



DSS Extension for Local Signature Computation Version 1.0

Committee Specification 02

\$6 March 2017

Specification URIs

This version:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Previous version:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Latest version:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Technical Committee:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Chairs:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Editors:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Additional artifacts:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Related work:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>
URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

Declared XML Namespaces:

URI: <https://www.oasis-open.org/committees/spec02/20170306-dss-extension-for-local-signature-computation-version-1.0/>

ÔÈÈÁÝ ^àÁQ[!{ Á -Ac@ÁÜã }Ü^•][]•^Á
ÔÈÁV [ÈÚc] ÁC[]!{ æ&@
ÔÈÈÁZQÛVÁÜ^~^•dJ^•][]•^Á
ÔÈÈÁÜÖÖUÞÖÁÜ^~^•dJ^•][]•^Á
ÖÁC& }[, |â*^ { ^ }•ÁÇ[] È[]!{ æã^D
ÒÁÜ^çãã } Áã ç !^ ÁÇ[] È[]!{ æã^D

1 Introduction

V@AUCEUWAÖä ääUä } æ |^AU^!çæ.Á } ^äæææ } ÄDSSCoreÁ çæ ääää^Áæ | | ç & |Á^ Á } @&@æææ } @ææ Á^ } áÁ | & { ^ } • ÁÇ |Á | & { ^ } • Áä^ • • DÄ Áæ^!ç^!Áæ } áÁ^ & ä^Áææ Áæä } æ |^Á } Á@á | & { ^ } • Ä |Á^ Á } @&@æææ } @ææ Á^ } áÁ | & { ^ } • ÁÇ |Á | & { ^ } • Áä^ • • DÄ } áÁæä } æ |^Á | Áæ •^!ç^!Áæ } áÁ^ & ä^Áææ Áæä } • , ^Á } Á } @&@æææ } æ |^Á^!äæ • Á@Á | & { ^ } • É

V@•^Á |^!ææ } • Áæ } Á^Á •^~ |Á Áææææ ç Á -Á | } ç ç Ä |Á çæ] |^Á@^ Áææ Áæ | , Á | } • Á | Áæ&•• Áæ • ä * |^Á |] | | æ Á^ Á | Á } ä * Á |^ • • Á |^ æ^ • Ä } æ & } çæä^ áÁæ&•• Á | } d [| Áæ äää * Áæ } áÁæ&çä * [-Á } æ |^Á^~^ • • ÄV@^ Áæ } Áæ [Áæ | , Á | } • Á | Á^æ Á } æ |^Á } ä@~ ç^ ää * Á | } | | çÁ | } ç [-ç æ^Áæ } -ä |^ææ } É

V@Ä | | -Á çæ } á • Á@AUCEUWAÖÜÜÄ | | ç & |Á^ & @æææ^ & |^Dä } æ |^É |^ææ } Á^çæ Áæ } ÁUÜÖÖ [| ÁÜÖÖÄ } á | Á@Áæä & Á | } d [| Á -Á@Á •^!Á Á •^áÁ | Á@Áæç æÁ | }]^ææ } Á -Á@Áæ ää } æ |^çæ^ ÄV@Áç & |^Dä } æ |^É |^ææ } Á^çæ Áæ } Á | } ç ç Ä } [| Á } ææ äÁæ@Á^!ç^!Áææ] |^ (^) • @ÄÜÜÄ | | ç & |É

V@Áç & |^Dä } æ |^É |^ææ } Á^çæ Áæ } Á@æ^Áä } ä^Á | -ç æ^Áæ } áÁ |^ | } ç & Áææ ääæä • Áæ } á @ } & Á@AUCEUWAÖÜÜÄ | } |] æ ç^!çæ^ Áæ } Á^Á •^áÁ } çæ } á | Á@Á | } |^çæä • Á -Á@Áç & |^Dä } æ |^É &^ææ } Áæ } áÁ | & { ^ } ç } ä |^ææ } É

1.1 Terminology

V@Á^ Á | | ä • ÁMUSTÉMUST NOTÉREQUIREDÉSHALLÉSHALL NOTÉSHOULDÉSHOULD NOTÉ RECOMMENDEDÉMAYÉ } áOPTIONALÁæ^Á | Á^Á ç |] |^ ç äæ Á • & ä^áÁ } RFC 2119É

V@•^Á^ , [| ä • Áæ^Áææ ää^ áÁ @) Á •^áÁç Á } æ ää [~ • | Á^] ^ææ Á^~ ä^ (^) • Á | ç^!Á | | ç & |^ææ |^ Áæ } áÁ^ @ææ } Á@ææ^ & @Á | } | |^æææ } ä^áÁ & | æ Á -Á | } |^ (^) ææ } • ÄV@^ Á@^ Á | | ä • æ^Á | } @ææ ää^ áÁ@^ Áæ^Á } äæ ç Á@áÁ æ | æææ *^ æ^Á) • É

V@Ä |^ äæææ } Á • • Á@Á | | , ä * Á | [* | æ } çÁ } ç } • Á Á ç ç Á ns:Element>É AttributeÉ DatatypeÉ OtherCodeÉ

Ç } Ánput documentÁ@æÁç Á | Á^Á } ^áÁæ } Á^Áæ } Á } ä & Á -Áæææ@Áæ } Á^Á •^áÁæ } ~ ç | Áæ • } æ |^ Áæ } |ææ } Áæ& | ää * Á | Á@ÄDSSCoreÁ |^ äæææ } Ä |^ ç } Áç Ä |^ & @Á | & { ^ } • Áç^ } áÁæä } æ |^ÁÇ |Á çæ } |^ Áæ } |^É çä ç * Á } æ |^ Áæ } Á^Á } } ç |É } ^áD | Á^ • çæ] É

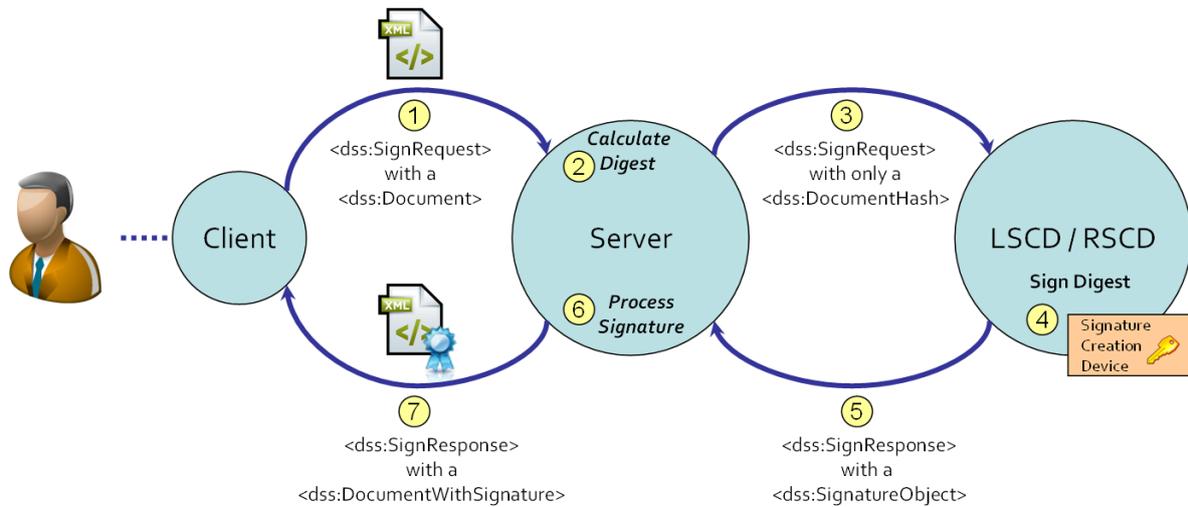
Ç } Ädigital signature valueÁ Áæäæ äÄ | | (Á -Áæä } æ |^É & | } | | ç } Áç Áæ<ds:Signature>Á | <dss:Base64Signature>Á } ç } Áæ dssSignatureObject>Á |^ (^) • Áæ& | ää * Á | Á@ÄDSSCoreá • |^ äæææ } Ä |^ ç } Áç Ä |^ @Áä ää^ } æ |^Áçæ^ Á^Á | }]^ ç^áÁ^ Á@Áç & |^Dä } æ |^É |^ææ } áÁçæ Á | Áæ } Áæ^ • ÄV@Áçææ ç Á -Á@Á | | • |] ää * Á^!çææ Á | } @& &^áÁ^ Á@Áçæ^ É

