



---

## 2 Electronic PostMark (EPM) Profile of the 3 OASIS Digital Signature Service

4 **2<sup>nd</sup> Committee Draft, 11 September, 2006 (WD-10)**

5 **Document identifier:**  
6 oasis-dss-1.0-profiles-epm-spec-cd-r2

7 **Location:**  
8 <http://docs.oasis-open.org/dss/v1.0/>

9 **Editors:**  
10 Ed Shallow, *Universal Postal Union*  
11 Paul Donohoe, *Universal Postal Union*

12 **Contributors:**  
13 Trevor Perrin, *individual*  
14 Nick Pope, *individual*  
15 Juan Carlos Cruellas, *individual*

16 **Abstract:**  
17 This draft defines a profile of the OASIS DSS protocol for the purpose of creating and verifying signatures  
18 and timestamps which support the extended features of the Universal Postal Union's Electronic  
19 PostMarking service.

20 **Status:**  
21 This is a **Public review Draft** produced by the OASIS Digital Signature Service Technical Committee.  
22 Comments may be submitted to the TC by any person by clicking on "Send A Comment" on the TC home  
23 page at:  
24 [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=dss](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dss)

25 For information on whether any patents have been disclosed that may be essential to implementing this  
26 specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights  
27 section of the Digital Signature Service TC web page at <http://www.oasis-open.org/committees/dss/ipr.php>.

---

## Table of Contents

29	
30	1 Introduction .....
31	1.1 Notation .....
32	1.2 Namespaces.....
33	2 Profile Features .....
34	2.1 Identifier .....
35	2.2 Scope.....
36	2.3 Relationship To Other Profiles.....
37	2.4 Signature Objects .....
38	2.5 Transport Binding .....
39	2.6 Security Binding.....
40	2.6.1 Security Requirements.....
41	2.7 Common Elements .....
42	2.7.1.1 Element <InputDocuments>.....
43	2.7.1.2 Element <DocumentWithSignature> .....
44	2.7.2 Element <PostMarkedReceipt> and the PostMarkedReceipt Signature .....
45	2.7.3 Output Element <TransactionKey>.....
46	2.7.4 Input Element <OrganizationID>.....
47	3 Profile of Signing Protocol .....
48	3.1 Element <SignRequest>.....
49	3.1.1 Constraints on Element <OptionallInputs> .....
50	3.1.1.1 Element SignatureType.....
51	3.1.1.2 Element <KeySelector> .....
52	3.1.1.3 Element <AddTimestamp> .....
53	3.1.1.4 Optional Input <Properties> .....
54	3.1.1.5 Optional Input <SignedReferences> .....
55	3.1.1.6 Optional Input <IncludeObject>.....
56	3.1.1.7 Optional Input <SignaturePlacement> .....
57	3.1.2 EPM-specific <OptionallInputs>.....
58	3.1.2.1 Element <DocumentContainsTemplate> .....
59	3.1.2.2 Element <TransactionKey> .....
60	3.1.2.3 Element <ClaimedIdentity> .....
61	3.1.2.4 Element <OrganizationID> .....
62	3.1.3 <OptionallInputs> Processing Directives .....
63	3.1.3.1 Element <IssuePostMarkedReceipt> .....
64	3.1.3.2 Element <StoreNonRepudiationEvidence> .....
65	3.1.3.3 Element <ReturnSignatureInfo> .....
66	3.1.3.4 Element <ReturnX509Info> .....
67	3.2 Element <SignResponse>.....
68	3.2.1 Element <Result>.....
69	3.2.2 Element <SignatureObject> .....
70	3.2.3 EPM-specific <OptionalOutputs>.....
71	3.2.3.1 Element <TransactionKey> .....
72	3.2.3.2 Element <PostMarkedReceipt>.....
73	3.2.3.3 Element <SignatureInfo>.....
74	3.2.3.4 Element <X509Info> .....

75	4 Profile of Verifying Protocol .....	17
76	4.1 Element <VerifyRequest>.....	17
77	4.1.1 Constraints on Element <OptionalInputs> .....	17
78	4.1.1.1 Element <InputDocuments>.....	17
79	4.1.1.2 SignatureObject .....	17
80	4.1.1.3 Element <AdditionalKeyInfo> .....	17
81	4.1.1.4 Element <ReturnProcessingDetails> .....	17
82	4.1.1.5 Element <ReturnSigningTime> .....	17
83	4.1.1.6 Element <ReturnSignerIdentity> .....	17
84	4.1.1.7 Element <VerifyManifests> .....	17
85	4.1.1.8 Element <ReturnUpdatedSignature> .....	17
86	4.1.1.9 Element <ReturnTransformedDocument>.....	17
87	4.1.2 EPM-specific <OptionalInputs>.....	18
88	4.1.2.1 Element <OrganizationID> .....	18
89	4.1.2.2 Element <IgnoreManifests> .....	18
90	4.1.2.3 Element <SignatureSelector> .....	18
91	4.1.2.4 Element <IssuePostMarkedReceipt> .....	19
92	4.1.3 <OptionalInputs> Processing Flags .....	20
93	4.1.3.1 Element <StoreNonRepudiationEvidence> .....	20
94	4.1.3.2 Element <ReturnSignatureInfo> .....	20
95	4.1.3.3 Element <ReturnX509Info>.....	20
96	4.2 Element <VerifyResponse> .....	20
97	4.2.1 Element <Result>.....	20
98	4.2.2 Element <SignatureObject>.....	20
99	4.2.3 Element <OptionalOutputs> .....	21
100	4.2.3.1 Element <DocumentWithSignature> .....	21
101	4.2.4 Element <EPM-specific OptionalOutputs> .....	21
102	4.2.4.1 Element <TransactionKey> .....	21
103	4.2.4.2 Element <PostMarkedReceipt>.....	21
104	4.2.4.3 Element <SignatureInfo>.....	21
105	4.2.4.4 Element <X509Info> .....	21
106	5 Signing Template Examples.....	22
107	6 PostMarkedReceipt Examples .....	26
108	7 Element cross-reference Table .....	32
109	8 References .....	38
110	8.1 Normative .....	38
111	Appendix A. Revision History.....	39
112	Appendix B. Notices.....	40
113		

---

## 114 1 Introduction

115 The Electronic Postmarking service is a Universal Postal Union (UPU) endorsed standard aimed at providing  
116 generalized signature creation, signature verification, timestamping, receipting, and evidence logging services for  
117 use by and across Postal Administrations and their target customers.

118 Although the total scope and functional coverage of the EPM's service offering are outside the immediate scope  
119 of the DSS initiative, the UPU wishes to offer its client base a DSS-compliant subset of the EPM for clients who  
120 wish to maintain OASIS compliance in the core areas of signature and timestamp, creation and verification. This  
121 profile can be used directly as the basis for implementing interoperable systems.

122 Implementers wishing to take their implementations of this profile to market are asked to do so through any of the  
123 Postal Administrations participating or wishing to participate in the global EPM initiative. Any client is free to  
124 develop service request calls which adhere to this interface and receive their corresponding service responses.

### 125 1.1 Notation

126 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT",  
127 "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC  
128 2119 [[RFC 2119](#)]. These keywords are capitalized when used to unambiguously specify requirements over  
129 protocol features and behavior that affect the interoperability and security of implementations. When these words  
130 are not capitalized, they are meant in their natural-language sense.

131 This specification uses the following typographical conventions in text: <ns:Element>, Attribute, Datatype,  
132 OtherCode.

### 133 1.2 Namespaces

134 The structures described in this specification are contained in the schema file [[EPM](#)]. All schema listings in the  
135 current document are excerpts from that schema file. In the case of a disagreement between the schema file and  
136 this document, the schema file takes precedence.

137 This schema is associated with the following XML namespace:

138 <http://www.docs.oasis-open.org/dss/2004/04/oasis-dss-1.0-profiles-EPM-wd-09#>

139 If a future version of this specification is needed, it will use a different namespace.

140 Conventional XML namespace prefixes are used in this document:

- 141 ➤ The prefix dss : (or no prefix) stands for the DSS core namespace [[Core-XSD](#)].
- 142 ➤ The prefix dsig: stands for the W3C XML Signature namespace [[XMLSig](#)].
- 143 ➤ The prefix xs : stands for the W3C XML Schema namespace [[Schema1](#)].
- 144 ➤ The prefix saml: stands for the OASIS SAML Schema namespace [[SAMLCore1.1](#)].
- 145 ➤ The prefix epm: stands for the EPM Schema namespace [[EPM](#)].
- 146 ➤ The prefix xades: stands for ETSI XML Advanced Electronic Signatures (XAdES) document [[XAdES](#)].

147 Applications MAY use different namespace prefixes, and MAY use whatever namespace defaulting/scoping  
148 conventions they desire, as long as they are compliant with the Namespaces in XML specification [[XML-ns](#)].

---

149 **2 Profile Features**

150 **2.1 Identifier**

151 urn:oasis:names:tc:dss:1.0:profiles:epm

152 **2.2 Scope**

153 This document profiles the DSS signing and verifying protocols defined in [DSSCore] and provides an OASIS  
154 DSS-compliant interface to selected services of the EPM. One of the primary intents of the EPM Profile is to  
155 simplify request and response processing for client callers by constraining [DSSCore] in several ways.

156 The EPM profile supports the creation and verification of both CMS/PKCS7 and [XMLSig] signature types.

157 Additional services within the EPM are supported through the extensibility mechanisms provided by the optional  
158 inputs and outputs of the [DSSCore]. This includes:

- 159 ➤ Easy to use EPM “Signing Templates”
- 160 ➤ PostMarked receipts
- 161 ➤ Same document signatures is the default and preferred mechanism for handling signature creation and  
162 verification
- 163 ➤ Certificate validation data
  - 164 ▪ Revocation references
  - 165 ▪ Certificate references
  - 166 ▪ Online Certificate Status Protocol (OCSP) responses
- 167 ➤ Timestamping from a CA-independent TimeStamp Authority

168 This profile constrains the <InputDocuments> element in that it may contain only one <Document> element.  
169 Additionally, this profile assumes that documents and their signatures are to be contained in the same XML  
170 document wherever possible. This considerably simplifies the amount of splicing that a client must perform when  
171 dealing with the protocol. This will be evident in the Input and Output constraints explained herein.

