



Service Metadata Publishing

Version 1.0.0



WP8 2009-12-23



Contents

1	Document information	4
1.1	Document history	4
1.2	Editor.....	5
2	Contributors (alphabetically)	5
3	Introduction	6
3.1	Goals and non-goals.....	6
3.2	Terminology	6
3.2.1	Notational conventions.....	6
3.2.2	Normative references	6
3.2.3	Non-normative references.....	6
3.3	Namespaces	6
4	The Service Discovery Process	7
4.1	Discovery flow.....	7
4.2	Discovering services associated with a Participant Identifier.....	8
4.3	Service Metadata Publisher Redirection	8
5	Interface model.....	10
6	Data model.....	10
6.1	On extension points	10
6.1.1	Semantics and use	10
6.2	ServiceGroup.....	12
6.2.1	Non-normative example	13
6.3	ServiceMetadata	14
6.3.1	Redirection.....	14
6.3.2	Non-normative example	17
6.4	SignedServiceMetadata	18
6.4.1	Non-normative example	18
6.4.2	Redirect, non-normative example	19
7	Service Metadata Publishing REST binding.....	21
7.1	The use of HTTP	21
7.1.1	HTTP status codes	21

7.2	The use of XML and encoding.....	21
7.3	Resources and identifiers.....	22
7.3.1	On the use of percent encoding	22
7.3.2	Using identifiers in the REST Resource URLs	22
7.3.3	Non-normative identifier example	23
7.3.4	Implementation considerations.....	24
7.4	Referencing the SMP REST binding.....	24
7.5	Security	24
7.5.1	Message signature	24
7.5.2	Verifying the signature.....	25
7.5.3	Verifying the signature of the destination SMP.....	25
8	Appendix A: Schema for the REST interface	26

1 Document information

1.1 Document history

Date	Version	Initials	Changes
2009-02-12	0.1.0	GS	Created document
2009-02-16	0.1.2	GS	Added WSDLs & included logical data model
2009-02-17	0.5.0	GS	Minor correction
2009-03-30	0.6.0	GS	Changed into a REST profile, UDDI and REST profiles collected into one doc, removed hashed
2009-04-01	0.7.0	GS	Merge completed, UDDI profile updated with current keys and identifiers, implied data model discussed, encryption certificate placed, non-normative examples up-to-date
2009-04-29	0.8.0	GS	Started changes for 0.8 version
2009-05-04	0.8.0	GS	Final content for 0.8 version
2009-06-25	0.8.2	GS	Updated with structural changes & extension points for 0.9 version
2009-07-01	0.8.3	GS	Updates to examples, schema, references. Moved identifier information to 'common definitions' document.
2009-08-28	0.8.4	GS	Updated some references & diagrams, changed the redirect scheme, changed 'Certificate' element to 'CertificateQN'
2009-08-31	0.9.0	GS	Updated naming, redirect scheme, diagrams, schemas, examples, proof-reading
2009-10-22	0.9.5	GS	Added an identifier for referencing the REST binding of the SMP (SMP TNS), updated all diagrams & descriptions to reflect the DNS-based SML, removed requirement for port 80, removed wsu:Id element, editorial updates.
2009-11-17	0.9.6	GS	Added support for MinimumAuthenticationLevel, removed ServiceGroupReference, changed semantics of 'href' redirection attribute, added full AP certificate to ServiceMetadata record, updated redirect seq. diagram, removed /smp/ part of resource paths, added flag for indicating that recipient requires business level signatures, misc editorial updates.
2009-11-24	1.0.0	GS	Restructured ServiceMetadata structure, changed identifiers, updated all schemas and examples, misc editorial changes. Updated identifiers from the NNN_NNN_NNN format to NNN-NNN-NNN
2009-12-23	1.0.0	MHB/GS	Updates based on Philip

1.2 Editor

Gert Sylvest, Avanade

2 Contributors (alphabetically)

Jens Jakob Andersen, NITA

Kenneth Bengtsson, Alfa1lab

Mikkel Hippe Brun, NITA

Mike Edwards, IBM

Paul Fremantle, WSO2

Thomas Gundel, IT Crew

Philip Helger, Bundesrechenzentrum

Joakim Recht, Trifork

Carl-Markus Piswanger, Bundesrechenzentrum

Bergþór Skúlason, NITA

Gert Sylvest, Avanade

1 **3 Introduction**

2 This document describes the REST (Representational State Transfer) interface for Service Metadata
3 Publication within the Business Document Exchange Network (BUSDOX). It describes the request/response
4 exchanges between a Service Metadata Publisher and a client wishing to discover endpoint information. A
5 client could be an end-user business application or an Access Point. It also defines the request processing
6 that must happen at the client.

7 **3.1 Goals and non-goals**

8 The goal of this document is to define the REST lookup interface that Service Metadata Publishers (“SMP”)
9 and clients must support. Decisions regarding physical data format and management interfaces are left to
10 implementers of such a service.

11 Service Metadata Publishers may be subject to additional constraints of agreements and governance
12 frameworks within instances of the BUSDOX infrastructure not covered in this specification, which only
13 addresses the technical interface of such a service.

