PEPPOL
Pan-European Public Procurement Online

PEPPOL Deliverable D1.1
Requirements for Use of Signatures in Public Procurement Processes

Part 6: OASIS DSS Interface Specification

Extended Validation Profile – Profiling and Extensions Specification

Version 1.2

PEPPOL WP1 2009-04-30

Borderless eProcurement
Let's make it happen!

cip

Table of Contents

1 Introduction

1.1 Scope and Structure of Deliverable D1.1

This document is a part of the multi-part deliverable D1.1 "Requirements for Use of Signatures in the Procurement Processes" issued by the PEPPOL[1] (Pan-European Public Procurement On-Line) project. PEPPOL is a three-year (May 2008 – May 2011) large scale pilot under the CIP (Competitiveness and Innovation Programme) initiative of the European Commission.

D1.1 consists of the following documents:

Part 1: Background and Scope

Part 2: E-tendering Pilot Specifications

Part 3: Signature Policies

Part 4: Architecture and Trust Models

Part 5: XKMS v2 Interface Specification

Part 6: OASIS DSS Interface Specification

Part 7: eID and eSignature Quality Classification

The D1.1 deliverable is the first version of **functional specifications** for cross-border interoperability of e-signatures in Europe. The specifications are specifically targeted at cross-border public procurement, the topic of PEPPOL. However, if the resulting solution is successful it is believed that it will be applicable also to other application areas in need of e-signature interoperability.

Signature interoperability in PEPPOL focuses on verification of e-signatures and their associated eIDs. Interoperability of signing solutions is not handled as it is assumed that all actors are capable of signing documents within their corporate infrastructure.

The specifications guide the implementation, testing, and piloting of e-signature interoperability solutions to be done by PEPPOL. The specifications are publicly available and comments from any interested party are most welcome. Note that since the specifications of D1.1 by necessity will evolve as a result of further work in PEPPOL, any party using or referring to the specifications must ensure that the latest version is used; contact the PEPPOL project for information.

1.2 Scope and Evolution of this Document

Cross-border interoperability for verification of e-signatures requires more information than merely an assessment that the signature is valid. Signature validity is just one aspect of signature acceptance, which is governed by the signature policy in force (see D1.1 part 3).

PEPPOL specifies validation services and their interfaces. A validation service must be able to assess and return information related to signature policy adherence, which necessitates a richer interface than merely OCSP or CRL for revocation checking. Two interfaces are specified:

- XKMS v2 for eID certificate validation (part 5 of D1.1);
- OASIS DSS for verification of entire signed documents (this document).

[1] http://www.peppol.eu

The interface specified by this document provides an opportunity for external verification of entire signed documents (all signatures on a document). A verification service may be a technical service providing functionality only, or it may be an *authority* issuing signed responses that can be viewed as "notary" statements on validity of signatures. The following evolution of this document is expected:

- Further alignment with D1.1 part 5 (XKMS) should be done in order to align structure and semantics of statements about eIDs.
- The alignment may be to the extent that the XKMS structure is embedded into the DSS structure as the certificate validation part, DSS adding only the signature specific items and the overall (aggregated) status information.
- At present, a difference between the XKMS and DSS profiles is that XKMS assumes that the caller is able to process certificate content while for DSS certificate content may be returned as "respond with" parameters assuming that the caller does not need to do this processing.
- The specification should be promoted as an OASIS DSS standard profile. PEPPOL will consider submission and follow up to OASIS; this process will necessarily lead to changes in specifications.
- Changes due to experience gained in PEPPOL and due to comments from external sources must be expected.

1.3 Version, List of Contributors

Version 1.0	2009/02/11	Complete version for internal quality assurance.
Version 1.1	2009/02/27	Submitted to PEPPOL project management, approved with comments at project management meeting 2009/03/27.
Version 1.2	2009/04/30	For publication, updated according to comments.

The following organizations, in alphabetical order, have contributed to Deliverable D1.1.