172 **2.3 Relationship To Other Profiles**

173 The profile in this document is based directly on the [DSSCore].

174 **2.4 Signature Objects**

175 This profile supports the creation and verification of XMLSig and CMS/PKCS7 signatures and timestamps as  
176 defined in [DSSCore].

177 **2.5 Transport Binding**

178 This profile is transported using either an XML-based HTTP payload POSTed to an implementation of this profile,  
179 or via a SOAP Transport Binding as defined in the OASIS EPM Profile Web Service Description language  
180 (WSDL).

181 **2.6 Security Binding**

182 **2.6.1 Security Requirements**

183 The TLS X.509 Server Authentication security binding as described in section 6.2.1 in [DSSCore] must be used.

184 Although outside the scope of this protocol, clients are expected to authenticate to an implementation of this  
185 specification. At a minimum HTTP Basic Authorization should be used to authenticate. Implementations are  
186 expected to validate the user and password contained in the HTTP header.

## 187 **2.7 Common Elements**

188 This section describes elements used and referenced within both the Sign and Verify protocols as either Input or  
189 Output elements.

### 190 **2.7.1.1 Element <InputDocuments>**

191 The EPM profile also constrains the <InputDocuments> element such that the EPM server presently accepts  
192 **only one** <Document> or <DocumentHash> element (i.e. equivalent of `maxOccurs="1"`). This may change in  
193 a subsequent version of the EPM profile. Multiple <Reference> elements are supported. Users wishing to  
194 create signatures with multiple <Reference> elements should use EPM signing templates. See section 3.1.2.1  
195 for details.

196 When the <Document> element is passed in by the user of this profile, it is assumed that it contains only the  
197 content to be signed or the signed document to be verified. When users wish to use the EPM's "signing template"  
198 mechanism, they must also pass in a <DocumentContainsTemplate> element directive. Please also refer to  
199 section 3.1.2.1 below.

200 On the Verify protocol the processing differs slightly from the core in that input documents containing "same  
201 document" signatures can be passed in on a Verify request via the `Document` element. This avoids having to use  
202 the `SignatureObject` in conjunction with the `SignaturePtr` choice of that element. Since only one  
203 occurrence of the `Document` element is allowed, `SignaturePtr` is not required.

204 The `dss:TransformedData` choice is not supported by this profile.

205 The <Document> element is also used to pass in a signature to be timestamped when the <SignatureType>  
206 specifies a timestamp type. The `MimeType` should specify `application/pkcs7-signature` when passing in  
207 a signature to be RFC 3161 timestamped.

### 208 **2.7.1.2 Element <DocumentWithSignature>**

209 This element is used in conjunction with the `SignatureObject` element for returning signed and verified  
210 documents. For Sign operations, this element will be initialized and returned for **[XMLSig]** based signatures when  
211 the signature is an enveloped or detached one. Additionally if the caller is using EPM signing templates and has  
212 passed in a signing template (See section 3.1.2.1) by specifying the `DocumentContainsTemplate` element,  
213 then this output element will contain the signed document.

214 For verify operations this output element is only initialized for **[XMLSig]** based signatures when the  
215 <IssuePostMarkedReceipt> option is specified with a `Location` attribute specified as `embedded`. In this  
216 case the signed and verified document is returned along with the embedded `PostMarkedReceipt` in this output  
217 element. See also <IssuePostMarkedReceipt>.

```
218 <xs:element name="DocumentWithSignature">
219   <xs:complexType>
220     <xs:sequence>
221       <xs:element ref="dss:Document" />
222     </xs:sequence>
223   </xs:complexType>
224 </xs:element>
```

## 225 **2.7.2 Element <PostMarkedReceipt> and the PostMarkedReceipt Signature**

226 A `PostMarkedReceipt` is a signature attesting to the validity of either the signature just created (Sign protocol) or  
227 the signature just verified (Verify protocol). It requires an additional profile element not part of **[DSSCore]** and that  
228 is the <PostMarkedReceipt> element. This element describes the EPM's receipt structure, which works in  
229 conjunction with the standard <TstInfo> element of **[DSSCore]**. A `PostMarkedReceipt` signature is returned

230 whenever the optional input <IssuePostMarkedReceipt> is included in either the Sign or Verify request. The  
231 PostMarkedReceipt is a superset of the DSS <Timestamp> element and carries specific meaning within the  
232 specific context of EPM service provisioning. Semantics as follows:

233 ➤ **Sign**

234 When a PostMarkedReceipt signature is issued as a result of a Sign operation, the EPM is attesting to  
235 the origin of the signature and the validity of the certificate used to create it.

236 ➤ **Verify**

237 Correspondingly, when the EPM issues a PostMarkedReceipt as a result of a Verify operation which  
238 requested an <IssuePostMarkedReceipt>, the EPM is attesting to the validity of both the verified  
239 signature as well as the validity (i.e. revocation status) of the public verification certificate contained  
240 therein.

241 See section 6 for a detailed example of a standalone PostMarkedReceipt signature returned after successful  
242 verification. The example illustrates a detached receipt signature representing the PostMark covering a signed  
243 and verified document. Additionally, all evidence surrounding this event is logged in the EPM's non-repudiation  
244 database when the StoreNonRepudiationInfo Optional Input is specified.

245 The EPM supports the issuance of conventional timestamps, both embedded and standalone. The EPM-specific  
246 notion of a PostMarked receipt applies in both the embedded and standalone scenarios. Both are valid within the  
247 Sign protocol.

248 All receipts are tied to a specific EPM operational transaction as specified by the enclosed <TransactionKey>  
249 element.

250 The <PostMarkedReceipt> element is similar to the <dss:Timestamp> when applied to XMLSig-based  
251 signatures.

252 PostMarkedReceipt signatures returned in XMLSig signatures scenarios, are exactly three (3)  
253 <dsig:Reference>'s which make up the signature associated with the PostMarkedReceipt. They are as  
254 follows:

- 255 ➤ <dsig:Reference> whose URI attribute references a <dsig:Object> containing the <TstInfo>  
256 ➤ <dsig:Reference> whose URI attribute references a <dsig:Object> containing the  
257 <epm:PostMarkedReceipt>  
258 ➤ <dsig:Reference> whose URI attribute references a <dsig:Object> containing the  
259 <dsig:SignatureValue> of the signature being PostMarked (Sign) or Verified and PostMarked  
260 (Verify)

261 EPM-produced <PostMarkedReceipt>'s, always bind the receipt to the signature just created or verified.

262 Please refer to the EPM documentation for additional policy and usage guidelines.

---

```
264 <xs:element ref="epm:PostMarkedReceipt"
265
266 <!-- imported from the EPM schema -->
267 <xs:element name="PostMarkedReceipt" type="epm:PostMarkedReceiptType">
268
269 <xs:complexType name="PostMarkedReceiptType">
270   <xs:sequence>
271     <xs:choice>
272       <xs:element name="PKCS7SignedReceipt" type="epm:PKCS7SignedReceiptType"/>
273       <xs:element name="XMLSignedReceipt" type="epm:QualifiedDataType"/>
274     </xs:choice>
275   </xs:sequence>
276 </xs:complexType>
277
278 <xs:complexType name="PKCS7SignedReceiptType">
279   <xs:sequence>
```

---

```

280             <xs:element name="Receipt" type="epm:ReceiptType" />
281             <xs:element name="ReceiptSignature" type="epm:QualifiedDataType"
282 nillable="true"/>
283         </xs:sequence>
284     </xs:complexType>
285     <xs:complexType name="ReceiptType">
286         <xs:sequence>
287             <xs:element name="TransactionKey" type="epm:TransactionKeyType" />
288             <xs:element name="Requester" type="xs:string" />
289             <xs:element name="Operation" type="xs:string" />
290             <xs:element name="TSAX509SubjectName" type="xs:string" />
291             <xs:element name="TimeStampValue" type="xs:string" />
292             <xs:element name="RevocationStatusQualifier" type="xs:string" />
293             <xs:element name="TimeStampToken" type="epm:QualifiedDataType" nillable="true"
294 minOccurs="0" maxOccurs="1"/>
295             <xs:element name="MessageImprint" type="xs:base64Binary" nillable="true" />
296             <xs:element name="PostMarkImage" type="epm:QualifiedDataType" nillable="true" />
297             <xs:element name="ReceiptMetadata" type="epm:ReceiptMetadataType"
298 nillable="true" minOccurs="0" maxOccurs="unbounded" />
299         </xs:sequence>
300     </xs:complexType>
301
302     <xs:complexType name="ReceiptMetadataType">
303         <xs:sequence>
304             <xs:element name="Name" type="xs:string" />
305             <xs:choice>
306                 <xs:element name="Value" type="xs:string" />
307                 <xs:element name="EncodedValue" type="epm:QualifiedDataType" />
308             <xs:choice>
309             </xs:sequence>
310         </xs:complexType>
311

```

312

313 **Note 1:** The ReceiptSignature child element of the PostMarkedReceipt is only used when processing  
 314 CMS/PKCS7 signatures where the receipt is standalone. It is simply used to protect the integrity of this  
 315 standalone XML structure which contains an encapsulated CMS/PKCS7 <TimeStampToken>.

316 **Note 2:** The binary <TimeStampToken> element above can be omitted for [XMLSig]-based  
 317 <SignatureType>'s since the PostMarkedReceipt is itself a signature which covers the <TstInfo>  
 318 structure. EPM implementations using TimeStamp Authorities (TSAs), are however free to initialize this element  
 319 with an RFC3161 Timestamp Token if they wish. The example in section 6 does not initialize the  
 320 <TimeStampToken> element.