14 **3.2 Terminology**

15 For a definition of terms, see [BDEN-CDEF].

16 **3.2.1 Notational conventions**

17 For notational conventions, see [BDEN-CDEF].

18 **3.2.2 Normative references**

19 [XML-DSIG] “XML Signature Syntax and Processing (Second Edition)”, <http://www.w3.org/TR/xmlldsig-core/>

20 [RFC3986] "Uniform Resource Identifier (URI): Generic Syntax", <http://tools.ietf.org/html/rfc3986>

21 [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/>

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

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

26 **3.2.3 Non-normative references**

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

29 [REST] “Architectural Styles and the Design of Network-based Software Architectures”,
30 <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

31 [BDEN-SML] Service Metadata Locator Profile, ServiceMetadataLocator.pdf

32 **3.3 Namespaces**

33 For a list of namespaces and prefixes used in this document, see [BDEN-CDEF].

34

35 **4 The Service Discovery Process**

36 The interfaces of the Service Metadata Locator service and the Service Metadata Publisher (SMP) service
37 cover both sender-side lookup and metadata management performed by SMPs. Business Document
38 Exchange Network (BUSDOX) mandates the following interfaces for these services:

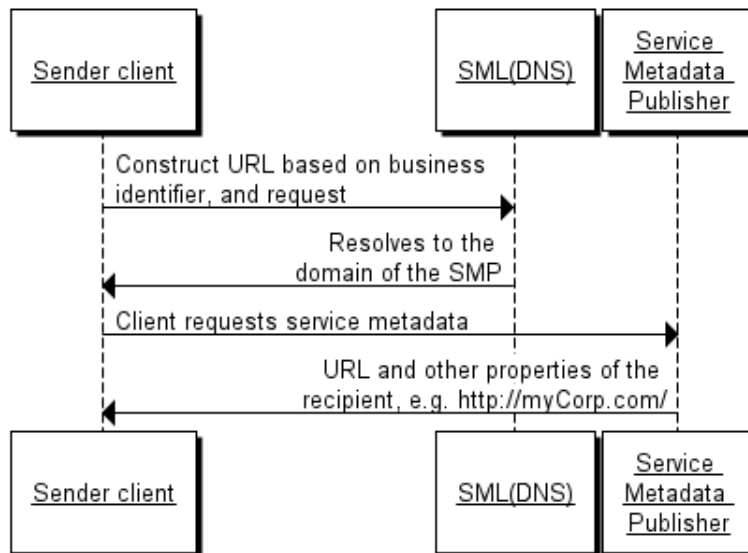
- 39 • Service Metadata Locator:
 - 40 ○ DNS-based resolve mechanism to locate individual SMPs
 - 41 ○ Management interface towards SMPs
- 42 • Service Metadata Publishers:
 - 43 ○ Discovery interface towards senders

44
45 This specification only covers the discovery interface for Service Metadata Publication services.

46 **4.1 Discovery flow**

47 For a sender, the first step in the Discovery process is to establish the location of the Service Metadata
48 relating to the particular Participant Identifier to which the sender wants to transmit a message. Each
49 participant identifier is registered with one and only one Service Metadata Publisher. The sender looks up
50 the endpoint for the Service Metadata Publisher using the DNS-based Service Metadata Locator service
51 (this is a regular DNS resolve). The sender can then retrieve the metadata associated with the Participant
52 Identifier. This metadata includes the information necessary to transmit the message to the recipient
53 endpoint.

54 The diagram below represents the lookup flow for a sender contacting both the Service Metadata Locator
55 and the SMP.



56
57 **Fig. 1. Endpoint lookup with Service Metadata**

58

59 **4.2 Discovering services associated with a Participant Identifier**

60 In addition to the direct lookup of Service Metadata based on participant identifier and document type, a
61 sender may want to discover what document types can be handled by a specific participant identifier. Such
62 discovery is relevant for applications supporting several equivalent business processes. Knowing the
63 capabilities of the recipient is valuable information to a sender application and ultimately to an end user.
64 Eg. the end user may be presented with a choice between a “simple” and a “rich” business process.

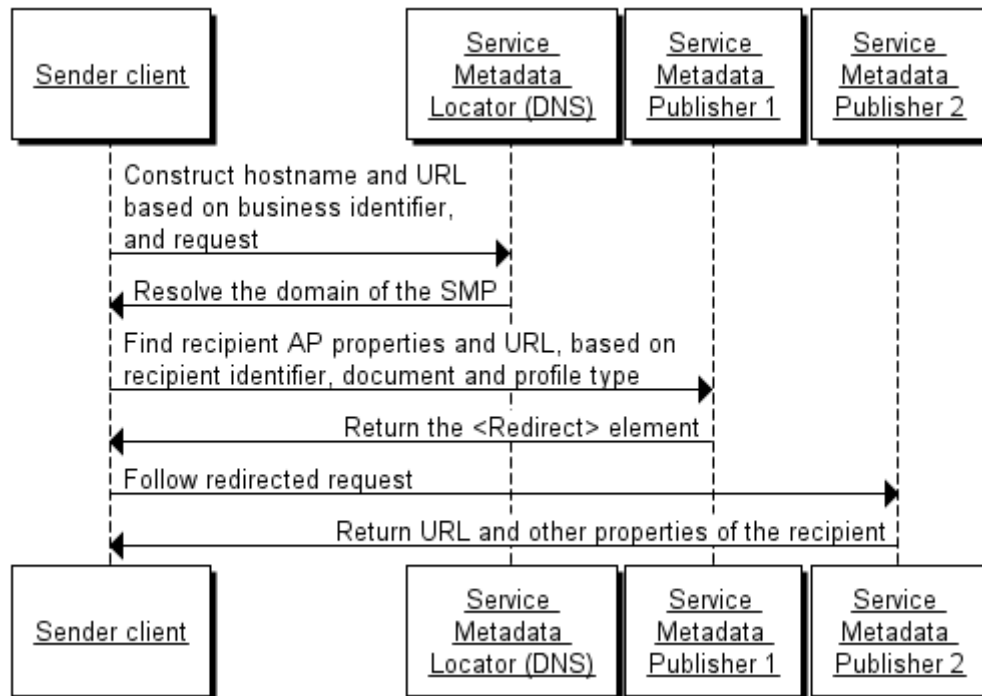
65 This is enabled by a pattern where the sender first retrieves the ServiceGroup entity, which holds a list of
66 references to the ServiceMetadata resources associated with it. The SignedServiceMetadata in turn holds
67 the metadata information that describes the capabilities associated with the recipient participant identifier

68 **4.3 Service Metadata Publisher Redirection**

69 For each participant identifier, the Service Metadata Locator may only point to a single Service Metadata
70 Publisher. There are cases however where the owner of a participant identifier may want to use different
71 Service Metadata Publishers for different document types or processes. This is supported by Service
72 Metadata Publisher Redirection.

