Draft ETSI TS 119 432 V0.0.5 (2018-07)

Electronic Signatures and Infrastructures (ESI); Protocols for remote digital signature creation

STABLE DRAFT

Send comments ONLY to E-SIGNATURES_COMMENTS@list.etsi.org

by 31 August 2018

CAUTION: This DRAFT document is provided for information and is for future development work within the ETSI Technical Committee ESI only. ETSI and its Members accept no liability for any further use/implementation of this Specification. Approved and published specifications and reports shall be obtained exclusively via the ETSI Documentation Service at http://www.etsi.org/standards-search

Reference
DTS/ESI-0019432

Keywords
e-commerce, electronic signature, public key, security, trust services, remote signature, protocol

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-préfecture de Grasse (06) N° 7803/88

Important notice

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This draft Technical Specification (TS) has been produced by ETSI Technical Committee Electronic Signatures and Infrastructures (ESI).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

Standards for digital signatures have generally been developed for a long time considering solutions tailored to the characteristics of devices such as desktop computers and laptops where all signature processing was done in one system locally to the user. These traditional signature solutions assume that the signer uses smart cards or tokens to create any required digital signatures. Given developments in distributed systems, cloud computing, mobile equipment and related technologies, solutions have been emerging in the last few years where the process of digital signature creation and construction of AdES format is done in a distributed way with different steps of the process carried out by different systems/services that may be controlled by different actors.

The present document specifies protocols and interfaces for components providing specific functionalities as part of a process for remote digital signatures creation and construction of AdES formats. The present document aims at supporting electronic signatures and electronic seals, including qualified electronic signatures and qualified electronic seals according to the Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC (the eIDAS Regulation).

1 Scope

The present document specifies protocols and interfaces applicable when the process of creating AdES digital signatures as defined by ETSI EN 319 102-1 [4] and/or Digital Signature Values, as result of Data To Be Signed Representations signatures, is carried out by a distributed solution comprised of two or more systems/services/components.

The present document is limited to remote server signing, i.e. the signing key is held in a remote shared service.

NOTE 1: Remote signature creation with local signing, i.e. the signing key is held with the signer's personal device but other steps in the signature creation are carried out by means of networked services, is a possible solution but protocols for such architecture are not covered in the present document.

2 References

2.1 Normative references

EDITORIAL NOTE: TO BE COMPLETED AND REVIEWED

References are either specific (identified by date of publication and/or edition number or version number) or nonspecific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] Cloud Signature Consortium Standard: "Architectures, Protocols and API Specifications for Remote Signature applications. Public pre-release version 0.1.7.9".

[2] OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 – Working Draft 05.

[3] OASIS Standard: Advanced Electronic Signature Profiles of the OASIS Digital Signature Service Version 2.0 – Working Draft 02.

[4] ETSI EN 319 102-1: "Electronic signatures and infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".

[5] OASIS Standard: Asynchronous Processing Abstract Profile of the OASIS Digital Signature Services Version 1.0.

[6] CEN EN 419 241-1: "Trustworthy Systems supporting Server Signing; Part 1: General System Security Requirements".

2.2 Informative references

EDITORIAL NOTE: TO BE COMPLETED AND REVIEWED

References are either specific (identified by date of publication and/or edition number or version number) or nonspecific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC.

[i.2] ETSI SR 019 020: "The framework for standardization of signatures; Standards for AdES digital signatures in mobile and distributed environment".

[i.3] ETSI TS 119 441: "Electronic signatures and infrastructures (ESI); Policy and security requirements for trust service providers providing AdES digital signature validation services".

[i.4] ETSI TS 119 442: "Electronic signatures and infrastructures (ESI); Protocol profiles for trust service providers providing AdES digital signature validation services".

[i.5] ETSI EN 319 122-1: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 1: Building blocks and CAdES baseline signatures".

[i.6] ETSI EN 319 122-2: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 2: Extended CAdES signatures".

[i.7] ETSI EN 319 132-1: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 1: Building blocks and XAdES baseline signatures".

[i.8] ETSI EN 319 132-2: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 2: Extended XAdES signatures".

[i.9] ETSI EN 319 142-1: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 1: Building blocks and PAdES baseline signatures".

[i.10] ETSI EN 319 142-2: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 2: Additional PAdES signatures profiles".

[i.11] ETSI TR 119 001: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures; Definitions and abbreviations".

[i.12] ETSI TS 119 312: "Electronic Signatures and Infrastructures (ESI); Cryptographic Suites".

[i.13] IETF RFC 6749: " The OAuth 2.0 Authorization Framework".

3 Definitions and abbreviations

3.1 Definitions

EDITORIAL NOTE: TO BE COMPLETED AND REVIEWED

For the purposes of the present document, the terms and definitions given in ETSI TR 119 001 [i.11] and the following apply:

AdES (digital) signature: digital signature that is either a CAdES signature, or a PAdES signature or a XAdES signature

digital signature: data appended to, or a cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

digital signature value: result of the cryptographic transformation of a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient

remote signature creation device: signature creation device used remotely from signer perspective and applying the signature activation protocol to provide control of signing operation on its behalf and guarantees with a high level of confidence that the signing keys are used under sole control of the signer

server signing application: application using a remote signature creation device to create a digital signature value on behalf of a signer

server signing application service component: TSP service component employing a server signing application

server signing application service provider: TSP operating a server signing application service component

signature creation application: application within the signature creation system that creates the AdES digital signature and relies on the SCDev to create a digital signature value

NOTE: The SCDev can be managed by the SSASC.

signature creation application service component: TSP service component employing a signature creation application

signature creation application service provider: TSP operating a signature creation application service component

signature creation constraint: criteria used when creating a digital signature

signature creation policy: set of **signature creation constraints** processed or to be processed by the SCA

signature creation service: TSP service implementing a signature creation application and / or a server signing application

signature creation service provider: a service provider offering a signature creation service

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CSC	Cloud Signature Consortium
DTBS	Data To Be Signed
DTBSF	Data To Be Signed (Formatted)
DTBSR	Data To Be Signed Representation
DSV	Digital Signature Value
SAM	Signature Activation Module
SCA	Signature Creation Application
SCASC	Signature Creation Application Service Component
SCASP	Signature Creation Application Service Provider
SCDev	Signature Creation Device (SCDev)
SCS	Signature Creation Service
SCSP	Signature Creation Service Provider
SD	Signer's Document
SDO	Signed Data Object
SDOC	Signed Data Object Composer
SDR	Signer's Document Representation
SSA	Server Signing Application
SSASC	Server Signing Application Service Component
SSASP	Server Signing Application Service Provider

4 Signature creation process, service decomposition

4.1 Signature creation process steps and data elements

Figure 1 below (derived from ETSI EN 319 102-1 [4] section 4.2.1) shows the various steps and the related data elements for a signature creation process. For remote signature creation, different steps of this process are carried out according to a decomposition into several components, which shall have access to or make available the corresponding data elements. The process illustrated in the figure below is limited to the buildings blocks and information needed for creating a signature without taking in consideration issues such as signer authentication, authorization to the signing key usage, signing certificate availability.

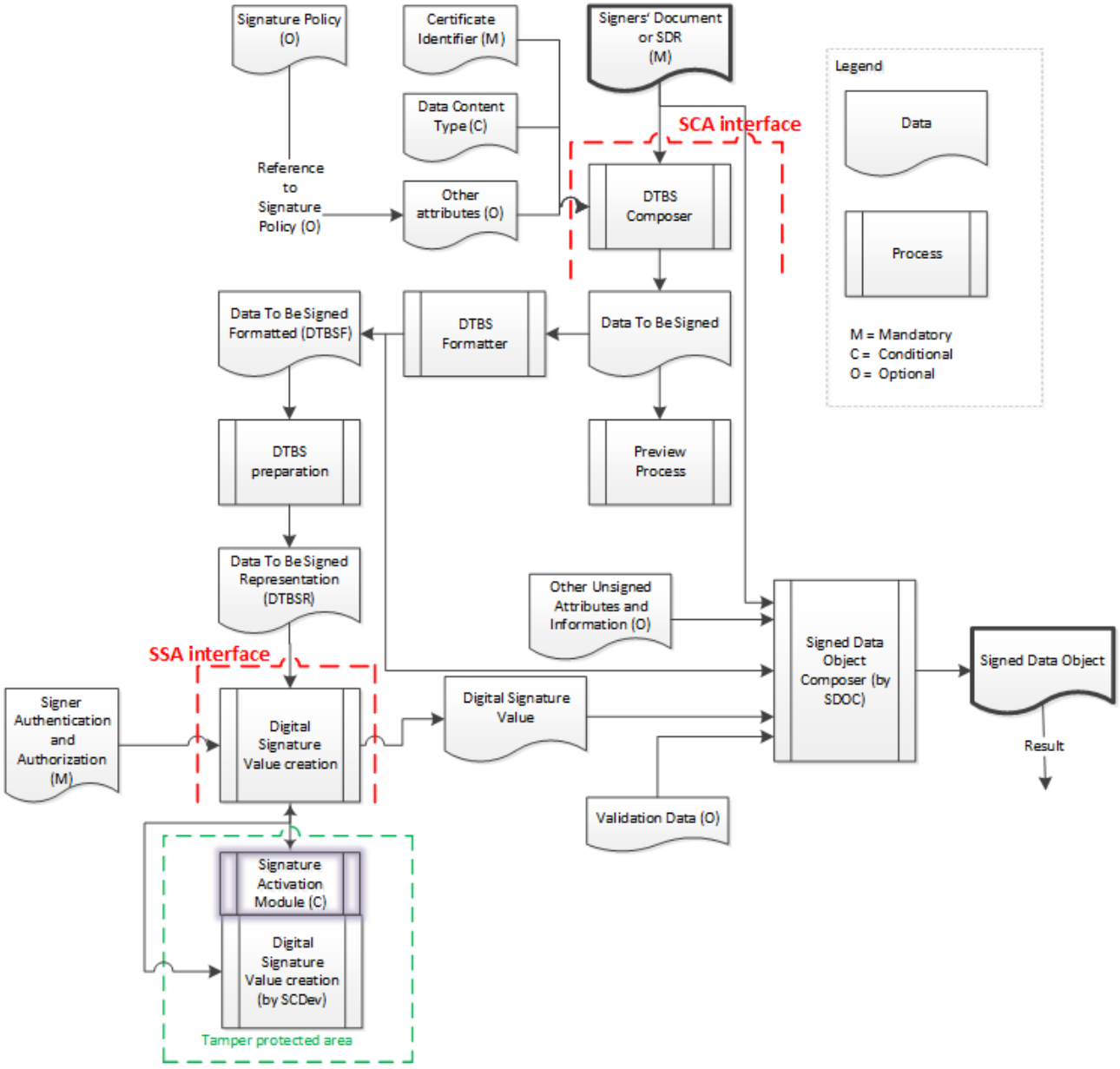


Figure 1: Process Steps and Data Elements in Signature Creation

4.2 Service main components and interfaces

The above process points out scenarios where the AdES and/or Digital Signature Value (DSV) are created using a signing key held within a cryptographic security module named Signature Creation Device (SCDev) operated by a Signature Creation Service Provider (SCSP).

Based on the different types of data managed in requests and responses, two main components can be identified in the above schema providing different interfaces for signing management: the Server Signing Application Service Component (SSASC) and the Signature Creation Application Service Component (SCASC) defined below.

The SSASC is the component supporting digital signature values creation. The SSASC is able to interact with the SCDev holding the signer's private key. When the SSASC uses the SCDev, the authorized signer is able to control the signing key with a certain level of confidence.

The SSASC interface has the Data To Be Signed Representation (DTBSR) and other parameters as main input and the digital signature value as main output.

The SCASC is the component supporting AdES digital signature creation and carrying out several specific parts of the signature creation process. The SCASC is able to interact with the SSASC for requesting digital signature values creation.

The SCASC interface has the document(s) to be signed (SD) or its (their) representation (SDR) and other parameters as main input and the signed document(s) or the digital signature(s) as main output.

By Signature Creation Service we denote TSP service implementing a signature creation application and / or a server signing application.

Some variants of these interface are possible depending on the functional split between the Signature Creation Service and the signer's local system.

The following clauses specify main information objects and processes in SCASC and SSASC.