## 321 2.7.3 Output Element <TransactionKey>

322 This complexType is a compound key made up of 3 elements uniquely identifying each event in the an EPM  
 323 Lifecycle. The EPM generates and returns a new and unique <TransactionKey> with all response operations.  
 324 The <Locator> element is used to identify the particular EPM instance when multiple EPM instances are  
 325 involved, as is the case with cross-border transactions. Please refer to EPM documentation for usage guidelines.

```

326 <xs:element ref="epm:TransactionKey"
327
328 <!-- imported from the EPM schema -->
329 <xs:element name=" TransactionKey" type="epm:TransactionKeyType" >
330 <xs:complexType name="TransactionKeyType" >
331     <xs:sequence>
332         <xs:element name="Locator" type="epm:LocatorType" />
333         <xs:element name="Key" type=" xs:string" />
334         <xs:element name="Sequence" type="xs:positiveInteger" />
335     </xs:sequence>
336 </xs:complexType>

```

```
337 <xs:complexType name="LocatorType">
338   <xs:sequence>
339     <xs:element name="CountryCode" type="xs:string"/>
340     <xs:element name="Version" type="xs:string"/>
341     <xs:element name="ServiceProvider" type="xs:string" nillable="true"/>
342     <xs:element name="Environment" type="xs:string" nillable="true"/>
343   </xs:sequence>
344 </xs:complexType>
```

345

## 346 **2.7.4 Input Element <OrganizationID>**

347 This element is used when the requester's organization name cannot or should not be derived from a public  
348 certificate (as would be the case with X509 Mutual Authentication). In those circumstances, this element should  
349 be initialized to the requester's organizational name as an `xs:string`. This value will be validated at  
350 authentication time by the EPM service against registration-time information.

```
351 <xs:element name="OrganizationID" type="xs:string" nillable="true"/>
```

352

---

353    **3 Profile of Signing Protocol**

354    **3.1 Element <SignRequest>**

355    **3.1.1 Constraints on Element <OptionalInputs>**

356    Details on the constraints and semantics which exist with respect to the optional inputs as described in  
357    **[DSSCore]** follow in this section. All <OptionalInputs> not explicitly mentioned in this section are supported  
358    as defined in **[DSSCore]**.

359    EPM-specific <OptionalInputs> are described below in the section entitled EPM-specific <OptionalInputs>.

360    **3.1.1.1 Element SignatureType**

361    The <SignatureType> element MUST be included in the EPM profile's SignRequest.

362    The following <SignatureType> URNs are supported:

363       ➤ Signature creation URNs:

- 364            ▪ urn:ietf:rfc:3275 (i.e. an XML Digital Signature)
- 365            ▪ urn:ietf:rfc:3369 (i.e. a CMS/PKCS7 binary Signature)

366       ➤ Timestamp creation URNs:

- 367            ▪ oasis:names:tc:dss:1.0:core:schema:XMLTimeStampToken
- 368            ▪ urn:ietf:rfc:3161 (i.e. a CMS/PKCS7 timestamp token)

369    The first 2 URNs instruct the EPM to create a signature. The last 2 URNs instruct the EPM to create a timestamp.  
370    The context and processing rules within which the EPM creates signatures is different than the context within  
371    which the EPM creates timestamps. These differences will be highlighted below as they apply to each optional  
372    input and output, as constrained by the <SignatureType> chosen above. If no restriction is mentioned below,  
373    one may assume that the optional input is valid for timestamp <SignatureType>'s as well.

374    **3.1.1.2 Element <KeySelector>**

375    The <KeySelector> optional input must be supported by EPM implementations of this profile, but is not  
376    required when calling the EPM service as a client user (i.e. is optional). If the EPM cannot derive the key to use  
377    for signing from the underlying authentication being used, or if the X509SubjectName is not readily available, the  
378    <KeySelector> can be used. When using EPM signing templates, users may initialize the <KeyInfo> element  
379    in the signing template with a valid <X509SubjectName> in the <KeyName> child element of <KeyInfo>. The  
380    EPM will utilize the specified certificate/key as defined. See Example 1 in section 5 for an example of signing  
381    templates.

382    **Note:** This optional input does not apply when users are requesting a timestamp <SignatureType>. EPM  
383    implementations are, by definition, TimeStamp Authorities and will use TSA-specific signing keys expressly for  
384    that purpose.

385    **3.1.1.3 Element <AddTimestamp>**

386    The EPM supports this <OptionalInputs> element when used in the Sign protocol. Processing is the same as  
387    that described in **[DSSCore]**.

388    See also section 3.1.3.1 <IssuePostMarkedReceipt> which delivers similar but different functionality than the  
389    <AddTimestamp> and results in the creation of an additional standalone <PostMarkedReceipt> structure  
390    which is a superset of a basic dss:XMLTimestamp. Both optional inputs are supported on a Sign operation and  
391    serve different purposes.

392 The <AddTimestamp> is also supported on the Verify operation, and will update the signature being verified.  
393 See also <IssuePostMarkedReceipt> which returns a timestamped receipt structure and has different  
394 semantic meaning. Please refer to section 4 covering the Verify.

395 **Note:** Content timestamps, created before signature generation, are currently not supported in the EPM profile  
396 (e.g. a timestamp created before signature generation and referenced as a signed property or attribute). They  
397 may however be added in a subsequent release of the EPM profile.

### 398 **3.1.1.4 Optional Input <Properties>**

399 This optional input element is not supported by this profile. If specialized signed or unsigned properties are  
400 required users are encouraged to use the EPM's "signing templates" facility.

### 401 **3.1.1.5 Optional Input <SignedReferences>**

402 This optional input element is not supported by this profile. If greater control over Reference element creation is  
403 required, users are encouraged to use the EPM's "signing templates" facility.

### 404 **3.1.1.6 Optional Input <IncludeObject>**

405 This optional input is supported by the EPM profile to produce enveloping signatures with the following  
406 constraints. The WhichDocument attribute is not required and hence not supported. The  
407 hasObjectTagsAndAttributesSet is also not supported. This profile supports only one <IncludeObject>  
408 in the request. The default and only behavior supported in this profile for this optional input is exactly as is  
409 described in the createReference attribute as specified in **[DSSCore]**.

### 410 **3.1.1.7 Optional Input <SignaturePlacement>**

411 This element is supported with the following constraints. Only the last attribute of this element from **[DSSCore]** is  
412 supported. That is, the CreateEnvelopedSignature attribute is the only attribute supported. Default signature  
413 placement in this profile for enveloped signatures is to place the ds:Signature as the last child of the input  
414 Document's root.

### 415 **3.1.2 EPM-specific <OptionalInputs>**

416 The following additional elements are specific to the EPM profile. Their specific usage and constraints are  
417 documented below.

#### 418 **3.1.2.1 Element <DocumentContainsTemplate>**

419 The <DocumentContainsTemplate> optional input element is a directive which tells the implementation that  
420 the <Document> element passed in on the request is a "signing template". It is used when users elect to utilize  
421 the EPM's "signing template" mechanism. EPM-supported signing templates contain not only the data to be  
422 signed, but also the format and directives of the signature to be created, expressed as valid **[XMLSig]** elements.  
423 In this fashion more elaborate signatures involving transforms, signed and unsigned properties, manifests, and  
424 multiple <Reference> elements can be supported without complex XML request constructs. **[XMLSig]** elements  
425 such as <SignatureValue>, <DigestValue>, and <X509Certificate> are populated by the EPM  
426 service based on the template provided. The user leaves these crypto-specific element tags empty, and the EPM  
427 service will automatically include the generated content and return the signed document in the  
428 <DocumentWithSignature> element of the <SignResponse>. See Example 1 in section 5 for an example of  
429 signing templates. More details are available in the EPM Systems Integrator's Guide and other EPM  
430 documentation available through the UPU.

431 **Note:** When using templates, all **[XMLSig]** References must resolve within the single <Document> element  
432 passed in on the request. This compromise was chosen to provide maximum flexibility and ease-of-use.

433 `<xs:element name="DocumentContainsTemplate" />`

434

435 **3.1.2.2 Element <TransactionKey>**

436 Please refer to the description in section 2.7.3 entitled [Element <TransactionKey>](#)

437 **3.1.2.3 Element <ClaimedIdentity>**

438 This optional complexType is an extension of the standard OASIS DSS <ClaimedIdentity> element. This  
439 extension to ClaimedIdentity utilizes the OASIS <SupportingInfo> to define EPM-specific additions  
440 required to support the authentication and assertion of the requester's identity. The default authentication  
441 mechanism of an EPM implementation is external to the EPM profile and is supported by the conventions used in  
442 that underlying binding. In this fashion EPM implementations are free to authenticate users using standard  
443 approaches like HTTP Basic Authentication (i.e. Authorization: Basic in the HTTP header), or may decide  
444 to use stronger techniques involving Digest Authentication, encrypted cookies, one-time password schemes, two-  
445 factor tokens, or any of several other authentications schemes they chose. However there are situations where  
446 the underlying binding may not support the representation or the transport of the desired token type. For this  
447 reason, the EPM profile allows the chosen token type to be passed as "Authentication Information" as an  
448 attestation of, in support of, in addition to, or instead of the underlying authentication scheme and its assertion of  
449 identity. As such, it is not used solely as additional authentication information, but rather could be used as an  
450 adjunct to the authentication mechanism itself. This scheme-specific authentication support is carried in the  
451 abstract <AlternateIdentity> type.

452 The <RequesterSignature> element is optional and is used in support of "Proof-of-Possession" or "Proof-of-  
453 Delivery" in the EPM's non-repudiation context. This element and its use-cases are further defined in the EPM  
454 Service Description documentation available through the Universal Postal Union.

```
455
456 <xs:element name="ClaimedIdentity">
457   <xs:complexType>
458     <xs:sequence>
459       <xs:element name="Name" type="saml:NameIdentifierType" />
460       <xs:element ref="epm:SupportingInfo">
461     </xs:sequence>
462   </xs:complexType>
463 </xs:element>
464
465 <!-- imported from the EPM schema -->
466 <xs:element name="SupportingInfo" type="epm:SupportingInfoType" />
467 <xs:complexType name="SupportingInfoType">
468   <xs:element name="BasicAuth" type="epm:BasicAuthType" nillable="true" />
469   <xs:element name="RequesterSignature" type="epm:QualifiedDataType" nillable="true" />
470   <xs:element name="AlternateIdentity" type="epm:AlternateIdentityType" nillable="true" />
471 </xs:complexType>
472
473 <xs:complexType name="epm:QualifiedDataType">
474   <xs:simpleContent>
475     <xs:extension base="xs:base64Binary">
476       <xs:attribute name="MimeType" type="xs:string" />
477     </xs:extension>
478   </xs:simpleContent>
479 </xs:complexType>
480
481 <xs:complexType name="BasicAuthType">
482   <xs:sequence>
483     <xs:element name="UserID" type="xs:string" />
484     <xs:element name="Password" type="xs:string" nillable="true" />
485   </xs:sequence>
486 </xs:complexType>
487
488 <xs:complexType name="AlternateIdentityType" abstract="true" >
489   <xs:sequence>
490     <xs:element name="IdentityToken" type="xs:anyType" />
491   </xs:sequence>
```

492 </xs:complexType>

493

#### 494 **3.1.2.4 Element <OrganizationID>**

495 Please refer to the description in section 2.7.4 entitled [Input Element <OrganizationID>](#)

### 496 **3.1.3 <OptionalInputs> Processing Directives**

497 This section describes the <OptionalInputs> that are simple processing directives for the EPM. Each  
498 directive or “flag” directs the EPM to perform specific functions and/or return specific response information. More  
499 detail on each processing option can be found in the EPM documentation.