73 In this pattern, the sender is redirected by the Service Metadata Publisher to a secondary, remote Service
74 Metadata Publisher where the actual SignedServiceMetadata can be found. A special element within the
75 SignedServiceMetadata record of the SMP points to the SMP that has the actual Service Metadata, and
76 certificate information for that SMP. The diagram below shows this flow:

77



78

79

Fig. 2: Service Metadata Redirection

80 Note that only one degree of redirect is allowed; clients are not required to follow more than one redirect,
81 i.e. a redirect resource cannot point to another redirect resource. Allowing one level of redirect permits the
82 described use case to be realized, while avoiding the possibility of cyclic references and long chains of
83 redirects

84

85 **5 Interface model**

86 This specification defines a REST-based interface for retrieving Service Metadata, but does not specify
87 interfaces for creating, updating, deleting and managing Service Metadata, or any internal data storage
88 formats.

89 The goal is to allow the interface in this specification to expose data from many different Service Metadata
90 back-ends, which may be based on any suitable technology such as for example RDBMS, LDAP, or UDDI.

91 Note that when adding or deleting Participant Identifiers in the SMP, an implementation of the SMP will
92 need to reflect its custody of a Participant Identifier in the SML. Please see the SML specification [BDEN-
93 SML] for a description of the processes and interfaces for doing this.

94 **6 Data model**

95 This section outlines the data model of the interface. The data model comprises the following main data
96 types:

- 97 • ServiceGroup
- 98 • ServiceMetadata / SignedServiceMetadata

99

100 Supporting data types for these main types are:

- 101 • ServiceInformation
- 102 • ServiceEndpointList
- 103 • ParticipantIdentifier
- 104 • DocumentIdentifier
- 105 • Redirect
- 106 • Process
- 107 • ProcessList
- 108 • Endpoint

109 Each of these data types is described in detail in the following sections.

110 **6.1 On extension points**

111 For each major entity, extension points have been added with the optional <smp:Extension> element.

112 **6.1.1 Semantics and use**

113 Child elements of the <smp:Extension> element are known as “custom extension elements”. Extension
114 points may be used for *optional* extensions of service metadata. This implies:

- 115 • Extension elements added to a specific Service Metadata resource **MUST** be ignorable by any client
116 of the transport infrastructure. The ability to parse and adjust client behavior based on an
117 extension element **MUST NOT** be a prerequisite for a client to locate a service, or to make a
118 successful request at the referenced service.

119 • A client MAY ignore any extension element added to specific service metadata resource instances.

120

121 **6.2 ServiceGroup**

122 The ServiceGroup structure represents a set of services associated with a specific participant identifier that
 123 is handled by a specific Service Metadata Publisher. The ServiceGroup structure holds a list of references to
 124 SignedServiceMetadata resources in the ServiceList structure.

125 Pseudo-schema for ServiceGroup:

```

126 <smp:ServiceGroup>
127   <ids:ParticipantIdentifier scheme="xs:string">
128     xs:string
129   </ids:ParticipantIdentifier>
130   <smp:ServiceMetadataReferenceCollection>
131     <smp:ServiceMetadataReference href="xs:anyURI" />*
132   </smp:ServiceMetadataReferenceCollection>
133   <smp:Extension>xs:any</smp:Extension?>
134 </smp:ServiceGroup>
  
```

135
 136 Description of the individual fields (elements and attributes).
 137

Field	Description
ServiceGroup	Document element
ParticipantIdentifier	Represents the business level endpoint key and key type, e.g. a DUNS or GLN number that is associated with a group of services. See the ParticipantIdentifier section of the 'Common Definitions' document [BDEN-CDEF] for information on this data type.
ServiceMetadataReferenceCollection	The ServiceMetadataReferenceCollection structure holds a list of references to SignedServiceMetadata structures. From this list, a sender can follow the references to get each SignedServiceMetadata structure.
ServiceMetadataReference (0..*)	Contains the URL to a specific <i>SignedServiceMetadata</i> instance - see the REST binding section for details on the URL format. Note that references MUST refer to SignedServiceMetadata records that are signed by the certificate of the SMP. It must not point to SignedServiceMetadata resources published by external SMPs..
Extension	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extended metadata to individual references to Service Metadata resources.

138
 139

140 6.2.1 Non-normative example

141 Non-normative example of a ServiceGroup resource.

```
142 <?xml version="1.0" encoding="utf-8" ?>
143 <!--
144   This sample assumes that the service metadata publisher resides at
145   "http://serviceMetadata.eu/".
146   It assumes that the business identifier is "0010:5798000000001".
147 -->
148 <ServiceGroup xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
149   xmlns:ids="http://busdox.org/transport/identifiers/1.0/">
150   <ids:ParticipantIdentifier scheme="busdox-actorid-upis">
151     0010:5798000000001
152   </ids:ParticipantIdentifier>
153   <ServiceMetadataReferenceCollection>
154     <ServiceMetadataReference href="http://serviceMetadata.eu/busdox-actorid-
155 upis%3A%3A0010%3A5798000000001/services/busdox-docid-
156 qns%3A%3Aurn%3Aaosis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
157 2%3A%3AInvoice%23%23UBL-2.0" />
158   </ServiceMetadataReferenceCollection>
159   <Extension>
160     <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
161   </Extension>
162 </ServiceGroup>
163
164
165
166
167
```

168 **6.3 ServiceMetadata**

169 This data structure represents Metadata about a specific electronic service. The role of the
170 ServiceMetadata structure is to associate a participant identifier with the ability to receive a specific
171 document type over a specific transport. It also describes which business processes a document can
172 participate in, and various operational data such as service activation and expiration times.

