Specification

OpenPeppol AISBL

Peppol Transport Infrastructure
ICT - Models

Service Metadata Publishing (SMP)

Version: 1.2.0

Status: In use

Editors:

Gert Sylvest (NITA/Avanade)
Jens Jakob Andersen (NITA)
Klaus Vilstrup Pedersen (DIFI)
Mikkel Hippe Brun (NITA)
Paul Fremantle (NITA/WSO2)

Revision History

Version	Date	Description of changes	Author
1.0.0	2010-02-15	First version (pending EC approval)	Mikkel Hippe Brun, NITA
1.0.1	2010-10-01	EC approved	Klaus Vilstrup Pedersen, DIFI
1.1.0	2012-08-15	Make room for alternative Transport Protocols e.g. AS2	Klaus Vilstrup Pedersen, DIFI
1.2.0	2021-02-24	Updated the references Improved layout Explicitly allowing Content-Type "application/xml" as it is equivalent to "text/xml" (chapter 5.1) Removing the requirement that the encoding attribute value is case sensitive (chapter 5.2) Change "is not" to "MUST NOT" in chapter 5.5 Replaced the references to the BusDox Common Definition document (BDEN-CEDF) Added clarifications on ServiceActivationDate and ServiceExpirationDate Linking peppol-smp-types-v1.xsd in the Appendix Fixed a typo in the name of the transformation Changed the Canonicalization Algorithm from "Exclusive" to "Inclusive"	Philip Helger, OpenPeppol OO

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Contributors

Organisations

DIFI (Direktoratet for forvaltning og IKT)[1], Norway, www.difi.no

NITA (IT- og Telestyrelsen)[2], Denmark, www.itst.dk

BRZ (Bundesrechenzentrum)[3], Austria, www.brz.gv.at

Consip, Italy

OpenPeppol

Persons

Bergthór Skúlason, NITA

Carl-Markus Piswanger, BRZ

Gert Sylvest, NITA/Avanade (editor)

Jens Jakob Andersen, NITA

Joakim Recht, NITA/Trifork

Kenneth Bengtsson, NITA/Alfa1lab

Klaus Vilstrup Pedersen, DIFI

Mike Edwards, NITA/IBM

Mikkel Hippe Brun, NITA

Paul Fremantle, NITA/WSO2

Philip Helger, BRZ/OpenPeppol OO

Thomas Gundel, NITA/IT Crew

---

[1] English: Agency for Public Management and eGovernment
[2] English: National IT- and Telecom Agency
[3] English: Austrian Federal Computing Centre

Table of contents

1 Introduction

1.1 Objective

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

1.2 Scope

This specification relates to the Technical Transport Layer i.e. BusDox specifications. The BusDox specifications can be used in many interoperability settings. In the Peppol context, it provides transport for procurement documents as specified in the Peppol Profiles.
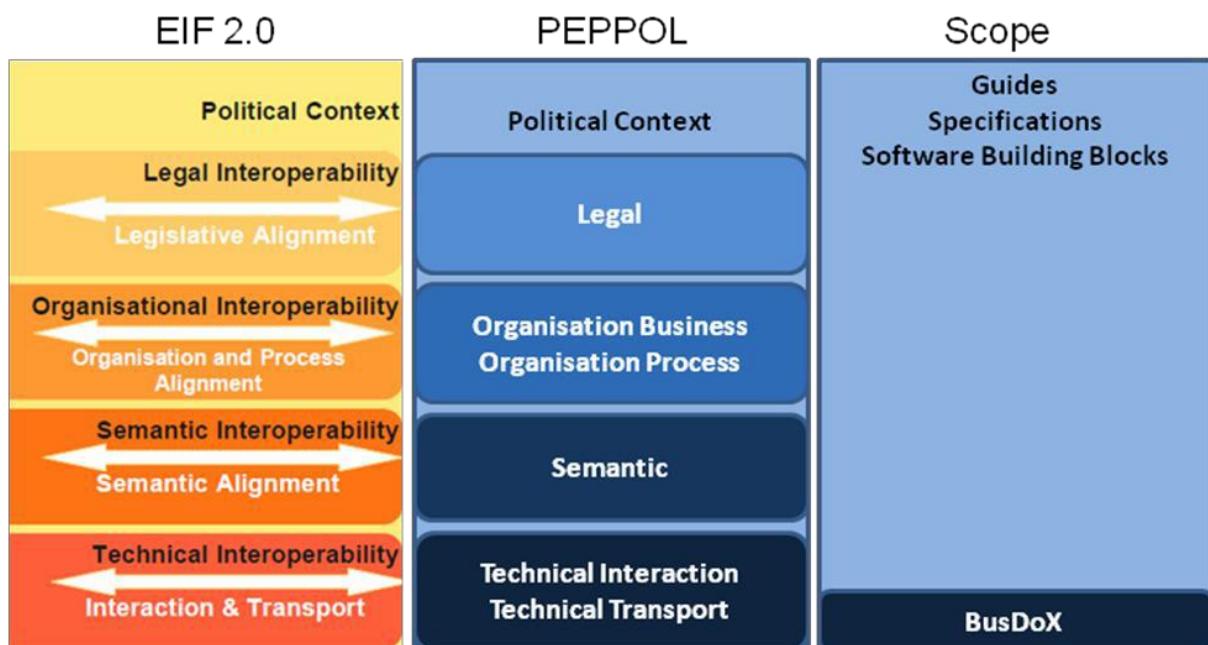
Fig. 1: Peppol Interoperability