Ç } Äelectronic signatureÁ Áæ } Á | } @áÁä ää^ } æ |^Áçæ^ Áæ } áÁæ } Áæ } & |^ áÁ | Á • çæ } & Áæ } |^É çæ] ç } Á |^ äæ^ Á@Á^ Á } ä * Dä } äD | Áç | &ææ } Á } | |^ææ } Áç } Á |^ äæ^ Á@Áçææ ç Á -Á@Á | | • |] ää * &^!çææ Áæ } áÁ^ • ç^áÁ [| ç Áæ } äD | Áç } |^ áÁæ^ Á^!çææ • D |^ ää } æ |^ Á@Á |^ & | } ä } æ |^ Á } |] |^ Á^Áæ^ áÁ^ Á@Áä ää^ } æ |^Á^!çæ^ Á@Á | | • |] ää * Á^!çææ Á çææÁç [Áç |^ áDÉ

1.2 Abbreviations

- “ SCDÄÜä } æ |^É |^ææ } Á^çæ ÄÖÄ^çæ^ Á@æ^ Áææ æ |^Á -Á^ææ * Áæä ää^ } æ |^Áçæ^ Á • ä * æ |^ææ Á^ Á@æ^ Áç |^ áÁ^ Á@Áçæ^ É
- “ SSCDIÁÜ & |^ÁÜä } æ |^É |^ææ } Á^çæ ÄÖÄ • } æ |^É |^ææ } Á^çæ^ Á , @&@ { ^ • Á@ |^~ ä^ (^) • Á | ææ Áæ [,] Áæ } Áç } ^çÁ@Ä -Á@ÄÖ | |]^ææ } ÁÖá & ç^ ÁFJJJHÖÖÁ -Á@ÄÖ | |]^ææ } Üæ | æ^ } • Áæ } áÁ@ÄÖ |] & Á | -ÁFHÄÖ & |^ ÁFJJJÁ | } ÁæÄÖ | { { } } æ Á^æ^ , [| Á | Á^Á & | } ä • } æ |^Á } 999/93/ECÉ

Figure 3. Delegation of the signature creation to the LSCD or RSCD



QÁ^}^!aÉ@ÁSUÖÖÁ!ÁUÜÖÖÁ@e^Áá a^Á[c a^Áa áÁ^!;|{ aÉ &ÁaÉ aÉááá•Áá áÁ@} &Á} a|Á^ •~]]|c^áÁ^ ÁeÁUÖÜÖÁÜÁ&| }|áÉ o^Á!çáÁÁ Á@é á!Á@Á&| }|Ácáá•Á Á@ÁG^&^!D^ á} aÉ !^É &^aÉ} ÉÁ^&c[]} ÁÁ á} aÉ !^Á!| &••á *Áá áÁÁ &{ ^} o^ á} á^|aÉ} ÉÖ^!c@!{ |ÁÉá Á^}^!aÉÁ@ SÜÖÖÁ!ÁUÜÖÖÁ á @Á| o^ÁÁ&Á••á!Á@~ * @ÁÖÜÜ&|!Á!|[] &|É

V@ÁSUÖÖÁ!ÁUÜÖÖÁ a|Á^Áa|ÁÁ Á!ç^ÁÁ^~^•o^ Áá} Áá á^} Áá^•cÁÁ Áe•{ ^áÁ@Á@Á c^!aÉ c^Á@Á&c aÁUÜÖÖÁÁ&Á••^áÁ@~ * @^}^Á Á@Á!••á!Ácá ááá•É^ &Áe Á@ÁUÖWÜÜÁ!FÍD [!Á@ÁÖÖÖ]a} o^UÜÖÖÁ G ÁÖÖPÁ!|ÉD cá áááÉÁ @, •Á@Á@Á!| áÁÁ^Á[o^Á^}áÁ} c@Á&c aÁ c^!aÉ Á Á@ÁUÜÖÖÖÁ Áe•{ ^áÁ@Á@!Á a|Á^Á[{ ^Á áá!^, a^Á@Áá•cÁÁ Á!{ c@Á^}á|É]^&áá} |Á^} cÉ} Á Á@ÁUÜÖÖÉ

V@Á d[á &c} Á Áá ÁSUÖÖÁ!ÁUÜÖÖÁ@e^Áá^!^} o^Á^Áe^•ÉÁ^}áá *Á} Á@Á&@[|^Áe ^||É|Á}cÉ &ÉÁ Á@ÁSUÖÖÁ c^!aÉ^ÁÁ ÁÁ [áá^ÁçáÁÁ}áÁ Á@ÁUÜÜ&|a} o^} }á *Á} Áá^Á^}c]|aÉ!| ÁÁ!Á Á@ÁSUÖÖ&|}^&c^áÁ Á@ÁUÜÜ&|a} c^Á@Á^cÁ^&c} Á!Á^} o^Á { a^!Á Á^ÁÁe^• c@ÁÁ^Á&|}•á^!áÁ!Á@Á!| áÉ

1.7.2 Use Cases

V@ÁaÉ ÁÁ^~^} &Á á@Á@Á^Áe^•ÁÁ Áe Á|, •ÉÖÁ^!Á ááe^•ÁÁ á} á *Á&c} ÁÁ Á^á} Á Áe &|a} o^} |áÉ} ÉV@Á!a} o^} |áÉ} Á ááe^•ÁÁ} Ü~^•o^ Á@ÁÖá ááÁ} aÉ !^Á!çáÁÁÁÉV@ Öá ááÁ} aÉ !^Á!çáÁÁ&c} |e^•Á@Áá^•ÁUÜÜá cÉ} Ááá ááÁ} aÉ !^Áçá^ÁÁ!{ ^Á^&c} á{ ÇÉÁ^aÉ^•Á@Á^&c[]} ÁÁ á} aÉ !^Á!{ Á@Áá ááÁ} aÉ !^Áçá^ÁÁ} áÁ!| &••^•ÁÁ&|!áá *Á Á@ !^~^•ÁUÜÜ@Á^~|c^Á^&c[]} ÁÁ á} aÉ !^Áçá áÁ @^ç^!Á^~^•c^Á@Á! &{ ^} cÁÁ^} o^Á Á c@Á!a} o^ Á@Á} Ü~^•}•^ÁÉ

