

This element contains information about a specific document that is waiting for collection.

/ lime:PageList/lime:EntryList/lime:Entry/@size

This required attribute captures the size of the document waiting for collection, in Kilobytes (multiples of 1024), as a rounded long integer numeric value.

/ lime:PageList/ lime:EntryList /lime:Entry/@creationTime

This required attribute whose value is an XML Schema dateTime captures the time that the document referenced by this Entry was stored by the access point.

/ lime:PageList/ lime:EntryList /lime:Entry/@messageBodyLocalName

This optional attribute captures the tag name of the first element of the document referred to by the Entry. In the case that the document is encrypted this will not be available to the Access Point and this attribute will not be present.

/ lime:PageList/ lime:EntryList /lime:Entry/@messageBodyNamespace

285 This optional attribute (type xs:anyURI) captures the namespace of the first element of the
286 document referred to by the Entry. In the case that the document is encrypted this will not be
287 available to the Access point and this attribute will not be present.

288 The EPR for listing messages from a channel SHOULD contain the Message Channel Identifier
289 (ChannelIdentifier), and the EPR Reference Parameters MUST be included in the SOAP Header
290 of any request messages.

291 Any EPRs offered by the channel for *listing* messages MUST NOT include the Reference Parameter
292 <ids:MessageIdentifier>

293

Example:

```
<ids:ChannelIdentifier wsa:isReferenceParameter=”true”>
          Channel Identifier
</ids:ChannelIdentifier>
<lime:PageIdentifier wsa:isReferenceParameter=”true”>
          5
</lime:PageIdentifier>
```

The WS-Transfer page list XML Schema is in the appendix.

For the purposes of this profile, the EndpointReferences returned in the sequence of Entries MUST contain the following two reference parameters:

```
<ids:ChannelIdentifier>xs:string<ids:ChannelIdentifier>
<ids:MessageIdentifier>xs:string</ids:MessageIdentifier>
```

Here is a sample XML response to the page listing GET request:

```
<?xml version="1.0" encoding="utf-8" ?>
<!--
  An sample XML response to the page listing GET request
-->
<lime:PageList
  xmlns:lime="http://busdox.org/transport/lime/1.0/"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:ids="http://busdox.org/transport/identifiers/1.0/"
  numberOfEntries="1">
  <lime:EntryList>
    <lime:Entry size="8295" creationTime="2009-02-18T12:33:45Z"
           messageBodyLocalName="Order"
messageBodyNamespace="http://busdox.org/ns/Order">
      <wsa:EndpointReference>
        <wsa:Address>
          http://LIME-AP.my-van.com:80/services/transfer
        </wsa:Address>
        <wsa:ReferenceParameters>
          <ids:ChannelIdentifier>55082098</ids:ChannelIdentifier>
          <ids:MessageIdentifier>uuid:45989-2429-
132412312</ids:MessageIdentifier>
        </wsa:ReferenceParameters>
      </wsa:EndpointReference>
    </lime:Entry>

  </lime:EntryList>

  <lime:NextPageIdentifier>
    <wsa:EndpointReference>
      <wsa:Address>
        http://LIME-AP.my-van.com:80/services/messagechannel
      </wsa:Address>
      <wsa:ReferenceParameters>
        <ids:ChannelIdentifier>55038353</ids:ChannelIdentifier>
        <!--
          NOTE: The 'PageIdentifier' may be replaced by element in
```

```
          any namespace that represents a system-specific ID of the next
page
        -->
        <PageIdentifier
xmlns="http://someNamespace.org">2</PageIdentifier>
      </wsa:ReferenceParameters>
    </wsa:EndpointReference>
  </lime:NextPageIdentifier>


</lime:PageList>
```

The LC MUST use document/literal binding to access the Channel WS-Transfer service.

If the message was transferred into the channel using a BUSDOX Transport profile, then the MessageIdentifier used as a reference parameter MUST be the same as the ids:MessageIdentifier of the message used to create the message in the channel. If no such MessageIdentifier exists, then the LIME-AP should create a guaranteed unique MessageIdentifier for the message.

4.4.2 Getting a message using WS-Transfer

Once an Endpoint Reference has been retrieved using the Get listing operation, the message may be retrieved using the WS-Transfer Get method.

All BUSDOX defined headers that were transferred to the LIME-AP MUST be included as SOAP Headers when the message is retrieved using Get.

4.4.3 Inclusion of SAML attributes

If the message being retrieved from the channel originated in another BUSDOX access point, then it will have had a SAML token attached at that point with an assurance level attribute. In order to support end-to-end traceability and assurance, the LIME-AP MUST include assurance level attribute in any messages that are made available in the inbound channel.

The following header contains the SAML attribute:

```
<lime:identityAssurance>
   <saml2:Attribute/>
</lime:identityAssurance>
```

The LC MUST include the <saml2:Attribute Name="urn:eu:busdox:attribute:assurance-level" > element. The LC MAY use this information to inform the business users of the BUSDOX assurance level.