1.3 Goals and non-goals

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

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

1.4 Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.4.1 Notational conventions

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>
```

1.4.2 Normative references

[XML-DSIG]   "XML Signature Syntax and Processing Version 1.1",
             https://www.w3.org/TR/xmldsig-core1/

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

[WSA-1.0]   "Web Services Addressing 1.0 – Core",
             http://www.w3.org/TR/2005/CR-ws-addrcore-20050817/
             and "Web Services Addressing 1.0 - SOAP Binding",
             http://www.w3.org/TR/wsaddr-soap/

[RFC2119]   "Key words for use in RFCs to Indicate Requirement Levels",
             http://www.ietf.org/rfc/rfc2119.txt

[PFUOI4]   "Peppol Policy for use of Identifiers 4.1.0",
             https://docs.peppol.eu/edelivery/policies/PEPPOL-EDN-Policy-for-use-of-identifiers-4.1.0-2020-03-11.pdf

1.4.3 Non-normative references

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

[REST]   "Architectural Styles and the Design of Network-based Software Architectures",
             https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[BDEN-SML]   "Peppol Service Metadata Locator (SML) 1.2.0",
             https://docs.peppol.eu/edelivery/sml/PEPPOL-EDN-Service-Metadata-Locator-1.2.0-2020-06-25.pdf

1.5 Namespaces

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

Prefix	Namespace URI

ds	http://www.w3.org/2000/09/xmldsig#
ids	http://busdox.org/transport/identifiers/1.0/
smp	http://busdox.org/serviceMetadata/publishing/1.0/
wsa	http://www.w3.org/2005/08/addressing
xs	http://www.w3.org/2001/XMLSchema

2 The Service Discovery Process

68 The interfaces of the Service Metadata Locator (SML) service and the Service Metadata Publisher
69 (SMP) service cover both sender-side lookup and metadata management performed by SMPs.
70 Business Document Exchange Network (BUSDOX) mandates the following interfaces for these
71 services:
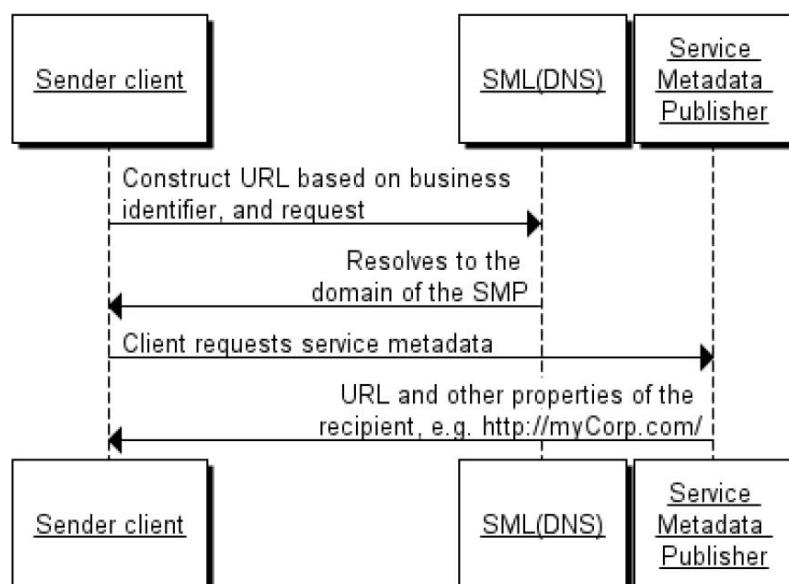
72 • Service Metadata Locator:
73 o DNS-based resolve mechanism to locate individual SMPs
74 o Management interface towards SMPs
75 • Service Metadata Publishers:
76 o Discovery interface towards senders

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

2.1 Discovery flow

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

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

Fig. 2: Endpoint lookup with Service Metadata

90 Note: For optimization reasons, the discovery doesn't have to be performed for every transfer if the
91 necessary information for transfer is already cached from previous transmissions. Though necessary
92 exception handling has to be in place i.e. new lookup has to be performed if the sending shows that
93 information is outdated e.g. old endpoint address.

| 94 | **2.1.1 Discovering services associated with a Participant Identifier** |

95 In addition to the direct lookup of Service Metadata based on participant identifier and document
96 type, a sender may want to discover what document types can be handled by a specific participant
97 identifier. Such discovery is relevant for applications supporting several equivalent business
98 processes. Knowing the capabilities of the recipient is valuable information to a sender application