#### 500 **3.1.3.1 Element <IssuePostMarkedReceipt>**

501 Including this empty directive element instructs the EPM to return a signed receipt attesting to the origin of the  
502 signature as well as the validity of the certificate used in the signature process. Inclusion of this element results in  
503 the return of either a standalone <PostMarkedReceipt> signature containing its signed  
504 <PostMarkedReceipt> and <TimeStampToken> or one embedded in the signature being created. This  
505 element contains a Location attribute instructing the EPM how to return the <PostMarkedReceipt>. This  
506 processing differs based on the <SignatureType> and the value of the Location attribute.

- 507 ➤ For a Location attribute value of standalone regardless of the <SignatureType>, processing is as  
508 follows:
  - 509 ▪ The <PostMarkedReceipt> XML element will be returned as a standalone optional output  
510 structure as defined in section 2.7.2. Standalone <PostMarkedReceipt>’s are self-contained  
511 and contain a timestamp signature which binds the receipt to the signature value of the signature  
512 being created as part of this Sign operation.
- 513 ➤ For a Location attribute value of embedded and a <SignatureType> value of urn:ietf:rfc:3275 (i.e.  
514 XMLSig), processing is as follows:
  - 515 ▪ The EPM will first create an [XMLSig] based “detached” signature covering the input document.  
516 The input document’s contents will be outside the produced signature and referenced by it. The  
517 EPM will then add a <PostMarkedReceipt> detached signature structure covering the  
518 <SignatureValue> of the first signature just created. The resulting signed and PostMarked  
519 document will be returned in the <DocumentWithSignature> element.
- 520 ➤ A Location attribute value of embedded with a <SignatureType> value of urn:ietf:rfc:3369 (i.e.  
521 CMS/PKCS7) is not supported.
  - 522 ▪ A signature timestamp (i.e. an RFC 3161 timestamp token) however can be embedded in a  
523 CMS/PKCS7 signature by using the <AddTimestamp> optional input described in section  
524 3.1.1.3. This timestamp bears the Issuer name of the Post’s TimeStamp Authority.

525

526 Please refer to section 6 for a detailed example of a <PostMarkedReceipt> signature.

```
527 <xs:element ref="epm:IssuePostMarkedReceipt">
528
529 <!-- imported from the EPM schema -->
530 <xs:element name="IssuePostMarkedReceipt" type="epm:IssuePostMarkedReceiptType">
531 <xs:complexType name="IssuePostMarkedReceiptType">
532   <xs:sequence>
533     <xs:element name="Location" type="epm:ValidLocation" minOccurs="0"/>
534     <xs:element name="PostMarkImage" type="epm:PostMarkImageType" minOccurs="0"/>
535   </xs:sequence>
536 </xs:complexType>
537
538 <xs:complexType name="PostMarkImageType">
```

```
539     <xs:simpleContent>
540         <xs:extension base="xs:boolean">
541             <xs:attribute name="Format" type="xs:string" default="JPG"/>
542             <xs:attribute name="Size" type="epm:ValidImageSize" default="Small"/>
543         </xs:extension>
544     </xs:simpleContent>
545 </xs:complexType>
546
```

### 547 **3.1.3.2 Element <StoreNonRepudiationEvidence>**

548 Including this empty directive element instructs the EPM to store evidence of the operation being performed. This  
549 evidentiary information can be subsequently retrieved and used to support challenges as to its authenticity.

```
550 <xs:element name="StoreNonRepudiationEvidence" />
```

551

### 552 **3.1.3.3 Element <ReturnSignatureInfo>**

553 Including this empty directive element instructs the EPM to return additional response information relating to the  
554 signing operation. This directive element results in the return of a <SignatureInfo> structure in the  
555 <OptionalOutputs>.

```
556 <xs:element name="ReturnSignatureInfo" />
```

557

### 558 **3.1.3.4 Element <ReturnX509Info>**

559 Including this empty directive element instructs the EPM to return additional response information relating to the  
560 certificate used for the signing. This directive element results in the return of an <X509Info> structure in the  
561 <OptionalOutputs>.

```
562 <xs:element name="ReturnX509Info" />
```

563

## 564 **3.2 Element <SignResponse>**

### 565 **3.2.1 Element <Result>**

566 This profile defines an additional <ResultMajor> code as follows:

567 urn:oasis:names:tc:dss:1.0:resultmajor:Warning

568 All EPM result codes are always accompanied by a <ResultMessage> element.

### 569 **3.2.2 Element <SignatureObject>**

570 If successful, the server will return a <ds:Signature> with the signature properties as defined in [DSSCore].  
571 Location of the generated signature will be determined based on signature type and envelope type. All  
572 CMS/PKCS7 signatures will be returned in the <SignatureObject> element. [XMLSig] based enveloping  
573 signatures will also be returned in the <SignatureObject> element. Enveloped and detached signatures are  
574 returned in the <DocumentWithSignature> element described below.

### 575 **3.2.3 EPM-specific <OptionalOutputs>**

576 The following additional elements are specific to the EPM profile. Their specific usage and constraints are  
577 documented below.

578   **3.2.3.1 Element <TransactionKey>**

579 Please refer to section 3.1.2.2 for a description of how the <TransactionKey> element which is used on both  
580 input and on output as an identification and retrieval mechanism and to support subsequent reference to specific  
581 service calls.

582   **3.2.3.2 Element <PostMarkedReceipt>**

583 If the <IssuePostMarkedReceipt> optional input is included, then this optional output will be returned. It is  
584 essentially a standalone receipt represented as an enveloping signature. See section 2.7.2 for details.

585   **3.2.3.3 Element <SignatureInfo>**

586 This structure can be returned on both the Sign and Verify operations and is returned whenever the  
587 <ReturnSignatureInfo> element is included in the request. Together with the <x509Info> element these  
588 elements provide more detail on the signature just created or being verified. The element <SignedContent> is  
589 used in conjunction with the Verify operation and allows users to extract the signed content from a CMS/PKCS7  
590 signature thus alleviating the need to parse the ASN.1 structure. See <VerifyResponse> below. The  
591 <SignedContent> element on a CMS/PKCS7 Sign response is empty as the user has just passed this content  
592 in to be signed, however the <SignedContent> element on an XMLDSig Sign request will contain the  
593 transformed content as it existed prior to digest calculation for the user's reference.

594 Detailed explanation of the other <SignatureInfo> elements can be found in the EPM System Integrator's  
595 Guide.

```
596 <xs:element ref="epm:SignatureInfo"
597
598 <!-- imported from the EPM schema -->
599 <xs:element name="SignatureInfo" type="epm:SignatureInfoType">
600 <xs:complexType name="SignatureInfoType">
601   <xs:sequence>
602     <xs:element name="SignedContent" type="epm:QualifiedDataType" nillable="true"/>
603     <xs:element name="ContentHash" type="xs:string" nillable="true"/>
604     <xs:element name="ContentHashAlgo" type="xs:string" nillable="true"/>
605     <xs:element name="ContentEncryptAlgo" type="xs:string" nillable="true"/>
606     <xs:element name="SigningTime" type="xs:string" nillable="true"/>
607     <xs:element name="PKCS1" type="epm:QualifiedDataType" nillable="true"/>
608   </xs:sequence>
609 </xs:complexType>
```

610 **Note:** This optional output does not apply when users have requested a timestamp <SignatureType>.

611   **3.2.3.4 Element <X509Info>**

612 This structure is returned whenever the <ReturnX509Info> element is included in the request. The  
613 <x509ValidationData> element is the default implementation of the abstract base type called  
614 GenericValidationDataType and contains a signed OCSP Validation Data element returned by the EPM.  
615 This element attests to the validity of the certificate used in the signing operation. It will contain signed content as  
616 per RFC 2560 and also described in RFC 3126 as returned by a standard OCSP Responder. If an EPM DSS  
617 implementation is not using an OCSP responder, then sufficient certificate chain and revocation references must  
618 be included here possibly as an alternate implementation of the abstarc base type. Additionally, many  
619 jurisdictions (e.g. the EU) require that this validation info be signed by the Certified Service Provider (CSP). This  
620 is not an issue when using an RFC 2560-compliant OCSP Responder. Definitions of the other elements are  
621 standard certificate fields and descriptions are also available in the EPM Systems Integrator's Guide.

```
622 <xs:element ref="epm:X509Info"
623
624 <!-- imported from the EPM schema -->
625 <xs:element name="X509Info" type="epm:X509InfoType">
626 <xs:complexType name="X509InfoType">
627   <xs:sequence>
```

```

628      <xs:element name="X509Subject" type="xs:string"/>
629      <xs:element name="X509Issuer" type="xs:string" nillable="true"/>
630      <xs:element name="X509Serial" type="xs:string" nillable="true"/>
631      <xs:element name="X509StatusSource" type="xs:string"/>
632      <xs:element name="X509ValidFrom" type="xs:string"/>
633      <xs:element name="X509ValidTo" type="xs:string"/>
634      <xs:element name="X509Certificate" type="xs:string" nillable="true"/>
635      <xs:element name="X509RevocationReason" type="xs:string" nillable="true"/>
636      <xs:element name="X509RevocationReasonString" type="xs:string" nillable="true"/>
637      <xs:element name="X509RevocationTime" type="xs:string" nillable="true"/>
638      <xs:element name="X509ValidationData" type="epm:X509ValidationDataType"
639      nillable="true"/>
640      </xs:sequence>
641  </xs:complexType>
642
643  <xs:complexType name="GenericValidationDataType" abstract="true">
644    <xs:sequence>
645      <xs:element name="GenericValidationData" type="xs:anyType" />
646    </xs:sequence>
647  </xs:complexType>
648
649  <xs:complexType name="X509ValidationDataType">
650    <xs:complexContent>
651      <xs:extension base="epm:GenericValidationDataType">
652        <xs:sequence>
653          <xs:element name="X509ValidationData"
654          type="epm:QualifiedDataType"/>
655        </xs:sequence>
656      </xs:extension>
657    </xs:complexContent>
658  </xs:complexType>
659
```

660  
661  
662

**Note:** This optional output does not apply when users have requested a timestamp <SignatureType>.

---

663 **4 Profile of Verifying Protocol**

664 **4.1 Element <VerifyRequest>**

665 **4.1.1 Constraints on Element <OptionalInputs>**

666 **4.1.1.1 Element <InputDocuments>**

667 Must be initialized when users wish to verify [XMLSig] based enveloped or detached signatures which contain the  
668 signed content. Presently constrained to one <Document> occurrence. This single <Document> must contain  
669 signature(s) to be verified as well as all signed content referenced by the signature(s) as part of a "same  
670 document" signature.