1.7.2.1 Use Case 1

V@Á^Áe^•{ ^•ÁÁ@Á!Á@Á!a} o^|aÉ!| ÉV@ÁG^&^!D^ á} aÉ!É^aÉ} ÁÁçáÁ&| ÁÁÁe •{ aÉaáÁ&|}^&c^áÁ Á@Á!a} o^|aÉ!| ÁÁ Á^á} Á ÁÁ{ aÉaáÁ^aÁ!ÉÁ^ÁÁ!|, •!ÁÁ^Á^ÁÁ Ác c@Á} áÁ^!Á&Á••Á@Á^áá} |áÉ} ÉV!Á}cÉ &ÁÁ&| &{ ^} o^ á} aÉ^} o^ÁU^•c{ ÁÖTÜÜV@Á^á á} |áÉ} Á!Á}cÉ &ÁÁÖTÜÜÖÁ]|Á^} c^Á@ÁÁUÜÜ!|[] &|ÁÁÁ|, ÁÁÁ^!ÁÁ} Á! &{ ^} o^É V[Áá} ÁGÁÁ &{ ^} cÉÁ@Á^!Á^&ÁÁÁ &{ ^} o^Á!{ Á á@Á Á@Á^áÁ} |áÉ} Á!Á}cÉ &ÁÁÖTÜÜÁ} áÁ@Á^áÁ} |áÉ} Á^} áÁ@Á! &{ ^} cÉÁ Á@ÁUÜÜÁ!ç^!Á@ÁUÜÜÁ!ç^!Á ááá •G[{ ^@, D@ÁÁá ááÁ} aÉ !^Áçá^Á&| } ^c^ÁÁ^Á@Á{ aÉaáÉÁ[c^Á@Á@ÁUÜÜÁ!ç^!Á&c}] c aÉÁ••Á@Á{ aÉaáÁ^aÁ!Á á@~^•aáá} aÁ^&c} á { •L!|[]|•aÁ Á!|ç^Á@Á Áe^•Á!Á^} c^ÁÁ Ü^&c} ÁÁ&c} ÁÉÉÉÉÜ!|[]^ÁÁ!|[]}•É

3.1.2.1 Element <dss:Result>

In case the end user cancelled the signing operation, the Digital Signature Service MUST return a <dss:ResultMajor> with the value RequesterError and a <dss:ResultMinor> with the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:user-cancelled
```

In case the Digital Signature Service detects a problem with the client runtime environment, the service returns a <dss:ResultMajor> with the value RequesterError and a <dss:ResultMinor> with the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:client-runtime-error
```

3.2 Two-Step Approach

This clause enables the client to obtain a signed document in two steps; the client MUST create of the digital signature value using a (secure) signature-creation device based on the document digest that is received from the Digital Signature Service. The interaction with the Digital Signature Service involves two <dss:SignRequest> requests:

- The FIRST <dss:SignRequest> MUST initiate the process by sending the document to the Digital Signature Service. (This is the *first step* in the two-step approach.) The Digital Signature Service MUST respond with the FIRST <dss:SignResponse>: it MUST contain the digest of the document. The Digital Signature Service MUST postpone processing of the document until the <dss:SignatureObject> is received in the SECOND <dss:SignRequest>.
- The SECOND <dss:SignRequest> MUST be sent to finalize the process by sending the <dss:SignatureObject> to the Digital Signature Service. (This is the *second step* in the two-step approach.) The Digital Signature Service MUST resume processing and MUST respond with the SECOND (and final) <dss:SignResponse> which contains the electronic signature or signed document, depending what has been requested.

3.2.1 Element <dss:SignRequest>

This clause profiles the <dss:SignRequest> element.

The [DSSCore] attribute Profile (Section 3.1) MUST have the value:

```
http://docs.oasis-open.org/dss-x/ns/localsig
```

Whenever needed, the [DSSCore] element <dss:AdditionalProfile> MAY be used to specify additional profiles for the proper creation of the resulting document(s) by the Digital Signature Service. The interpretation of the additional profile(s) is determined by the corresponding specification(s).

The [DSSCore] element <dss:InputDocuments> (Section 2.4) MUST be present in the FIRST request and MAY be present in the SECOND request. If present in the SECOND request, it MUST contain the document as used in the FIRST request.

3.2.1.1 Element <dss:OptionalInputs>

This clause profiles Optional Input elements.

3.2.1.1.1 Element <dss:ServicePolicy>

The [DSSCore] element <dss:ServicePolicy> (Section 2.8.1) MUST be present and MUST have the value:

```
http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
```

This policy instructs the Digital Signature Service that two request/response pairs are used to obtain the digital signature value from the (secure) signature-creation device of the end user.

3.2.1.1.2 Element <localsig:RequestDocumentHash>

The new element <localsig:RequestDocumentHash> MUST be present in the FIRST request. The type of the element is defined as follows (taken from [LocalSigXSD]):

```
<xs:element name="RequestDocumentHash">
  <xs:complexType>
    <xs:sequence>
      <xs:element
        minOccurs="0" maxOccurs="1"
        ref="ds:DigestMethod"/>
    </xs:sequence>
    <xs:attribute
      name="MaintainRequestState"
      use="optional"
      type="xs:boolean"/>
  </xs:complexType>
</xs:element>
```

The element <localsig:RequestDocumentHash> instructs the Digital Signature Service to return the digest of the document and to postpone further processing.

The attribute `MaintainRequestState` MAY be used to instruct the Digital Signature Service to maintain the state of the request, until the SECOND request is received or a certain predefined timeout has been reached (the timeout is determined by the Digital Signature Service). If the attribute is not specified, the assumed value is `false`. If the attribute `MaintainRequestState` is not used or `false`, then the SECOND request MUST contain the same document as specified in the FIRST request.

The [XMLSig] element <ds:DigestMethod> (Section 4.3.3.5) MAY be used to instruct the Digital Signature Service to use the specified type of digest method. If no method is specified, the Digital Signature Service will use a default digest method.

Note. The state is only useful (i) in case the document digest cannot be easily re-created, for instance if a signature has to be incorporated into a PDF document, or (ii) the document has to be transferred only once.

3.2.1.1.3 Element <dss:SignatureObject>

The [DSSCore] element <dss:SignatureObject> (Section 2.5) MUST be used in the SECOND request to provide the digital signature value, obtained from the (secure) signature-creation device at the client. The Digital Signature Service creates and processes the electronic signature based on the <dss:SignatureObject> and the request.

A <dss:Base64Signature> element MUST be used for the digital signature value together with a specified value for the `type` attribute. For OID's a urn according to [RFC 3061] and [draft-larmouth-oid-iri-04] MAY be used.

3.2.1.1.4 Element <localsig:CorrelationID>

The new element <localsig:CorrelationID> MUST only be used in the SECOND request if and only if the FIRST response contained the element <localsig:CorrelationID>. (Their value MUST be the same.)

The type of the element is defined as follows (taken from [LocalSigXSD]):

```
<xs:element name="CorrelationID" type="xs:NCName"/>
```

3.2.2 Element <dss:SignResponse>

This clause profiles the <dss:SignResponse> element.