173 The *ServiceMetadata* resource contains all the metadata about a service that a sender Access Point needs
174 to know in order to send a message to that service..

175 **6.3.1 Redirection**

176 For recipients that want to associate more than one SMP with their participant identifier, they may redirect
177 senders to an alternative SMP for specific document types. To achieve this, the ServiceMetadata element
178 defines the optional element 'Redirect'. This element holds the URL of the alternative SMP, as well as the
179 Subject Unique Identifier of the destination SMPs certificate used to sign its resources.

180 In the case where a client encounters such a redirection element, the client **MUST** follow the first redirect
181 reference to the alternative SMP. If the SignedServiceMetadata resource at the alternative SMP also
182 contains a redirection element, the client **SHOULD NOT** follow that redirect. It is the responsibility of the
183 client to enforce this constraint.

184 Pseudo-schema for this data type:

```
185 <smp:ServiceMetadata>  
186   [<smp:ServiceInformation /> | <smp:Redirect />]  
187 </smp:ServiceMetadata>
```

188

189 Pseudo-schema for the 'ServiceInformation' data type:

```
190 <smp:ServiceInformation>  
191   <ids:ParticipantIdentifier scheme="xs:string">xs:string  
192     </ids:ParticipantIdentifier>  
193   <ids:DocumentIdentifier scheme="xs:string" />  
194   <smp:ProcessList>  
195     <smp:Process>+  
196       <ids:ProcessIdentifier scheme="xs:string" />  
197       <smp:ServiceEndpointList>  
198         <smp:Endpoint transportProfile="xs:string">+  
199           <wsa:EndpointReference />  
200           <smp:RequireBusinessLevelSignature>xs:boolean  
201             </smp:RequireBusinessLevelSignature>  
202           <smp:MinimumAuthenticationLevel>xs:string  
203             </smp:MinimumAuthenticationLevel >?  
204           <smp:ServiceActivationDate>xs:dateTime  
205             </smp:ServiceActivationDate>?  
206           <smp:ServiceExpirationDate>xs:dateTime  
207             </smp:ServiceExpirationDate>?  
208           <smp:Certificate>xs:string</smp:Certificate>
```

```

209         <smp:ServiceDescription>xs:string
210             </smp:ServiceDescription>
211         <smp:TechnicalContactUrl>xs:anyURI
212             </smp:TechnicalContactUrl>
213         <smp:TechnicalInformationUrl>xs:anyURI
214             </smp:TechnicalInformationUrl?>
215         <smp:Extension>xs:any</smp:Extension?>
216     </smp:Endpoint>
217 </smp:ServiceEndpointList>
218 <smp:Extension>xs:any</smp:Extension?>
219 </smp:Process>
220 </smp:ProcessList>
221 <smp:Extension>xs:any</smp:Extension?>
222 </smp:ServiceInformation>
223

```

224 Pseudo-schema for the 'Redirect' data type:

```

225
226 <smp:Redirect href="xs:anyURI">
227     <smp:CertificateUID>xs:string</smp:CertificateUID>
228     <smp:Extension />? <!-- may contain any type -->
229 </smp:Redirect>
230

```

231 The extension element may contain any XML element. Clients MAY ignore this element. It can be used to
232 add extension metadata to the service metadata.

233 The 'href' attribute of the Redirect element contains the full address of the destination SMP record that the
234 client is redirected to.

235 For example, assume that an SMP called "SMP1" has the address "http://smp1.eu", and another SMP called
236 "SMP2" has the address "http://smp2.eu", and a client requests a resource with the following URL (note
237 that these examples have been percent encoded using the rules from [BDEN-CDEF]):

```

238 http://smp1.eu/busdox-actorid-upis%3A%3A0010%3A5798000000001/services/busdox-
239 docid-
240 qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
241 2%3A%3AInvoice%23%23UBL-2.0

```

242 We now assume that the owner of these metadata has moved them to SMP2. SMP1 would then return a
243 *SignedServiceMetadata* resource with a *Redirect* child element that has the "href" attribute set to

```

244 http://smp2.eu/busdox-actorid-upis%3A%3A0010%3A5798000000001/services/busdox-
245 docid-
246 qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
247 2%3A%3AInvoice%23%23UBL-2.0
248

```

249 For the list of endpoints under each <Endpoint> element in the ServiceEndpointList, each endpoint MUST
250 have different values of the transportProfile attribute, i.e. represent bindings to different transports.

251 Description of the individual fields (elements and attributes).

252

Field	Description
ServiceMetadata	Document element
/Redirect	The direct child element of <i>ServiceMetadata</i> is either the <i>Redirect</i> element or the <i>ServiceInformation</i> element. The <i>Redirect</i> element indicates that a client must follow the URL of the <i>href</i> attribute of this element.
/Redirect/CertificateUID	Holds the Subject Unique Identifier of the certificate of the destination SMP. A client SHOULD validate that the Subject Unique Identifier of the certificate used to sign the resource at the destination SMP matches the Subject Unique Identifier published in the redirecting SMP.
/Redirect/Extension	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extension metadata to the <i>Redirect</i> .
/ServiceInformation	The direct child element of <i>ServiceMetadata</i> is either the <i>Redirect</i> element or the <i>ServiceInformation</i> element. The <i>ServiceInformation</i> element contains service information for an actual service registration, rather than a redirect to another SMP.
ServiceInformation/ ParticipantIdentifier	The participant identifier. Comprises the identifier, and an identifier scheme. This identifier MUST have the same value of the {id} part of the URI of the enclosing <i>ServiceMetadata</i> resource. See the <i>ParticipantIdentifier</i> section of the 'Common Definitions' document [BDEN-CDEF] for information on this data type.
ServiceInformation/ DocumentIdentifier	Represents the type of document that the recipient is able to handle. The document is represented by an identifier (identifying the document type) and an identifier scheme, which the format of the identifier itself. See the <i>DocumentIdentifier</i> section of the 'Common Definitions' document [BDEN-CDEF] for information on this data type.
ServiceInformation/ ProcessList	Represents the processes that a specific document type can participate in, and endpoint address and binding information. Each process element describes a specific business process that accepts this type of document as input and holds a list of endpoint addresses (in the case that the service supports multiple transports) of services that implement the business process, plus information about the transport used for each endpoint. See the <i>Process</i> section of the 'Common Definitions' document [BDEN-CDEF] for information on the identifier format.
/ProcessList/Process/ ProcessIdentifier	The identifier of the process. See the 'Common Definitions' document for a definition of process identifiers [BDEN-CDEF]
/ProcessList/Process/ ServiceEndpointList	List of one or more endpoints that support this process.
ServiceInformation/ ProcessList/./Endpoint	Endpoint represents the technical endpoint and address type of the recipient, as an URL.