4.3 Signature Creation Application

4.3.1 Signer's document and hashing

The signature creation process starts with the signer's document (SD), which is to be signed. The SD is represented (SDR) by a hash value in the Data To Be Signed (DTBS). The following observations are made:

- The creation of the SDR (the hashing) can be done where the SD is stored or by the SCASC. In the former case, the SDR shall be transferred to the SCASC while in the latter case, the SD shall be transferred to the SCASC.
- The SD is part of the final Signed Data Object (SDO). Part of the Signed Data Object Composer (SDOC) function (building of the final AdES format) is to relate the digital signature value to the SD.

An important design decision for remote signature creation services is where the SD, and thus its content, needs to be available. Making available only the SDR limits threats to confidentiality but may result in limitations in the functionality of the remote signature creation solution (i.e. when enveloping or enveloped signatures need to be created, or when visual representation of the signature needs to be included).

4.3.2 DTBS composition and formatting

In the two processes of DTBS composition and formatting, which in the context of this specification are seen together, the SDR (hash of the document to be signed) and hashes of all signed attributes are assembled into the Data To Be Signed Formatted (DTBSF). In addition to a certificate identifier (hash of signing certificate, possibly also of further certificates in a certificate chain) as indicated in the figure, further signed attributes are required or allowed by the ETSI standard signature formats (C/X/PAdES). For example all baseline CAdES and XAdES variants require the presence of the signed attributes "document type" (of SD) and "claimed signing time".

The signed attributes, whose presence is needed in the DTBS, or their hash values are available to the SCASC when the DTBSF is created by the SCASC.

4.3.3 DTBS preparation

This step consists of creating the DTBSR from DTBSF. The SCASC prepares the entire DTBSF, calculates the hash, and sends the hash value (DTBSR) as input to a SSASC.

4.3.4 SDO composer

As the final step, the SDO (the AdES format) is constructed. This consists of combining the digital signature value with other parameters into the requested format. Depending on the format, the digital signature made available for the SD is named:

- Enveloped: The signature is added to the SD (e.g. PAdES signature),
- Enveloping: The signature wraps the SD (e.g. certain CAdES formats),
- Detached: The signature is a separate object linked to the SD..

The SDO composing is done by a separate service instance or integrated with other functions in the SCASC.

4.4 Server Signing Application

4.4.1 Signature creation

The purpose of the signature creation process is to take DTBSR and create a digital signature value under the control of the signer. In the context of this specification, the creation of the digital signature value is managed by a SSASC that uses a signing key, held within a cryptographic security module (SCDev), that the signatory can activate by means of a secure authorization and activation process.

4.4.1.1 Signature activation

The SSASC uses a remote SCDev in order to generate, maintain and use the signing keys under the sole control of their authorized signer. The authorized signer remotely controls the signing key with a high level of confidence by means of the Signature Activation Module (SAM) that is a software component using the Signature Activation Data (SAD) to authenticate the signer and gain its authorisation to activate its signing key for the purpose of signing the DTBSR. This process ensures confidence that the signing keys are under the sole control of the signer.

Two different levels of confidence of the control of the signing key, as defined in [6], are considered in the present document:

- Sole control assurance level 1 (SCAL1):
    - The signing keys are used, with a low level of confidence, under the sole control of the signer.
    - The authorised signer's use of its key for signing is enforced by the SSA which authenticates the signer. The activation of the signing key can remain for a given period and/or for a given number of signatures.

  It is not expected that such implementations would meet the requirements of sole control as it would be expected for a stand-alone QSCD as defined in the eIDAS [i.1] Regulation.

- Sole control assurance level 2 (SCAL2):
    - The signing keys are used, with a high level of confidence, under the sole control of the signer.
    - The authorised signer's use of its key for signing is enforced by the Signature Activation Module by means of Signature Activation Data provided, by the signer, using a Signature Activation Protocol, in order to enable the use of the corresponding signing key to sign specific documents.

4.4.1.2 Signature creation by SCDev

Signature creation process is performed by SCDev. In the context of this specification, only architectures where the signature creation process is carried out by a remote SCDev are considered. According to the above sole control assurance levels the signing key performs the digital signature value creation after a successful signer authentication by the SSASC (SCAL1) or after a successful SAD verification by the SAM.

5 Architectures for server signing

EDITORIAL NOTE: TO BE COMPLETED AND REVIEWED.

5.1 Overview

This clause describes the architectures of systems supporting remote server signing pointing out the fundamental interactions of Signature Creation Application (SCA) and Server Signing Application (SSA) services with the other parties involved in remote signature processes and taking in consideration the level of confidence of the control of the signing keys.

A typical schema for representing systems supporting remote server signing includes a server Creation Application (SCA) that is connected to a Server Signing Application (SSA) hosting the remote Signature Creation Device (providing QES and/or AdES). Services such as CA/RA, OCSP and CRLs, timestamping and authentication and/or authorization servers are considered external to the schema.

Two main scenarios are considered:

- the signature application sends to the Signature Creation Service Provider requests to generate one or more AdESs;
- the signature application sends to the Signature Creation Service Provider requests to generate one or more Digital Signature Values and then completes the creation of the AdES structure.

The defined protocols allow both SCA and SSA services to implement batch signing for documents, hashes of documents and DTBSRs. In such processes of batch signatures of documents the signer is not requested to explicitly approve each document signature.

5.2 Introduction to architectures

Two different architectures will be presented in the following clauses in which SCA and SSA implement different authentication and authorization mechanisms according to the level of confidence of the control of the signing keys, taking in consideration that the TSP managing SCA and/or SSA services can delegate the authentication and authorization processes to an external party (e.g. to an identity and/or an authentication provider).

The architectures include two main environments: the signer's environment and the TSP protected environment.

NOTE: in order to meet the higher assurance level for SCAL2, the TSP protected environment shall also include a tamper protected device.

The signer's environment is local to the signer and its protection is under responsibility of the signer. The TSP protected environment is operated according to the security policy chosen by the TSP for securing the operations of the SCS and can store in a protected form, signing key(s) and link(s) between key(s) and signer(s).

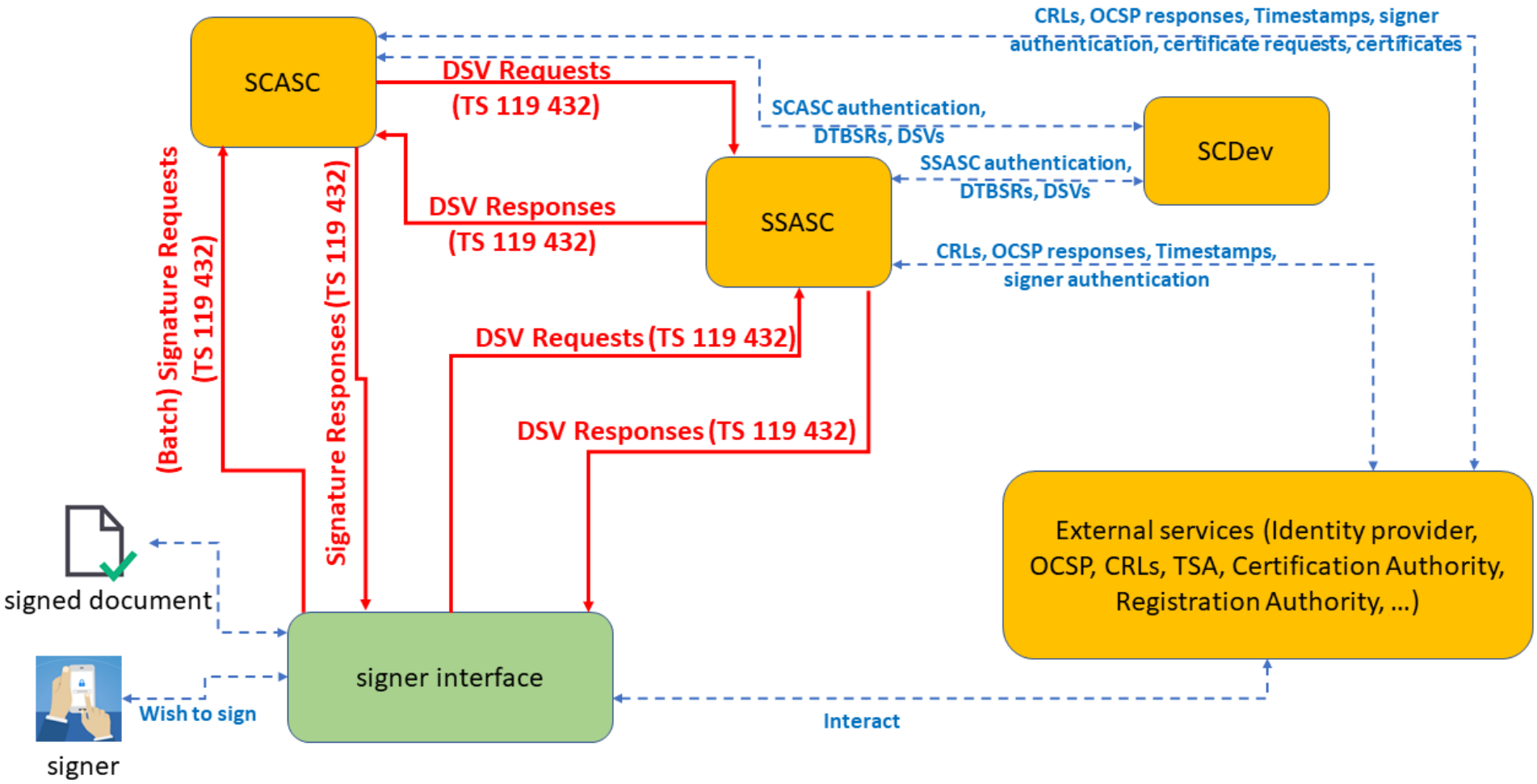
In the following models the dotted lines are used to represent data streams that are not part of the protocols defined in the document and are present only for the purpose of specifying any possible features of the illustrated services.

5.3 Remote signing services with SCAL1

In this model the signing key confidentiality and integrity are ensured by the SCDev that can be activated by the SSASC. Such activation can remain for a given period and/or for a given number of signatures.

The signer can be authenticated by a SCASC or a SSASC depending on whether the SCSP is hosting a SCASC and/or a SSASC. If the SCSP is hosting only the SSASC then the SCASC can be provided i.e. directly in the signer environment or by a different SCSP. SCASC and SSASC can delegate signer authentication to an external party. When the signer authentication succeeds, the corresponding signing key may be used for signature operations on behalf of the signer within a certain time frame and/or a certain amount of signature operations thus allowing the management of bulk/batch signature operations.