4.4.4 Deleting messages

It is the responsibility of the LC to delete messages once they have been retrieved. The WS-Transfer DELETE operation SHOULD be used. Both reference parameters (ChannelIdentifier and MessageIdentifier) MUST be used to delete messages.

In the case there is an error during the Delete (for example a dropped connection) it is the responsibility of the LIME client to retry until there is a confirmation that the resource no longer exists.

4.5 The Outbound Message Channel

The Outbound Message Channel (OutMC) provides a simple model where the LC may transfer messages to a LIME-AP. These messages are then transferred to other BUSDOX Access Points using business addressing information stored in the business message.

4.6 Message Sending

In this exchange, the LC and the LIME-AP implement a reliable delivery model to ensure that messages are delivered once-only. This is known as the CreatePut model.

In order to implement a simple reliable idempotent model for relaying messages outbound, the LC implements a two-stage message sending process:

1. In the first stage, the LC creates an *empty* resource in the OutMC, not containing the real message. This is done using the CREATE request and with no business message
   <wst:Create/>
   The response to this is:
   <wxf:ResourceCreated>endpoint-reference</wxf:ResourceCreated>
   which contains an EndpointReference of the resource that will be used to transmit the message. The EndpointReference will contain a unique ids:MessageIdentifier reference parameter.
   If this operation is incomplete or the response is dropped at the network level, the LC should retry. In this case, there may be an extra unused Resource and EPR available on the LIME-AP (from the first failed CREATE request). The LIME-AP SHOULD keep such resources available for up to one hour, to allow for timing issues in the LC. The LIME-AP SHOULD garbage collect/delete any such resources that remain empty for extended periods of time.

   The Create message MUST include the following BUSDOX defined headers in the SOAP Header: *ids:RecipientIdentifier, ids:ChannelIdentifier, ids:SenderIdentifier, ids:DocumentIdentifier, ids:ProcessIdentifier*

   These headers MUST be used by the OutMC when sending this message onwards.

   In addition the ids:MessageIdentifier header will be included as one of the reference parameters.

   The client MAY include other headers in the message. Any headers that are not defined in the BUSDOX-namespace and are not part of the Reference Parameters SHOULD be stored and SHOULD be relayed onward with the business message.

2. In the second stage the LC uses the WS-Transfer Put operation to transfer the actual message to the EPR returned in stage 1.

   The Put message request SHOULD NOT include the BUSDOX defined headers in the SOAP Header, except the reference parameters. Other non-BUSDOX headers SHOULD NOT be included. Any headers that are not defined in the BUSDOX-namespace and are not part of the Reference Parameters SHOULD be dropped by the Access Point before transferring to another Access Point.

The LC SHOULD repeat this second step as often as required until it gets a successful response. If there is a long period of time between step 1 and step 2, it is possible that the LIME-AP has deleted the resource. In this case the LIME-AP will return a fault to the LC indicating an unknown EPR. In this case, the LC SHOULD restart at step 1.

The LC SHOULD log the PUT request message and PUT response message for proof-of-delivery. The LIME-AP MUST include the WS-Addressing RelatesTo header.

This profile explicitly defines the format of the EndpointReferences used to Create resources in the Outbound Message Channel. The <wsa:MessageID> from the Initial Create message MUST be used to create the EndpointReference in the following way:

```
<wsa:EndpointReference>
   <wsa:Address>
      http://LIME-AP.my-van.com:80/services/messagechannel
   </wsa:Address>
   <wsa:ReferenceParameters>
          <ids:ChannelIdentifier>outbound</ids:ChannelIdentifier>
   </wsa:ReferenceParameters>
</wsa:EndpointReference>
```

When the Channel returns a new endpoint reference from the Create operation, it adds a unique MessageIdentifier to the ReferenceParameters, e.g.:

```
<wxf:ResourceCreated>
  <wsa:EndpointReference>
   <wsa:Address>
          http://LIME-AP.my-van.com:80/services/messagechannel
   </wsa:Address>
   <wsa:ReferenceParameters>
          <ids:ChannelIdentifier>outbound</ids:ChannelIdentifier>
          <ids:MessageIdentifier>uuid:45989-2429-
132412313</ids:MessageIdentifier>
   </wsa:ReferenceParameters>
  </wsa:EndpointReference>
</wxf:ResourceCreated>
```

Once the LIME-AP receives a complete successful Put operation it can relay the message on to the final recipient. The LIME-AP should treat subsequent Puts of the same message as correct, as these indicate that the LC has not yet received a successful response and will keep retrying. The LIME-AP SHOULD use the <ids:MessageIdentifier> of the Put request when relaying messages onward.