671 **4.1.1.2 SignatureObject**

672 Since "same document" signatures are common place and InputDocuments can be used as an input element  
673 on a Verify, the SignatureObject input element is not required on the Verify operation by this profile.

674 **4.1.1.3 Element <AdditionalKeyInfo>**

675 This optional input element is not required by the EPM.

676 **4.1.1.4 Element <ReturnProcessingDetails>**

677 This optional input element is not supported by the EPM.

678 **4.1.1.5 Element <ReturnSigningTime>**

679 This optional input element is not required by EPM implementations as this information is returned to the caller in  
680 the <SignatureInfo> element which can be optionally requested by including the <ReturnSignatureInfo>  
681 optional input.

682 **4.1.1.6 Element <ReturnSignerIdentity>**

683 This optional input element is not required by the EPM as this information is returned in the <X509Info>  
684 element which can be optionally requested by including the <ReturnX509Info> optional input.

685 **4.1.1.7 Element <VerifyManifests>**

686 This optional input element is not supported by the EPM. By default, the EPM verifies Manifest references and  
687 returns results in the same fashion as normal References. The EPM has a separate and related optional input  
688 which allows callers to suppress Manifest verification. See below.

689 **4.1.1.8 Element <ReturnUpdatedSignature>**

690 This optional input is not supported by the EPM. The only signature update supported is when the caller specifies  
691 <AddTimestamp>. This produces an embedded timestamp similar to the one produced as part of the Sign  
692 protocol and described in section 3.1.1.3 entitled Element <AddTimestamp>. The RFC3161 compliant timestamp  
693 token is included in the signature as an unauthenticated attribute of the verified signature. This conventionally  
694 timestamped CMS/PKCS7 signature, now updated, will be returned in the <SignatureObject>.

695 **4.1.1.9 Element <ReturnTransformedDocument>**

696 This optional input element is not supported by the EPM.

697 **4.1.2 EPM-specific <OptionalInputs>**

698 **4.1.2.1 Element <OrganizationID>**

699 See section 3.1.2.4 for a detailed explanation of this elements usage.

700 **4.1.2.2 Element <IgnoreManifests>**

701 This EPM-specific optional input allows callers to specify that Manifests be ignored during verification processing.

702 `<xs:element name="IgnoreManifests" />`

703 **4.1.2.3 Element <SignatureSelector>**

704 This optional SignatureSelector element qualifies the XMLDSIG signature(s) to be verified by the EPM. This  
705 element may also serve useful if the user is unsure of exactly what has been verified, and wishes to control the  
706 verification process more explicitly.

707 If the user wishes to Verify a particular signature or signatures, they have two choices has to how they may  
708 specify the dsig:Signature nodes to be verified. Each choice is a sub-element of the  
709 SignatureSelectorType below.

710 The **First** method allows users to specify any ancestor (parent) node of the signature(s) to be verified and are  
711 specified by including these names as `NodeName` element(s). The value is expressed as a string. A namespace  
712 URI qualifier may precede the actual signature `NodeName` value.

713 .

714 `<xs:element name="SignatureSelector" type="epm:SignatureSelectorType" />`  
715 `<xs:complexType name="SignatureSelectorType">`  
716 `<xs:sequence>`  
717 `<xs:choice>`  
718 `<xs:element name="NodeName" type="xs:string" minOccurs="1"`  
719 `maxOccurs="unbounded"/>`  
720 `<xs:element name="XPathSelector" type="epm:XPathSelectorType" />`  
721 `</xs:choice>`  
722 `</xs:sequence>`  
723 `</xs:complexType>`  
724  
725 `<xs:complexType name="XPathSelectorType">`  
726 `<xs:sequence>`  
727 `<xs:element name="XPath" type="xs:string" />`  
728 `<xs:element name="NameSpace" type="xs:string" nillable="true" />`  
729 `<xs:element name="Qualifier" type="xs:string" nillable="true" />`  
730 `</xs:sequence>`  
731 `</xs:complexType>`

732

733 **EXAMPLE** The user would specify string values of `lgl:Party1` and/or `lgl:Party2` to explicitly instruct the  
734 EPM what to Verify. By default the EPM will search for signature nodes specified as `<dsig:Signature>`, which  
735 appear as descendants of the document root.

736 `<lgl:Party1>`  
737 `<dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">`  
738 `...`  
739 `</dsig:Signature>`  
740 `</lgl:Party1>`  
741 `<lgl:Party2>`  
742 `<dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">`  
743 `...`  
744 `</dsig:Signature>`  
745 `</lgl:Party2>`

746

747 The **Second** method involves specifying an XPath expression which when evaluated will return the target  
748 <dsig:Signature> nodes to be verified. The actual Xpath expression is included in  
749 the XPath element and any required namespace and qualifier can be specified in the  
750 NameSpace and Qualifier elements.

751 *EXAMPLE Using an XPath expression to select the target <dsig:Signature> nodes*

```
752 <lgl:Document xmlns:lgl="http://www.lgl.org/SomeService"  
753     xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">  
754     <lgl:Signatures>  
755         <dsig:Signature>1st</dsig:Signature>  
756         ...  
757         <dsig:Signature>2nd</dsig:Signature>  
758         ...  
759         <dsig:Signature>3rd</dsig:Signature>  
760         ...  
761     </lgl:Signatures>  
762 </lgl:Document>
```

763 In the example above a value of //lgl:Signatures//dsig:Signature[position=2] would select only  
764 the second signature to be verified.

765 A value of //lgl:Signatures//dsig:Signature in the XPath element would cause all signatures to be  
766 verified.

767 In both examples a value of http://www.lgl.org/SomeService and lgl should be specified for the  
768 NameSpace and Qualifier elements respectively in order to allow the XPath string expression to evaluate.  
769

#### 770 **4.1.2.4 Element <IssuePostMarkedReceipt>**

771 This optional input instructs the EPM to issue a PostMarkedReceipt signature as attestation of successful  
772 verification of the incoming signature(s). A <PostMarkedReceipt> signature will not be returned if the incoming  
773 signature(s) do not verify successfully or the revocation status of the public verification certificate is not zero.

774 When specifying this element on a Verify operation, the EPM will use a <SignatureSelector> element if it is  
775 present. The <PostMarkedReceipt> will cover the signature(s) that have been verified.

776 Processing differs based on the <SignatureType> and the value of the Location attribute.

- 777 ➤ For a Location attribute value of standalone regardless of the <SignatureType>, processing is as  
778 follows:

- 779 ▪ The <PostMarkedReceipt> XML element will be returned as a standalone optional output  
780 structure as defined in section 2.7.2. Standalone <PostMarkedReceipt>'s are self-contained  
781 and contain a timestamp signature which binds the receipt to the signature value of the signature  
782 being verified as part of this Verify operation.

- 783 ➤ For a Location attribute value of embedded and a <SignatureType> value of urn:ietf:rfc:3275 (i.e.  
784 XMLSig), the incoming <Document> containing the signature(s) **must** be a **detached** XMLSig based  
785 signature. Processing is as follows:

786 The incoming signed document will contain an **[XMLSig]** based “detached” signature covering the  
787 required content within the input document. The input document's signed content will be outside the  
788 signature and referenced by it. The EPM will verify this signature. If the signature(s) verify successfully,  
789 the EPM will then add a <PostMarkedReceipt> detached signature structure covering the  
790 <SignatureValue>'s of the signature(s) just verified.

- 791 ▪ The resulting PostMarked document will be returned in the <DocumentWithSignature>  
792 element and will include the <PostMarkedReceipt> attesting to its validity.

- 793 ➤ A Location attribute value of embedded with a <SignatureType> value of urn:ietf:rfc:3369 (i.e.

794 CMS/PKCS7) is not supported.

795     ▪ A signature timestamp (i.e. an RFC 3161 timestamp token) however can be embedded in a  
796       CMS/PKCS7 signature by using the <AddTimestamp> optional input described in section  
797       3.1.1.3. This timestamp bears the Issuer name of the Post's TimeStamp Authority.

798 Please refer to section 6 for a detailed example of a <PostMarkedReceipt> signature.

```
799 <xs:element ref="epm:IssuePostMarkedReceipt">  
800  
801 <!-- imported from the EPM schema -->  
802 <xs:complexType name="IssuePostMarkedReceiptType">  
803   <xs:sequence>  
804     <xs:element name="Location" type="epm:ValidLocation" minOccurs="0"/>  
805     <xs:element name="PostMarkImage" type="epm:PostMarkImageType" minOccurs="0"/>  
806   </xs:sequence>  
807 </xs:complexType>  
808  
809 <xs:complexType name="PostMarkImageType">  
810   <xs:simpleContent>  
811     <xs:extension base="xs:boolean">  
812       <xs:attribute name="Format" type="xs:string" default="JPG"/>  
813       <xs:attribute name="Size" type="epm:ValidImageSize" default="Small"/>  
814     </xs:extension>  
815   </xs:simpleContent>  
816 </xs:complexType>  
817
```

818

### 819 **4.1.3 <OptionalInputs> Processing Flags**

820 This section describes the <OptionalInputs> that are simple processing directives for the EPM. Each flag  
821 directs the EPM to perform specific functions and/or return specific response information. More detail on each  
822 processing option can be found in the EPM documentation.