99 and ultimately to an end user. E.g. the end user may be presented with a choice between a "simple"
100 and a "rich" business process.

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

2.2 Service Metadata Publisher Redirection

106 For each participant identifier, the SML may only point to a single SMP. There are cases however
107 where the owner of a participant identifier may want to use different SMPs for different document
108 types or processes. This is supported by Service Metadata Publisher Redirection.

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

113
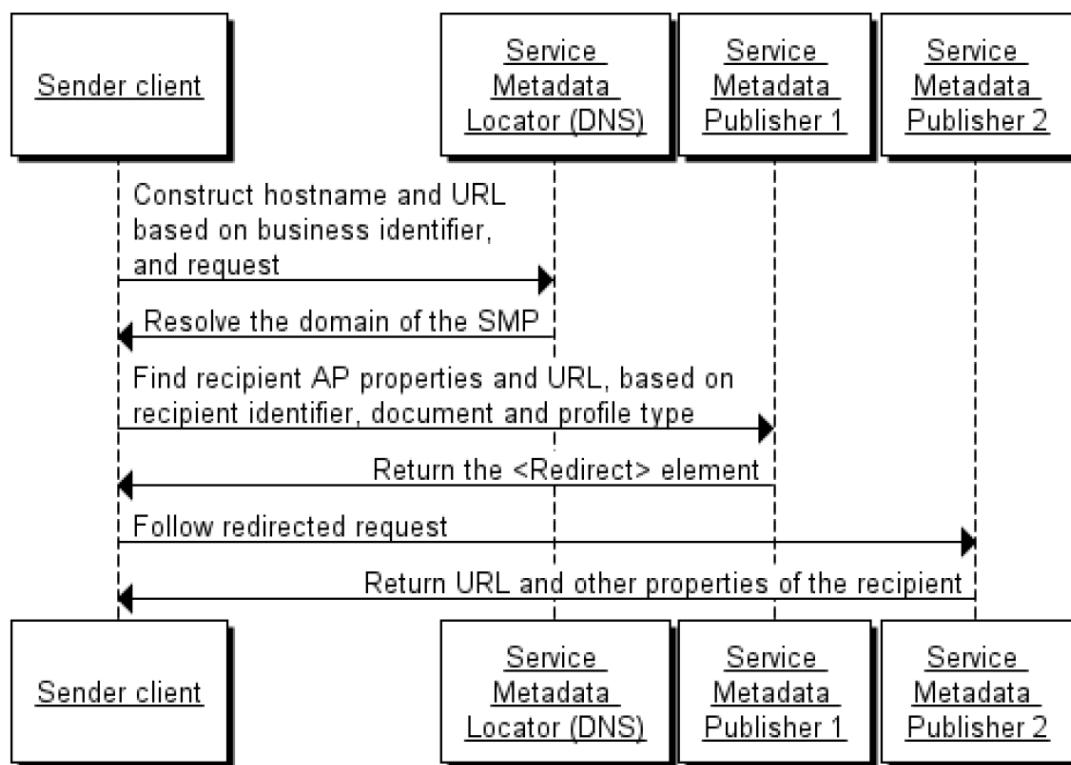
114 <center>Fig. 3: Service Metadata Redirection</center>

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

3 Interface model

119

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

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

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

129   4   Data model

130   This section outlines the data model of the interface. The data model comprises the following main
131   data types:

132   • ServiceGroup
133   • ServiceMetadata / SignedServiceMetadata

134   Supporting data types for these main types are:

135   • ServiceInformation
136   • ServiceEndpointList
137   • ParticipantIdentifier
138   • DocumentIdentifier
139   • Redirect
140   • Process
141   • ProcessList
142   • Endpoint

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

144   4.1   On extension points

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

147   4.1.1   Semantics and use

148   Child elements of the `<smp:Extension>` element are known as "custom extension elements".
149   Extension points may be used for optional extensions of service metadata. This implies:

150   • Extension elements added to a specific Service Metadata resource MUST be ignorable by any
151   client of the transport infrastructure. The ability to parse and adjust client behaviour based
152   on an extension element MUST NOT be a prerequisite for a client to locate a service, or to
153   make a successful request at the referenced service.
154   • A client MAY ignore any extension element added to specific service metadata resource
155   instances.

156   4.2   ServiceGroup

157   The *ServiceGroup* structure represents a set of services associated with a specific participant
158   identifier that is handled by a specific SMP. The *ServiceGroup* structure holds a list of references to
159   *SignedServiceMetadata* resources in the *ServiceList* structure.

160   Pseudo-schema for *ServiceGroup*:

```
<smp:ServiceGroup>
  <ids:ParticipantIdentifier scheme="xs:string">
    xs:string
  </ids:ParticipantIdentifier>
  <smp:ServiceMetadataReferenceCollection>
    <smp:ServiceMetadataReference href="xs:anyURI" />*
  </smp:ServiceMetadataReferenceCollection>
  <smp:Extension>xs:any</smp:Extension>?
</smp:ServiceGroup>
```

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

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 [PFUOI4] for information on this data type.
ServiceMetadataReferenceCollection	This 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 *SignedServiceMetadata* 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.

4.2.1 Non-normative example

Non-normative example of a *ServiceGroup* resource:

```
<?xml version="1.0" encoding="utf-8" ?>
<!--
This sample assumes that the service metadata publisher resides at
"http://serviceMetadata.eu/".
It assumes that the business identifier is "0010:5798000000001".
-->
<ServiceGroup xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
xmlns:ids="http://busdox.org/transport/identifiers/1.0/">
  <ids:ParticipantIdentifier scheme="busdox-actorid-upis">
    0010:5798000000001
  </ids:ParticipantIdentifier>
  <ServiceMetadataReferenceCollection>
    <ServiceMetadataReference href="http://serviceMetadata.eu/busdox-actorid-
upis%3A%3A0010%3A5798000000001/services/busdox-docid-
qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
2%3A%3AInvoice%23%23UBL-2.0"/>
  </ServiceMetadataReferenceCollection>
  <Extension>
    <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
  </Extension>
</ServiceGroup>
```

4.3 ServiceMetadata

This data structure represents Metadata about a specific electronic service. The role of the *ServiceMetadata* structure is to associate a participant identifier with the ability to receive a specific

197 document type over a specific transport. It also describes which business processes a document can
198 participate in, and various operational data such as service activation and expiration times.

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

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

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

210 Pseudo-schema for this data type:

```
211  <smp:ServiceMetadata>
212     [<smp:ServiceInformation /> | <smp:Redirect />]
213  </smp:ServiceMetadata>
```

214 Pseudo-schema for the `ServiceInformation` data type:

```
215  <smp:ServiceInformation>
216    <ids:ParticipantIdentifier scheme="xs:string">xs:string
217    </ids:ParticipantIdentifier>
218    <ids:DocumentIdentifier scheme="xs:string" />
219    <smp:ProcessList>
220      <smp:Process>+
221        <ids:ProcessIdentifier scheme="xs:string" />
222        <smp:ServiceEndpointList>
223          <smp:Endpoint transportProfile="xs:string">+
224            <wsa:EndpointReference />
225            <smp:RequireBusinessLevelSignature>xs:boolean
226            </smp:RequireBusinessLevelSignature>
227            <smp:MinimumAuthenticationLevel>xs:string
228            </smp:MinimumAuthenticationLevel >?
229            <smp:ServiceActivationDate>xs:dateTime
230            </smp:ServiceActivationDate>?
231            <smp:ServiceExpirationDate>xs:dateTime
232            </smp:ServiceExpirationDate>?
233            <smp:Certificate>xs:string</smp:Certificate>
234            <smp:ServiceDescription>xs:string
235            </smp:ServiceDescription>
236            <smp:TechnicalContactUrl>xs:anyURI
237            </smp:TechnicalContactUrl>
238            <smp:TechnicalInformationUrl>xs:anyURI
239            </smp:TechnicalInformationUrl>?
240            <smp:Extension>xs:any</smp:Extension>?
241          </smp:Endpoint>
242        </smp:ServiceEndpointList>
243        <smp:Extension>xs:any</smp:Extension>?
244      </smp:Process>
245    </smp:ProcessList>
246    <smp:Extension>xs:any</smp:Extension>?
247  </smp:ServiceInformation>
```

248 Pseudo-schema for the `Redirect` data type:

```
249  <smp:Redirect href="xs:anyURI">
250    <smp:CertificateUID>xs:string</smp:CertificateUID>
251    <smp:Extension>xs:any</smp:Extension>?
252  <smp:Redirect>
```

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

255 The `href` attribute of the `Redirect` element contains the full address of the destination SMP
256 record that the client is redirected to.

257 For example, assume that an SMP called "SMP1" has the address `http://smp1.eu`, and another
258 SMP called "SMP2" has the address `http://smp2.eu`, and a client requests a resource with the
259 following URL (note that these examples have been percent encoded):

```
260  http://smp1.eu/busdox-actorid-
261  upis%3A%3A0010%3A5798000000001/services/busdox-docid-
262  qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice
263  - 2%3A%3AInvoice%23%23UBL-2.0
```

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