- **bremen online services, Germany,** http://www.bos-bremen.de
- **CNIPA, Italy** http://www.cnipa.it
- **DGME, French Ministry of Finance** http://www.references.modernisation.gouv.fr/
- **DNV, Norway** http://www.dnv.com

The following persons (alphabetical ordering for each participating organization) have contributed to the work:

Jörg Apitzsch	bos	Uwe Trostheide	bos	Dr. Daniele Tatti	CNIPA
Markus Ernst (co-editor)	bos	Jens Wothe	bos	Mario Terranova	CNIPA
Mark Horstmann	bos	Martine Schiavo	DGME	Anette Andresen	DNV
André Jens	bos	Stefano Arbia	CNIPA	Dr. Leif Buene	DNV
Dr. Jan Pelz	bos	Giovanni Manca	CNIPA	Jon Ølnes (editor)	DNV
Marco von der Pütten	bos	Adriano Rossi	CNIPA		

1.4 Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]**Fehler! Verweisquelle konnte nicht gefunden werden.**.

This profile uses the following typographical conventions in text:

<Element>, **<ns:ForeignElement>**, **Attribute**, **Type**, **OtherCode**

```
Listings of Extended Validation profile schema like this.
```

1.5 References

[DNV01] A. Andresen, DNV Validation Authority XML Schema description, http://www.dnv.com/binaries/XML_schema%20description_v1.0_tcm4-243234.pdf

[DNV02] Ølnes, Jon et al.: Making Digital Signatures Work across National Borders. Paper published in Pohlmann, Reimer, Schneider: ISSE/Secure 2007, Securing Electronic Busines Processes, pp. 287-296, October 2007, ISBN 978-3-8348-0346-7.

[DSSCore] S.Drees et al., Digital Signature Service Core Protocols and Elements OASIS, February 2007, http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-spec-v1.0-os.html

[XKMS] XML Key Management Specification (XKMS 2.0) Version 2.0, W3C Recommendation, 28 June 2005, http://www.w3.org/TR/2005/REC-xkms2-20050628/

[XMLDSIG] World Wide Web Consortium. XML-Signature Syntax and Processing (Second Edition), W3C Recommendation, 10 June 2008; http://www.w3.org/TR/xmldsig-core/

[XMLSchema] World Wide Web Consortium. XML Schema, Parts 0, 1, and 2 (Second Edition). W3C Recommendation, 28 October 2004; http://www.w3.org/TR/xmlschema-0/, http://www.w3.org/TR/xmlschema-1/, and http://www.w3.org/TR/xmlschema-2/

[RFC2119] S.Bradner, Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119, March 1997

1.6 Namespaces

Prefix	XML Namespace	Specification
ds	**http://www.w3.org/2000/09/xmldsig#**	[XMLDSIG]
dss	**http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-schema-v1.0-os.xsd** **urn:oasis:names:tc:dss:1.0:core:schema**	[DSSCore]
evdss	**http://www.peppol.eu/....**	This document
xkms	**http://www.w3.org/2002/03/xkms#**	[XKMS]
xkmsDe2008	**http://www.peppol.eu/2008/12/xkmsExtDE#**	D1.1 part 5
xs	**http://www.w3.org/2001/XMLSchema**	[XMLSchema]

Table 1: Namespaces

2 Overview

This document defines a profile of the verifying protocol defined in “Digital Signature Services Core Protocol and Elements” [DSSCore]. The profile “Extended Validation profile” extends the verifying protocol with quality assessments and information regarding the signatures, the belonging certificates and the verification process.

The profile supports multi signature documents. However it only supports one document per request. The signature format used is not considered or limited by this profile; hence it should be possible to support any signature format.

The quality assessment for eID quality and signature quality is based on the quality assessment and classification scheme described in part 7 of the D1.1 deliverable.

Information about the signature and the belonging certificates is given in a new Optional Input / Optional Output. This is information retrieved from the different fields in a X509 certificate(s) and from the signature(s). For each signature verification the status of that particular verification is reported in addition to an overall status for all signature verifications.

The time of verification of the signatures and forming of the response are set as required in this profile.

The profile opens up for asynchronous communication. Communication through a gateway is also possible. A gateway may be beneficial e.g. when it is desirable to have all requests from a company towards a validation service, from all systems inside a company, go through a common channel towards the validation service. The gateway can then sign all requests on behalf of the company and common policies, e.g. quality levels, can be enforced by the gateway. The gateway can also remove document content to avoid sending the content to the validation service (for confidentiality reasons or simply to save bandwidth), forwarding only signature fields and corresponding hash values. The concept of a gateway is briefly discussed in [DNV02].

3 XML Quality concrete profile

3.1 Profile Features

3.1.1 Identifier

urn:oasis:names:tc:dss:1.0:profiled:extended-validation

3.1.2 Scope

This document profiles the DSS verifying protocol defined in [DSSCore]. This profile extends the core specification with quality assessment and more information in the response.