#### 823 **4.1.3.1 Element <StoreNonRepudiationEvidence>**

824 See section 0 for a detailed explanation of this elements usage.

#### 825 **4.1.3.2 Element <ReturnSignatureInfo>**

826 See section 3.1.3.3 or a detailed explanation of this elements usage.

#### 827 **4.1.3.3 Element <ReturnX509Info>**

828 See section 3.1.3.4 for a detailed explanation of this elements usage.

### 829 **4.2 Element <VerifyResponse>**

#### 830 **4.2.1 Element <Result>**

831 This profile defines an additional <ResultMajor> code as follows:

832 urn:oasis:names:tc:dss:1.0:resultmajor:Warning

833 All EPM result codes are always accompanied by a <ResultMessage> element.

#### 834 **4.2.2 Element <SignatureObject>**

835 This element is only returned when the <AddTimestamp> optional input is included. Please refer to section  
836 4.1.1.8 for details.

837 **4.2.3 Element <OptionalOutputs>**

838 **4.2.3.1 Element <DocumentWithSignature>**

839 If the <IssuePostMarkedReceipt> optional input is included and its Location attribute specifies embedded,  
840 then this optional output will be returned. See the scenario described in the 2<sup>nd</sup> bullet within section 4.1.2.4 above  
841 for more details.

842 **4.2.4 Element <EPM-specific OptionalOutputs>**

843 The following additional elements are specific to the EPM profile. Their specific usage and constraints are  
844 documented below.

845 **4.2.4.1 Element <TransactionKey>**

846 Please refer to section 3.1.2.2 for a description of how the <TransactionKey> element is used on both input  
847 and on output as both an identification mechanism and to support the concept of a multi-event LifeCycle.

848 **4.2.4.2 Element <PostMarkedReceipt>**

849 If the <IssuePostMarkedReceipt> optional input is included in the Verify request and its Location attribute  
850 specifies standalone, then this optional output will be returned. It is essentially a standalone receipt signature.  
851 See also section 0 above.

852 **4.2.4.3 Element <SignatureInfo>**

853 See section 0 for a detailed explanation of this element's usage.

854 **4.2.4.4 Element <X509Info>**

855 See section 3.2.3.4 for a detailed explanation of this element's usage.

856

## 5 Signing Template Examples

857

This section reproduces a few illustrative Sign template examples from the EPM Signature generation service. For full details on features and options of the EPM XML Digital Signature signing templates, please consult the UPU EPM System Integrator's Guide.

859

### Example 1:

This first example is a simple enveloped signature template which uses the standard enveloped-signature transform and the illustrated digest method. Note how the `<SignatureValue>` element is simply left empty. The EPM Service will expand all valid empty element tags with appropriate content. This particular example also requests that selected `<X509Data>` elements be completed. This is accomplished by including empty `<X509Certificate>`, `<X509SubjectName>`, and `<X509IssuerSerial>` elements.

```
865 <?xml version="1.0" encoding="UTF-8"?>
866 <DocumentWithTemplate/>
867 <Document>
868   <Data>
869     <SubData1>
870       <SubSubData1MimeType="text/plain">This is the data to be signed.</SubSubData1>
871       <SubSubData2MimeType="text/plain">This is the data to be signed.</SubSubData2>
872       <SubSubData3MimeType="text/plain">This is the data to be signed.</SubSubData3>
873     </SubData1>
874     <SubData2>This is the data to be signed.</SubData2>
875     <SubData3>This is the data to be signed.</SubData3>
876   </Data>
877   <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
878     <SignedInfo>
879       <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
880       <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
881       <Reference URI="">
882         <Transforms>
883           <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
884         </Transforms>
885         <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
886         <DigestValue></DigestValue>
887       </Reference>
888     </SignedInfo>
889     <SignatureValue>
890     </SignatureValue>
891     <KeyInfo>
892       <KeyName>C=CA, S=Ontario, L=Ottawa, O=CPC, OU=eServices, CN=Ed Test, E=ed.shallow@rogers.com</KeyName>
893       <X509Data>
894         <X509Certificate></X509Certificate>
895         <X509SubjectName></X509SubjectName>
896       </X509Data>
```

```

897             <X509IssuerSerial>
898                 </X509IssuerSerial>
899             </X509Data>
900         </KeyInfo>
901     </Signature>
902 </Document>

903

904 Example 2:
905 This example is similar to the first however an Xpointer is used within the <Reference> element's URI attribute. This approach is useful when
906 specific subsets of the document require signing. Again certificate information is added to the produced signature.

907
908 <?xml version="1.0" encoding="UTF-8"?>
909 <DocumentWithTemplate/>
910 <Document>
911     <Data>
912         <SubData1>
913             <SubSubData1MimeType="text/plain">This is the data to be signed.</SubSubData1>
914             <SubSubData2MimeType="text/plain">This is the data to be signed.</SubSubData2>
915             <SubSubData3MimeType="text/plain">This is the data to be signed.</SubSubData3>
916         </SubData1>
917         <SubData2>This is data.</SubData2>
918         <SubData3>This is data.</SubData3>
919     </Data>
920     <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
921         <SignedInfo>
922             <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
923             <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
924             <Reference URI="#xpointer(/Document/Data/SubData1)">
925                 <Transforms>
926                     <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
927                 </Transforms>
928                 <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
929                 <DigestValue></DigestValue>
930             </Reference>
931         </SignedInfo>
932         <SignatureValue>
933             </SignatureValue>
934         <KeyInfo>
935             <X509Data>
936                 <X509Certificate></X509Certificate>
937                 <X509SubjectName></X509SubjectName>
938                 <X509IssuerSerial>
939                     </X509IssuerSerial>
940                 </X509Data>
941             </KeyInfo>

```

```

942      </Signature>
943  </Document>
944
945 Example 3:
946 This is a more complicated example using intersect and subtract XPath Filters. This 3rd example illustrates step 1 in a multi-party contract signing
947 workflow. This template controls the scope of data to be signed by the first party. A similar template would be used by the second party after the
948 first party has signed the document. This second template would simply change the "subtract" value in the transform filter. Again certificate
949 information is added to the produced signature.
950
951 <?xml version="1.0"?>
952 <DocumentWithTemplate/>
953 <Document>
954   <Contract>
955     <TermsMimeType="text/plain">This is the data to be signed by both parties</Terms>
956     <ConditionsMimeType="text/plain">This is the data to be signed by both parties</Conditions>
957     <ObligationsMimeType="text/plain">This is the data to be signed by both parties</Obligations>
958     <Party1>
959       <TermsMimeType="text/plain">This is the data to be signed by party 1</Terms>
960       <ConditionsMimeType="text/plain">This is the data to be signed by party 1</Conditions>
961       <ObligationsMimeType="text/plain">This is the data to be signed by party 1</Obligations>
962     </Party1>
963     <Party2>
964       <TermsMimeType="text/plain">This is the data to be signed by party 2</Terms>
965       <ConditionsMimeType="text/plain">This is the data to be signed by party 2</Conditions>
966       <ObligationsMimeType="text/plain">This is the data to be signed by party 2</Obligations>
967     </Party2>
968   </Contract>
969   <dsig:Signature xmlns:dsig-xpath="http://www.w3.org/2002/06/xmldsig-filter2">
970     <dsig:SignedInfo>
971       <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
972       <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
973       <dsig:Reference URI="">
974         <dsig:Transforms>
975           <dsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
976           <dsig:Transform Algorithm="http://www.w3.org/2002/06/xmldsig-filter2">
977             <dsig-xpath:XPath Filter="intersect" //Contract</dsig-xpath:XPath>
978             <dsig-xpath:XPath Filter="subtract" //Party2</dsig-xpath:XPath>
979           </dsig:Transform>
980         </dsig:Transforms>
981         <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
982         <dsig:DigestValue></dsig:DigestValue>
983       </dsig:Reference>
984     </dsig:SignedInfo>
985     <dsig:SignatureValue>
986   </dsig:SignatureValue>

```

```
987     <dsig:KeyInfo>
988         <dsig:X509Data>
989             <dsig:X509Certificate></dsig:X509Certificate>
990             <dsig:X509SubjectName></dsig:X509SubjectName>
991             <dsig:X509IssuerSerial></dsig:X509IssuerSerial>
992         </dsig:X509Data>
993     </dsig:KeyInfo>
994 </dsig:Signature>
995 </Document>
996
```

997

## 6 PostMarkedReceipt Examples

998 PostMarked receipts are normally returned to the application as standalone XML structures, whether they are of type CMS/PKCS7 or XMLSig.  
 999 Upon request however <PostMarkedReceipt>'s can be embedded in the incoming signed document. This is true for both the Sign protocol as  
 1000 well as the Verify protocol. The first example below is a standalone <PostMarkedReceipt>, and the second example is one that is embedded  
 1001 into the signed document.

1002 **Example 1 Standalone PostMarkedReceipt:**

1003 This is an example of a PostMarkedReceipt. It is essentially a conventional XMLSig enveloping signature over the <SignatureValue> of the  
 1004 target signature being PostMarked. It contains three (3) <Reference> elements pointing to each of the following:

- 1005   ➤ a standard <dss:TstInfo> as per **[DSSCore]**
- 1006   ➤ an <epm:PostMarkedReceipt> element from the **[EPM]** schema
- 1007   ➤ the <SignatureValue> element of the target signature being PostMarked