Remote signing services architecture with SCAL1

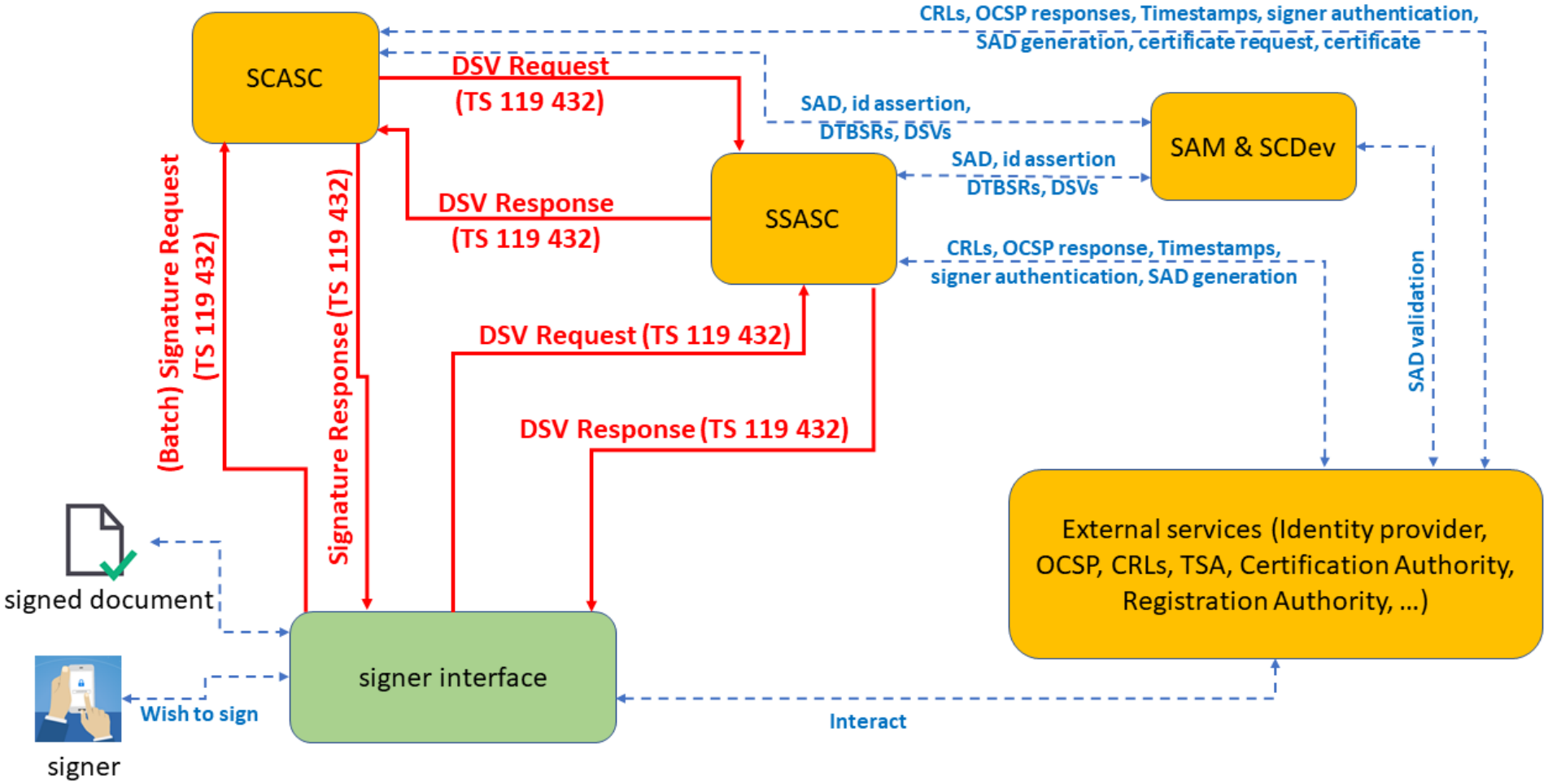


5.4 Remote signing services with SCAL2

In this model a third main environment is defined in addition to signer's and TSP protected environments: the tamper protected environment. It is operated within the TSP protected environment, protects the use of signing keys and enforces signature activation to be under the signer control with a greater degree of confidence than in the previous model.

In this model the signing key confidentiality and integrity are ensured by the SAM that can be activated by the SSA. The SAM verifies the SAD in order to be able to authorize the requested signature operation. The SAM can delegate signer authentication to an external party. When the SAD validation succeeds, the corresponding signing key may be used for signature operations on behalf of the signer.

Remote signing services architecture with SCAL2



5.5 Security, integrity and confidentiality

ETSI TS 119 431-2 provides requirements for TSPs operating a SCASC supporting AdES digital signature creation.

ETSI TS 119 431-1 provides requirements for TSPs operating a SSASC supporting digital signature value creation.

6 Protocol profiles specification

EDITORIAL NOTE: TO BE CHECKED THE TERMS USED IN CEN DOCUMENTS. THEY SHOULD BE THE SAME UNLESS …

6.1 Introduction

The present document specifies the semantics of a protocol for requesting the creation of digital signatures to a remote server and for receiving the related response.

For the aforementioned semantics the present document specifies two bindings, each one in a different format (XML and JSON).

The profiles specified in the present document take as starting point a number of OASIS DSS and DSS-X Technical Committees' specifications, namely "OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 – Working Draft 05" [2], "OASIS Standard: Advanced Electronic Signature Profiles of the OASIS Digital Signature Service Version 2.0 – Working Draft 02" [3] and the Cloud Signature Consortium specification "Architectures, Protocols and API Specifications for Remote Signature applications. Public pre-release version 0.1.7.9" [1].

The rest of the document is organized as follows:

- Sub-clauses 6.2 and 6.3 provide general remarks on the XML and JSON protocols relying on OASIS DSS and DSS-X Technical Committees' and CSC's protocols.
- Clause 7 specifies the components of the protocols for remote digital signature creation (XML and JSON) used by SSASC and SCASC.
- Clause 8 specifies the profiles implemented by SSASC and SCASC by using the components defined in clause 7.

For each component of the aforementioned protocols, the present document:

- Defines requirements for the semantics of the component (i.e. its mandatory contents, its optional contents, etc). These requirements are defined in clauses "Component semantics".
- Defines requirements for the XML component of the XML protocol relying on OASIS DSS and DSS-X Technical Committees' protocols, which is able to fulfil the semantic requirements already defined. These requirements are defined in clauses named "XML related component".
- Defines requirements for the JSON component of the JSON protocol relying on CSC's protocols, which is able to fulfil the semantic requirements already defined. These requirements are defined in clauses named "JSON related component".

6.2 OASIS DSSX TC XML related protocol

The structures described in this specification are contained in the schema files [DSS_Core_XSD], [AdES_XSD], [ASYN_XSD], [SIG_POL_XSD], and the xml schema file [ETSI_SIG_CORE_XSD]. The new elements and types defined in that schema are defined within the XML namespace whose URI value is: http://uri.etsi.org/19432/v1.1.1#

EDITORIAL NOTE: the schema files above are not yet available

Table 1 shows the URI values of other XML namespaces and their corresponding prefixes used in the aforementioned schema file and within the present document.

URI value of the XML Namespace	Prefix
http://uri.etsi.org/19342/v1.1.1#	etsisig
urn:oasis:names:tc:dss:1.0:core:schema	dss
urn:oasis:names:tc:dss:1.0:profiles:AdES:schema#	dssades
urn:oasis:names:tc:dss:1.0:profiles:asynchronousprocessing:1.0	dssasyn
urn:oasis:names:tc:dss:1.0:profiles: VisibleSignatures:schema#	dssvs
http://www.w3.org/2000/09/xmldsig#	ds
http://uri.etsi.org/01903/v1.3.2	xades
http://uri.etsi.org/01903/v1.4.1	xadesv141
urn:oasis:names:tc:SAML:2.0:assertion	saml2
http://docs.oasis-open.org/dss-x/ns/base	dsb
http://docs.oasis-open.org/dss-x/ns/core	dss2

The present document will reference components in the aforementioned documents and further profiles some of them.

In the absence of any further requirement defined in the present document, the requirements defined in the aforementioned documents for each element present in this profile, shall apply.

The present document also specifies elements that are not specified in the aforementioned documents. For these elements, the present document also defines the processing model that the server shall implement. This processing model is specified below the indication *Processing model* within each clause that specifies one of these elements.

6.3 CSC JSON related protocol

The structures described in this specification are contained in the schema files [ETSI_SIG_CORE_JSCHEMA].

EDITORIAL NOTE: the schema files above are not yet available

The present document also specifies elements that are not specified in the CSC specifications document. For these elements the present document also defines the processing model that the server shall implement. This processing model is specified below the indication *Processing model* within each clause that specifies one of these elements.

7 Protocol Components Definitions

7.1 Introduction

This clause defines the components of the protocol for remote digital signature creation. The components represent the data that can be passed to or returned by the SCS in order to request and execute the SCS functionalities. Clause 8 defines the profiles implemented by the SCS that make use of the components defined in this clause.

7.2 Component for asynchronous/synchronous operation mode selection

7.2.1 Component semantics

With the term synchronous operation mode it is intended that the client after sending any request to the SCS will wait for it to finish before moving on to another task. In such operation mode the SCS will perform the requested operation and return the corresponding outcomes to the client. With the term asynchronous operation mode it is intended that the client after sending any request to the SCS, can move on to another task before it finishes. In such operation mode the SCS will accept the request and return a notification informing the client about the acceptance of its request. The client will be able to request the corresponding outcomes to the SCS later.

This component shall be used to require the synchronous or asynchronous operation mode from the SCS. This component is one element string parameter.

When this component is not defined by the client, the default operation mode applied by the SCS is synchronous.

When the asynchronous mode is selected the 'Component for identification of the request' defined in the clause 7.7 shall be used to retrieve the signature response later.

7.2.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

- `operationMode` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
operationMode	Required	String	The type of operation mode requested to the SCS. It can take one of the following values: "A": an asynchronous operation mode is requested "S": a synchronous operation mode is requested.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.2.3 XML related component

The element for asynchronous/synchronous operation mode selection shall be `etsisig:OperationMode`, child element of the `dss2:OptionalInputs` element. If the `OperationMode` element is omitted, the operation mode selection is implicitly synchronous.

The `OperationMode` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="OperationMode" type="etsisig:OperationModeType" />

<xs:simpleType name="OperationModeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Synchronous" />
    <xs:enumeration value="Asynchronous" />
  </xs:restriction>
</xs:simpleType>
```

7.3 Component for correlating response to corresponding request

7.3.1 Component semantics

This component shall be used to correlate the received response with the request sent. The request shall include the 'Component for identification of the request' defined in clause 7.7

7.3.2 JSON related component

The component for correlating the received response with the request sent shall be represented by the following

- `requestID` string type result;

specified according to the following table

Parameter	Presence	Type	Description
requestID	Required	String	Arbitrary data generated by the client application in order to handle a transaction identifier. It is unique within SCS.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.3.3 XML related component

The element for correlating a response to a request shall be the `RequestID` attribute of the `dss2:SignResponse` root element.

7.4 Component for credential authorization

7.4.1 Component semantics

This component shall contain the information needed to authorize the use of the signing key.

When the sole control assurance level 1 (SCAL1) is implemented this component contains the data used to control, with a low level of confidence, that a given signature operation is performed on behalf of the signer under sole control of the signer.

When the sole control assurance level 2 (SCAL2) is implemented this component contains the data used to control, with a high level of confidence, that a given signature operation is performed on behalf of the signer under sole control of the signer.

7.4.2 JSON related component

The component for submitting the credential authorization shall be represented by the following

- `SAD` string type parameter;

specified according to the following table

Parameter	Presence	Type	Description
SAD	Required	String	Authentication data used to authorize the use of the signing key

7.4.3 XML related component

The element for credential authorization shall be `etsisig:SignatureActivationData`, child element of the `dss2:OptionalInputs element`. If the value to be submitted is not of string type, it shall be encoded into a string value using Base64 encoding.

The SignatureActivationData element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="SignatureActivationData" type="xs:string" />
```

7.5 Component for defining optional data to be returned

7.5.1 Component semantics

This component shall be used by the client to define optional data to be returned from the SCS. This element shall contain a list of element names of the optional data requested. If the SCS doesn't recognize or can't handle any optional data to be returned, it shall reject the request and return an error.

7.5.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

- `optionalData` array of string type parameter

specified according to the following table

Parameter	Presence	Type	Description
optionalData	Required conditional	Array of String	List of element names identifying the data that are requested to be returned.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.5.3 XML related component

The element for requesting optional data to be returned shall be the optional element `OptionalData,` contained in the `dss2:OptionalInputs` element.

The `OptionalData` element is defined in XML Schema file "[?]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="OptionalData" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
```

EDITORIAL NOTE: To be discussed if we should follow OASIS DSS structure?`<ReturnSpecificData>true</ReturnSpecificData>` then deciding which data can be returned.

7.6 Component for defining the validity period for asynchronous requests

7.6.1 Component semantics

This component shall be used to specify a maximum period of time within which the asynchronous request outcome(s) retrieval shall be completed. The validity period shall be calculated starting from the moment of the request acceptance.

7.6.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

- `validity_period` number type parameter

specified according to the following table

Parameter	Presence	Type	Description
validity_period	Required	Integer	Maximum period of time, starting from the asynchronous request acceptance and expressed in milliseconds, within which the asynchronous request outcome(s) retrieval shall be completed.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.6.3 XML related component

The element for defining the validity period for asynchronous requests shall be `etsisig:ValidityPeriod`, child element of the `dss2:OptionalInputs` element. The value specifies the maximum period of time, starting from the asynchronous request acceptance and expressed in milliseconds, within which the asynchronous request outcome(s) retrieval shall be completed.

The `ValidityPeriod` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="ValidityPeriod" type="xs:int" />
```

7.7 Component for identification of the request

7.7.1 Component semantics

This component shall be used to identify a request. It is needed in order to correlate requests with responses by means of the component defined in clause 7.3.