/ServiceEndpointList/ Endpoint/EndpointReference	The address of an endpoint, as an WS-Addressing Endpoint Reference (EPR)
ServiceInformation/ ProcessList/./Endpoint/ @transportProfile	Indicates the type of BUSDOX transport that is being used between access points, e.g. the BUSDOX START profile. This specification defines the following identifier URI which denotes the BUSDOX START transport: "busdox-transport-start"
ServiceInformation/ ProcessList/./Endpoint/ RequireBusinessLevelSignature	Set to 'true' if the recipient requires business-level signatures for the message, meaning a signature applied to the business message before the message is put on the transport. This is independent of the transport-level signatures that a specific transport profile, such as the START profile, might mandate. This flag does not indicate which type of business-level signature might be required. Setting or consuming business-level signatures would typically be the responsibility of the final senders and receivers of messages, rather than a set of APs.
ServiceInformation/ ProcessList/./Endpoint/ MinimumAuthenticationLevel	Indicates the minimum authentication level that recipient requires. The specific semantics of this field is defined in a specific instance of the BUSDOX infrastructure. It could for example reflect the value of the "urn:eu:busdox:attribute:assurance-level" SAML attribute defined in the START specification.
ServiceInformation/ ProcessList/./Endpoint/ ServiceActivationDate	Activation date of the service. Senders should ignore services that are not yet activated. Format of ServiceActivationDate date is xs:dateTime
/ProcessList/./Endpoint/ ServiceExpirationDate	Expiration date of the service. Senders should ignore services that are expired. Format of ServiceExpirationDate date is xs:dateTime.
/ProcessList/./Endpoint/ Certificate	Holds the complete signing certificate of the recipient AP, as a PEM base 64 encoded X509 DER formatted value.
/ProcessList/./Endpoint/ ServiceDescription	A human readable description of the service
/ProcessList/./Endpoint/ TechnicalContactUrl	Represents a link to human readable contact information. This might also be an email address.
/ProcessList/./Endpoint/ TechnicalInformationUrl	A URL to human readable documentation of the service format. This could for example be a web site containing links to XML Schemas, WSDLs, Schematrons and other relevant resources.
/Process/Extension	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extension metadata to the process metadata block as a whole.
/ServiceInformation/ Extension	The extension element may contain any XML element. Clients MAY ignore this element. It can be used to add extension metadata to the service metadata.

253

254 **6.3.2 Non-normative example**

255 For a non-normative example of a ServiceMetadata resource, see the SignedServiceMetadata non-
256 normative example below.

257 **6.4 SignedServiceMetadata**

258 The SignedServiceMetadata structure is a ServiceMetadata structure that has been signed by the
259 ServiceMetadataPublisher, according to governance policies that are not covered by this document.
260 Pseudo-schema for this data type:

```
261 <smp:SignedServiceMetadata>  
262   <smp:ServiceMetadata />  
263   <ds:Signature />  
264 </smp:SignedServiceMetadata>  
265
```

- 266 • **ServiceMetadata** : The ServiceMetadata element covered by the signature.
- 267 • **Signature** represents an enveloped XML signature over the SignedServiceMetadata element.