1008 Selected element contents have been deliberately truncated for brevity and clarity.

```

1009
1010 <?xml version="1.0" encoding="UTF-8"?>
1011 <dsig:Signature Id="PostMarkedReceiptSignature" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
1012   <dsig:SignedInfo>
1013     <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1014     <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1015     <dsig:Reference URI="#TstInfo">
1016       <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1017       <dsig:DigestValue>jWkUFR6epvkrtaxTiQ33DiWy+18=</dsig:DigestValue>
1018     </dsig:Reference>
1019     <dsig:Reference URI="#Receipt">
1020       <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1021       <dsig:DigestValue>9JWKdLh/8Cs9Slu2QmZixOJl+x0=</dsig:DigestValue>
1022     </dsig:Reference>
1023     <dsig:Reference URI="#PostMarkedSignatures">
1024       <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1025       <dsig:DigestValue>MOBYPfrllMBCJz6yojbhrwH9KP4=</dsig:DigestValue>
1026     </dsig:Reference>
1027   </dsig:SignedInfo>
1028   <dsig:SignatureValue>qnBvJoSgo4OoiYYaE3AwL5/EDq7BhTT6 ... Qw11HK+zxy66I=</dsig:SignatureValue>
1029   <dsig:KeyInfo>
1030     <dsig:KeyName>C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, E= ... </dsig:KeyName>
1031     <dsig:X509Data>
1032       <x509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIEUDC ... EwZOBg==</x509Certificate>
1033     <x509SubjectName xmlns="http:// ... xmldsig#"> C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, E=... </x509SubjectName>
1034     <x509IssuerSerial xmlns="http://www.w3.org/2000/09/xmldsig#">
```

```

1035      <X509IssuerName>C=CA, O=CPC, OU=EPM Service, CN=Electronic PostMark CA, E=... </X509IssuerName>
1036      <X509SerialNumber>25</X509SerialNumber>
1037    </X509IssuerSerial>
1038    </dsig:X509Data>
1039  </dsig:KeyInfo>
1040  <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
1041    <dss:TstInfo xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema" Id="TstInfo">
1042      <SerialNumber>1847365279</SerialNumber>
1043      <CreationTime>2004-03-27T17:47:18.750</CreationTime>
1044      <Policy/>
1045      <ErrorBound/>
1046      <Ordered/>
1047      <TSA>C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, E= ... </TSA>
1048    </dss:TstInfo>
1049  </dsig:Object>
1050  <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="Receipt">
1051    <epm:PostMarkedReceipt xmlns:epm="http://www.upu.int/EPMService/schemas">
1052      <Receipt>
1053        <TransactionKey>
1054          <Locator>
1055            <CountryCode>CA</CountryCode>
1056            <Version>114</Version>
1057            <ServiceProvider>ePost Corporation</ServiceProvider>
1058            <Environment xsi:nil="true" />
1059          </Locator>
1060          <Key>1234567890</Key>
1061          <Sequence>1</Sequence>
1062        </TransactionKey>
1063        <Requester>CN=Joe Public, O=VeriSign Class 1 Certificate, C=CA, E=joe.public@rogers.com</Requester>
1064        <Operation>Verify</Operation>
1065        <TSAX509SubjectName> ... </TSAX509SubjectName>
1066        <MessageImprint> ... </MessageImprint>
1067        <PostMarkImage> ... </PostMarkImage>
1068        <RevocationStatusQualifier>CRL Checked</RevocationStatusQualifier>
1069        <TimeStampToken Mime-Type="application/pkcs7-signature"></TimeStampToken>
1070        <ReceiptMetadata>
1071          <Name> ... </Name>
1072          <Value>... </Value>
1073        </ReceiptMetadata>
1074      </Receipt>
1075    </epm:PostMarkedReceipt>
1076  </dsig:Object>
1077  <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
1078    <epm:PostMarkedContent xmlns:epm="http://www.upu.int/EPMService/schemas" Id="PostMarkedSignatures">
1079      <epm:PostMarkedSignatureValue>1NiHC2bBKfT ... AlfhecQo=</epm:PostMarkedSignatureValue>
1080    </epm:PostMarkedContent>
1081  </dsig:Object>
1082</dsig:Signature>
```

```

1083
1084 If the standalone PostMarkedReceipt covers more than one signature, the 3rd Referenced Object would look like this:
1085
1086 <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig# Id="PostMarkedSignatures">
1087   <epm:PostMarkedContent xmlns:epm="http://www.upu.int/EPMService/schemas">
1088     <PostMarkedSignatureValue>1NiHC2bBKfT ... AlfcGhecQo=</PostMarkedSignatureValue>
1089     <PostMarkedSignatureValue>aqw95gB/Tz5 ... n0qRqMHJ5c=</PostMarkedSignatureValue> ...
1090     ... would include as many other PostMarkedSignatureValue elements as may be present in the PostMarked document
1091     ...
1092   </epm:PostMarkedContent>
1093 </dsig:Object>
1094
1095

```

1096 **Note:** Similarly, when the <PostMarkedReceipt>'s signature scope simply covers data (as opposed to a SignatureValue), then the 3<sup>rd</sup>  
1097 <Reference> will be to an <Object> containing the hash of the data to be PostMarked with base64 encoding specified.

```

1098 <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig# Id="PostMarkedData">
1099   <epm:PostMarkedContent xmlns:epm="http://www.upu.int/EPMService/schemas"
1100     Encoding="http://www.w3.org/2000/09/xmldsig#base64">RGF0YSBgdG8gcQ...gVGkgMTUgMTI6MTA=</PostMarkedContent>
1101 </dsig:Object>
1102

```

### 1103 Example 2 Embedded PostMarkedReceipt:

1104 This is an example of an embedded <PostMarkedReceipt> returned after a successful Verify operation. It is a conventional XMLSig detached  
1105 signature over the <SignatureValue> of the target signature(s) being PostMarked. It contains three (3) <Reference> elements pointing to  
1106 each of the following:

- 1107 ➤ a standard <dss:TstInfo> as per **[DSSCore]**
- 1108 ➤ an <epm:PostMarkedReceipt> element from the **[EPM]** schema
- 1109 ➤ the <SignatureValue> element of the target signature(s) being PostMarked

1110 Note that depending on the value of the optional SignatureSelector element within the Verify request, the <PostMarkedReceipt> can  
1111 potentially cover all <SignatureValue>'s in the signed document when the document contains multiple signatures.

1112 Selected element contents have been deliberately truncated for brevity and clarity.

```

1113 <?xml version="1.0" encoding="UTF-8"?>
1114 <!DOCTYPE Document [
1115   <!ATTLIST Object Id ID #IMPLIED>
1116 ]>
1117 <Document>
1118 <!-- Beginning of PostMarkedReceipt signature -->
1119   <dsig:Signature Id="PostMarkedReceiptSignature" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
1120     <dsig:SignedInfo>
1121

```

```

1122 <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1123 <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1124 <dsig:Reference URI="#TstInfo">
1125     <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1126     <dsig:DigestValue>3Lk/6TE71dqeXZFUJ9qqaPInm24=</dsig:DigestValue>
1127 </dsig:Reference>
1128 <dsig:Reference URI="#Receipt">
1129     <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1130     <dsig:DigestValue>430zTvcoa9r8Rpr5DiVZf7IPv18=</dsig:DigestValue>
1131 </dsig:Reference>
1132 <dsig:Reference URI="">
1133     <dsig:Transforms>
1134         <dsig:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
1135             <dsig:XPath xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
1136                 ancestor-or-self::dsig:SignatureValue[@Id!="PostMarkedReceiptSignature"]
1137             </dsig:XPath>
1138         </dsig:Transform>
1139     </dsig:Transforms>
1140     <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1141     <dsig:DigestValue>LRAX6mCfAq8hprb8UMU1H35PTYw=</dsig:DigestValue>
1142 </dsig:Reference>
1143 </dsig:SignedInfo>
1144 <dsig:SignatureValue>qnBvJoSgo4OoiYYaE3AwbL5/EDq7BhTT6 ... Qw11HK+zxy66I=</dsig:SignatureValue>
1145 <dsig:KeyInfo>
1146     <dsig:KeyName>C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, E= ... </dsig:KeyName>
1147     <dsig:X509Data>
1148         <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIEUDC ... EwZOBg==</X509Certificate>
1149 <x509SubjectName xmlns="http://... xmldsig#"> C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, E=... </x509SubjectName>
1150 <x509IssuerSerial xmlns="http://www.w3.org/2000/09/xmldsig#">
1151     <x509IssuerName>C=CA, O=CPC, OU=EPM Service, CN=Electronic PostMark CA, E=... </x509IssuerName>
1152     <x509SerialNumber>25</x509SerialNumber>
1153 </x509IssuerSerial>
1154 </dsig:X509Data>
1155 </dsig:KeyInfo>
1156 <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="TstInfo">
1157     <dss:TstInfo xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema">
1158         <SerialNumber>1847365279</SerialNumber>
1159         <CreationTime>2004-03-27T17:47:18.750</CreationTime>
1160         <Policy/>
1161         <ErrorBound/>
1162         <Ordered/>
1163         <TSAX509SubjectName>C=CA, O=CPC, OU=EPM Service, CN=EPM Signature, ... </TSAX509SubjectName>
1164     </dss:TstInfo>
1165 </dsig:Object>
1166 <dsig:Object xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="Receipt">
1167     <epm:PostMarkedReceipt xmlns:epm="http://www.upu.int/EPMService/schemas">
1168         <Receipt>
1169             <TransactionKey>
```

```

1170      <Locator>
1171          <CountryCode>CA</CountryCode>
1172          <Version>114</Version>
1173          <ServiceProvider>ePost Corporation</ServiceProvider>
1174          <Environment xsi:nil="true"/>
1175      </Locator>
1176      <Key>1234567890</Key>
1177      <Sequence>1</Sequence>
1178  </TransactionKey>
1179  <Requester>CN=Joe Public, O=VeriSign Class 1 Certificate, C=CA, E=joe.public@rogers.com</Requester>
1180  <Operation>Verify</Operation>
1181  <TSAX509SubjectName> ... </TSAX509SubjectName>
1182  <MessageImprint> ... </MessageImprint>
1183  <PostMarkImage> ... </PostMarkImage>
1184  <RevocationStatusQualifier>CRL Checked</RevocationStatusQualifier>
1185  <TimeStampToken MimeType="application/pkcs7-signature"></TimeStampToken>
1186  <ReceiptMetadata>
1187      <Name> ... </Name>
1188      <Value> ... </Value>
1189  </ReceiptMetadata>
1190  </Receipt>
1191  </epm:PostMarkedReceipt>
1192  </dsig:Object>
1193 </dsig:Signature>
1194 <!-- End of PostMarkedReceipt signature -->
1195 <!-- Beginning of signed document being PostMarked -->
1196  <Object Id="DetachedDataBeingSigned">
1197      <PersonalData>
1198          <Name>Ed Smith</Name>
1199          <StreetAddress>1234 Mockingbird Lane</StreetAddress>
1200          <City>Yellowknife</City>
1201          <PostalCode>W1C6J3</PostalCode>
1202          <SocialInsuranceNumber>123456789</SIN>
1203      </PersonalData>
1204  </Object>
1205  <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#" Id="TargetSignature">
1206      <dsig:SignedInfo>
1207          <dsig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1208          <dsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1209          <dsig:Reference URI="#DetachedDataBeingSigned">
1210              <dsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1211              <dsig:DigestValue>Po3vwPXh8kdpRUAzMGjzlua065I=</dsig:DigestValue>
1212          </dsig:Reference>
1213      </dsig:SignedInfo>
1214      <dsig:SignatureValue>KyKUMJKW ... Yi7swX0FjLkDDZNs=</dsig:SignatureValue>
1215      <dsig:KeyInfo>
1216          <dsig:KeyName>C=CA, O=Acme Corp, CN=Joe Public, E= ... </dsig:KeyName>
1217          <dsig:X509Data>
```

```
1218     <X509Certificate xmlns="http://www.w3.org/2000/09/xmldsig#">MIIE ... EwZOBg==</X509Certificate>
1219     <X509SubjectName> C=CA, O=Acme Corp, CN=Joe Public, E= ... </X509SubjectName>
1220     <X509IssuerSerial>
1221         <X509IssuerName>C=CA, O=Partner CA, O=For Test Use Only, CN=Partner CA, E= ... </X509IssuerName>
1222         <X509SerialNumber>25</X509SerialNumber>
1223     </X509IssuerSerial>
1224     </dsig:X509Data>
1225     </dsig:KeyInfo>
1226   </dsig:Signature>
1227   <!-- End of signed document being PostMarked -->
1228 </Document>
1229
```

1230

## 7 Element cross-reference Table

1231

1232

The following tables provide a summary of the Input elements, options, and corresponding Output elements for each of the usage scenarios. Comments are also provided.