7.7.2 JSON related component

The component for the identification of the request shall be represented by the following

- `requestID` string type parameter.

specified according to the following table

Parameter	Presence	Type	Description
requestID	Required	String	Arbitrary data from the client application used to handle a transaction identifier. It shall be unique within SCS.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.7.3 XML related component

The element for identification of the request shall be the `RequestID` attribute of the `dss2:SignRequest` root element.

7.8 Component for identifying signature credentials

7.8.1 Component semantics

This component is used to uniquely identify the signer's private key and corresponding certificate to be used for signature creation.

7.8.2 JSON related component

The component for requesting the operation mode selection shall be represented by the following

- `credentialID` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
credentialID	Required	String	The identifier associated to the private key and corresponding certificate.

7.8.3 XML related component

The element for identifying signature credentials shall be the `dss2:KeySelector`, child element of `dss2:OptionalInputs` element.

7.9 Component for language and region selection

7.9.1 Component semantics

This component shall be used to request a preferred language of the response and shall be specified according to RFC5646.

The service may provide language-specific responses using the requested language. In the case the requested language is not supported then no error shall be raised and the responses shall be produced in a default language that can be specified in the language output parameter.

7.9.2 JSON related component

The component for selecting language and region settings shall be represented by the following

- `lang` string type parameter.

specified according to the following table

Parameter	Presence	Type	Description
lang	Required	String	The preferred language of the SCS responses, specified according to RFC5646. The SCS should provide language-specific responses using the specified language. If the specified language is not supported then SCS shall provide responses in its own default language.

7.9.3 XML related component

The element for language and culture selection shall be `dsb:Language`, child element of the `dss2:OptionalInputs` element.

7.10 Component for specifying the contents from certificate info to be returned

7.10.1 Component semantics

This component is used to specify which contents of the signing certificate chain shall be returned. If this component isn't defined only the end-entity certificate will be returned.

7.10.2 JSON related component

The component for specifying the contents from certificate chain to be returned shall be represented by the following

- `returnCertificates` string type parameter
- `authInfo` boolean type parameter
- `certInfo` boolean type parameter

specified according to the following table

Parameter	Presence	Type	Description
`returnCertificates`	Optional	String	Specifies which certificates from the certificates chain shall be returned in the CSC response. • "none": no certificate is returned. • "single": only the end entity certificate is returned. • "chain": the full certificate chain is returned. The default value is "single".
`authInfo`	Optional	Boolean	Specifies if the information on the end entity certificate shall be returned as printable strings. The default value is "false", so if the parameter is omitted then the information will not be returned.
`certInfo`	Optional	Boolean	Specifies if the information on the authorization mechanisms supported by this credential (PIN and OTP groups) shall be returned. The default value is "false", so if the parameter is omitted then the information will not be returned.

7.10.3 XML related component

The element for list the certificate chain shall be `etsisig:ReturnSigningCertificate`, child element of the `dss2:OptionalInputs` element.

The `ReturnSigningCertificate` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="ReturnSigningCertificate" type="ReturnSigningCertificateType" default="single"/>

<xs:complexType name="ReturnSigningCertificateType">
    <xs:sequence>
        <xs:restriction base="xs:string">
               <xs:enumeration value="none"/>
               <xs:enumeration value="single"/>
               <xs:enumeration value="chain"/>
        </xs:restriction>
        <xs:element name="authInfo" type="xs:boolean"/>
```

```
        <xs:element name="certInfo" type="xs:boolean"/>
    </xs:sequence>
</xs:complexType>
```

7.11 Component for managing digital signatures transactions

7.11.1 Component semantics

This component may be used to initiate and manage a sequence of an agreed number of signatures to be regarded as a single unit of work (named transaction in the present clause) between the client and the SCS.

The transaction shall be identified by the component defined in clause 7.7 that will handle a transaction identifier unique within the SCS.

NOTE 1: a typical example of a working transaction is the case in which a signer needs to put in a document a certain number of PAdES signatures. In such case a new DTBS is computed before every new signature. By using a working transaction to complete all the signatures allows the signer to have a better control of the whole signature process.

NOTE 2: another example is the case in which a signer is requesting to sign a very large number of documents, where each document is being included in the request in its entirety and each document is fairly big. This component can be used to prevent the request from being very big. Instead of sending all the documents to the SCASC at the same time, the documents can be sent one at a time, and they would be signed with the same signatory action. The session key would be used to be sure that subsequent documents belong to the same transaction.

NOTE 3: the definition of the authorization protocol is out of scope of the present document.

7.11.2 JSON related component

The component for initiating and managing a transaction shall be represented by the following

1) `numSignatures` number type parameter

specified according to the following table

Parameter	Presence	Type	Description
numSignatures	Required	Integer	The number of signatures to be performed in the context of the transaction. The SCS will check this value in the context of the transaction.

7.11.3 XML related component

The element for managing digital signature transaction shall be the `Transaction` element, child element of `dss2:OptionalInputs`.

`Transaction` is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="Transaction" type="TransactionType"/>
<xs:complexType name="TransactionType">
    <xs:choice>
        <xs:element name="SessionKey" type="xs:string"/>
        <xs:element name="NumberOfSignatures" type="xs:integer"/>
    </xs:choice>
</xs:complexType>
```

7.12 Component for notifying operation result(s)

7.12.1 Component semantics

This component shall contain information representing the outcome of the request.

Note: When the requesting mode is synchronous it shall return the outcome of the processing. When the requesting mode is asynchronous it shall return the confirmation or not of the acceptance of the request.

EDITORIAL NOTE: Add list of additional results specific to this profile.

7.12.2 JSON related component

When the HTTP status of the request is different from 200 OK, the outcome of the requested operation shall be returned in the body of the HTTP response using the "application/json" media type. The json structure includes the following

- `error` string type retrun;
- `error_description` string type retrun

specified according to the following table

Parameter	Presence	Type	Description
error	Required	String	An error code string.
error_description	Optional	String	A human readable description, written in a language that considers the `lang` parameter specification, providing additional error information to assist the client application in understanding the error that occurred.

7.12.3 XML related component

The element for notifying the result shall be the `dsb:Result` with values of child elements `dss2:ResultMajor` and `dss2:ResultMinor` as specified in [2].

EDITORIAL NOTE: Define new major and minor results for this protocol as required.

7.13 Component for optional signature attributes/properties selection

7.13.1 Component semantics

The request can include this component if there are certain signed or unsigned attributes/properties that the SCASC is requested to include in the signature. The client can pass to the SCASC a particular value to be used for each attribute/property or leave the value up to the SCASC to be determined. If no value is passed and the SCASC cannot calculate it the corresponding attribute/property shall not be included in the signature. The SCASC can include additional attributes/properties in the signature, even if these ones aren't explicitly requested by the client (i.e. because such attributes/properties are mandated by the signature profiles).

7.13.2 JSON related component

The component for the selection of the attributes/properties to be included in the signature shall be represented by the following

1) `attribute_name` string type parameter;
1) `attribute_value` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
attribute_name	Required	Array of String	Names or OIDs of the attributes/properties to be included in the signature.
attribute_value	Conditional	Array of String	Values to be used for the corresponding attributes/properties to be included in the signature. When it is not defined the SCASC shall calculate it, if needed.

7.13.3 XML related component

The element for optional signature attributes/properties shall be `dss2:Properties`, child element to the `dss2:OptionalInputs` element.

7.14 Component for polling signature results

7.14.1 Component semantics

This component is used to identify the signature request whose results are requested by the client. The values returned in the component specified in clause 7.2 shall be used in order to identify the signature request whose results are requested.

7.14.2 JSON related component

The component for identifying the signature request whose results are requested shall be represented by the following

- `pendingRequestID` string type parameter;

specified according to the following table

Parameter	Presence	Type	Description
pendingRequestID	Required	String	Arbitrary data generated by the client application and passed to SCS in the asynchronous request in order to identify uniquely the corresponding request.

7.14.3 XML related component

The element for submitting the identification of the previous signature request shall be included in the `requestID` attribute of the element `dss2:PendingRequest` root element.

7.15 Component for protocol identifier

7.15.1 Component semantics

This component can be used by the client to communicate to the server which protocol is being used to communicate to the server itself. The value of this component shall be an identifier notifying that the request has been built using the profile defined by the present document.

The identifier for the profile defined by the present document shall be:

http://uri.etsi.org/19342/v1.1.1/creationprofile#

The request may contain additional components whose values are identifiers of other profiles that have also been used for building the request.

This component shall be used to notify the server that the client expects processing of the request according to the profile defined by the present document.

7.15.2 JSON related component

The component for notifying to SCS the information of the protocol that is being used by the client shall be represented by the following

- `profile` string type parameter;

specified according to the following table

Parameter	Presence	Type	Description
profile	Required	String	String that identifies the protocol being used by the client to communicate to SCS. The value should be the one specified in clause 7.15.1.

7.15.3 XML related component

The element used to notify the server of the profile shall be the `dsb:Profile` element of the `dss2:SignRequest`. Exactly one such element shall hold the value of the profile identifier as specified in 7.15.1.

7.16 Component for requesting specific signature formats

7.16.1 Component semantics

This component is used to request a specific signature format. The signature format and conformance level shall be the same for each document that will be signed within the identified signature request.

The conformance levels of the "baseline profiles" standards should be used.

EDITORIAL NOTE: Add reference to ETSI TS 103 171, ETSI TS 103 172 and ETSI TS 103 173.

EDITORIAL NOTE: This component seems to combine two things: signature format and signature placement. Should we split this component into two? Are there any ETSI defined URI identifiers for the twelve AdES baseline profiles we can use?

Signature format
CAdES
PAdES
XAdES

Conformance Level:
B
T

LT
LTA

According type signature selected

Signature format:	**Signed envelope property**
CAdES	Detached Attached Parallel
PAdES	Certification Revision
XAdES	Enveloped Enveloping Detached

7.15.2 JSON related component

The component for requesting a specific signature format shall be represented by the following parameters

1) `signature_format` string type parameter;
2) `conformance_level` string type parameter;
3) `signed_envelope_property` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
signature_format	Required	String	The required signature format: • "CAdES" shall be used to request the creation of a CAdES signature; • "XAdES" shall be used to request the creation of a XAdES signature; • "PAdES" shall be used to request the creation of a PAdES signature.
conformance_level	Optional	String	The required baseline level format: • "B" shall be used to request the creation of a baseline level B signature; • "T" shall be used to request the creation of a baseline level T signature; • "LT" shall be used to request the creation of a baseline level LT signature; • "LTA" shall be used to request the creation of a baseline level LTA signature. The parameter is optional. The default baseline level is B in case it is omitted.

			If a timestamp is needed its request and inclusion is managed by the SCS according to SCS configuration and policies.
signed_envelope_property	Optional Conditional	String	The required property concerning the signed envelope.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.15.3 XML related component

The element for requesting specific signature formats shall be `SignatureFormat`, child element of the `dss:OptionalInputs` element.

The `SignatureFormat` element shall be defined as in XML Schema file "[TO-BE-DEFINED]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="SignatureFormat" type="SignatureFormatType"/>

<xs:complexType name="SignatureFormatType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="ConformanceLevel" type="xs:string" use="required" />
        <xs:attribute name="WhichDocument" type="xs:IDREF" use="required" />
      </xs:extension>
    </xs:simpleContent>
 </xs:complexType>
```

EDITORIAL NOTE: In OASIS DSS2, the SignatureType element can be used but this uses an URI value. For signature placement/packaging, OASIS DSS2 provides the IncludeObject, IncludeEContent and SignaturePlacement elements.

7.17 Component for returning credential authorization mode

7.17.1 Component semantics

This component specifies the authorization mode required by the identified signature credential.

The possible returned values are:

1) “implicit”: the authorization process is managed by the SCS autonomously
2) “explicit”: the client shall collect needed factors of security elements
3) “authorizationCode”: the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on authorization code (RFC 6749)
4) “identificationToken”: the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on implicit grant (RFC 6749)

7.17.2 JSON related component

The component for returning the authorization mode of the identified signature credential shall be represented by the following

- `authMode` string type result;

specified according to the following table

Parameter	Presence	Type	Description
authMode	Required	String	Specifies one of the authorization modes: “implicit”: the authorization process is managed by the SCS autonomously.

			"explicit": the client application shall collect the needed levels of security elements. "authorizationCode": the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on authoritzation code as described of RFC 6749 [i.13]. "identificationToken": the authorization process is managed by the SCS, for example using an OAuth 2.0 mechanism based on implicit grant as described in RFC 6749 [i.13].

7.17.3 XML related component

The element for returning credential authorization mode shall be `etsisig:AuthorizationMode`, child element of the `dss2:OptionalOutputs` element.

The `etsisig:AuthorizationMode` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="AuthorizationMode" type="AuthorizationModeType"/>
<xs:simpleType name="AuthorizationModeType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Implicit"/>
        <xs:enumeration value="Explicit"/>
        <xs:enumeration value="AuthorizationCode"/>
        <xs:enumeration value="IdentificationToken"/>
    </xs:restriction>
</xs:simpleType>
```

7.18 Component for returning digital signature value(s)

7.18.1 Component semantics

This component shall contain a list of base64 encoded signature values corresponding to the hashes passed in the DTBSR(s) component specified in clause 7.33.