3.2.2.1 Element <dss:Result>

In response to a successful processing of a <dss:SignRequest> that specified the element <localsig:RequestDocumentHash>, the <dss:ResultMajor> contains the value Success and the <dss:ResultMinor> the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:document-hash
```

In response to a successful processing of a <dss:SignRequest> that specified the element <localsig:SignatureObject>, the <dss:ResultMajor> and <dss:ResultMinor> follow the [DSSCore] specification, Section 2.6.

3.2.2.2 Element <dss:OptionalOutputs>

This profile defines the Optional Output elements.

3.2.2.2.1 Element <dss:DocumentHash>

The [DSSCore] element <dss:DocumentHash> (Section 2.4.4) MUST be returned in response to a <dss:SignRequest> that specified the element <localsig:RequestDocumentHash>. (Note that [DSSCore] specifies the use of this element as part of the OptionalInputs.)

The client uses the document digest for further processing by the (secure) signature-creation device.

3.2.2.2.2 Element <localsig:CorrelationID>

The new element <localsig:CorrelationID> MUST be returned in response to a <dss:SignRequest> if and only if the element <localsig:RequestDocumentHash> element was present within the SignRequest and its MaintainRequestState attribute was set to true. The type of the element is defined as follows (taken from [LocalSigXSD]):

```
<xs:element name="CorrelationID" type="xs:NCName"/>
```

The Digital Signature Service will generate a suitable value on its own behalf so that a client can refer to the state of its FIRST request.

The client MUST use this value in the SECOND request to refer to this state.

3.3 Third-Party

This clause enables the client to obtain an electronic signature (or signed document whenever requested) from the Digital Signature Service.

3.3.1 Element <dss:SignRequest>

This clause profiles the <dss:SignRequest> element.

The [DSSCore] attribute Profile (Section 3.1) MUST have the value:

```
http://docs.oasis-open.org/dss-x/ns/localsig
```

Whenever needed, the [DSSCore] element <dss:AdditionalProfile> MAY be used to specify additional profiles for the proper creation of the resulting document(s) by the Digital Signature Service. The interpretation of the additional profile(s) is determined by the corresponding specification(s).

3.3.1.1 Element <dss:OptionalInputs>

This clause profiles the Optional Inputs elements.

3.3.1.1.1 Element <dss:ServicePolicy>

The [DSSCore] element <dss:ServicePolicy> (Section 2.8.1) MUST be present and MUST have the value:

```
http://docs.oasis-open.org/dss-x/ns/localsig/delegation
```

This policy instructs the Digital Signature Service to delegate the creation of the signature to a third-party.

3.3.1.1.2 Element <ds:DigestMethod>

The [DSSCore] element <ds:DigestMethod> (Section 4.3.3.5) MAY be present to specify which digest method has to be used by the Digital Signature Service.

3.3.1.1.3 Element <localsig:ChallengeCode>

The new element <localsig:ChallengeCode> MUST be present in the request and MUST have a random value that can easily be read by a person. The type of the element is defined as follows (taken from [LocalSigXSD]):

```
<xs:element name="ChallengeCode" type="xs:NCName" />
```

The client has to show the challenge code to the end user. The end user MUST be able to compare the code with the value that is shown on the mobile device before it confirms the computation of the digital signature value by the mobile device (the challenge code is sent to the third-party as well).

3.3.1.1.4 Element <localsig:ResponseCode>

The new element <localsig:ResponseCode> MAY be present in the request. If present it MUST have a random value that can easily be read and entered by a person. The type of the element is defined as follows (taken from [LocalSigXSD]):

```
<xs:element name="ResponseCode" type="xs:NCName" />
```

The client has to show the response code to the end user. The end user MUST be able to enter the response code at the mobile device before it confirms the computation of the digital signature value by the mobile device; the third-party MUST return the response code to the Digital Signature Service. The Digital Signature Service only creates the requested electronic signature if the response code matches the value that was given by the client in the request.

3.3.2 Element <dss:SignResponse>

This clause profiles the <dss:SignResponse> element.

The <dss:SignResponse> contains (in according to the [DSSCore] specification, Section 3.2) either the electronic signature or the signed document, or an error message.

3.3.2.1 Element <dss:Result>

If the signature creation is cancelled by the end user the <dss:ResultMajor> contains the value for ResponderError and the <dss:ResultMinor> the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:user-cancelled
```

If the third-party is not able to handle the signature creation the <dss:ResultMajor> contains the value for ResponderError and the <dss:ResultMinor> the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:delegation-failed
```

If the response code that is returned by the third-party does not correspond to the value that is provided in the request, the <dss:ResultMajor> contains the value for ResponderError and the <dss:ResultMinor> the value:

urn:oasis:names:tc:dss-x:profiles:localsig:incorrect-responsecode

4 Protocol Bindings

OASIS DSS bindings are categorized under either transport bindings or security bindings as defined under Section 6 of [DSSCore]. The DSS signing protocol messages inherently assume that all security aspects are covered by the transport binding and appropriate security binding.

In Section [Section 3.1, "User Agent"](#) a transport binding is assumed by which a session is established between the Digital Signature Service and the HTTP User Agent. This document profiles a binding for a HTTP User Agent at the client using a web form.

In Section [Section 3.2, "Two-Step Approach"](#) it is assumed that a session (transaction) is created when a request contains the element `<localsig:RequestDocumentHash>` with the attribute `MaintainRequestState="true"`. The session is not profiled in this document.

4.1 WEB FORM Transport Binding

A WEB FORM binding is defined as a mechanism by which OASIS DSS protocol messages may be transmitted within the base64-encoded content of a HTML form control.

The reference URI for this binding is:

```
urn:oasis:names:tc:dss-x:profiles:localsig:bindings:web-form
```

4.1.1 Overview

The WEB FORM binding is intended for those cases where a requester and responder need to communicate using a HTTP User Agent (as defined in HTTP 1.1 [RFC 2616]) as an intermediary. This may be necessary, for example, if the communicating parties do not share a direct path of communication. It may also be needed if the responder requires an interaction with the User Agent in order to fulfill the request, such as when the User Agent must authenticate itself.

The OASIS DSS `<dss:SignRequest>` and `<dss:SignResponse>` XML messages are encoded into a HTML form, as described in the next Section.

4.1.2 Message Encoding using a HTML form

The HTML document MUST adhere to the XHTML 1.0 specification [XHTML] to ease parsing.

The `action` attribute of the HTML form MUST be the HTTP endpoint of the recipient: either the OASIS Digital Signature Service in case of the `<dss:SignRequest>` or the client in case of the `<dss:SignResponse>`.

The `method` attribute of the HTML form MUST be `POST`.

A `<dss:SignRequest>` or a `<dss:SignResponse>` message MUST be base64-encoded and placed in a hidden HTML form control within the HTML form (as defined by [HTML401] Section 17). The base64-encoded value MAY be line-wrapped at a reasonable length in accordance with common practice.

- If the message contains an OASIS DSS `<dss:SignRequest>` then the hidden HTML form control MUST be named `signrequest`.
- If the message contains an OASIS DSS `<dss:SignResponse>`, then the hidden HTML form control MUST be named `signresponse`.

The `clienturl` attribute MAY be used to provide the Digital Signature Service with a client URL. This URL will be used by the Digital Signature Service for the transmission of the response HTML form. If the `clienturl` is not specified, either an error is returned to the HTTP agent or a predefined URL is used by the Digital Signature Service.

Any additional HTML form controls or presentation MAY be included to allow the recipient to process the message.

Any technique supported by the User Agent MAY be used to cause the submission of the HTML form, and any HTML form content necessary to support this MAY be included, such as submit controls and client-side scripting commands. However, the Digital Signature Service MUST be able to process the message regardless for the mechanism by which the form submission is initiated.

Note that any HTML form control values included MUST be transformed so as to be safe to include in the XHTML document. This includes transforming characters such as quotes into HTML entities, etc.

4.1.3 HTTP and Caching Considerations

HTTP proxies and the User Agent intermediary should not cache OASIS DSS protocol messages. To ensure this, the following rules SHOULD be followed. When returning OASIS DSS protocol messages using HTTP 1.1, HTTP responders SHOULD:

- Include a `Cache-Control` header field set to "no-cache, no-store".
- Include a `Pragma` header field set to "no-cache".

There are no other restrictions on the use of HTTP headers.

4.2 Security Binding (Non-Normative)

4.2.1 Security Considerations

Before deployment, each combination of profile, transport binding, and security binding SHOULD be analyzed for vulnerability in the context of the specific protocol exchange and the deployment environment. Below we illustrate some of the security concerns that often come up with protocols of this type, but we stress that this is not an exhaustive list of concerns.

As the OASIS DSS protocol defined in this document is similar to the SAML protocol most of the security considerations defined in [SAMLCore] also apply to the OASIS DSS protocol.

The creation of document signatures using the OASIS DSS protocol yields additional attack vectors, due to possible manipulations of the document that is being transferred between DSS client and a DSS server. If the end user signs a different document as assumed by the DSS client, the impact could be huge. Therefore, it is of eminent importance to properly secure the OASIS DSS protocol request message that is transferred from DSS client to the DSS server via an intermediate User Agent.

4.2.2 TLS Security

The [Section 4.1, "WEB FORM Transport Binding"](#) SHOULD be used with the TLS Security Bindings as defined under Section 6.3 of [DSSCore].

4.2.3 Claimed Identity

The DSS Client can include the optional `<dss:ClaimedIdentity>` element as defined in [DSSCore] Section 2.8.2 to indicate the identity of the client who is making the request. The information provided by `<dss:ClaimedIdentity>` can be used to further personalize the interface presented to the end user by the DSS server.

In case the DSS server detects a problem with the claimed identity, the service returns in `<dss:SignResponse>` a `<dss:ResultMajor>` with the value `RequesterError` and a `<dss:ResultMinor>` with the value:

```
urn:oasis:names:tc:dss-x:profiles:localsig:resultminor:claimed-identity
```

5 Conformance

There are three conformance profiles, one for each approach.

- Profile A: Stateless Two-Step Approach
- Profile B: Stateful Two-Step Approach
- Profile C: User Agent Approach
- Profile D: Third Party Approach

5.1 Conformance Profile A - Stateless Two-Step Approach

This conformance profile only applies to a Two-Step approach; see [Section 3.2, "Two-Step Approach"](#).

5.1.1 Conformance Target: Server

The subject of the conformance test is the server that implements the Digital Signature Service.

5.1.1.1 Level 1

The conformance level states that the Digital Signature Service cannot maintain state information between subsequent requests in the Two-Step approach.

The request **MUST** contain the element `<dss:ServicePolicy>` with the value

```
http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
```

If a request contains the element `<localsig:RequestDocumentHash>` and the attribute value of `MaintainRequestState` is either absent or has a value of "false", then compute the digest value of the given input document. In case of success, the response **MUST** contain the element `<dss:DocumentHash>` with the digest value.

If a request contains the element `<dss:SignatureObject>` (and optionally, the `<dss:DocumentHash>`), then the server **MUST** create the signed document by means of the given `<dss:SignatureObject>` and the given input document. In case of success, the response **MUST** contain the signed document.

The element `<localsig:CorrelationID>` is ignored in a request; the element `<localsig:CorrelationID>` **MUST NOT** be returned in a response.

5.1.2 Conformance Target: Client

The subject of the conformance test is the client of the Digital Signature Service.

5.1.2.1 Level 1

The conformance level states that the client of the Digital Signature Service is capable of computing a digital signature value based on a given digest value of a document

The requests **MUST** contain the element `<dss:ServicePolicy>` with the value

```
http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
```

The **FIRST** request **MUST** contain the document to be signed as well as the element `<localsig:RequestDocumentHash>` for which the attribute `MaintainRequestState` is either absent or has a value of "false".

The `<dss:DocumentHash>` is obtained from the FIRST response and MUST be used to compute the digital signature value.

The SECOND request MUST contain the document to be signed, as provided in the FIRST request, as well as the `<dss:SignatureObject>` that contains the digital signature value.

If the FIRST request contained the `<dss:DocumentHash>` element, the SECOND request MUST contain that element too with the same value.

5.2 Conformance Profile B - Stateful Two-Step Approach

This conformance profile only applies to a Two-Step approach; see [Section 3.2, "Two-Step Approach"](#).

5.2.1 Conformance Target: Server

The subject of the conformance test is the server that implements the Digital Signature Service.

5.2.1.1 Level 1

The conformance level 'Stateful' states that the Digital Signature Service is able to maintain state information between two subsequent requests (in the Two-Step approach).

The element `<dss:ServicePolicy>` MUST be present with the value

```
http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
```

The FIRST request MUST contain the element `<localsig:RequestDocumentHash>` and the attribute `MaintainRequestState="true"` in the `OptionalInputs`. In case of success, the FIRST response MUST contain a reference to the state, by means of the element `<localsig:CorrelationID>` and MUST contain the digest value of the input document by means of the element `<dss:DocumentHash>`.

The SECOND request MUST contain the `<dss:SignatureObject>` and MUST contain the element `<localsig:CorrelationID>` that refers to the state of a corresponding (FIRST) request. The Digital Signature Service MUST use the referred state to create the signed document. In case of success, the SECOND response MUST contain the electronic signature or signed document, based on the input document found in the FIRST request and the referred state.

5.2.2 Conformance Target: Client

The subject of the conformance test is the client of the Digital Signature Service.

5.2.2.1 Level 1

The conformance level states that the client of the Digital Signature Service is capable of computing a digital signature value based on a given digest value of a document

The FIRST request MUST contain the element `<localsig:RequestDocumentHash>`; the attribute `MaintainRequestState` is either absent or has a value of `"false"`.

The FIRST request MUST NOT contain the element `<localsig:CorrelationID>`

The `<dss:DocumentHash>` obtained from the FIRST response MUST be used to compute the digital signature value.

The SECOND request MUST contain the `<dss:SignatureObject>` that contains the digital signature value.

The SECOND request MUST contain the element `<localsig:CorrelationID>` with the value that is obtained from the FIRST response.

If the FIRST request contained the `<dss:DocumentHash>` element, the SECOND request MUST contain that element too with the same value.

5.3 Conformance Profile C - User Agent

This conformance level only applies to the User Agent approach; see [Section 3.1, "User Agent"](#).

5.3.1 Conformance Target: Server

The subject of the conformance test is the server that implements the Digital Signature Service.

5.3.1.1 Level 1

The conformance profile states that the Digital Signature Service is capable of using a HTTP User Agent.

The element `<dss:ServicePolicy>` MUST be present with the value