268 **6.4.1 Non-normative example**

269 Non-normative example of a SignedServiceMetadata resource.

```
270 <?xml version="1.0" encoding="utf-8" ?>  
271 <!--  
272   This sample assumes that the service metadata publisher resides at  
273   "http://serviceMetadata.eu/".  
274   It assumes that the business identifier is "0010:5798000000001".  
275 -->  
276 <SignedServiceMetadata xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"  
277 xmlns:ids="http://busdox.org/transport/identifiers/1.0/">  
278   <ServiceMetadata  
279     xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"  
280     xmlns:wssu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
281 wssecurity-utility-1.0.xsd">  
282     <ServiceInformation>  
283       <ids:ParticipantIdentifier scheme="busdox-actorid-upis">  
284         0010:5798000000001  
285       </ids:ParticipantIdentifier>  
286       <ids:DocumentIdentifier scheme="busdox-docid-qns">  
287         urn:oasis:names:specification:ubl:schema:xsd:Invoice-2::Invoice##UBL-2.02  
288       </ids:DocumentIdentifier>  
289     <ProcessList>  
290       <Process>  
291         <ids:ProcessIdentifier scheme="cenbii-procid-ubl">BII04  
292         </ids:ProcessIdentifier>  
293       <ServiceEndpointList>  
294         <Endpoint transportProfile="busdox-transport-start">  
295           <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">  
296             <Address>http://busdox.org/sampleService/</Address>  
297           </EndpointReference>  
298           <RequireBusinessLevelSignature>>false  
299           </RequireBusinessLevelSignature>  
300           <MinimumAuthenticationLevel>2</MinimumAuthenticationLevel>  
301           <ServiceActivationDate>2009-05-01T09:00:00</ServiceActivationDate>  
302           <ServiceExpirationDate>2016-05-01T09:00:00</ServiceExpirationDate>  
303
```

```

304         <Certificate>TlRMTVNTUAABAAAAt7IY4gk...</Certificate>
305         <ServiceDescription>invoice service</ServiceDescription>
306         <TechnicalContactUrl>https://example.com</TechnicalContactUrl>
307         <TechnicalInformationUrl>http://example.com/info
308             </TechnicalInformationUrl>
309     </Endpoint>
310 </ServiceEndpointList>
311 </Process>
312
313 <Process>
314     <ids:ProcessIdentifier scheme="cenbii-procid-ubl">BII07
315     </ids:ProcessIdentifier>
316     <ServiceEndpointList>
317         <Endpoint transportProfile="busdoux-transport-start">
318             <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
319                 <Address>http://busdoux.org/sampleService/</Address>
320             </EndpointReference>
321             <RequireBusinessLevelSignature>true
322             </RequireBusinessLevelSignature>
323             <MinimumAuthenticationLevel>1</MinimumAuthenticationLevel>
324             <ServiceActivationDate>2009-05-01T09:00:00</ServiceActivationDate>
325             <ServiceExpirationDate>2016-05-01T09:00:00</ServiceExpirationDate>
326             <Certificate>TlRMTVNTUAABAAAAt7IY4gk...</Certificate>
327             <ServiceDescription>invoice service</ServiceDescription>
328             <TechnicalContactUrl>https://example.com</TechnicalContactUrl>
329             <TechnicalInformationUrl>http://example.com/info
330                 </TechnicalInformationUrl>
331             <Extension>
332                 <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
333             </Extension>
334         </Endpoint>
335     </ServiceEndpointList>
336     <Extension>
337         <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
338     </Extension>
339 </Process>
340 </ProcessList>
341
342     <Extension>
343         <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
344     </Extension>
345 </ServiceInformation>
346 </ServiceMetadata>
347
348 <!-- Message signature, details omitted for brevity -->
349 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#" />
350
351 </SignedServiceMetadata>

```

352 6.4.2 Redirect, non-normative example

```

353 <?xml version="1.0" encoding="utf-8" ?>
354 <!--
355     This sample assumes that the user contacts a service metadata publisher that
356     resides at "http://serviceMetadata.eu/",
357     but is redirected to a service metadata publisher that resides at
358     "http://serviceMetadata2.eu/".
359     -->
360 <SignedServiceMetadata
361     xmlns="http://busdoux.org/serviceMetadata/publishing/1.0/">

```

```
362     <ServiceMetadata
363         xmlns="http://busdox.org/serviceMetadata/publishing/1.0/">
364         <Redirect xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
365 href="http://serviceMetadata2.eu/busdox-actorid-
366 upis%3A%3A0010%3A5798000000001/services/busdox-docid-
367 qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
368 2%3A%3AInvoice%23%23UBL-2.0">
369             <CertificateUID>PID:9208-2001-3-279815395</CertificateUID>
370             <Extension>
371                 <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
372             </Extension>
373         </Redirect>
374     </ServiceMetadata>
375     <!-- Message signature, details omitted for brevity -->
376     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#" />
377 </SignedServiceMetadata>

378
379
```

380 **7 Service Metadata Publishing REST binding**

381 This section describes the REST binding of the Service Metadata Publishing interface.

382 **7.1 The use of HTTP**

383 A service implementing the REST binding MUST set the HTTP “content-type” header, and give it a value of
384 “text/xml”. A service implementing the REST profile MUST NOT use TLS (Transport Layer Security) or SSL
385 (Secure Sockets Layer). An instance of the BUSDOX infrastructure MAY set restrictions on what ports are
386 allowed.

387 **7.1.1 An implementation of the SMP might choose to manage resources through the HTTP**
388 **POST, PUT and DELETE verbs. It is however up to each implementation to choose how to**
389 **manage records, and use of HTTP POST, PUT and DELETE is not mandated or regulated**
390 **by this specification.HTTP status codes**

391 HTTP GET operations MUST return the following HTTP status codes:

HTTP Status Code	Meaning
200	Must be returned if the resource is retrieved correctly.
404	Code 404 must be returned if a specific resource could not be found. This could for example be the result of a request containing a participant identifier that does not exist.
500	Code 500 must be returned if the service experiences an internal processing error.

392

393 The service MAY support other HTTP status codes as well.

394 The service SHOULD NOT use redirection in the manner indicated by the HTTP 3xx codes. Clients are not
395 required to support active redirection.

396 **7.2 The use of XML and encoding**

397 XML document returned by HTTP GET MUST be UTF-8 encoded. They MUST contain a document type
398 declaration starting with “<?xml” which includes the ‘encoding’ attribute set to “UTF-8”. Please observe
399 that the content of the encoding attribute is case sensitive. Version 1.0 of XML is used.

400

401 **7.3 Resources and identifiers**

402 The REST interface comprises 2 types of resources.

Resource	URI	Method	XML resource root element	HTTP Status	Description of returned content
ServiceGroup	/ {identifier scheme} :: {id}	GET	<ServiceGroup>	200; 500; 404	Holds the participant identifier of the recipient, and a list of references to individual ServiceMetadata resources that are associated with that participant identifier.
SignedService Metadata	/ {identifier scheme} :: {id} / services / {docType} See section below for {docType} format	GET	<SignedServiceMetadata>	200; 500; 404	Holds all of the metadata about a Service, or a redirection URL to another Service Metadata Publisher holding this information.

403 Fig. 3: Table of resources and identifiers

404 A service implementing the REST binding MUST support these resource types. It MUST provide access to
405 these using the URI scheme of table in Fig. 3.