The digital signature value(s) position into the list shall be the same of the hashes included in DTBSR(s) component.

This component can be specified according to possible alternative behaviours of the SCS:

1) When one or more of the requested signatures fail, this component is not returned and an error code is returned as signature creation result outcome.
2) When one or more of the requested signatures fail, the corresponding DSV(s) are returned as empty values.

7.18.2 JSON related component

The component for returning the DSV(s) shall be represented by the following

- `signatures` array of string type result

specified according to the following table

Parameter	Presence	Type	Description
signatures	Required	Array of String	One or more base64-encoded signature value(s). In case of multiple signatures, the signatures values shall be returned in the same order of the corresponding hashes provided as an input parameter.

7.18.3 XML related component

The element for returning digital signature value(s) shall be `dss2:SignatureObject`, child element of the `dss2:SignResponse` root element. There shall be one `dss2:SignatureObject` for each DSV, according to the ETSI remote signing profile.

7.19 Component for returning sole control assurance level required

7.19.1 Component semantics

This component specifies the sole control assurance level required by the identified signature credential, as defined in [6]. Only two values of sole control assurance level shall be supported: "SCAL1" and "SCAL2". The "SCAL1" value indicates that at least a basic authorization is required, the "SCAL2" values indicates that at least a two-factors authorization is required.

7.19.2 JSON related component

The component for returning the sole control assurance level required by the identified signature credential shall be represented by the following

1) `SCAL` string type result

specified according to the following table

Parameter	Presence	Type	Description
SCAL	Required	String	Specifies the Sole Control Assurance Level required by the credential, as defined in CEN EN 419 241-1 [6]: • "1": at least a basic authorization is required (SCAL1). This level does not require to pass a Signature Activation Data to the signHash profile. • "2": at least a two-factor authorization is required (SCAL2). This level requires to pass the Signature Activation Data to the signHash profile.

7.19.3 XML related component

The element for returning SCAL level required shall be `SoleControlAssuranceLevel`, child element of the `dss2:OptionalOutputs` element.

The `SoleControlAssuranceLevel` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="SoleControlAssuranceLevel" type="SoleControlAssuranceLevelType"/>

<xs:simpleType name="SoleControlAssuranceLevelType">
        <xs:restriction base="xs:string">
              <xs:enumeration value="SCAL1"/>
              <xs:enumeration value="SCAL2"/>
        </xs:restriction>
 </xs:simpleType>
```

7.20 Component for returning service information

7.20.1 Component semantic

This element shall contain:

1) a general description of the service;
2) a general name identifying the service;
3) a pointer to the logo of the service;
4) the country where the service is operating;
5) information concerning the protocol versions supported;
6) a list of service policies implemented by the SCS;
7) a list of signature policies implemented by the SCS;
8) a list of supported languages;
9) a list of supported authentication modes;
10) a list of accepted operation modes;
11) a list of names of all the API methods implemented and supported by the SCS.

7.20.2 JSON related component

The component for returning service information shall be represented by the following

1) `description` string type result;
2) `name` string type result;
3) `logo` string type result;
4) `region` string type result;
5) `protocol` string type result;
6) `versions` array of string type result;
7) `lang` array of string type result;
8) `signaturePolicies` an array of strings result;
9) `servicePolicies` an array of strings result;
10) `operationModes` an array of strings result;
11) `authType` an array of strings result;
12) `methods` an array of strings result.

specified according to the following table

Parameter	Presence	Value	Description
description	Required	String	A short description of the service.
name	Required	String	The name of the service.
logo	Required	String	The URI of the image file containing the logo of the Service. The image shall be in either JPEG or PNG format and not larger than 256x256 pixels.
region	Required	String	The ISO 3166-1 alpha-2 code where the service is operating.
protocol	Required	String	The name of the protocol supported by the SCS.
versions	Required	Array of String	The versions of the protocol specifications supported by the SCS. The format of the strings is Major.Minor.x.
lang	Required	Array of String	List of the supported languages in the service responses, specified according to RFC5646.
signaturePolicies	Required	Array of String	List of the supported signature policies names.
servicePolicies	Required	Array of String	List of the supported service policies names.
operationModes	Required	Array of String	List of the supported operation modes.
authType	Required	Array of String	List of the authentication mechanisms supported by the service for API methods access.

methods	Required	Array of String	List of the supported API methods names.

7.20.3 XML related component

The element for returning service information shall be `etsisig:ServiceInformation` child element of the `dss2:OptionalOutputs`.

The `ServiceInformation` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="ServiceInformation" type="ServiceInformationType"/>

<xs:complexType name="ServiceInformationType">
          <xs:sequence>
                 <xs:element name="Description" type="xs:string"/>
                 <xs:element name="Version" type="xs:string"/>
                 <xs:element name="Logo" type="xs:anyURI"/>
                 <xs:element name="Region" type="xs:string"/>
                 <xs:element name="SupportedProtocol" type="xs:anyURI"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedLanguage" type="xs:language"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedSignaturePolicy" type="xs:anyURI"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedServicePolicy" type="xs:anyURI"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedOperationMode" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedAuthMode" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
                 <xs:element name="SupportedMethod" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
 </xs:complexType>
```

7.21 Component for returning signed documents or signatures

7.21.1 Component semantics

This component shall be used to return the requested signatures. The protocol shall allow returning the signatures in two different containers according to the following rules:

1. If the signature is enveloped within the signed document, it shall be included in a specific container identified as the container for the signed document.
2. If the signature is not enveloped then it shall be included in a specific container identified as the container that encloses the signature.

7.21.2 JSON related component

The component for returning the requested signed documents or signatures shall be represented by the following

1) `DocumentWithSignature: JSON` array of string type result;
2) `SignatureObject: JSON` array of string type result;

specified according to the following table

Parameter	Presence	Type	Description
DocumentWithSignature	Required Conditional	Array of String	Base64-encoded signatures enveloped within the documents.
SignatureObject	Required Conditional	Array of String	Base64-encoded signatures detached from the the documents.

7.21.3 XML related component

The element for returning signatures not enveloped within the documents shall be `dsb:Base64Signature`, child element of the `dss2:SignatureObject`.

In the case when returning signatures enveloped within the documents the following two elements shall be used:

1. `dss2:DocumentWithSignature`, child element of the `dss2:OptionalOutputs`, containing the document that envelopes the signature
2. `dss2:SignaturePtr`, child element of the `dss2:SignatureObject`, that shall point to the former `dss2:DocumentWithSignature`.

EDITORIAL NOTE: In our XSD for the protocol, `dss2:SignatureObject` and `dss2:DocumentWithSignature` will be redefined so that several instances can be included in the Sign Response.

7.22 Component for returning signing certificate information

7.22.1 Component semantics

This component is used for returning signing chain/certificate/credential information.

This component shall contain information about the signing credential and the signing certificate/chain used or to be used in the operation of DSV(s) creation.

This component shall contain the following data:

1) the signing X.509 certificate/chain;
2) information about the signing key:
    a. status
    b. algo
    c. len
    d. curve
3) the signer certificate attribute details:
    a. status
    b. validFrom
    c. validTo
    d. issuerDN
    e. serialNumber
    f. subjectDN
4) credential supporting multiple signatures creation with a single authorization request specification

The inclusion of this component in the response is required by the client using the "Component for defining optional data to be returned" defined in clause 7.5.

7.22.2 JSON related component

The component for returning signing certificate/credential information shall be represented by the following

1) `cert/certificates`: array of string type result;
2) `key/status`: string type result;

3) `key/algo`: array of string type result;
4) `key/len`: integer type result;
5) `key/curve`: string type result;
6) `cert/status`: string type result;
7) `cert/validFrom`: string type result;
8) `cert/validTo`: string type result;
9) `cert/IssuerDN`: string type result;
10) `cert/serialNumber`: string type result;
11) `cert/subjectDN`: string type result;
12) `multisign`: boolean type result;

specified according to the following table

Parameter	Presence	Type	Description
cert/certificates	Required Conditional	Array of String	Contains one or more Base64-encoded X.509v3 certificates from the certificate chain. If the certificates parameter defined in 7.10.1.2 clause is "chain", the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the certificates parameter is "single", only the end entity certificate shall be returned. If the certificates parameter is "none", this parameter shall not be returned.
key/status	Required	String	enabled | disabled The status of enablement of the signing key of the credential: • "enabled": the signing key is enabled and can be used for signing. • "disabled": the signing key is disabled and cannot be used for signing. This may occur when the owner has disabled it or when the SCS has detected that the associated certificate is expired or revoked.
key/algo	Required	String	The list of OIDs of the supported key algorithms. For example: 1.2.840.113549.1.1.1 = RSA encryption, 1.2.840.10045.4.3.2 = ECDSA with SHA256
key/len	Required	Number	The length of the cryptographic key in bits.
key/curve	Required Conditional	String	The OID of the ECDSA curve. The value shall only be returned if keyAlgo is based on ECDSA.
cert/status	Optional	String	valid | expired | revoked | suspended The status of validity of the end entity certificate. The value is optional. The SCS shall only return a value that is accurate and consistent with the actual validity status of the certificate at the time the response is generated.
cert/validFrom	Required	String	The validity start date from the X.509v3 signing certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. "YYYYMMDDHHMMSSZ"). This parameter shall be returned when certInfo defined in clause 7.10 is "true".
cert/validTo	Required	String	The validity end date from the X.509v3 signing certificate in printable string format, encoded as GeneralizedTime format (RFC 2459) (e.g. "YYYYMMDDHHMMSSZ"). This parameter shall be returned when certInfo defined in clause 7.10 is "true".
cert/IssuerDN	Required Conditional	String	The Issuer Subject Distinguished Name from the X.509v3 end entity certificate in printable string format, UTF-8-encoded according to

			RFC 2253. This parameter shall be returned when certInfo defined in clause 7.10 is "true".
cert/serialNumber	Required Conditional	String	The Serial Number from the X.509v3 certificate in hex encoded format. This parameter shall be returned when certInfo defined in clause 7.10 is "true"
cert/subjectDN	Required Conditional	String	The Distinguished Name from the X.509v3 certificate in printable string format, UTF-8-encoded according to RFC 2253. This parameter shall be returned when certInfo defined in clause 7.10 is "true"
multisign	Required Conditional	Boolean	Specifies if the credential supports multiple signatures to be created with a single authorization request

7.22.3 XML related component

The element for returning signing certificate/credential/chain shall be `etsisig:KeyInfo`, child element of the `dss2:OptionalOutputs` root element.

The element for returning signing certificate/credential/chain information shall be `ds:X509Data`, child element of the `etsisig:KeyInfo` element. To include additional attributes as listed in point 3 in the semantics section above, the `X509Details` element shall be used as a child element to `ds:X509Data`.