```
http://docs.oasis-open.org/dss-x/ns/localsig/user-agent
```

The request follows the [\[DSSCore\]](#) specification, Section 3.1. The response contains the electronic signature or signed document according to the request, as defined by the [\[DSSCore\]](#) specification, Section 3.2.

The server MUST implement the protocol binding according to [Section 4.1, "WEB FORM Transport Binding"](#).

The server MUST be able to receive the digital signature value that has been computed by the user agent. The server MUST use the received digital signature value to create the signed document.

5.4 Conformance Profile D - Third Party

This conformance level only applies to the Third Party approach; see [Section 3.3, "Third-Party"](#).

5.4.1 Conformance Target: Server

The subject of the conformance test is the server that implements the Digital Signature Service.

5.4.1.1 Level 1

The conformance profile states that the Digital Signature Service is capable of delegating the signature creation to a third-party.

The element `<dss:ServicePolicy>` MUST be present in both requests with the value

```
http://docs.oasis-open.org/dss-x/ns/localsig/delegation
```

The element `<localsig:ChallengeCode>`, MUST be present

The server MUST create a new `SignRequest` and send it to the Third Party. The `SignRequest` MUST contain the element `<localsig:ChallengeCode>`. and MUST NOT contain the element `<localsig:ResponseCode>`.

If the request contains the element `<ds:DigestMethod>` it must instruct the Third Party to use the specified digest method.

The server MUST wait for the `SignResponse` of the Third Party. In case of success, the `SignResponse` MUST contain the element `<dss:SignatureObject>`.

In case of success, the response MUST contain the electronic signature or signed document as requested, as defined by the [DSSCore] specification, Section 3.2, based on the input document and the <dss:SignatureObject> from the SignResponse of the Third Party.

5.4.1.2 Level 2

The conformance profile states the same capabilities a level 1 and the server is capable acting upon a response code.

The request MUST contain the element <localsig:ResponseCode>.

The SignResponse from the Third Party MUST contain an element <localsig:ResponseCode> with the value that has been entered by the user.

When the server receives the SignResponse from the Third Party, the SignResponse MUST contain the element <localsig:ResponseCode>. The electronic signature or signed document MUST ONLY be created if and only if the value of the element <localsig:ResponseCode>, obtained from the Third Party, is the same as the value of the element <localsig:ResponseCode>, obtained from the client.

5.4.2 Conformance Target: Client

The subject of the conformance test is the client of the Digital Signature Service.

5.4.2.1 Level 1

The conformance profile states that the client is capable of creating a challenge code and presenting the challenge code to the user.

The client MUST create a random value, the challenge code, that can easily be read and entered by a person.

The client MUST present the challenge code to the user. Information MUST be displayed to the user about the use of the code: the user has to compare the presented code with the code that is received from the Third Party. If the codes do not match, the user must cancel the operation.

The request MUST contain the element <localsig:ChallengeCode> with the challenge code.

The client MUST be able to create a request and process a response, as defined by the [DSSCore] specification.

5.4.2.2 Level 2

The conformance profile states the same capabilities a level 1 and the client is capable of creating a response code.

The client MUST create a random value, the response code, that can easily be read and entered by a person.

The client MUST present the response code to the user. Information MUST be displayed to the user about the use of the code: the user has to enter the code into the application of the Third Party.

The request MUST contain the element <localsig:ResponseCode> with the response code.

Appendix A XML Schema Definition (Non-Normative)

A.1 Schema

The structures described in this specification are contained in the schema file which is part of [LocalSigXSD]. The xml schema definitions present within this document are copy of the XML schema file and must be considered as informative text, and that in case of discrepancy, definitions within the XML schema prevail.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  XSD for "DSS Extension for Local Signature Computation Version 1.0"
  Committee Specification 02
  6 March 2017
  Copyright (c) OASIS Open 2017. All Rights Reserved.
-->
<xs:schema
  xmlns:localsig="http://docs.oasis-open.org/dss-x/ns/localsig"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  targetNamespace="http://docs.oasis-open.org/dss-x/ns/localsig"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:annotation>
    <xs:documentation>
      XSD for "DSS Extension for Local Signature Computation Version 1.0"
      Committee Specification 02
      6 March 2017
      Copyright (c) OASIS Open 2017. All Rights Reserved.

      Declared XML Namespace:
      http://docs.oasis-open.org/dss-x/ns/localsig

      XSD Schema Location:
      http://docs.oasis-open.org/dss-x/localsig/v1.0/csprd03/schemas/localsig-v1.0.xsd
    </xs:documentation>
  </xs:annotation>
  <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
    schemaLocation=
      "http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
  <xs:import namespace="urn:oasis:names:tc:dss:1.0:core:schema"
    schemaLocation=
      "http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-schema-v1.0-os.xsd"/>
  <xs:element name="RequestDocumentHash">
    <xs:annotation>
      <xs:documentation>
        This element is part of the Two-Step approach
        in a SignRequest (as part of the OptionalInputs).
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element
          minOccurs="0"

```

```

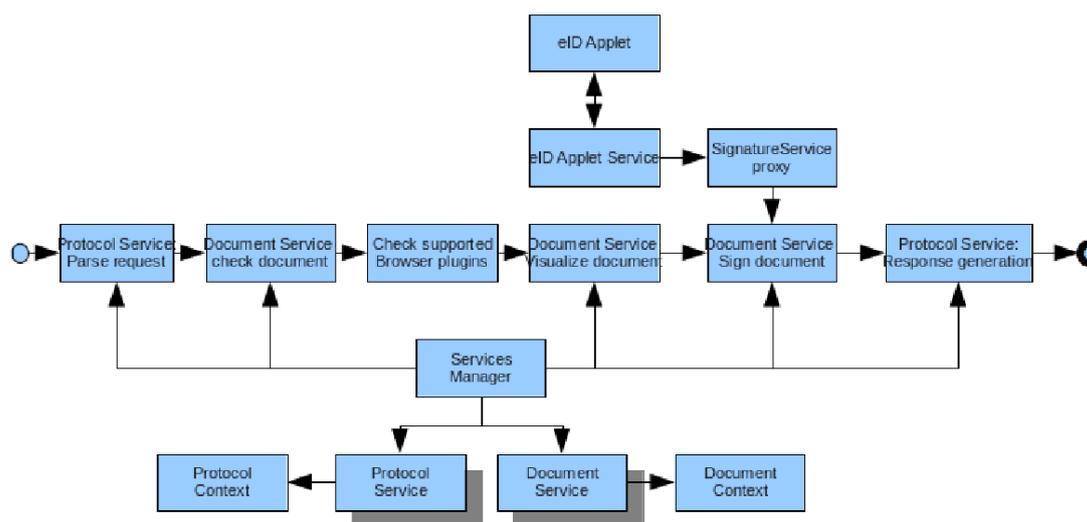
        maxOccurs="1"
        ref="ds:DigestMethod"/>
    </xs:sequence>
    <xs:attribute
        name="MaintainRequestState"
        use="optional"
        type="xs:boolean"/>
</xs:complexType>
</xs:element>
<xs:element name="CorrelationID" type="xs:NCName">
    <xs:annotation>
        <xs:documentation>
            This element is part of the Two-Step approach.
            The CorrelationID is obtained in the first step, from
            a SignResponse (as part of the OptionalOutputs)
            and is provided in the second step, in a SignRequest
            (as part of the OptionalInputs).
        </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="ChallengeCode" type="xs:NCName">
    <xs:annotation>
        <xs:documentation>
            This element is part of the Third-Party approach
            in a SignRequest (as part of the OptionalInputs).
        </xs:documentation>
    </xs:annotation>
</xs:element>
<xs:element name="ResponseCode" type="xs:NCName">
    <xs:annotation>
        <xs:documentation>
            This element is part of the Third-Party approach
            in a SignRequest (as part of the OptionalInputs).
        </xs:documentation>
    </xs:annotation>
</xs:element>
<!--
    The element <dss:DocumentHash> as defined by the
    DSS-core xml schema definition (see corresponding import).
    The element is part of the Two-Step approach in the
    first step, in a SignResponse (as part of the OptionalOutputs).
-->
<!--
    The element <dss:SignatureObject> as defined by the
    DSS-core xml schema definition (see corresponding import).
    The element is part of the Two-Step approach in the
    second step, in a SignRequest (as part of the OptionalInputs).
-->
</xs:schema>