```
267  http://smp2.eu/busdox-actorid-
268  upis%3A%3A0010%3A5798000000001/services/busdox-docid-
269  qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice
270  - 2%3A%3AInvoice%23%23UBL-2.0
```

271 For the list of endpoints under each `Endpoint` element in the `ServiceEndpointList`, each
272 endpoint MUST have different values of the `transportProfile` attribute, i.e. represent bindings
273 to different transports.

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

Field	Description
/ServiceMetadata	Document element
ServiceMetadata/Redirect	The direct child element of `ServiceMetadata` is either the `Redirect` element or the `ServiceInformation` element. The `Redirect` element indicates that a client must follow the URL of the `href` 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 `Redirect`.
ServiceMetadata/ServiceInformation	The direct child element of `ServiceMetadata` is either the `Redirect` element or the

Field	Description
	`ServiceInformation` element. The `ServiceInformation` 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 *ServiceMetadata* resource. See the ParticipantIdentifier section of the 'Policy for use of identifiers' document [PFUOI4] for information on this data type.
ServiceInformation/DocumentIdentifier	Represents the type of document that the recipient is able to handle. The document type is represented by an identifier (identifying the document type) and an identifier scheme, which the format of the identifier itself. See the DocumentTypeIdentifier section of the 'Policy for use of identifiers' document [PFUOI4] 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 Process section of the 'Policy for use of identifiers' document [PFUOI4] for information on the identifier format.
Process/ProcessIdentifier	The identifier of the process. See the 'Policy for use of identifiers' document for a definition of process identifiers [PFUOI4]
Process/ServiceEndpointList	List of one or more endpoints that support this process.
ServiceEndpointList/Endpoint	`Endpoint` represents the technical endpoint and address type of the recipient, as an URL.
Endpoint/EndpointReference	The address of an endpoint, as a WS-Addressing Endpoint Reference (EPR).
Endpoint/@transportProfile	Indicates the type of transport protocol that is being used between access points, e.g. the Peppol AS4 profile (`peppol-transport-as4-v2_0`). A list of

Field	Description
	valid transport protocols is referenced from the 'Policy for use of identifiers' document [PFUOI4].
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 Peppol AS4 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.
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.
Endpoint/ServiceActivationDate	Activation date of the service. Senders MUST ignore services that are not yet activated.
	A missing activation date MUST be interpreted as "valid since forever".
	Format of `ServiceActivationDate` is `xs:dateTime`.
Endpoint/ServiceExpirationDate	Expiration date of the service. Senders MUST ignore services that are expired.
	A missing expiration date MUST be interpreted as "valid until eternity".
	Format of `ServiceExpirationDate` is `xs:dateTime`.
Endpoint/Certificate	Holds the complete signing certificate of the recipient AP, as a PEM (base 64) encoded X509 DER formatted value.
Endpoint/ServiceDescription	A human readable description of the service.
Endpoint/TechnicalContactUrl	Represents a link to human readable contact information. This might also be an email address.
Endpoint/TechnicalInformationUrl	A URL to human readable documentation of the service format. This could for example be a web site

Field	Description
	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.

4.3.1   Non-normative example

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

4.4   SignedServiceMetadata

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

```
<smp:SignedServiceMetadata>
  <smp:ServiceMetadata />
  <ds:Signature />
</smp:SignedServiceMetadata>
```

- `ServiceMetadata` is the `ServiceMetadata` element covered by the signature.
- `Signature` represents an enveloped XML signature over the `SignedServiceMetadata` element.

4.4.1   Non-normative example

Non-normative example of a *SignedServiceMetadata* resource.

```
<?xml version="1.0" encoding="utf-8" ?>
<!--
This sample assumes that the service metadata publisher resides at
"http://serviceMetadata.eu/".
It assumes that the business identifier is "0010:5798000000001".
-->
<SignedServiceMetadata xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
xmlns:ids="http://busdox.org/transport/identifiers/1.0/">
  <ServiceMetadata xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wsswssecurity-
utility-1.0.xsd">
    <ServiceInformation>
      <ids:ParticipantIdentifier scheme="busdox-actorid-
upis">0010:5798000000001</ids:ParticipantIdentifier>
      <ids:DocumentIdentifier scheme="busdox-docid-
qns">urn:oasis:names:specification:ubl:schema:xsd:Invoice-2::Invoice##UBL-
2.02</ids:DocumentIdentifier>
      <ProcessList>
        <Process>
```