3.1.3 Relationship to Other Profiles

The profile in this document is based on the [DSSCore]. It MAY be used with other profiles of the [DSSCore].

3.2 Profile of Verifying Protocol

3.2.1 Element <VerifyRequest>

This clause profiles the **<VerifyRequest>** element.

3.2.1.1 Attribute RequestID

This attribute is REQUIRED.

The attribute **RequestID** SHOULD be a globally unique ID.

One solution for the content of the **RequestID** is to combine the **<Name>** element from **<RequesterIdentity>** with a 80-bit random number to make the ID approximately unique.

3.2.1.2 Attribute Profile

The attribute is REQUIRED.

urn:oasis:names:tc:dss:1.0:profiles:extended-validation

3.2.1.3 Element <OptionalInputs>

3.2.1.3.1 New Optional Inputs

3.2.1.3.1.1 Element <QualityLevelRequirements>

This element is REQUIRED.

In cases where there are no requirements regarding quality, the requested quality SHALL be set to 0.

The element **<QualityLevelRequirements>** is based on the type **QualityLevelType**. Information about the quality assessment and a complete description is found in part 7 of the D1.1 deliverable, and enumerations are defined in part 5 of D1.1.

The **QualityLevelType** has the following child elements:

<CertificateQuality> [REQUIRED]

This element contains the minimum required certificate quality level.

<IndependentAssurance> [REQUIRED]

This element is the minimum required independent assurance level.

<HashAlgQuality> [REQUIRED]

This element contains the minimum accepted hash algorithm quality level.

<PublicKeyAlgQuality> [REQUIRED]

This element contains the minimum accepted public key algorithm quality level.

```
<!-- QualityLevelRequirements -->
<xs:element name="QualityLevelRequirements" type="QualityLevelType"/>
<xs:complexType name="QualityLevelType">
 <xs:sequence>
  <xs:element name="CertificateQuality" type="xs:string"/>
  <xs:element name="IndependentAssurance" type="xs:string"/>
  <xs:element name="HashAlgQuality" type="xs:string"/>
  <xs:element name="PublicKeyAlgQuality" type="xs:string"/>
 </xs:sequence>
</xs:complexType>
```

3.2.1.3.1.2 Element <RespondWith>

The **<RespondWith>** element is REQUIRED. The values **Subject**, **IssuerName** and **CertificateSerialNumber** are the minimum of the **RespondWithEnum** values that MUST be used.

```
<!-- RespondWith -->
<xs:element name="RespondWith" type="RespondWithEnum"/>
<xs:simpleType name="RespondWithEnum">
 <xs:restriction base="xs:string">
  <xs:enumeration value="Subject" />
  <xs:enumeration value="IssuerName"/>
  <xs:enumeration value="CertificateSerialNumber"/>
  <xs:enumeration value="KeyValue"/>
  <xs:enumeration value="HashAlgorithm"/>
  <xs:enumeration value="X509CertificateChain"/>
  <xs:enumeration value="X509CRL"/>
  <xs:enumeration value="SKI"/>
  <xs:enumeration value="OCSP"/>
  <xs:enumeration value="Timestamp"/>
  <xs:enumeration value="SignHash"/>
  <xs:enumeration value="ContentHash"/>
  <xs:enumeration value="Content"/>
  <xs:enumeration value="KeyUsage"/>
  <xs:enumeration value="ExtendedKeyUsage"/>
  <xs:enumeration value="BasicConstraints"/>
  <xs:enumeration value="ValidFrom"/>
  <xs:enumeration value="ValidTo"/>
  <xs:enumeration value="CRLUrl"/>
  <xs:enumeration value="CRLNumber"/>
 </xs:restriction>
</xs:simpleType>
```

The meanings of the different **RespondWithEnum** values are listed in the table below.