1233

### Sign Protocol

		Optionality	As Used in Sig Type		Elements Affected	Comments
Input / Request Elements			CMS/PKCS7	XMLSIG		
	OrganizationID	M	✓	✓		Must match the string specified at registration time.
	SignatureType	M	✓	✓		Tells the EPM whether this is a sign request, or a timestamp request, and also specifies CMS/PKCS7 or XMLSig.
	KeySelector	O	✓	✓		Optional since the key can usually can be derived from the underlying authentication mechanism. Also not required when using signing templates, in which case the key may be specified in <KeyInfo>. Can be used when non-default handling is required.
	SignedReferences	n/a				Not required by the EPM. This functionality is covered by signing templates in the EPM Profile.
	InputDocuments	M	✓	✓		Presently constrained to one <Document> occurrence.
	SignaturePlacement	n/a				Not required. Default handling of placement is supported by the EPM. Signature placement

					can be controlled as required by using signing templates. Placement of <code>&lt;PostMarkedReceipt&gt;</code> 's can be controlled via the Locator attribute.
	DocumentContainsTemplate	O		✓	Signatures produced will be returned in <code>&lt;DocumentWithSignature&gt;</code>
	ClaimedIdentity	O	✓	✓	Optionally used for alternate authentication schemes or when "Proof of Delivery" is required.
	<b>Processing Option Flags</b>				
	AddTimestamp	O	✓		Attribute not req'd. Produces a conventional timestamp as opposed to a <code>&lt;PostMarkedReceipt&gt;</code> .
	IssuePostMarkedReceipt	O	✓		Returns a standalone <code>&lt;PostMarkedReceipt&gt;</code> element.
	IssuePostMarkedReceipt	O		✓	Returns a standalone <code>&lt;PostMarkedReceipt&gt;</code> element in the response if the Location attribute is specified as standalone. If Location specifies embedded, the receipt will be embedded and returned in <code>&lt;DocumentWithSignature&gt;</code> .
	StoreNonRepudiationEvidence	O			The EPM will log the original request as well as the response and all result structures as evidence in the event of a dispute.
	ReturnSignatureInfo	O	✓	✓	Returns a <code>&lt;SignatureInfo&gt;</code>

					structure.	
	ReturnX509Info	O	✓	✓	Returns a <X509Info> structure.	
<b>Output / Response Elements</b>						
	Result	M	✓	✓		As per [DSSCore]. <ResultMajor>, <ResultMinor>, and <ResultMessage> will all be initialized and returned
	TransactionKey	M	✓	✓		This element is returned as part of the SignResponse and contains the unique identifier. Always initialized and returned.
	SignatureObject	M	✓	✓	Initialized for CMS/PKCS7 signatures and for XMLSig enveloping signatures. See also <DocumentWithSignature>	
	DocumentWithSignature	O		✓	Only initialized for XMLSig based enveloped and detached signatures. Also initialized when signing templates are used. See also <SignatureObject>.	
	PostMarkedReceipt	O	✓	✓	See <IssuePostMarkedReceipt> above.	
	SignatureInfo	O	✓	✓	Returned when <ReturnSignatureInfo> has been specified.	
	X509Info	O	✓	✓	Returned when <ReturnX509Info> has been specified.	

1234

1235

1236

1237

1238

**Verify Protocol**

		Optionality	As Used in Sig Type		Elements Affected	Comments
Input / Request Elements			CMS/PKCS7	XMLSIG		
	OrganizationID	M	✓	✓		Must match the string specified at registration time.
	SignatureObject	M	✓			Required when verifying CMS/PKCS7 detached signatures.
	InputDocuments	O	✓			Default location of "same document" signature to be verified.
	InputDocuments	M		✓		Presently constrained to one <Document> occurrence. Must contain signature(s) to be verified along with any referenced signed content.
	SignatureSelector	O		✓		Optionally used to specify the ancestor node containing the target signature to be verified (NodeName) or the XPath expression to select the signatures to be verified. May apply if more than one signature is present in the InputDocuments.
	ClaimedIdentity	O	✓	✓		Optionally used for alternate authentication schemes or when "Proof of Delivery" is required.
Processing Option Flags						
	AddTimestamp	O	✓		Updated CMS/PKCS7 signature, now containing an embedded RFC 3161 timestamp token, is returned in <SignatureObject>.	Allows for the inclusion of an RFC3161 embedded timestamp into the verified CMS/PKCS7 signature.

	IssuePostMarkedReceipt	O	✓		Returns a standalone <PostMarkedReceipt> element.	
	IssuePostMarkedReceipt	O		✓	Returns a standalone <PostMarkedReceipt> element in the response if the Location attribute is specified as standalone. If Location specifies embedded, the receipt will be embedded and returned in <DocumentWithSignature>. The SignatureSelector element optionally controls the scope of the PostMarkedReceipt signature.	
	StoreNonRepudiationEvidence	O				The EPM will log the original request as well as the response and all result structures as evidence in the event of a dispute.
	ReturnSignatureInfo	O	✓	✓	Returns a <SignatureInfo> structure.	
	ReturnX509Info	O	✓	✓	Returns a <X509Info> structure.	
<b>Output / Response Elements</b>						
	Result	M	✓	✓		As per [DSSCore]. <ResultMajor>, <ResultMinor>, and <ResultMessage> will all be initialized and returned
	TransactionKey	M	✓	✓		This element is returned as part of the VerifyResponse and contains the unique identifier. Always initialized and returned.
	PostMarkedReceipt	O	✓	✓	See <IssuePostMarkedReceipt> above.	
	SignatureObject	O	✓		Initialized for CMS/PKCS7	

					signatures when <AddTimestamp> has been specified. See also <DocumentWithSignature>	
	DocumentWithSignature	O		✓	Only initialized for XMLSig based signatures when <IssuePostMarkedReceipt> with a Location attribute specified as embedded. See also <IssuePostMarkedReceipt>.	
	SignatureInfo	O	✓	✓	Returned when <ReturnSignatureInfo> has been specified.	
	X509Info	O	✓	✓	Returned when <ReturnX509Info> has been specified.	

1239

1240

---

1241

## 8 References

1242

### 8.1 Normative

1243

[Core-XSD] T. Perrin et al. *DSS Schema*. OASIS, (MONTH/YEAR TBD)

1244

[DSSCore] T. Perrin et al. *Digital Signature Service Core Protocols and Elements*. OASIS, (MONTH/YEAR TBD)

1245

[RFC 2119] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. IETF RFC 2396, August 1998.

1246

<http://www.ietf.org/rfc/rfc2396.txt>.

1247

[TS 101733] Advanced Electronic Signatures. ETSI TS 101 733.

1248

[XAdES] XML Advanced Electronic Signatures. ETSI TS 101 903, February 2002 (shortly to be reissued).

1249

[XML-ns] T. Bray, D. Hollander, A. Layman. *Namespaces in XML*. W3C Recommendation, January 1999.

1250

<http://www.w3.org/TR/1999/REC-xml-names-19990114>

1251

[XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing*. W3C Recommendation, February 2002.

1252

<http://www.w3.org/TR/1999/REC-xml-names-19990114>

1253

[RFC 2634] P. Hoffman (ed.). Enhanced Security Services for S/MIME, June 1999.

1254

[RFC 3369] Message Syntax (CMS). R. Housley. August 2002.

1255

[EPM] Universal Postal Union, Electronic PostMark Web Service Description Language (WSDL)

1256

the UPU's Postal Technology Centre <http://www.ptc.upu.int/>.

## Appendix A. Revision History

Rev	Date	By Whom	What
wd-01	2004-07-27	Ed Shallow	Initial version
wd-02	2004-08-18	Ed Shallow	Update
wd-03	2004-09-06	Ed Shallow	Update
wd-04	2004-10-14	Ed Shallow	Juan-Carlos and Trevor's changes
wd-05	2004-11-21	Ed Shallow	Changes for Public Draft
wd-06	2004-11-30	Ed Shallow	Changes for Public Draft
wd-07	2004-12-12	Ed Shallow	Changes to Appendix examples (not released)
wd-08	2006-04-19	Ed Shallow	Changes to reflect core updates
wd-09	2006-08-28	Ed Shallow	Updated schema includes from :epm

1260

## Appendix B. Notices

1261 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be  
1262 claimed to pertain to the implementation or use of the technology described in this document or the extent to  
1263 which any license under such rights might or might not be available; neither does it represent that it has made any  
1264 effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications  
1265 can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances  
1266 of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the  
1267 use of such proprietary rights by implementers or users of this specification, can be obtained from the OASIS  
1268 Executive Director.

1269 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other  
1270 proprietary rights which may cover technology that may be required to implement this specification. Please  
1271 address the information to the OASIS Executive Director.

1272 Copyright © OASIS Open 2006. *All Rights Reserved.*

1273 This document and translations of it may be copied and furnished to others, and derivative works that comment  
1274 on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in  
1275 whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are  
1276 included on all such copies and derivative works. However, this document itself does not be modified in any way,  
1277 such as by removing the copyright notice or references to OASIS, except as needed for the purpose of  
1278 developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
1279 Property Rights document must be followed, or as required to translate it into languages other than English.

1280 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or  
1281 assigns.

1282 This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL  
1283 WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE  
1284 USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES  
1285 OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.