```

Appendix B Sample Application (Non-Normative)

Figure B.1, “eID DSS Signature Pipeline” shows the design of a digital signature service that uses the Belgian eID card as client signing token.

Figure B.1. eID DSS Signature Pipeline

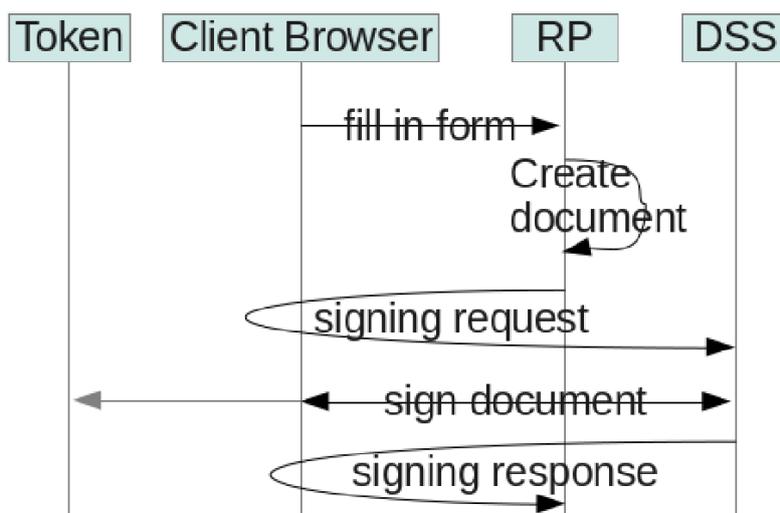


An end user enters the DSS signature pipeline via some protocol. First of all the appropriate protocol service parses the request. At this step the mime type of the incoming document is determined. Via the mime type the appropriate document service can be selected. The document service will first check the incoming document (syntax,...). Next the web browser capabilities are being queried in order for the document service to be able to correctly visualize the received document. After the user's consent the document service will orchestrate the document signing process using a web browser Java applet component. Finally the signed document is returned via the protocol service that also handled the incoming protocol request.

The advantage of such a generic signature pipeline architecture is that one can easily add new document formats by providing a new document service implementation. Because the protocol handling is also isolated in protocol services, one can also easily add new DSS protocols to the platform. Another advantage of such a signature pipeline is that every Relying Party (RP), in the role of a DSS client, that uses the platform is guaranteed that the user followed a certain signature ceremony and is fully aware of the content of the signed document. This guarantee can be interesting from a legal point of view.

A sample protocol flow is shown in [Figure B.2, “Sequence diagram of a simple protocol flow”](#).

Figure B.2. Sequence diagram of a simple protocol flow



Here the client navigates via a web browser to the web application of the relying party. As part of the business work flow, the client fills in a web form. The relying party's web application converts the received form data into a document that needs to be signed by the client. Now the relying party's web application redirects the client web browser to the DSS web application. The DSS web application takes care of the signing ceremony using Java applet technology to connect to the client's token. Finally the DSS web application redirects the client's web browser back to the relying party. The relying party can now further process the signed document as part of the implemented business work flow.

In such scenarios it is difficult to use the existing OASIS DSS protocol messages as is, because the OASIS DSS protocol does not provide the security mechanisms required to secure the communication between relying parties and the DSS in the context of web browsers. Various MITM attacks are possible at different points during the signature ceremony. Similar to the OASIS SAML Browser POST profile, we need to define additional wrapper messages to be able to guarantee secure transportation of the DSS requests and responses via web browsers.

A disadvantage of the simple protocol shown is that the entire document is being transferred between relying party and DSS (and back) using the client's web browser. Given the upload limitation of most client's internet connection, this might result in a bad end user experience when trying to sign a large document. So additionally we should define some form of artifact binding. Here the relying party sends the to be signed document via a SOAP DSS web service to the DSS. The DSS stores the document in some temporary document repository. The relying party receives back a document identifier which it passes as parameter when redirecting the client's web browser towards the DSS. At the end of the protocol flow, the relying party can fetch the signed document from the DSS web service using the document identifier.

Appendix C Examples (Non-Normative)

C.1 User Agent

C.1.1 Web Form of the SignRequest

The client sends a request to the Digital Signature Service via the HTTP agent (a web browser) by means of a HTML form.

```
<html>
<head>SignRequest</head>
<body>
  <form
    id="dss-request-form"
    method="post"
    action="http://localsig.digitalsignatureservice.co.uk">
    <input type="hidden" name="signrequest"
      value=
        "JVBERi0xLjYNJJeLjz9MNCjI4IDAgb2JqDTw8L0ZpbHRlcid
        ZW5ndGggMTY1L04gMi9UeXB1L09ialN0bT4+c3RyZWFT
        i4GBiYFBiYEZTDKBSUZGRrlJUHEgR72YAQz+/wcAlBAG
        eHJlZgo3NjA0MQolJUVPRgo=" />
    <input type="hidden" name="clienturl"
      value="http://localsig.digid.nl/654528644274424/" />
  </form>
  <script type="text/javascript">
    document.getElementById( 'dss-request-form' ).submit();
  </script>
</body>
</html>
```

C.1.2 Web Form of the SignResponse

The Digital Signature Service sends the result back to the client, via the HTTP agent (a web browser, by using the URL of the client `http://localsig.digid.nl/654528644274424/` in the HTML form. The URL was provided in the request by the attribute `clienturl`.

```
<html>
<head>SignResponse</head>
<body>
  <form
    id="dss-response-form"
    method="post"
    action="http://localsig.digid.nl/654528644274424/">
    <input type="hidden" name="signresponse"
      value=
        "eHJlZgo3NjA0MQolJUVPRgoDAGb2JqDTw8L0ZpbHRlcid
        i4GBiYFBiYEZTDKBSUZGRrlJUHEgR72YAQz+/wcAlBAG
        ZW5ndGggMTY1L04gMi9UeXB1L09ialN0bT4+c3RyZWFT
        JVBERi0xLjYNJJeLjz9MNCjI4I=" />
  </form>
  <script type="text/javascript">
    document.getElementById( 'dss-response-form' ).submit();
  </script>
</body>
</html>
```