```
310        <ids:ProcessIdentifier scheme="cenbii-procid-
311 ubl">BII04</ids:ProcessIdentifier>
312        <ServiceEndpointList>
313          <Endpoint transportProfile="busdox-transport-start">
314            <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
315              <Address>http://busdox.org/sampleService/</Address>
316            </EndpointReference>
317            <RequireBusinessLevelSignature>false</RequireBusinessLevelSignature>
318            <MinimumAuthenticationLevel>2</MinimumAuthenticationLevel>
319            <ServiceActivationDate>2009-05-01T09:00:00</ServiceActivationDate>
320            <ServiceExpirationDate>2016-05-01T09:00:00</ServiceExpirationDate>
321            <Certificate>TlRMTVNTUAABAAAAt7IY4gk....</Certificate>
322            <ServiceDescription>invoice service</ServiceDescription>
323            <TechnicalContactUrl>https://example.com</TechnicalContactUrl>
324        <TechnicalInformationUrl>http://example.com/info</TechnicalInformationUrl>
325          </Endpoint>
326        </ServiceEndpointList>
327      </Process>
328      <Process>
329        <ids:ProcessIdentifier scheme="cenbii-procid-
330 ubl">BII07</ids:ProcessIdentifier>
331        <ServiceEndpointList>
332          <Endpoint transportProfile="busdox-transport-start">
333            <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
334              <Address>http://busdox.org/sampleService/</Address>
335            </EndpointReference>
336            <RequireBusinessLevelSignature>true</RequireBusinessLevelSignature>
337            <MinimumAuthenticationLevel>1</MinimumAuthenticationLevel>
338            <ServiceActivationDate>2009-05-01T09:00:00</ServiceActivationDate>
339            <ServiceExpirationDate>2016-05-01T09:00:00</ServiceExpirationDate>
340            <Certificate>TlRMTVNTUAABAAAAt7IY4gk....</Certificate>
341            <ServiceDescription>invoice service</ServiceDescription>
342            <TechnicalContactUrl>https://example.com</TechnicalContactUrl>
343        <TechnicalInformationUrl>http://example.com/info</TechnicalInformationUrl>
344            <Extension>
345              <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
346            </Extension>
347          </Endpoint>
348        </ServiceEndpointList>
349        <Extension>
350          <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
351        </Extension>
352      </Process>
353    </ProcessList>
354    <Extension>
355      <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
356    </Extension>
357  </ServiceInformation>
358 </ServiceMetadata>
359 <!-- Message signature, details omitted for brevity -->
360 <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"/>
361 </SignedServiceMetadata>
```

4.4.2   Redirect, non-normative example

```
363 <?xml version="1.0" encoding="utf-8" ?>
364 <!--
365 This sample assumes that the user contacts a service metadata publisher that
```

```
366  resides at "http://serviceMetadata.eu/",
367  but is redirected to a service metadata publisher that resides at
368  "http://serviceMetadata2.eu/".
369  -->
370  <SignedServiceMetadata xmlns="http://busdox.org/serviceMetadata/publishing/1.0/">
371    <ServiceMetadata xmlns="http://busdox.org/serviceMetadata/publishing/1.0/">
372      <Redirect xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
373  href="http://serviceMetadata2.eu/busdox-
374  actoridupis%3A%3A0010%3A5798000000001/services/busdox-
375  docidqns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AInvoice-
376  2%3A%3AInvoice%23%23UBL-2.0">
377        <CertificateUID>PID:9208-2001-3-279815395</CertificateUID>
378        <Extension>
379          <ex:Test xmlns:ex="http://test.eu">Test</ex:Test>
380        </Extension>
381      </Redirect>
382    </ServiceMetadata>
383    <!-- Message signature, details omitted for brevity -->
384    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"/>
385  </SignedServiceMetadata>
```

5   Service Metadata Publishing REST binding

387   This section describes the REST binding of the SMP interface.

5.1   The use of HTTP

389   A service implementing the REST binding MUST set the HTTP `Content-Type` header, and give it a
390   value of `text/xml` or `application/xml`. A service implementing the REST profile MUST NOT
391   use TLS (Transport Layer Security) or SSL (Secure Sockets Layer). An instance of the BUSDOX
392   infrastructure MAY set restrictions on what ports are allowed.

393   An implementation of the SMP might choose to manage resources through the HTTP POST, PUT and
394   DELETE verbs. It is however up to each implementation to choose how to manage records, and use of
395   HTTP POST, PUT and DELETE is not mandated or regulated by this specification.

396   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.

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

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

5.2   The use of XML and encoding

401   XML document returned by HTTP GET MUST be UTF-8 encoded. They MUST contain a document type
402   declaration starting with `<?xml` which includes the `encoding` attribute set to `UTF-8`. Please
403   observe that the content of the encoding attribute is not case sensitive. Version 1.0 of XML is used.

5.3   Resources and identifiers

405   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.

SignedServiceMet adata	/{identifier scheme}::{id}/ser vices/{docType} See section below for {docType} format	GET	<SignedSe rviceMeta data>	200; 500; 404	Holds all of the metadata about a Service, or a redirection URL to another Service Metadata Publisher holding this information.

406   **Fig. 4: Table of resources and identifiers**

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

5.3.1   On the use of percent encoding

410   When any types of BUSDOX identifiers are used in URLs, each section between slashes MUST be
411   percent encoded according to [RFC3986] individually, i.e. section by section.

412   For example, this implies that for an URL in the form of `/{identifier`
413   `scheme}::{id}/services/{docType}` the slash literals MUST NOT be URL encoded.

5.3.2   Using identifiers in the REST Resource URLs

415   This section describes specifically how participant and document identifiers are used to reference
416   *ServiceGroup* and *SignedServiceMetadata* REST resources. For a general definition on how to
417   represent participant and document identifiers in URLs, see [PFUOI4].

418   For the URL referencing a *ServiceGroup* resource, the `{identifier scheme}::{id}` part
419   follows the participant identifier format described in the "ParticipantIdentifier" section of the 'Policy
420   for use of identifiers' document [PFUOI4].

421   The following URL format is used:

422   `/{identifier scheme}::{id}`

423   In the reference to the *SignedServiceMetadata* or *Redirect* resources
424   (`/{id}/services/{docType}`), the `{docType}` part consists of `{document type`
425   `identifier scheme}::{document type identifier}`. For information on the format
426   of `{document type identifier}`, see the DocumentIdentifier section of the 'Policy for use of
427   identifiers' document [PFUOI4].

5.3.3   Non-normative identifier example

429   We assume an SMP can be accessed at the URL `http://serviceMetadata.eu`.

430   A business with the participant identifier `0010:5798000000001` would have the following
431   identifier for the *ServiceGroup* resource:

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

433   After percent encoding:

434   `http://serviceMetadata.eu/busdox-actorid-upis%3a%3a0010%3a5798000000001`

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

437   • Identifier format type: `busdox-docid-qns`
438   • Root namespace:
439   `urn:oasis:names:specification:ubl:schema:xsd:Order-2`

440 • Document element local name: `Order`
441 • Subtype identifier: `UBL-2.0` (since several versions of the Order schema may use the same
442 namespace + document element name)

443 The document type identifier will then be:

```
444   busdox-docid-qns::urn:oasis:names:specification:ubl:schema:xsd:Order-
445   2::Order##UBL-2.0
```

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

```
448   busdox-docid-
449   qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AOrder-
450   2%3A%3AOrder%23%23UBL-2.0
```

451 The entire URL reference to a *SignedServiceMetadata* or *Redirect* resource thus has the form

```
452   {URL to server}/{identifier scheme}::{id}/services/{document identifier
453   type}::{rootNamespace}::{documentElementLocalName}[##{Subtype identifier}]
```

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

```
455   http://serviceMetadata.eu/busdox-actorid-
456   upis%3a%3a0010%3a5798000000001/services/busdox-docid-
457   qns%3A%3Aurn%3Aoasis%3Anames%3Aspecification%3Aubl%3Aschema%3Axsd%3AOrder-
458   2%3A%3AOrder%23%23UBL-2.0
```

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

5.3.4 Implementation considerations

462 When a client is redirected to an SMP using the DNS-based SML scheme described in [BDEN-SML],
463 the HTTP `Host` header will be set to a value originating from the CNAME alias set in the SML
464 (http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.23). Implementations should be
465 prepared to accept requests with this "host" header value.

5.4 Referencing the SMP REST binding

467 For referencing the SMP REST binding, for example from SML records, the following identifier should
468 be used for the version 1.0 of the SMP REST binding:

```
469   http://busdox.org/serviceMetadata/publishing/1.0/
```

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

5.5 Security

472 At the transport level, the service MUST NOT be secured.

5.5.1 Message signature

474 The message returned by the service is signed by the Service Metadata Publisher with XML-Signature
475 according to [XML-DSIG].

476 The signature MUST be an enveloped XML signature represented via a `ds:Signature` element
477 embedded in the `SignedServiceMetadata` element. The `ds:Signature` element MUST be
478 constructed according to the following rules:

479	•	The <Reference> MUST use exactly one <Transform> being:
480		`http://www.w3.org/2000/09/xmldsig#enveloped-signature`
481	•	The <ds:KeyInfo> element MUST contain an <ds:X509Data> element with an
482		<ds:X509Certificate> sub-element containing the signer's X.509 certificate as PEM (base 64)
483		encoded X509 DER value.
484	•	The canonicalization algorithm MUST be
485		`http://www.w3.org/TR/2001/REC-xml-c14n-20010315`
486	•	The SignatureMethod MUST be
487		`http://www.w3.org/2000/09/xmldsig#rsa-sha1`
488	•	The DigestMethod MUST be
489		`http://www.w3.org/2000/09/xmldsig#sha1`

490   **5.5.2   Verifying the signature**

491   When verifying the signature, the consumer has access to the full certificate as a PEM (base 64)
492   encoded X509 DER value within the `dsSignature` element. The consumer may verify the signature
493   by

494   a)   extracting the certificate from the `ds:X509Data` element,
495   b)   verify that it has been issued by the trusted root,
496   c)   perform a validation of the signature, and
497   d)   perform the required certificate validation steps (which might include checking
498   expiration/activation dates and revocation lists).

499   **5.5.3   Verifying the signature of the destination SMP**

500   For the redirect scheme, the unique identifier of the destination SMP signing certificate is stored at
501   the redirecting SMP. In addition to the regular signature validation performed by the client of the
502   destination SMP resources, the client SHOULD also validate that the identifier of the destination SMP
503   signing certificate corresponds to the unique identifier which the redirecting SMP claims belongs to
504   the destination SMP.

6 Appendix A: Schema for the REST interface

6.1 peppol-smp-types-v1.xsd (non-normative)

This section defines the XML Schema for all the resources of the REST interface. The normative version of the XML Schema is packaged together with this specification.

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="ServiceMetadataPublishing"
targetNamespace="http://busdox.org/serviceMetadata/publishing/1.0/"
elementFormDefault="qualified"
xmlns="http://busdox.org/serviceMetadata/publishing/1.0/"
xmlns:ids="http://busdox.org/transport/identifiers/1.0/"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <xs:import schemaLocation="xmldsig-core-schema.xsd"
namespace="http://www.w3.org/2000/09/xmldsig#"/>
  <xs:import schemaLocation="oasis-200401-wss-wssecurity-utility-1.0.xsd"
namespace="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"/>
  <xs:import schemaLocation="ws-addr.xsd"
namespace="http://www.w3.org/2005/08/addressing"/>
  <xs:import schemaLocation="peppol-identifiers-v1.xsd"
namespace="http://busdox.org/transport/identifiers/1.0/"/>

  <xs:element name="ServiceGroup" type="ServiceGroupType"/>
  <xs:element name="ServiceMetadata" type="ServiceMetadataType"/>
  <xs:element name="SignedServiceMetadata" type="SignedServiceMetadataType"/>

  <xs:complexType name="SignedServiceMetadataType">
    <xs:sequence>
      <xs:element ref="ServiceMetadata"/>
      <xs:element ref="ds:Signature"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ServiceMetadataType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="ServiceInformation" type="ServiceInformationType"/>
        <xs:element name="Redirect" type="RedirectType"/>
      </xs:choice>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ServiceInformationType">
    <xs:sequence>
      <xs:element ref="ids:ParticipantIdentifier"/>
      <xs:element ref="ids:DocumentIdentifier"/>
      <xs:element name="ProcessList" type="ProcessListType"/>
      <xs:element name="Extension" type="ExtensionType" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="ProcessListType">
    <xs:sequence>
      <xs:element name="Process" type="ProcessType" maxOccurs="unbounded"/>
```

```
560        </xs:sequence>
561      </xs:complexType>
562
563      <xs:complexType name="ProcessType">
564        <xs:sequence>
565          <xs:element ref="ids:ProcessIdentifier"/>
566          <xs:element name="ServiceEndpointList" type="ServiceEndpointList"/>
567          <xs:element name="Extension" type="ExtensionType" minOccurs="0"/>
568        </xs:sequence>
569      </xs:complexType>
570
571      <xs:complexType name="ServiceEndpointList">
572        <xs:sequence>
573          <xs:element name="Endpoint" type="EndpointType" maxOccurs="unbounded"/>
574        </xs:sequence>
575      </xs:complexType>
576
577      <xs:complexType name="EndpointType">
578        <xs:sequence>
579          <xs:element ref="wsa:EndpointReference"/>
580          <xs:element name="RequireBusinessLevelSignature" type="xs:boolean"/>
581          <xs:element name="MinimumAuthenticationLevel" type="xs:string"
582   minOccurs="0"/>
583          <xs:element name="ServiceActivationDate" type="xs:dateTime" minOccurs="0"/>
584          <xs:element name="ServiceExpirationDate" type="xs:dateTime" minOccurs="0"/>
585          <xs:element name="Certificate" type="xs:string"/>
586          <xs:element name="ServiceDescription" type="xs:string"/>
587          <xs:element name="TechnicalContactUrl" type="xs:anyURI"/>
588          <xs:element name="TechnicalInformationUrl" type="xs:anyURI" minOccurs="0"/>
589          <xs:element name="Extension" type="ExtensionType" minOccurs="0"/>
590        </xs:sequence>
591        <xs:attribute name="transportProfile" type="xs:string"/>
592      </xs:complexType>
593
594      <xs:complexType name="ServiceGroupType">
595        <xs:sequence>
596          <xs:element ref="ids:ParticipantIdentifier"/>
597          <xs:element name="ServiceMetadataReferenceCollection"
598   type="ServiceMetadataReferenceCollectionType"/>
599          <xs:element name="Extension" type="ExtensionType" minOccurs="0"/>
600        </xs:sequence>
601      </xs:complexType>
602
603      <xs:complexType name="ServiceMetadataReferenceCollectionType">
604        <xs:sequence>
605          <xs:element name="ServiceMetadataReference"
606   type="ServiceMetadataReferenceType" minOccurs="0" maxOccurs="unbounded"/>
607        </xs:sequence>
608      </xs:complexType>
609
610      <xs:complexType name="ServiceMetadataReferenceType">
611        <xs:attribute name="href" type="xs:anyURI"/>
612      </xs:complexType>
613
614      <xs:complexType name="RedirectType">
615        <xs:sequence>
616          <xs:element name="CertificateUID" type="xs:string"/>
```

```
617              <xs:element name="Extension" type="ExtensionType" minOccurs="0"/>
618          </xs:sequence>
619          <xs:attribute name="href" type="xs:anyURI"/>
620      </xs:complexType>
621
622      <xs:complexType name="ExtensionType">
623          <xs:sequence>
624              <xs:any/>
625          </xs:sequence>
626      </xs:complexType>
627  </xs:schema>
```