RespondWithEnum	Required / Optional	Meaning
Subject	REQUIRED	This is the distinguished name (DN) of the holder of the certificate (subject field in a certificate)
IssuerName	REQUIRED	This is the DN of the certificate issuer
CertificateSerialNumber	REQUIRED	This is the serial number in the certificate
KeyValue	OPTIONAL	This is the certificate holder's public key
HashAlgorithm	OPTIONAL	For a signature verification, this hash algorithm is extracted from the signature, while in a certificate validation this hash algorithm is found in the signature algorithm field in the certificate
X509CertificateChain	OPTIONAL	This is the certificate chain for the certificate
SKI	OPTIONAL	The SKI (Subject Key Identifier) is the SKI from the certificate. This is an extension in the certificate, and it provides a means of identifying certificates that contain a particular public key
KeyUsage	OPTIONAL	This is the key usage from the certificate
ExtendedKeyUsage	OPTIONAL	This is the extended key usage from the certificate
BasicConstraints	OPTIONAL	This is the basic constraints for the certificate.
ValidFrom	OPTIONAL	The certificate is valid from this date
ValidTo	OPTIONAL	The certificate is valid to this date
SignHash	OPTIONAL	This is the decrypted hash of the signature
ContentHash	OPTIONAL	This is the hash of the content sent in the request
Content	OPTIONAL	This is the content sent in the request
X509CRL	OPTIONAL	This is the CRL used to validate the end user's certificate. If other methods than CRLs are used for validation of certificates, this value will be N/A. Note: The return of CRLs may reduce the response time of a request due to the size of the CRL

RespondWithEnum	Required / Optional	Meaning
OCSP	OPTIONAL	If OCSP is used in the validation of a certificate, the entire OCSP response is given here. If OCSP is not used, this value will be N/A
CRLUrl	OPTIONAL	This is the URL from which the CRL was downloaded
CRLNumber	OPTIONAL	This is the CRLNumber of the CRL used for validation of the certificate
Timestamp	OPTIONAL	This is the timestamp of a signature. If a time stamp is not used, the value will be N/A

3.2.1.3.1.3 Element <GatewayRequester>

To support the use of a gateway forwarding requests to a validation authority (VA) service the element **<GatewayRequester>** has been defined. The element uses the **IdentityType** type which is based on the **<RequesterIdentity>** element. The **<GatewayRequester>** element is OPTIONAL, but it is REQUIRED when a gateway is being used to forward a request to a VA service.

<Name> [REQUIRED]

This is the name of the gateway the verification request is sent through. This element is of the type **saml:NameIdentifierType**. This SHOULD be filled with the CN from the gateway's client authentication certificate used in the two-way SSL connection.

<SupportingInfo> [OPTIONAL]

This element MAY include an URI to the requester or responder to support an asynchronous version of this protocol.

```
<!-- GatewayRequesterIdentity -->
<xs:element name="GatewayRequesterIdentity" type="IdentityType"/>
<xs:complexType name="IdentityType">
 <xs:sequence>
  <xs:element name="Name" type="saml:NameIdentifierType"/>
  <xs:element name="SupportingInfo" type="dss:AnyType" minOccurs="0"/>
 </xs:sequence>
</xs:complexType>
```

3.2.1.3.2 Optional Inputs Already Defined in the Core

The below are Optional Inputs already defined in the core that this profile is using.

3.2.1.3.2.1 Optional Input <UseVerificationTime>

The element **<UseVerificationTime>** is defined in [DSSCore]. The element consists of two child elements; **<CurrentTime>** and **<SpecificTime>**. All elements are OPTIONAL. When the **<UseVerificationTime>** element is not used it is up to the server's policy to determine what time to use, e.g. current time or the time stamp in the SDO.

3.2.1.3.2.2 Optional Input <ReturnVerificationTimeInfo>

The element is defined in [DSSCore]. This element is made REQUIRED in this profile.

3.2.1.4 Element <InputDocuments>

This profile supports only one document in one request, but the document may have an unlimited number of signatures. The document is added using an appropriate child element of the **<InputDocuments>** element following the definitions in [DSSCore]. This element is REQUIRED in this profile. Instead of sending the entire document, a document hash and the signatures may be used.

3.2.1.5 Element <SignatureObject>

The **<SignatureObject>** element is only used in relation with the **<DocumentHash>** element in **<InputDocuments>**.

3.2.1.6 Other Core Elements

3.2.1.6.1 Optional Input <RequesterIdentity>

The element **<RequesterIdentity>** defined in the [DSSCore] is used to identify the requester of the verification. The element is REQUIRED in this profile.

The child elements are as follow:

<Name> [REQUIRED]

This is the name of the requester who requested the verification. This element SHOULD be the same as the CN in the client authentication certificate used in the security bindings.