C.2 Two-Step Approach

C.2.1 FIRST Request/Response

The client application initiates a `<dss:SignRequest>` to request the digest of the document.

```
<?xml version="1.0" encoding="utf-8"?>
<dss:SignRequest
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:localsig="http://docs.oasis-open.org/dss-x/ns/localsig"
  RequestID="example1-initial-request"
  Profile="http://docs.oasis-open.org/dss-x/ns/localsig">
  <dss:OptionalInputs>
    <dss:ServicePolicy>
      http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
    </dss:ServicePolicy>
    <localsig:RequestDocumentHash MaintainRequestState="true">
      <ds:DigestMethod
        Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
      </localsig:RequestDocumentHash>
    </dss:OptionalInputs>
    <dss:InputDocuments>
      <dss:Document>
        <dss:Base64Data MimeType="application/pdf">
          JVBERi0xLjYNJcLjz9MNCjI4IDAgb2JqD
          [...]
          eHJlZgo3NjA0MQolJUVPRgo=
        </dss:Base64Data>
      </dss:Document>
    </dss:InputDocuments>
  </dss:SignRequest>
```

The `<dss:SignResponse>` contains the `<localsig:CorrelationID>` to be used in the subsequent `<dss:SignRequest>`.

```
<?xml version="1.0" encoding="utf-8"?>
<dss:SignResponse
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  xmlns:localsig="http://docs.oasis-open.org/dss-x/ns/localsig"
  RequestID="example1-initial-request"
  Profile="http://docs.oasis-open.org/dss-x/ns/localsig">
  <dss:Result>
    <dss:ResultMajor>
      urn:oasis:names:tc:dss:1.0:resultmajor:Success
    </dss:ResultMajor>
    <dss:ResultMinor>
      urn:oasis:names:tc:dss:1.0:resultminor:documentHash
    </dss:ResultMinor>
  </dss:Result>
  <dss:OptionalOutputs>
    <localsig:CorrelationID>
      82962C67-F8D6-4480-A130-9280AB1F11A7
    </localsig:CorrelationID>
    <dss:DocumentHash>
      <dss:DocumentHash>
        <ds:DigestMethod
```

```

        Algorithm="http://www.w3.org/2001/04/xmlenc#sha256" />
        <ds:DigestValue>
            In6GUzH+gMFR5q4WpUTyPa+1b4s=
        </ds:DigestValue>
    </dss:DocumentHash>
</dss:DocumentHash>
</dss:OptionalOutputs>
</dss:SignResponse>

```

The client application uses the digest for the (secure) signature-creation device to obtain the signature.

C.2.2 SECOND Request/Response

The client application initiates a `<dss:SignRequest>` to provide the Digital Signature Service with the signed digest (a `<dss:SignatureObject>` element) and request for the final document.

```

<?xml version="1.0" encoding="utf-8"?>
<dss:SignRequest
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  xmlns:localsig="http://docs.oasis-open.org/dss-x/ns/localsig"
  RequestID="example1-final-request"
  Profile="http://docs.oasis-open.org/dss-x/ns/localsig">
  <dss:OptionalInputs>
    <dss:ServicePolicy>
      http://docs.oasis-open.org/dss-x/ns/localsig/two-step-approach
    </dss:ServicePolicy>
    <localsig:CorrelationID>
      82962C67-F8D6-4480-A130-9280AB1F11A7
    </localsig:CorrelationID>
    <dss:SignatureObject>
      <dss:Base64Signature>
        MIAGCSqGSIb3DQEHAqCAMIIRdQIBATE
        DQEHAaCCD74wggWAMIIEaKADAgECAg
        [...]
        DQEBAQUABEA3YkuiPSDVaAhaAza49UT
        DZWtCGVc0LCc5QR1BOc54ZrVGp6AA==
      </dss:Base64Signature>
    </dss:SignatureObject>
  </dss:OptionalInputs>
</dss:SignRequest>

```

The final `<dss:SignResponse>` contains the signed document.

```

<?xml version="1.0" encoding="UTF-8"?>
<dss:SignResponse
  xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
  RequestID="example1-final-request"
  Profile="http://docs.oasis-open.org/dss-x/ns/localsig" >
  <dss:Result>
    <dss:ResultMajor>
      urn:oasis:names:tc:dss:1.0:resultmajor:Success
    </dss:ResultMajor>
    <dss:ResultMinor>
      urn:oasis:names:tc:dss:1.0:resultminor:valid:signature:OnAllDocuments
    </dss:ResultMinor>
    <dss:ResultMessage lang="en-us" />
  </dss:Result>
  <dss:OptionalOutputs>
    <dss:DocumentWithSignature>

```

```
<dss:Document>
  <dss:Base64Data MimeType="application/pdf">
    JVBERi0xLjYNJeLjz9MNCjI4IDAgb2JqDTw
    [...]
    qmMAG///AYYzCwcKZW5kc3RyZWFTcmVCg==
  </dss:Base64Data>
</dss:Document>
</dss:DocumentWithSignature>
</dss:OptionalOutputs>
</dss:SignResponse>
```

Appendix D Acknowledgements (Non-Normative)

The following persons have participated in the creation of this specification and are gratefully acknowledged (in alphabetical order):

- Andreas Kuehne, Individual.
- Detlef Huehnlein, Individual.
- Ernst Jan van Nigtevecht, Sonnenglanz Consulting BV.
- Ezer Farhi, ARX
- Frank Cornelis, Fedict
- Juan Carlos Cruellas, Departamento de Arquitectura de Computadores, Univ Politecnica de Cataluna.
- Oscar Burgos, CATCert-Agencia Catalana de Certificacio.
- Pim van der Eijk, Sonnenglanz Consulting BV.
- Stefan Hagen, Individual.

Appendix E Revision History (Non-Normative)

Revision 0.1	13 Januari 2014	E.J. Van Nigtevecht
Creation of the CSPRD 01 based on the CSD 01.		
Revision 0.2	30 June 2014	E.J. Van Nigtevecht
Processed the comments on CSD 01; see " https://www.oasis-open.org/committees/document.php?document_id=53473&wg_abbrev=dss-x ". Renamed localsig:ReturnDocumentHash into localsig:RequestDocumentHash (in Section 3.2.1.1.2 in the code).		
Revision 0.3	13 October 2016	E.J. Van Nigtevecht
Corrected url in Section 1.3 under [LocalSigXSD].		
Revision 0.4	13 October 2016	E.J. Van Nigtevecht
Corrected the XSD that was published under CS01: changed element ReturnDocumentHash into element RequestDocumentHash. Updated comments in the XSD and the import statement for http://www.w3.org/2000/09/xmlsig# .		
Revision 0.5	13 October 2016	E.J. Van Nigtevecht
Added a description/indication for the use of the element AdditionalProfile in Section 3.1.1, Section 3.2.1 and Section 3.3.1.		
Revision 0.6	13 February 2017	E.J. Van Nigtevecht
Editorial updates for CSPRD03.		
Revision 0.7	6 March 2017	E.J. Van Nigtevecht
Editorial updates for CSPRD03 into CS02.		