406 **7.3.1 On the use of percent encoding**

407 See the ‘Common Definitions’ document for requirements for percent encoding [BDEN-CDEF].

408 **7.3.2 Using identifiers in the REST Resource URLs**

409 This section describes specifically how participant and document identifiers are used to reference
410 <ServiceGroup> and <SignedServiceMetadata> REST resources. For a general definition on how to
411 represent participant and document identifiers in URLs, see the ‘Common Definitions’ document [BDEN-
412 CDEF].

413 For the URL referencing a <ServiceGroup> resource, the “ {identifier scheme} :: {id} ” part follows the
414 participant identifier format described in the “ParticipantIdentifier” section of the ‘Common Definitions’
415 document [BDEN-CDEF].

416

417 The following URL format is used:

418 /{identifier scheme}::{id}

419 In the reference to the SignedServiceMetadata or Redirect elements ({id}/services/ {docType}), the
420 {docType} part consists of {document identifier scheme}::{document identifier}. For information on the
421 format of {document identifier}, see the DocumentIdentifier section of the 'Common Definitions' document
422 [BDEN-CDEF].

423 7.3.3 Non-normative identifier example

424 We assume a Service Metadata Publisher can be accessed at the URL "http://serviceMetadata.eu".

425 A business with the participant identifier "0010:5798000000001" would have the following identifier for
426 the ServiceGroup resource:

427 http://serviceMetadata.eu/busdox-actorid-upis::0010:5798000000001

428 After percent encoding:

429 **http://serviceMetadata.eu/busdox-actorid-upis%3a%3a0010%3a5798000000001**

430 In the case of a NES-UBL order, a SignedServiceMetadata or Redirect resource can then be identified by

- 431 • **Identifier format type:** busdox-docid-qns
- 432 • **Root namespace:** urn:oasis:names:specification:ubl:schema:xsd:Order-2
- 433 • **Document element local name:** Order
- 434 • **Subtype identifier:** UBL-2.0 (since several versions of the Order schema may use the same
435 namespace + document element name)

436 The document type identifier will then be:

437 busdox-docid-qns::urn:oasis:names:specification:ubl:schema:xsd:Order-2::Order##UBL-2.0

438 The document type identifier MUST be percent encoded as described in [RFC3986]. The above, non-
439 normative example is thus encoded to

440 *busdox-docid-qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AOrder-*
441 *2%3A%3AOrder%23%23UBL-2.0*

442

443

444 The entire URL reference to a SignedServiceMetadata or Redirect element thus has the form

445 {URL to server}/{identifier scheme}::{id}/services/{document identifier
446 type}::{rootNamespace}::{documentElementLocalName}[#{Subtype identifier}]

447 The percent-encoded form of the identifier using the above example will then be

448 ***[http://serviceMetadata.eu/busdox-actorid-upis%3a%3a0010%3a579800000001/services/busdox-docid-
qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AOrder-
2%3A%3AOrder%23%23UBL-2.0](http://serviceMetadata.eu/busdox-actorid-upis%3a%3a0010%3a579800000001/services/busdox-docid-
449 qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AOrder-
450 2%3A%3AOrder%23%23UBL-2.0)***

451 Note that the forward slashes delimiting the individual parts of the REST resource identifier URL are *not*
452 percent encoded, since they are part of the URL.

453 **7.3.4 Implementation considerations**

454 When a client is redirected to an SMP using the DNS-based SML scheme described in [BDEN-SML], the HTTP
455 “host” header will be set to a value originating from the CNAME alias set in the SML
456 (<http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.23>). Implementations should be
457 prepared to accept requests with this “host” header value.

458 **7.4 Referencing the SMP REST binding**

459 For referencing the SMP REST binding, for example from Service Metadata Locator records, the following
460 identifier should be used for the version 1.0 of the SMP REST binding:

461 <http://busdox.org/serviceMetadata/publishing/1.0/>

462 This is identical to the target namespace of the SMP schema.

463 **7.5 Security**

464 At the transport level, the service is not secured.

465 **7.5.1 Message signature**

466 The message returned by the service is signed by the Service Metadata Publisher with XML-Signature
467 according to the standard <http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/>.

468 The signature MUST be an enveloped XML signature represented via an <ds:Signature> element embedded
469 in the <SignedServiceMetadata> element. The <ds:Signature> element MUST be constructed according to
470 the following rules:

- 471 • The <Reference> MUST use exactly one Transform being:
472 “<http://www.w3.org/2000/09/xmldsig#envelopedsignature>”
 - 473 • The <ds:KeyInfo> element MUST contain an <ds:X509Data> element with an <ds:X509Certificate>
474 sub-element containing the signer’s X.509 certificate as PEM base 64 encoded X509 DER value.
 - 475 • The canonicalization algorithm MUST be <http://www.w3.org/2001/10/xml-exc-c14n#>
 - 476 • The SignatureMethod MUST be <http://www.w3.org/2000/09/xmldsig#rsa-sha1>
 - 477 • The DigestMethod MUST be <http://www.w3.org/2000/09/xmldsig#sha1>
- 478

479 **7.5.2 Verifying the signature**

480 When verifying the signature, the consumer has access to the full certificate as a PEM base 64 encoded
481 X509 DER value within the <Signature> element. The consumer may verify the signature by a) extracting
482 the certificate from the <ds:X509Data> element, b) verify that it has been issued by the trusted root, c)
483 perform a validation of the signature, and d) perform the required certificate validation steps (which might
484 include checking expiration/activation dates and revocation lists).

485 **7.5.3 Verifying the signature of the destination SMP**

486 For the redirect scheme, the unique identifier of the destination SMP signing certificate is stored at the
487 redirecting SMP. In addition to the regular signature validation performed by the client of the destination
488 SMP resources, the client SHOULD also validate that the identifier of the destination SMP signing certificate
489 corresponds to the unique identifier which the redirecting SMP claims belongs to the destination SMP.