<SupportingInfo> [OPTIONAL]

This element may be used in an asynchronous communication between the client and the server. This element will then contain a respond address element

3.2.2 Element <VerifyResponse>

3.2.2.1 Attribute RequestID

This attribute is REQUIRED in this profile. The **RequestID** must be the same as the **RequestID** in the request.

3.2.2.2 Attribute Profile

The element is REQUIRED in this profile.

3.2.2.3 Element <Result>

The values defined for **<ResultMajor>** in the core specification are used as specified there.

3.2.2.3.1 Element <ResultMinor>

It is not defined any new **<ResultMinor>** codes for this profile at the moment. Refinements of some of the codes are given below.

For the **<ResultMajor>** code **Success** the following **<ResultMinor>** codes are specified with example usage:

- **IncorrectSignature**: This code should be used when one of the signatures fails to verify and the **OverallAssertionStatus** attribute in the **<ContentVerifyInfo>** element is **NotTrusted**.
- **ValidMultiSignatures**: This code should be used when the signature verification are valid for all signatures

For the **<ResultMajor>** code **InsufficientInformation** the following **<ResultMinor>** codes are specified with example usage:

- **CrlNotAvailable**: This code is used if the CRL is unavailable during verification of any of the signatures resulting in an **OverallAssertionStatus** attribute in the **<ContentVerifyElement>** set to Indeterminate
- **OcspNotAvailable**: This code is used if the OCSP responder is unavailable during verification of any of the signatures resulting in an **OverallAssertionStatus** attribute in the **<ContentVerifyElement>** set to **Indeterminate**

3.2.2.4 Element <OptionalOutputs>

3.2.2.4.1 New OptionalOutputs

3.2.2.4.1.1 Element <ResponderIdentity>

The element **<ResponderIdentity>** is required in a response from a verification service using this profile. The element is of type **IdentityType**.

<Name> [REQUIRED]

This element SHOULD be populated with the CN of the service's signing certificate or the SSL Server authentication certificate used in the SSL connection.

<SupportingInfo> [OPTIONAL]

Not in use in this profile.

```
<!-- Responderidentity -->
<xs:element name="ResponderIdentity" type="IdentityType"/>
<xs:complexType name="IdentityType">
 <xs:sequence>
  <xs:element name="Name" type="saml:NameIdentifierType"/>
  <xs:element name="SupportingInfo" type="dss:AnyType" minOccurs="0"/>
 </xs:sequence>
</xs:complexType>
```

3.2.2.4.1.2 Element <QualityLevel>

The different quality levels SHALL always be returned when using the extended validation profile of DSS, hence the **<QualityLevel>** element is REQUIRED. The **<QualityLevel>** is of type **QualityLevelType**. The **<QualityLevel>** element is used in the **<VerifyInfo>** element described in 3.2.2.4.1.4.1. The **QualityLevel** consist of the following:

<CertificateQuality> [REQUIRED]

This is the actual quality of the certificate(s) used for signing.

<IndependentAssurance> [REQUIRED]

This is the independent assurance level of the certificate(s) used for signing.

<HashAlgQuality> [REQUIRED]

This is the quality of the hash algorithm used

<PublicKeyAlgQuality> [REQUIRED]

This is the quality of the public key algorithm used

```
<!-- QualityLevel -->
<xs:element name="QualityLevel" type="QualityLevelType"/>
<xs:complexType name="QualityLevelType">
 <xs:sequence>
  <xs:element name="CertificateQuality" type="xs:string"/>
  <xs:element name="IndependentAssurance" type="xs:string"/>
  <xs:element name="HashAlgQuality" type="xs:string"/>
  <xs:element name="PublicKeyAlgQuality" type="xs:string"/>
 </xs:sequence>
</xs:complexType>
```

3.2.2.4.1.3 Element <ResponseItem>

The **<ResponseItem>** element is of type **ResponseItemType**, and it is REQUIRED in this profile. This Optional Output is used in the **<VerifyInfo>** element. This element returns all the requested **RespondWithEnum** parameters. The **<ResponseItem>** element consists of the following elements and attribute:

<IntValue> [OPTIONAL]

The **<IntValue>** element is of type **integer** defined in [XMLSchema]. This element may be used for any **RespondWithEnum** item that has an integer value as result.

<Value> [OPTIONAL]

The **<Value>** element is of type **string** defined in [XMLSchema]. This element is for future use, and it can have multiple values against one **RespondWithEnum** item.