The `X509Details` element is defined in XML Schema file "[?]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="X509Details" type="X509DetailsType"/>

<xs:complexType name="X509DetailsType">
            <xs:sequence>
                  <xs:element name="Status" type="CertificateStatusType"/>
                  <xs:element name="NotBefore" type="xs:dateTime"/>
                  <xs:element name="NotAfter" type="xs:dateTime"/>
            </xs:sequence>
  </xs:complexType>

<xs:simpleType name="CertificateStatusType">
            <xs:restriction base="xs:string">
                  <xs:enumeration value="Valid"/>
                  <xs:enumeration value="Expired"/>
                  <xs:enumeration value="Revoked"/>
                  <xs:enumeration value="Suspended"/>
            </xs:restriction>
   </xs:simpleType>
```

7.23 Component for returning the list of the signing certificate(s)

7.23.1 Component semantics

This component shall be used for returning available signing certificate(s) of the signer. Each signing certificate may also include its chain.

7.23.2 JSON related component

The component for returning available signing certificate(s) of the signer shall be represented by the following

- `credentialIDs` array of string type parameters;
- `certificates` array of string type parameters;

specified according to the following table

Parameter	Presence	Type	Description
credentialIDs	Required	Array of String	One or more credentialID associated with the provided or implicit signer identification.
certificates	Required Conditional	Array of String	Contains one or more Base64-encoded X.509v3 certificates from the signing certificate chain. If the `returnCertificates` parameter defined in 7.10.1.2 clause is "chain", the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the certificates parameter is "single", only the end entity certificate shall be returned. If the `returnCertificates` parameter is "none", this parameter shall not be returned.

7.23.3 XML related component

The element for returning the list of the signing certificate(s) shall be `etsisig:ListKeyResponse`, child element of the `dss2:OptionalOutputs` root element.

7.24 Component for service authentication

7.24.1 Component semantics

This component shall contain information to authenticate the client to access to the service.

NOTE: the way a client authenticates to the service is out of scope of this document.

7.24.2 JSON related component

The authorization component shall be included into the Authorization HTTP header of every call and shall be in "Bearer" format.

7.24.3 XML related component

The element for service authentication shall be `dss2:ClaimedIdentity`, child element of the `dss2:OptionalInputs` element.

7.25 Component for service policy identification

7.25.1 Component semantics

This component shall be used to return the name of the service policy used by the server to perform the requested operation.

The inclusion of this component in the response is required by the client using the "Component for defining optional data to be returned" defined in clause 7.5.

7.25.2 JSON related component

The component for returning the service policy identification shall be represented by the following

- `policy` string type result

specified according to the following table

Parameter	Presence	Type	Description
policy	Required	String	The element that identifies a particular policy associated with the SCS.

7.25.3 XML related component

The element for service policy identification shall be `dsb:AppliedPolicy`, child element of the `dss2:OptionalOutputs` element.

7.26 Component for service policy selection

7.26.1 Component semantics

This component shall contain a non-ambiguous identifier of the service policy under which the server shall perform the requested operation.

7.26.2 JSON related component

The component for specifying the identifier of the service policy under which the server shall perform the requested operation shall be represented by the following

- `policy` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
policy	Required	String	The element that identifies a particular policy associated with the SCS. The policy element may be used to select a specific service policy if a SCS supports multiple policies, or as a sanity-check to make sure the SCS implements the service policy the client expects.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.26.3 XML related component

The element for service policy selection shall be `dsb:ServicePolicy`, child element of the `dss2:OptionalInputs` element.

7.27 Component for signature creation policy identification

7.27.1 Component semantics

This component shall contain information that has been used to generate signature(s) and/or DSV(s).

The information shall contain:

a. The identification of the signature creation policy used while creating the DSV(s)
b. Other optional parameters containing the locations of the signature creation policy document

The inclusion of this component in the response is required by the client using the “Component for defining optional data to be returned” defined in clause 7.5.

7.27.2 JSON related component

The component for returning the signature creation policy identification shall be represented by the following

- `signaturePolicyID` string type parameter;
- `signaturePolicyLocations` array of string type parameter.

specified according to the following table

Parameter	Presence	Type	Description
signaturePolicyID	Required	String	The element that identifies a particular policy associated with the SCS. It shall have the value of a unique identifier of the signature creation policy as an URI. If the identifier of the signature creation policy is an OID, then the value of this element shall be an URN indicating the value of the aforementioned OID as specified in RFC 3061.
signaturePolicyLocations	Optional	Array of String	Every string element shall have the value of one location where the signature creation policy document can be accessed, as an URI value.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.27.3 XML related component

For signature creation policy identification, the OASIS standard for Signature Policy Profile [X?] shall be used. The XML related signature creation policy protocol shall satisfy the requirements specified in [X?].

EDITORIAL NOTE: Add reference to http://docs.oasis-open.org/dss-x/profiles/sigpolicy/oasis-dssx-1.0-profiles-sigpolicy.pdf. Ask OASIS to update profile.

7.28 Component for signature creation policy selection

7.28.1 Component semantics

This component shall contain information that the service requires to generate signature(s). The information that shall be supplied is the identification of the signature policy that shall be used while signing the DTBSR(s).

7.28.2 JSON related component

The component for specifying the signature creation policy identification to be used by the SCS shall be represented by the following

- `signaturePolicyID` string type parameter

or, alternatively, the following two parameters:

- `signAlgo` string type parameter
- `signAlgoParams` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
signaturePolicyID	Required Conditional	String	The element that identifies a particular policy associated with the SCS. It shall have the value of a unique identifier of the signature creation policy as an URI. If the identifier of the signature creation policy is an OID, then the value of this element shall be an URN indicating the value of the aforementioned OID as specified in RFC 3061.
signAlgo	Required Conditional	String	The element specifies the algorithm OID used for signing.
signAlgoParams	Optional Conditional	String	The element specifies the Base64-encoded of DER-encoded ASN.1 signature parameters, if required by the signature algorithm like, for example, the RSA-PSS [RFC 3447].

7.28.3 XML related component

The element for signature algorithm selection shall be `dss2:SignatureAlgorithm`, child element of the `dss2:OptionalInputs` element.

EDITORIAL NOTE: DSS2 does not currently support including parameters to the signature algorithm (e.g. RSA-PSS). We have asked OASIS how this can be achieved.

7.29 Component for signer identification

7.29.1 Component semantics

This component is used to uniquely identify the signer within the SCS. This component represents the signer identifier associated to the signer identity within the SCS.

A request shall contain one of these or both components:

- component for signer identification, it shall be specified when there is no service authentication
- component for service authentication, specified in clause 7.24

In the case both components are passed, they shall be congruent. If a signer identification different from the implicit one defined in the service authentication is specified the latter one shall be considered.

7.29.2 JSON related component

The component for specifying the signer identification shall be represented by the following

- `userId` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
userId	Required	String	The user identifier associated to the signer identity. If the service authorization is user-specific the userID is already implicit in the service access token passed in the Authorization header. In this case, it shall not be possible to specify a different userID i.e. to obtain the list of credentials associated to another user, and the SCS shall return an error.

7.29.3 XML related component

The element for signer identification shall be the `etsisig:SignerIdentity` element, child element of `dss2:OptionalInputs`.

The `etsisig:SignerIdentity` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="SignerIdentity" type="dss2:ClaimedIdentityType"/>
```

7.30 Component for specifying response URL

7.30.1 Component semantics

With this component a client can communicate to the SCS a destination URL where the client expects to receive a notification when the SCS has completed the requested operation.

This component may accept relative URL; the server may convert the relative URL to an absolute URL before sending the response to the client.

7.30.2 JSON related component

The component for specifying the response URL (s) where the notification of the requested operation completion is to be returned shall be represented by the following

- `response_uri` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
response_uri	Required	String	The element shall have the value of one location where the SCS will notify the signature creation operation completion, as an URI value.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.30.3 XML related component

The element for specifying response URL shall be `ResponseURL`, child element to the `dss2:OptionalInputs` element.

The `ResponseURL` element shall be defined as in XML schema file "[???]", whose location is detailed in clause ? and is copied below for information.

```
<xs:element name="ResponseURL" type="xs:anyURI"/>
```

7.31 Component for submitting document(s) or hash(es) of document(s) to be signed

7.31.1 Component semantics

The protocol shall allow including the document(s) or a list of hashes for which generating signature(s) in two different containers.

When using hashes, the information concerning the digest algorithm used to calculate the hash of the document at the client side shall be supplied too.

The information that shall be supplied is:

a. The content(s) of the document(s) to be signed, included in a specific container identified as the container for the document(s).
b. The hash(es) of the document(s) to be signed and the digest algorithm used to calculate such hash(es), included in a specific container identified as the container for the hash(es). 7.32.2JSON related component

7.31.2 JSON related component

The component for submitting document(s) or hash(es) to be signed shall be represented by the following (mutually exclusive) parameters

- `documents`: array of string type parameter;
- `documentDigests`: array of JSON objects containing the following parameters:
    a. `hash`: array of string type parameter;
    b. `hashAlgorithmOID`: string type parameter.

specified according to the following table

Parameter	Presence	Type	Description

documents	Required Conditional	Array of String	Base64-encoded document(s) content(s), to be signed.
hash	Required Conditional	Array of String	Base64-encoded document(s) hash(es), to be signed.
hashAlgorithmOID	Required Conditional	String	Hashing algorithm OID used to calculate document hash(es).

The element for submitting the hash representation(s) of documents to be signed shall conforms to the definition of clause 7.32.2.

EDITORIAL NOTE: Not present in CSC defined by the editor

7.31.3 XML related component

The element for submitting document(s) or hash(es) of document(s) to be signed shall be `dss2:InputDocuments`, child element of `dss2:SignRequest` root element.

For sending documents the element `dss2:Document` shall be used. The MimeType attribute of dss2:Base64Data, child element of dss2:Document, shall be set.

The element for submitting hash(es) of document(s) to be signed shall conform to the definition of clause 7.32.3.EDITORIAL NOTE: Should we mention something about what appears to be a mistake in the dss2 xsd? Where the child elements to InputDocuments are in a choice instead of in a sequence.

7.32 Component for submitting DTBSR(s)

7.32.1 Component semantics

This component shall be used in order to provide to the SCS the list of DTBSR(s). The protocol shall allow the inclusion into the DTBSR(s) of the following information:

a. a list of hashes that shall have been calculated using the same algorithm.
b. the identification of the hash algorithm used to calculate the hashes contained in the DTBSR(s).

NOTE: ETSI TS 119 312 [i.12] should be considered for the choice of the hashing algorithms to be used.

A list of hashing algorithms recommended in ETSI TS 119 312 [i.12] is listed in the following table:

Hash algorithm OID	Hash algorithm name	Hash algorithm URI
2.16.840.1.101.3.4.2.1	SHA-256	http://www.w3.org/2001/04/xmlenc#sha256
2.16.840.1.101.3.4.2.2	SHA-384	http://www.w3.org/2001/04/xmldsig-more#sha384
2.16.840.1.101.3.4.2.3	SHA-512	http://www.w3.org/2001/04/xmlenc#sha512
2.16.840.1.101.3.4.2.8	SHA3-256	http://www.w3.org/2007/05/xmldsig-more#sha3-256
2.16.840.1.101.3.4.2.9	SHA3-384	http://www.w3.org/2007/05/xmldsig-more#sha3-384
2.16.840.1.101.3.4.2.10	SHA3-512	http://www.w3.org/2007/05/xmldsig-more#sha3-512

7.32.2 JSON related component

The component for submitting the representation(s) of the documents to be signed shall be represented by the following parameters:

- `hash` array of string type parameter

- `hashAlgorithmOID` string type parameter

specified according to the following table

Parameter	Presence	Type	Description
hash	Required	Array of String	Base64-encoded document(s) hash(es), to be signed.
hashAlgorithmOID	Required	String	Hashing algorithm OID used to calculate document hash(es).

7.32.3 XML related component

The element for submitting DTBSR(s) shall be `dss2:InputDocuments`, child element of `dss2:SignRequest` root element.

For sending hashes of documents the element `dss2:DocumentHash` shall be used. The client shall incorporate the base64 encoding of this digest value into `ds:DigestInfos` and the hash algorithm as an URI into `ds:DigestMethod`, child element of the `dss2:DocumentHash` element.

7.33 Component for returning signing credential authorization information

7.33.1 Component semantics

This component is used for returning signing credential authorization information.

This component shall contain information about the signing credential to be used in the operation of DSV(s) and/or signatures creation.

This component shall contain the following data:

1) the signing key PIN attribute details:
    a. an indication if a PIN is required or not;
    b. the PIN format;
    c. a label to be used in the user interface in order to request PIN collection;
    d. a description of the PIN;
2) the signing key second factor authentication OTP details:
    a. an indication if an OTP is required or not;
    b. the OTP type;
    c. the OTP format;
    d. a label to be used in the user interface in order to request OTP collection;
    e. a description of the OTP mechanism;
    f. an identifier of OTP device or application;
    g. the provider of OTP device or application.