The LIME-AP MUST generate unique Message IDs for the endpoint reference returned in the CreateResponse.

The LIME-AP MUST NOT attempt to deliver empty messages.

4.6.1 Faults

The DestAP can fault in four circumstances on the OutMC. Firstly, it may have a "full channel". This indicates that the client should retry at a later time. Secondly, the endpoint may not be recognized. Thirdly, there may be a security error. Finally, there may be an internal server fault (Server Error). The faults used are as follows:

Channel Full Fault

[action]	http://busdox.org/2010/02/channel/fault
Code	s:Sender
Subcode	lime:ChannelFull
Reason	The channel is not accepting messages for this destination
Detail	As detailed by the AP

Unknown Endpoint

[action]	http://busdox.org/2010/02/channel/fault
Code	s:Sender
Subcode	lime:UnknownEndpoint
Reason	The endpoint is not known
Detail	As detailed by the AP

Security Error

[action]	http://busdox.org/2010/02/channel/fault
Code	s:Sender
Subcode	lime:SecurityFault
Reason	There is a security error in processing this request
Detail	As detailed by the AP

Server Error

[action]	http://busdox.org/2010/02/channel/fault
Code	s:Sender

Subcode	lime:ServerError
Reason	ServerError
Detail	As detailed by the AP

4.7 Use of HTTP

Please see the Common Definitions document for use of HTTP [BDEN-CDEF].

4.8 Use of MTOM

The Message Transmission Optimization Mechanism is a way of effectively encoding binary data in SOAP messages. LIME Clients MAY use MTOM to send messages. If an LC supports MTOM, it MAY use an MTOM packaging to issue a WS-Transfer GET request. In this case the LIME-AP MUST respond with an MTOM encapsulated message. The LIME-AP MUST support MTOM on the LIME services.

5 Security

It is up to the LIME-AP to manage the user access to channels, based on the HTTP Basic Authentication or other authentication credentials provided to the LIME-AP by the LC.

It is important to note that the security of the Lightweight Message Exchange Profile is only point-to-point and not end-to-end. This means that the credentials used to authenticate the LC to the LIME-AP need not be acceptable by other BUSDOX Access Points. The credentials are only required to be accepted by the LIME-AP. For example, the LIME-AP may run its own user channel for small companies and map these credentials into tokens acceptable by other BUSDOX Access Points.

The minimum required security for LIME is to use:

- HTTP Basic authentication
- Transport Layer Security for encryption

Appendix C – XML Schema for Lime Types

XSD for the Lime Types:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
  targetNamespace="http://busdox.org/transport/lime/1.0/"
  elementFormDefault="qualified"
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:tns="http://busdox.org/transport/lime/1.0/"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:ids="http://busdox.org/transport/identifiers/1.0/"
  version="1.0.0">

  <import namespace="http://www.w3.org/2005/08/addressing"
schemaLocation="ws-addr.xsd" />
  <import schemaLocation="Identifiers-1.0.xsd"
namespace="http://busdox.org/transport/identifiers/1.0/" />

  <element name ="MessageUndeliverable"
type="tns:MessageUndeliverableType" />

  <complexType name="MessageUndeliverableType">
    <sequence>
      <element ref="ids:MessageIdentifier" />
      <element name="ReasonCode" type="tns:ReasonCodeType" />
      <element name="Details" type="string" />
    </sequence>
  </complexType>

  <simpleType name="ReasonCodeType">
    <restriction base="string">
      <enumeration value="METADATA_ERROR" />
      <enumeration value="TRANSPORT_ERROR" />
      <enumeration value="SECURITY_ERROR" />
      <enumeration value="OTHER_ERROR" />
    </restriction>
  </simpleType>

  <element name="PageList" type="tns:PageListType" />

  <complexType name="PageListType">
    <sequence>
      <element name="EntryList" type="tns:EntryListType" />
      <element name="NextPageIdentifier"
type="tns:NextPageIdentifierType" minOccurs="0" />
    </sequence>
    <attribute name="numberOfEntries" type="long"
use="optional"></attribute>
  </complexType>

  <complexType name="EntryListType">
    <sequence>
```

```
      <element name="Entry" type="tns:Entry" minOccurs="0"
maxOccurs="unbounded" />
    </sequence>
  </complexType>

  <complexType name="Entry">
    <sequence>
      <element ref="wsa:EndpointReference" />
    </sequence>
    <attribute name="size" type="long"></attribute>
    <attribute name="creationTime" type="dateTime"
use="optional"></attribute>
    <attribute name="messageBodyLocalName" type="string"
use="optional"></attribute>
    <attribute name="messageBodyNamespace" type="anyURI"
use="optional"></attribute>
  </complexType>

  <complexType name="NextPageIdentifierType">
    <sequence>
      <element ref="wsa:EndpointReference"/>
    </sequence>
  </complexType>
</schema>

```