<Base64Value> [OPTIONAL]

The **<Base64Value>** element is of type **base64Binary** defined in [XMLSchema]. This is used when the result of the **RespondWithEnum** item can be returned in base64 format. This value is for example used when the **<ResponseItem> Id** is “KeyValue”.

<StringValue> [OPTIONAL]

The **<StringValue>** element is of type **string** defined in [XMLSchema]. This is used when the **ResponseItem** element is presented in string format.

Id [Optional]

The **Id** attribute is of type **RespondWithEnum**, and it is an optional attribute. This attribute is used to identify the specified **<ResponseItem>**. See the table in 3.2.1.3.1.2 for a description of the different id’s.

```
<!-- ResponseItem -->
<xs:element name="ResponseItem" type="ResponseItemType"/>
<xs:complexType name="ResponseItemType">
 <xs:choice>
  <xs:element name="IntValue" type="xs:integer"/>
  <xs:element name="Value" type="xs:string" maxOccurs="unbounded"/>
  <xs:element name="Base64Value" type="xs:base64Binary"
maxOccurs="unbounded"/>
  <xs:element name="StringValue" type="xs:string"/>
 </xs:choice>
 <xs:attribute name="Id" type="RespondWithEnum"/>
</xs:complexType>
```

3.2.2.4.1.4 Element <ContentVerifyInfo>

The **<ContentVerifyInfo>** element is used to return information about all the signatures and the belonging certificates used in the signed document. The element is REQUIRED in a response using this profile. The element is of type **ContentVerifyInfoType** and it consists of the following child elements and attribute:

<VerifyInfo> [REQUIRED]

The **<VerifyInfo>** element is described in more detail in 3.2.2.4.1.4.1 .

<Reason> [OPTIONAL]

The **<Reason>** element is only used when the **OverallAssertionStatus** attribute is set to either **NotTrusted** or **Indeterminate**. The element will point to the signature(s) that failed verification, and this MAY be done using the **Id** attribute in the **<VerifyInfo>** element like this:

<Reason> ID1, ID3 </Reason> - where the first and third signatures fails verification. ID1 and ID3 are retrieved from the **Id** attribute in the different **<VerifyInfo>** elements used in the specific verification transaction.

OverallAssertionStatus [REQUIRED]

The **OverallAssertionStatus** attribute is of type **ContentStatusEnum**. This attribute provides the overall result of the signature verification of the signed document. The table below lists the different status messages and their meaning.

```
<!-- ContentStatusEnum -->
<xs:simpleType name="ContentStatusEnum">
 <xs:restriction base="xs:string">
  <xs:enumeration value="Trusted"/>
  <xs:enumeration value="NotTrusted"/>
  <xs:enumeration value="Indeterminate"/>
 </xs:restriction>
</xs:simpleType>
```

ContentStatusEnum	Meaning
Trusted	The overall result is trusted. All signature verifications succeeded.
NotTrusted	The overall result is not trusted. One or more of the signatures failed verification.
Indeterminate	The overall result is indeterminate. The verification process could not determine if this is a trusted or not trusted request.

The following rules apply regarding the usage of **OverallAssertionStatus**:

- If any of the signatures have the attribute **AssertionStatus** set to **Invalid**, the **OverallAssertionStatus** SHALL be set to **NotTrusted**
- If one or more **Indeterminate** signatures are present, and none **Invalid**, the OverallAssertionStatus attribute SHALL be set to **Indeterminate**
- If one or more **InsufficientQuality** signatures are present, and none **Invalid** and **Indeterminate**, the **OverallAssertionStatus** attribute SHALL be set to **NotTrusted**.

- If all signatures have the attribute **`AssertionStatus`** set to **`Valid`**, the **`OverallAssertionStatus`** SHALL be set to **`Trusted`**

```
<!-- ContentVerifyInfo -->
<xs:element name="ContentVerifyInfo" type="ContentVerifyInfoType"/>
 <xs:complexType name="ContentVerifyInfoType">
  <xs:sequence>
   <xs:element ref="VerifyInfo" maxOccurs="unbounded"/>
   <xs:element name="Reason" type="xs:string" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="OverallAssertionStatus" type="ContentStatusEnum"
use="required"/>
 </xs:complexType>
```

3.2.2.4.1.4.1 Element <VerifyInfo>

The **`<VerifyInfo>`** element is of type **`VerifyInfoType`**, and this is a REQUIRED element in the extended validation profile of DSS. Each signature in the signed document will have its own **`<VerifyInfo>`** element, and this element will be returned as many times as there are signatures on the signed document. The **`<VerifyInfo>`** consist of the following elements and attributes:

`<QualityLevel>` [REQUIRED]

See 3.2.2.4.1.2 for a detailed explanation of this element.