The inclusion of this component in the response can be required by the client using the "Component for defining optional data to be returned" defined in clause 7.5.

7.33.2 JSON related component

The component for returning signing certificate/credential information shall be represented by the following

1) `PIN` JSON object type result:
    a. `presence` result;
    b. `format` result;
    c. `label` result;
    d. `description` result;
2) `OTP` JSON object type result;
    a. `presence` result;

b. `type` result;
c. `format` result;
d. `label` result;
e. `description` result;
f. `ID` result;
g. `provider` result;

specified according to the following table

Parameter	Presence	Type	Description
PIN/presence	Required Conditional	String	true | false | optional Specifies if a text-based PIN is required or not, or optional. This result shall be present only when authMode is "explicit".
PIN/format	Required Conditional	String	Specifies the format of the PIN: • "A": the PIN contains alphanumeric text; • "N": the PIN contains numeric text. This result shall be present only when authMode is "explicit" and PIN/presence is not "false"..
PIN/label	Optional Conditional	String	Specifies an optional label for the data field used to collect the PIN in the user interface, in the language specified in the lang parameter. This result can be present only when authMode is "explicit" and PIN/presence is not "false".
PIN/description	Optional Conditional	String	Specifies a description of the PIN in the language specified in the lang parameter. This result can be present only when authMode is "explicit" and PIN/presence is not "false".
OTP/presence	Required Conditional	String	true | false | optional Specifies if a text-based OTP is required or not, or optional. This result shall be present only when authMode is "explicit".
OTP/type	Required Conditional	String	offline | online Specifies the type of the OTP. This result shall be present only when authMode is "explicit" and OTP/presence is not "false".
OTP/format	Required Conditional	String	Specifies the data format of the OTP: • "A": the OTP contains alphanumeric text; • "N": the OTP contains numeric text. This result shall be present only when authMode is "explicit" and OTP/presence is not "false".
OTP/label	Optional Conditional	String	Specifies an optional label for the data field used to collect the OTP in the user interface, in the language specified in the lang parameter. This result can be present only when authMode is "explicit" and OTP/presence is not "false".
OTP/description	Optional Conditional	String	Specifies a description of the OTP mechanism in the language specified in the lang parameter. This result can be present only when authMode is "explicit" and OTP/presence is not "false".
OTP/ID	Optional Conditional	String	Specifies the identifier of the OTP device or application. This result shall be present only when authMode is "explicit" and OTP/presence is not "false".

OTP/provider	Optional Conditional	String	Specifies the provider of the OTP device or application. This result can be present only when authMode is “explicit” and OTP/presence is not “false”.

7.33.3 XML related component

The element for returning signing credential authorization information shall be `etsisig:KeyInfo`, child element of the `Other` element of the `dss2:OptionalOutputs` root element.

The element for returning signing credential authorization information shall be `ds:AuthData`, child element of the `etsisig:KeyInfo` element. To include additional attributes, the `PinDetails` and `OtpDetails` elements shall be used as a child element to `ds:AuthData`.

The `PinDetails` and `OtpDetails` elements are defined in XML Schema file "[?]", whose location is detailed in clause ?, and are copied below for information.

```
<xs:element name="PinDetails" type="PinDetailsType"/>

<xs:element name="OtpDetails" type="OtpDetailsType"/>

<xs:complexType name="PinDetailsType">
            <xs:simpleContent>
                  <xs:element name="presence" type="xs:string" use="required"/>
                  <xs:element name="format" "xs:string" use="required"/>
                  <xs:element name="label" type="xs:string"/>
                  <xs:element name="description" "xs:string"/>
            </xs:simpleContent>
<xs:complexType>

<xs:complexType name="OtpDetailsType">
            <xs:simpleContent>
                  <xs:element name="presence" type="xs:string" use="required"/>
                  <xs:element name="type" "xs:string" use="required"/>
                  <xs:element name="format" "xs:string" use="required"/>
                  <xs:element name="label" type="xs:string"/>
                  <xs:element name="description" "xs:string"/>
                  <xs:element name="ID" type="xs:string" use="required"/>
                  <xs:element name="provider" "xs:string"/>
            </xs:simpleContent>
<xs:complexType>
```

8 Profiles for remote signature creation

8.1 Introduction

In this document the profiles are intended to represent the interfaces by which any client can interact with any SCS conforming to this specification. Technically speaking, profiles represent the way for accessing a web-based service. A profile can be considered a software-to-software programming interface enabling applications and services to talk to each other. The following clauses define the profiles to be provided by SCSs. For any profile it is specified which of the components defined in the previous clause can be used, by means of a table where the following information are provided:

- the reference to the clause where the component is specified;
- a brief description of the component;
- the presence of the component.

The value included in the column “Presence” of the table has the following meaning:

Value	Description
M	The component shall be included in the request to or response from the SCS.
O	The component may be included in the request to or response from the SCS.
C	The component shall be included in the request or response based on the occurrence of certain conditions.

8.2 Profile for signature request (A)

8.2.1 Profile semantics

The mandatory input parameter for this profile is either the signer’s document or the SDR (signer’s document representation, for example the hash value of the document). In practice, other parameters are usually also needed, such as:

other signature attributes, signed or unsigned (e.g. document type),

signature creation policy identification or signature policy parameters such as resulting signature format (AdES type),

signer identification, usually also authentication of signer,

identification of the signing key to be used, when more than one alternative exists for the user.

For user authentication, several alternatives exist, including use of identity assertions (SAML, Oauth2, OpenID Connect etc.) issued by trusted authentication and identity provision services.

The message for requesting the creation of AdES signature(s) to an SCASC shall contain the following components:

1) One component for submitting the document(s) or document representation(s) to be signed. Clause 7.31 specifies semantic requirements for this component.

NOTE 1: The message may contain one or more documents and/or one or more document representations for signature creation. An AdES signature will be created for each of these input components. Other components are used to select the type of signature that will be created for each document or document hash.

2) One component for notifying the server that the protocol profile defined by the present document is to be used. Clause 7.15 specifies semantic requirements for this component.

NOTE 2: The profile defined by the present document may be combined with other profiles to request additional features or functionality provided by the SCASC as long as these profiles do not conflict with the requirements specified in this document.

3) One component for identifying the signing key to be used by the server for computing the requested signature operations. Clause 7.8 specifies semantic requirements for this component.

The message for requesting the creation of AdES signature(s) to the SCASC may contain other components for requesting additional features. Clause 7.5 lists these optional components and contain references to clauses that specify semantic requirements for each component.

This profile includes the following components:

Ref.	Component for	Presence
7.2	asynchronous/synchronous operation mode selection	**O**
7.4	credential authorization	**O**
7.5	defining optional data to be returned	**O**
7.6	defining the validity period for asynchronous requests	**O**
7.7	identification of the request	**O**
7.8	identifying signature credentials	**M**
7.9	language and region selection	**O**
7.11	managing digital signatures transactions	**O**
7.13	optional signature attributes/properties selection	**O**
7.15	protocol identifier	**M**
7.16	requesting specific signature formats	**O**
7.24	service authentication	**O**
7.26	service policy selection	**O**
7.28	signature creation policy selection	**O**
7.30	specifying response URL	**O**
7.31	submitting document(s) or hash(es) of document(s) to be signed	**M**

8.2.2 JSON related component

The element to request the signature of a document or a document representation shall be the set of required and optional parameters to invoke the document signature by means of the `signatures/signDoc` method.

Processing model.

The server shall process the components received with the `signatures/signDoc` in order to calculate the remote digital signature of one or multiple document(s) or SDR(s) as indicated in the clause 8.2.1 of the present document. This method can receive credential authorization in the form of Signature Activation Data (SAD).

8.2.3 XML related component

The element that shall be the main component for requesting the creation of AdES signature(s) shall be the root element of the message `dss:SignRequest` as specified in [2].

Processing model.
The server shall process the components in the `dss:SignRequest` as indicated in the corresponding clauses of [2].

The server shall process each child of the `dss:OptionalInputs` and `dss:InputDocuments` components as indicated in the corresponding clause of the present document if the child is not specified in any of the referenced OASIS documents. Otherwise, the server shall follow the processing model defined in the corresponding OASIS document.

8.3 Profile for signature response (B)

8.3.1 Profile semantics

The mandatory response returned by this profile is the result of the signature operation requested by the client. In practice, other responses are usually also returned, such as:

signed documents or signatures,

an identifier for correlating response to corresponding request, quite important if the asynchronous operating mode has been set in the corresponding request.

This profile includes the following components:

Ref	Component for	Presence
7.3	correlating response to corresponding request	**O**
7.12	notifying operation result(s)	**M**
7.21	returning signed documents or signatures	**O**
7.22	returning signing certificate information	**O**
7.25	service policy identification	**O**
7.27	signature creation policy identification	**O**

8.3.2 JSON related component

The element to respond to the signature of the document(s) request shall be the set of components as indicated in the clause 8.3.1 of the present document.

8.3.3 XML related component

The element that shall be the main component for responding to the creation of AdES signature(s) request shall be the root element of the message `dss2:SignResponse` as specified in [2].

8.4 Profile for DSVs creation request (C)

8.4.1 Profile semantics

In this scenario the SCASC or an application in the signer's environment prepares the DTBS(s) that are sent to the SSASC along with other information required to compute the signature.

Therefore the mandatory input parameter for this profile is the DTBS(s). In practice, as in the case of profile defined in clause 8.2, other parameters are usually also needed, such as:

signature creation policy identification,

signer identification, usually also authentication of signer,

identification of the signing key to be used, when more than one alternative exists for the user.

The SSASC creates the digital signature value(s) using the signer's private key held on the SCDev and returns the outcome of the signature operation and information to retrieve DSVs.

This profile includes the following components:

Ref	Component for	Presence
7.2	asynchronous/synchronous operation mode selection	**O**
7.4	credential authorization	**O**

7.5	defining optional data to be returned	**O**
7.6	defining the validity period for asynchronous requests	**O**
7.7	identification of the request	**O**
7.8	identifying signature credentials	**M**
7.9	language and region selection	**O**
7.11	managing digital signatures transactions	**O**
7.15	protocol identifier	**M**
7.24	service authentication	**O**
7.26	service policy selection	**O**
7.28	signature creation policy selection	**O**
7.30	specifying response URL	**O**
7.32	submitting DTBSR(s)	**M**

8.4.2 JSON related component

The element to request the signature on a DTBSR shall be the set of required parameters to invoke the hash signature by means of the `signatures/signHash` method as specified in CSC [1].

Processing model

The server shall process the components as indicated in the `signatures/signHash` description of the CSC [1].

The server shall process each component as indicated in the clause 8.4.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

8.4.3 XML related component

The element for requesting the hash signature(s) shall be the root element of the message `dss2:SignRequest` as specified in the present clause.

Processing model

The server shall process the components inherited from `dsb:RequestBaseType` as indicated in the clause (Processing for XML Signature) in particular in the variant for `<DocumentHash>` of OASIS Standard: Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0 - Working Draft 05 [2].

8.5 Profile for digital signature value response (D)

8.5.1 Profile semantics

This profile returns the results of the signatures generated by the SSASC using the information contained in the components received with the request message.

The mandatory response returned by this profile is the result of the signature operation requested by the client. In practice, other responses are usually also returned, such as:

DSV(s) generated by the SSASC,

an identifier for correlating response to corresponding request, quite important if the asynchronous operating mode has been set in the corresponding request.

This profile includes the following components:

Ref	Component for	Presence
7.3	correlating response to corresponding request	**O**

7.12	notifying operation result(s)	**M**
7.18	returning DSV	**O**
7.22	returning signing certificate information	**O**
7.25	service policy identification	**O**
7.27	signature creation policy identification	**O**

8.5.2 JSON related component

The element to respond to the signature of the hash(es) request shall be the set of parameters returned in the response of the `signatures/signHash` method as specified in CSC [1].

8.5.3 XML related component

The element to respond to the DTBS(s) signature request shall be the `dss2:SignResponse` root element as indicated in the clause 8.5.1 and in [2] specifications.

8.6 Profile for asynchronous processing (E)

8.6.1 Profile semantics

This profile shall manage a request that shall notify to the server to return the response corresponding to a previously sent (initial) request. Requests of this type are named pending-requests hereinafter.