490

491

492 **8 Appendix A: Schema for the REST interface**

493 This section defines the XML Schema for all the resources of the REST interface.

```
494 <?xml version="1.0" encoding="utf-8"?>
495 <xs:schema id="ServiceMetadataPublishing"
496           targetNamespace="http://busdox.org/serviceMetadata/publishing/1.0/"
497           elementFormDefault="qualified"
498           xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
499           xmlns:ids="http://busdox.org/transport/identifiers/1.0/"
500           xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
501           xmlns:xs="http://www.w3.org/2001/XMLSchema"
502           xmlns:wsa="http://www.w3.org/2005/08/addressing">
503   <xs:import schemaLocation="xmldsig-core-schema.xsd"
504   namespace="http://www.w3.org/2000/09/xmldsig#" />
505   <xs:import schemaLocation="oasis-200401-wss-wssecurity-utility-1.0.xsd"
506   namespace="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-
507   1.0.xsd" />
508   <xs:import schemaLocation="ws-addr.xsd"
509   namespace="http://www.w3.org/2005/08/addressing" />
510   <xs:import schemaLocation="Identifiers-1.0.xsd"
511   namespace="http://busdox.org/transport/identifiers/1.0/" />
512
513   <xs:element name="ServiceGroup" type="ServiceGroupType"/>
514   <xs:element name="ServiceMetadata" type="ServiceMetadataType"/>
515   <xs:element name="SignedServiceMetadata" type="SignedServiceMetadataType"/>
516
517   <xs:complexType name="SignedServiceMetadataType">
518     <xs:sequence>
519       <xs:element ref="ServiceMetadata"/>
520       <xs:element ref="ds:Signature" />
521     </xs:sequence>
522   </xs:complexType>
523
524   <xs:complexType name="ServiceMetadataType">
525     <xs:sequence>
526       <xs:choice>
527         <xs:element name="ServiceInformation" type="ServiceInformationType"/>
528         <xs:element name="Redirect" type="RedirectType"/>
529       </xs:choice>
530     </xs:sequence>
531   </xs:complexType>
532
533   <xs:complexType name="ServiceInformationType">
534     <xs:sequence>
535       <xs:element ref="ids:ParticipantIdentifier" />
536       <xs:element ref="ids:DocumentIdentifier" />
537       <xs:element name="ProcessList" type="ProcessListType" />
538       <xs:element name="Extension" type="ExtensionType" minOccurs="0" />
539     </xs:sequence>
540   </xs:complexType>
541
542   <xs:complexType name="ProcessListType">
543     <xs:sequence>
544       <xs:element name="Process" type="ProcessType" maxOccurs="unbounded" />
545     </xs:sequence>
546   </xs:complexType>
547
548   <xs:complexType name="ProcessType">
549     <xs:sequence>
550       <xs:element ref="ids:ProcessIdentifier" />
```

```

551     <xs:element name="ServiceEndpointList" type="ServiceEndpointList"/>
552     <xs:element name="Extension" type="ExtensionType" minOccurs="0" />
553   </xs:sequence>
554 </xs:complexType>
555
556 <xs:complexType name="ServiceEndpointList">
557   <xs:sequence>
558     <xs:element name="Endpoint" type="EndpointType" maxOccurs="unbounded" />
559   </xs:sequence>
560 </xs:complexType>
561
562 <xs:complexType name="EndpointType">
563   <xs:sequence>
564     <xs:element ref="wsa:EndpointReference"/>
565     <xs:element name="RequireBusinessLevelSignature" type="xs:boolean" />
566     <xs:element name="MinimumAuthenticationLevel" type="xs:string" minOccurs="0" />
567     <xs:element name="ServiceActivationDate" type="xs:dateTime" minOccurs="0" />
568     <xs:element name="ServiceExpirationDate" type="xs:dateTime" minOccurs="0" />
569     <xs:element name="Certificate" type="xs:string" />
570     <xs:element name="ServiceDescription" type="xs:string" />
571     <xs:element name="TechnicalContactUrl" type="xs:anyURI" />
572     <xs:element name="TechnicalInformationUrl" type="xs:anyURI" minOccurs="0" />
573     <xs:element name="Extension" type="ExtensionType" minOccurs="0" />
574   </xs:sequence>
575   <xs:attribute name="transportProfile" type="xs:string" />
576 </xs:complexType>
577
578 <xs:complexType name="ServiceGroupType">
579   <xs:sequence>
580     <xs:element ref="ids:ParticipantIdentifier" />
581     <xs:element name="ServiceMetadataReferenceCollection"
582 type="ServiceMetadataReferenceCollectionType" />
583     <xs:element name="Extension" type="ExtensionType" minOccurs="0" />
584   </xs:sequence>
585 </xs:complexType>
586 <xs:complexType name="ServiceMetadataReferenceCollectionType">
587   <xs:sequence>
588     <xs:element name="ServiceMetadataReference" type="ServiceMetadataReferenceType"
589 minOccurs="0" maxOccurs="unbounded" />
590   </xs:sequence>
591 </xs:complexType>
592 <xs:complexType name="ServiceMetadataReferenceType">
593   <xs:attribute name="href" type="xs:anyURI" />
594 </xs:complexType>
595 <xs:complexType name="RedirectType">
596   <xs:sequence>
597     <xs:element name="CertificateUID" type="xs:string" />
598     <xs:element name="Extension" type="ExtensionType" minOccurs="0" />
599   </xs:sequence>
600   <xs:attribute name="href" type="xs:anyURI" />
601 </xs:complexType>
602 <xs:complexType name="ExtensionType">
603   <xs:sequence>
604     <xs:any />
605   </xs:sequence>
606 </xs:complexType>
607 </xs:schema>
608

```