`<ResponseItem>` [REQUIRED]

See 3.2.2.4.1.3 for a detailed explanation of this element.

`<FailureReason>` [OPTIONAL]

The **`<FailureReason>`** element will be used if the **`AssertionStatus`** attribute is either **`Invalid`**, **`Indeterminate`** or **`InsufficientQuality`**. The element SHOULD return a code explaining the reason for failure.

`Id` [Required]

The **`Id`** attribute is used to identify the different signatures used on the signed document, hence it is the id of the signature.

`AssertionStatus` [Required]

The **`AssertionStatus`** element gives the status for every signature verification.

```
<!-- SignatureStatusEnum -->
<xs:simpleType name="SignatureStatusEnum">
 <xs:restriction base="xs:string">
  <xs:enumeration value="Valid"/>
  <xs:enumeration value="Invalid"/>
  <xs:enumeration value="Indeterminate"/>
  <xs:enumeration value="InsufficientQuality"/>
 </xs:restriction>
</xs:simpleType>
```

SignatureStatusEnum	Meaning
`Valid`	The signature is valid.

SignatureStatusEnum	Meaning
Invalid	The signature is invalid. The failure reason will be given in the **<FailureReason>** element
Indeterminate	It is not possible to determine the status of the signature due to for example no available CRL or OCSP responder.
InsufficientQuality	The signature is valid, but the quality of either the certificate used and/or the signature algorithms (hash and public key) are lower than the requested quality.

```
<!-- VerifyInfo -->
<xs:element name="VerifyInfo" type="VerifyInfoType"/>
<xs:complexType name="VerifyInfoType">
 <xs:sequence>
  <xs:element ref="QualityLevel"/>
  <xs:element ref="ResponseItem" maxOccurs="unbounded"/>
  <xs:element name="FailureReason" type="xs:string" minOccurs="0"/>
 </xs:sequence>
 <xs:attribute name="Id" type="xs:string" use="required"/>
 <xs:attribute name="AssertionStatus" type="SignatureStatusEnum"
use="required"/>
</xs:complexType>
```

3.2.2.4.2 Optional Outputs Already Defined in the Core

3.2.2.4.2.1 Optional Output <VerificationTimeInfo>

The element **<VerificationTimeInfo>** is defined in [DSSCore]. This element is made required in this profile. The **<VerificationTimeInfo>** consists of the two elements **<VerificationTime>** and **<AdditionalTimeInfo>**. The **<VerificationTime>** is REQUIRED and it is also used here as described in the [DSSCore]. The **<AdditionalTimeInfo>** element is made REQUIRED as well in this profile. This element shall be populated with the time the response message was formed while the **<VerificationTime>** is the time of verification.

New value for the **Type** attribute in **<AdditionalTimeInfo>**:

urn:oasis:names:tc:dss:1.0:additionaltimeinfo:responseTimeInstant

The **Ref** attribute are not used in this profile.

3.3 Profile Bindings

3.3.1 Transport Bindings

This profile SHOULD support both HTTP POST and SOAP 1.2 transport bindings as defined in [DSSCore].

3.3.2 Security Bindings

3.3.2.1 Security Requirements

The profile MUST authenticate the requester to the DSS Server and the DSS Server to the requester. The DSS server supporting this profile MUST support signed requests and responses to and from the

service to maintain the integrity of the request and response, but only signed responses are mandatory to use.

3.3.2.2 TLS X.509 Mutual Authentication

This profile is secured using the TLS X.509 Mutual Authentication Binding defined in [DSSCore].