In asynchronous processing one client usually sends an initial request to the server. The initial request shall contain, among other things, a request identifier generated by the client, as specified in clause 7.7.

The server can return a response indicating that the signature creation request has been accepted but it has not yet been completed. Within this initial response, the server shall convey a response identifier, as specified in clause 7.3. Both client and server can correlate the response identifier to the request identifier.

Under this processing model the client, after a certain time from the initial request, can send a pending-request to the server. This pending-request shall include the response identifier previously returned by the server. This response identifier allows the server to correlate this pending-request to the initial request and can return the signature creation results or return again an indication of "not yet finished".

If this is the case, the client can send subsequent requests until the server returns a response with the signature creation result. Each subsequent request shall include the request identifier included in the initial request and returned by the server in the response to the initial request.

For managing asynchronous processing, a component for identifying the request as a pending-request associated to an initial request is required. This component is specified in clause 7.14

Other optional components may also be required by the SSASC. These optional components are defined in the clauses 7.9 and 7.24.

The main output is the DSV(s) or the signed document(s) or signature(s). This profile includes the following components:

Ref	Component for	Presence
7.9	language and region selection	**O**
7.14	polling signature results	**M**
7.15	protocol identifier	**M**
7.24	service authentication	**O**

8.6.2 JSON related component

The element that shall indicate to the SCS that the client is requesting the response corresponding to a previously sent (initial) request (as part of an asynchronous protocol) shall be the set of required and optional parameters defined in clause 8.6.1 to invoke the `signatures/signPolling` method.

Processing model.

The server shall process each component as indicated in the clause 8.6.1 of the present document.

The server shall check the completion of the operations related to the request identified by the attribute requestID and respond accordingly returning the generated DSV(s) or signed document(s) or signature(s) or the indication that the operations have not yet finished or an error code (for example because the request cannot be completed or because the time allowed for requesting the results has expired).

8.6.3 XML related component

The element that shall indicate to the SCS that the client is requesting the response corresponding to a previously sent (initial) request (as part of an asynchronous protocol) shall be the `dss2:PendingRequest` element as specified in [2].

8.7 Profile for signing certificates list request (F)

8.7.1 Profile semantics

This profile shall be used to request the credentials list of a user identified with the component for "signer identification". A user may have one or multiple credentials associated within a single user identifier.

The mandatory input parameter for this profile is the signer identification.

This profile includes the following components:

Ref	Component for	Presence
7.7	identification of the request	**O**
7.9	language and region selection	**O**
7.10	contents from certificate chain to be returned	**O**
7.15	protocol identifier	**M**
7.24	service authentication	**O**
7.29	signer identification	**M**

8.7.2 JSON related component

The element to retrieve credentials shall be the set of parameters required by the `credentials/list` method as specified in CSC [1].

Processing model

The server shall process the components as indicated in the `credentials/list` description of the CSC [1].

The server shall process each component as indicated in the clause 8.7.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

8.7.3 XML related component

The element that shall be the main component for requesting a signing certificates list shall be the root element of the message `etsisig:InformationRequest`.

The etsisig:InformationRequest element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationRequest" type="etsisig:InformationRequestType"/>

<xs:complexType name="InformationRequestType">
    <xs:complexContent>
      <xs:extension base="dsb:RequestBaseType">
        <xs:sequence>
          <xs:element name="OptionalInputs" type="dss2:OptionalInputsType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The etsisig:GetSigningCertificatesList, child element of the dss2:OptionalInputs element shall be set to true.

```
<xs:element name="GetSigningCertificatesList" type="xs:boolean"/>
```

The etsisig:SignerIdentity, child element of the dss2:OptionalInputs element shall be set in order to identify the signer for which the list of the signing credentials is being requested.

Processing model

The server shall process the components inherited from the etsisig:SignerIdentity element in order to retrieve and return the list of credentials associated with the user identifier.

8.8 Profile for signing certificates list response (G)

8.8.1 Profile semantics

This profile shall be used to return the credentials list of a user.

This profile includes the following components:

Ref	Component for	Presence
7.12	notifying operation result(s)	**M**
7.23	returning the list of the signing certificate(s)	**O**

8.8.2 JSON related component

The element to respond to retrieve user's credentials list shall be the set of results returned in the response of the credentials/list method as specified in CSC [1].

8.8.3 XML related component

The element that shall be the main component for responding with signing certificates list shall be the root element of the message etsisig:InformationResponse.

The etsisig:InformationResponse element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationResponse" type="etsisig:InformationResponseType"/>
  <xs:complexType name="InformationResponseType">
    <xs:complexContent>
      <xs:extension base="dsb:ResponseBaseType">
        <xs:sequence>
          <xs:element name="OptionalOutputs" type="dss2:OptionalOutputsType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
```

```
    </xs:complexContent>
  </xs:complexType>
```

The etsisig:ListKeyResponse, child element of the dss2:OptionalOutputs element will return the requested signing certificates list.

8.9 Profile for certificate information retrieval request (H)

8.9.1 Profile semantics

This profile shall be used to request some information about a signing certificate identified with the component for "identifying signing credential".

The mandatory input parameter for this profile is the signing credential identification.

This profile includes the following components:

Ref	Component for	Presence
7.7	identification of the request	**O**
7.8	identifying signature credentials	**M**
7.9	language and region selection	**O**
7.10	list the certificate chain	**O**
7.15	protocol identifier	**M**
7.24	service authentication	**O**

8.9.2 JSON related component

The element to request credentials information shall be the set of parameters required by the credentials/info method as specified in CSC [1].

Processing model

The server shall process the components as indicated in the credentials/info description of the CSC [1].

The server shall process each component as indicated in the clause 8.9.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

8.9.3 XML related component

The element that shall be the main component for requesting certificate information retrieval shall be the root element of the message etsisig:InformationRequest.

The etsisig:InformationRequest element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationRequest" type="etsisig:InformationRequestType"/>

<xs:complexType name="InformationRequestType">
    <xs:complexContent>
      <xs:extension base="dsb:RequestBaseType">
        <xs:sequence>
          <xs:element name="OptionalInputs" type="dss2:OptionalInputsType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The etsisig:GetCertificateInformation, child element of the dss2:OptionalInputs element shall be set to true.

```
<xs:element name="GetCertificateInformation" type="xs:boolean"/>
```

The dss2:KeySelector, child element of the dss2:OptionalInputs element shall be set in order to identify the signing credential whose information are needed to be returned as main output of the profile.

8.10 Profile for certificate information retrieval response (I)

8.10.1 Profile semantics

This profile shall be used to return some information about a signing certificate.

This profile includes the following components:

Ref	Component for	Presence
7.12	notifying operation result(s)	**M**
7.17	returning credential authorization mode	**O**
7.19	returning SCAL level required	**O**
7.22	returning signing certificate information	**O**
7.33	returning signing credential authorization information	**O**

8.10.2 JSON related component

The element to respond to retrieve user's credential list shall be the set of parameters returned in the response of the credentials/info method as specified in CSC [1].

8.10.3 XML related component

The element that shall be the main component for responding with certificate information retrieval shall be the root element of the message etsisig:InformationResponse.

The etsisig:InformationResponse element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationResponse" type="etsisig:InformationResponseType"/>
  <xs:complexType name="InformationResponseType">
    <xs:complexContent>
      <xs:extension base="dsb:ResponseBaseType">
        <xs:sequence>
          <xs:element name="OptionalOutputs" type="dss2:OptionalOutputsType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

The etsisig:KeyInfo, child element of the dss2:OptionalOutputs element will return the requested signing certificate and key information (not defined in DSS-X, to be newly specified).

8.11 Profile for service information request (J)

8.11.1 Profile semantics

This component may returns several information about the SCS and the list of the functionalities implemented and supported by it.

Ref	Component for	Presence
7.10	language and culture selection	**O**

8.11.2 JSON related component

The component to request service information shall be the set of required parameters required by the `info` method as specified in CSC [1].

Processing model

The server shall process the components as indicated in the `info` description of the CSC [1].

The server shall process each component as indicated in the clause 8.11.1 of the present document if the child is not specified in any of the referenced CSC [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC [1].

8.11.3 XML related component

The element that shall be the main component for requesting service information shall be the root element of the message `etsisig:InformationRequest`.

The `etsisig:InformationRequest` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationRequest" type="etsisig:InformationRequestType"/>

<xs:complexType name="InformationRequestType">
    <xs:complexContent>
      <xs:extension base="dsb:RequestBaseType">
        <xs:sequence>
          <xs:element name="OptionalInputs" type="dss2:OptionalInputsType"
minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
</xs:complexType>
```

The `etsisig:GetServiceInformation`, child element of the `dss2:OptionalInputs` element shall be set to true.

```
<xs:element name="GetServiceInformation" type="xs:boolean"/>
```

The `dsb:Language`, child element of the `dss2:OptionalInputs` element can be set for language and culture selection.

8.12 Profile for service information response (K)

8.12.1 Profile semantics

This profile includes the following components:

Ref	Component for	Presence
7.20	returning service information	**M**

8.12.2 JSON related component

The component to respond to service information request list shall be the set of values returned in the response of the `info` method as specified in CSC [1].

8.12.3 XML related component

The element that shall be the main component for responding with service information shall be the root element of the message `etsisig:InformationResponse`.

The `etsisig:InformationResponse` element is defined in XML Schema file "[???]", whose location is detailed in clause ?, and is copied below for information.

```
<xs:element name="InformationResponse" type="etsisig:InformationResponseType"/>
  <xs:complexType name="InformationResponseType">
    <xs:complexContent>
      <xs:extension base="dsb:ResponseBaseType">
        <xs:sequence>
          <xs:element name="OptionalOutputs" type="dss2:OptionalOutputsType"
minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
```

The `etsisig:ServiceInformation`, child element of the `dss2:OptionalOutputs` element will return the requested service information (not defined in DSS-X, to be newly specified).

8.13 Component use summary

A = Profile for signature request
B = Profile for signature response
C = Profile for digital signature value request
D = Profile for digital signature value response
E = Profile for asynchronous processing
F = Profile for signing certificates list request
G = Profile for signing certificates list response
H = Profile for certificate information retrieval request
I = Profile for certificate information retrieval response
J = Profile for service information request
K = Profile for service information response

Ref.	Component for:	A	B	C	D	E	F	G	H	I	J	K
7.2	asynchronous/synchronous operation mode selection	O		O								
7.3	correlating response to corresponding request		O		O							
7.4	credential authorization	O		O								
7.5	defining optional data to be returned	O		O								
7.6	defining the validity period for asynchronous requests	O		O								
7.7	identification of the request	O		O			O		O			
7.8	identifying signature credentials	M										
7.9	language and region selection	O		O		O	O		O		O	
7.10	list the certificate chain								O			
7.11	managing digital signatures transactions	O		O								
7.12	notifying operation result(s)		M		M			M		M		
7.13	optional signature attributes/properties selection	O										
7.14	polling results					M						
7.15	protocol identifier	M		M		M	M		M			
7.16	requesting specific signature formats	O										
7.17	returning credential authorization mode									O		
7.18	returning DSV				O							
7.19	returning SCAL level required									O		
7.20	returning service information											M
7.21	returning signed documents or signatures		O									
7.22	returning signing certificate information		O		O					O		
7.23	returning the list of the signing certificate(s)							O				

7.24	service authentication	**O**		**O**		**O**	**O**		**O**			
7.25	service policy identification		**O**		**O**							
7.26	service policy selection	**O**		**O**								
7.27	signature creation policy identification		**O**		**O**							
7.28	signature creation policy selection	**O**		**O**								
7.29	signer identification						**M**					
7.30	specifying response URL	**O**		**O**								
7.31	submitting document(s) or hash(es) of document(s) to be signed	**M**										
7.32	submitting DTBSR(s)			**M**								
7.33	returning signing credential authorization information									**O**		

***

History

Document history		
v0.0.0.a	December 2017	For discussions between STF539 members.
v0.0.1	February 2018	First draft
v0.0.2	June 2018	New draft to be discussed at ESI#63 meeting
v0.0.3	June 2018	New draft to be discussed at ESI#63 meeting, added some previously missing subclauses
V0.0.4	June 2018	New draft after comments received at ESI#63 meeting
V0.0.5	July 2018	Stable draft for public review