A.1 XML Schema

The extended validation profile of the OASIS DSS version 1.0 is attached here.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:oasis:names:tc:dss:1.0:profiles:VA:schema#"
xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
targetNamespace="urn:oasis:names:tc:dss:1.0:profiles:VA:schema#"
elementFormDefault="qualified" attributeFormDefault="unqualified">
   <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="http://www.w3.org/XML/1998/namespace"
schemaLocation="http://www.w3.org/2001/xml.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:import namespace="urn:oasis:names:tc:SAML:1.0:assertion"
schemaLocation="http://www.oasis-
open.org/committees/download.php/3408/oasis-sstc-saml-schema-protocol-
1.1.xsd"/>
   <!-- QualityLevel -->
   <xs:element name="QualityLevelRequirements" type="QualityLevelType"/>
   <xs:element name="QualityLevel" type="QualityLevelType"/>
   <xs:complexType name="QualityLevelType">
      <xs:sequence>
         <xs:element name="CertificateQuality" type="xs:string"/>
         <xs:element name="IndependentAssurance" type="xs:string"/>
         <xs:element name="HashAlgQuality" type="xs:string"/>
         <xs:element name="PublicKeyAlgQuality" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>
   <!-- RespondWith -->
   <xs:element name="RespondWith" type="RespondWithEnum"/>
   <xs:simpleType name="RespondWithEnum">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Subject"/>
         <xs:enumeration value="IssuerName"/>
         <xs:enumeration value="CertificateSerialNumber"/>
         <xs:enumeration value="KeyValue"/>
         <xs:enumeration value="HashAlgorithm"/>
         <xs:enumeration value="X509CertificateChain"/>
         <xs:enumeration value="X509CRL"/>
         <xs:enumeration value="SKI"/>
         <xs:enumeration value="OCSP"/>
         <xs:enumeration value="Timestamp"/>
         <xs:enumeration value="SignHash"/>
         <xs:enumeration value="ContentHash"/>
         <xs:enumeration value="Content"/>
         <xs:enumeration value="KeyUsage"/>
         <xs:enumeration value="ExtendedKeyUsage"/>
         <xs:enumeration value="BasicConstraints"/>
         <xs:enumeration value="ValidFrom"/>
         <xs:enumeration value="ValidTo"/>
         <xs:enumeration value="CRLUrl"/>
         <xs:enumeration value="CRLNumber"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:element name="ResponseItem" type="ResponseItemType"/>
   <xs:complexType name="ResponseItemType">
      <xs:choice>
```

```
         <xs:element name="IntValue" type="xs:integer"/>
         <xs:element name="Value" type="xs:string" maxOccurs="unbounded"/>
         <xs:element name="Base64Value" type="xs:base64Binary"
maxOccurs="unbounded"/>
         <xs:element name="StringValue" type="xs:string"/>
      </xs:choice>
      <xs:attribute name="Id" type="RespondWithEnum"/>
   </xs:complexType>
   <!-- Identity -->
   <xs:element name="ResponderIdentity" type="IdentityType"/>
   <xs:element name="GatewayRequesterIdentity" type="IdentityType"/>
   <xs:complexType name="IdentityType">
      <xs:sequence>
         <xs:element name="Name" type="saml:NameIdentifierType"/>
         <xs:element name="SupportingInfo" type="dss:AnyType"
minOccurs="0"/>
      </xs:sequence>
   </xs:complexType>
   <!--ContentVerifyInfo -->
   <xs:simpleType name="SignatureStatusEnum">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Valid"/>
         <xs:enumeration value="Invalid"/>
         <xs:enumeration value="Indeterminate"/>
         <xs:enumeration value="InsufficientQuality"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="ContentStatusEnum">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Trusted"/>
         <xs:enumeration value="NotTrusted"/>
         <xs:enumeration value="Indeterminate"/>
      </xs:restriction>
   </xs:simpleType>
   <xs:element name="VerifyInfo" type="VerifyInfoType"/>
   <xs:complexType name="VerifyInfoType">
      <xs:sequence>
         <xs:element ref="QualityLevel"/>
         <xs:element ref="ResponseItem" maxOccurs="unbounded"/>
         <xs:element name="FailureReason" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="Id" type="xs:string" use="required"/>
      <xs:attribute name="AssertionStatus" type="SignatureStatusEnum"
use="required"/>
   </xs:complexType>
   <xs:element name="ContentVerifyInfo" type="ContentVerifyInfoType"/>
   <xs:complexType name="ContentVerifyInfoType">
      <xs:sequence>
         <xs:element ref="VerifyInfo" maxOccurs="unbounded"/>
         <xs:element name="Reason" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="OverallAssertionStatus" type="ContentStatusEnum"
use="required"/>
   </xs:complexType>
</xs:schema>
```