ETSI TS 119 432 V1.1.1 (2019-03)

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

Reference
DTS/ESI-0019432

Keywords
electronic signature, protocol, remote, security,
trust services

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

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.
The content of the PDF version shall not be modified without the written authorization of ETSI.
The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2019.
All rights reserved.

DECT™, **PLUGTESTS**™, **UMTS**™ and the ETSI logo are trademarks of ETSI registered for the benefit of its Members.
3GPP™ and **LTE**™ are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.
oneM2M™ logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners.
GSM® and the GSM logo are trademarks registered and owned by the GSM Association.

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables 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 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 [i.1] (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 TS 119 102-1 [i.7] 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: 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.

Finally, the present document specifies two bindings, each one in a different syntax (XML and JSON), for each of the protocols mentioned above.

As far as it has been possible and suitable, the protocols have re-used constructs of CSC JSON and OASIS DSS-X XML specifications. When this has not been possible the present document specifies new components semantically and also syntactically in the two formats: XML and JSON.

The authorized signer's use of its key for signing requires users to provide multiple proofs of their claimed identity before being granted access to the needed set of resources. The way in which the user identity verification process is carried out by the service provider or any suggestion concerning the usage of multi-factor authentication mechanisms is out of the scope of the present document.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. 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 https://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 (Version 1.0.3.0): "Architectures and protocols for remote signature applications".

NOTE: Available at https://cloudsignatureconsortium.org/wp-content/uploads/2019/02/CSC_API_V1_1.0.3.0.pdf.

[2] OASIS Standard: "Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0", Committee Specification Draft 03 / Public Review Draft 03.

NOTE: Available at https://www.oasis-open.org/committees/download.php/64707/dss-core-v2.0-wd12-package-for-CSD03-PRD01.zip.

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

NOTE: Available at https://www.oasis-open.org/committees/download.php/63125/oasis-dssx-2.0-profiles-ades%20WD%2002.docx.

[4] IETF RFC 5280: "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".

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

NOTE: Available at https://docs.oasis-open.org/dss/v1.0/oasis-dss-profiles-asynchronous_processing-spec-cs-v1.0-r1.pdf.

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

[7] IETF RFC 5646: "Tags for Identifying Languages".

[8] IETF RFC 4514: "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names".

[9] IETF RFC 3061: "A URN Namespace of Objects Identifiers".

[10] IETF RFC 7468: "Textual Encodings of PKIX, PKCS, and CMS Structures".

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

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

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

[14] ISO 3166-1: "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. 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] CEN EN 419 221-5: "Protection profiles for TSP Cryptographic modules – Part 5: Cryptographic Module for Trust Services".

[i.3] ETSI TR 119 001: "Electronic Signatures and Infrastructures (ESI); The framework for standardization of signatures; Definitions and abbreviations".

[i.4] ETSI TS 119 312: "Electronic Signatures and Infrastructures (ESI); Cryptographic Suites".

[i.5] IETF RFC 6749: "The OAuth 2.0 Authorization Framework".

[i.6] IETF RFC 7519: "JSON Web Token (JWT)".

[i.7] ETSI TS 119 102-1 (V1.2.1): "Electronic signatures and infrastructures (ESI); Procedures for Creation and Validation of AdES Digital Signatures; Part 1: Creation and Validation".

[i.8] IETF RFC 8017: "PKCS #1: RSA Cryptography Specifications Version 2.2".

[i.9] ETSI TS 103 173: "Electronic Signatures and Infrastructures (ESI); CAdES Baseline Profile".

[i.10] ETSI TS 103 171: "Electronic Signatures and Infrastructures (ESI); XAdES Baseline Profile".

[i.11] ETSI TS 103 172: " Electronic Signatures and Infrastructures (ESI); PAdES Baseline Profile".

[i.12] ETSI TS 119 431-1: "Electronic signatures and infrastructures (ESI); Policy and security requirements for trust service providers; Part 1: TSP service components operating a remote QSCD / SCDev".

[i.13] ETSI TS 119 431-2: "Electronic signatures and infrastructures (ESI); Policy and security requirements for trust service providers; Part 2: TSP service components supporting AdES digital signature creation".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TR 119 001 [i.3] and the following apply:

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

client application: application running in a signer's environment that accesses the services made available by the SCASC and/or the SSASC

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 providing control of signing operation on the signer's behalf

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 device: configured software or hardware used to implement the signature creation data and to create a digital signature value

signature creation policy: set of signature creation constraints processed or to be processed by the SCASC or the SSASC

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

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

NOTE: As in CEN EN 419 241-1 [6].

signature credential: set of the signing key and the corresponding signing certificate

3.2 Symbols

Void.

3.3 Abbreviations

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

API	Application Program Interface
ASN	Abstract Syntax Notation
CA	Certification Authority
CEN	European Committee for Standardization
CRL	Certificate Revocation List
CSC	Cloud Signature Consortium
DSS-X	Digital Signature Services eXtended
DSV	Digital Signature Value
DTBS	Data To Be Signed
DTBSF	Data To Be Signed Formatted
DTBSR	Data To Be Signed Representation
ECDSA	Elliptic Curve Digital Signature Algorithm
EN	European Norm
HTTP	Hyper Text Transfer Protocol
ISO	International Organization for Standardization
JPEG	Joint Photographic Experts Group
JSON	Java Script Object Notation
JWT	JSON Web Token
LT	Long Term
LTA	Long Term Archival
OASIS	Organization for the Advancement of Structured Information Standards
OCSP	Online Certificate Status Protocol
OID	Object Identifier
PNG	Portable Network Graphics
QES	Qualified Electronic Signature
QSCD	Qualified electronic Signature Creation Device
RA	Registration Authority
RSA	Rivest, Shamir, & Adleman
SAD	Signature Activation Data
SAM	Signature Activation Module
SAML	Security Access Markup Language
SCA	Signature Creation Application
SCAL	Sole Control Assurance Level
SCAL1	Sole Control Assurance Level 1

NOTE: As defined in CEN EN 419 241-1 [6].

SCAL2	Sole Control Assurance Level 2

NOTE: As defined in CEN EN 419 241-1 [6].

SCASC	Signature Creation Application Service Component
SCDev	Signature Creation Device
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
TSP	Trust Service Provider
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
XML	Extensible Markup Language
XSD	XML Schema Definition

4 Signature creation process, service decomposition

4.1 Signature creation process steps and data elements

Figure 1 (derived from ETSI TS 119 102-1 [i.7], clause 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 will 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 or signing certificate availability. The signature activation module in the tamper protected area is needed only when the Signature Creation Service (SCS) complies to the sole control assurance level 2 (SCAL2) signature activation mechanism.

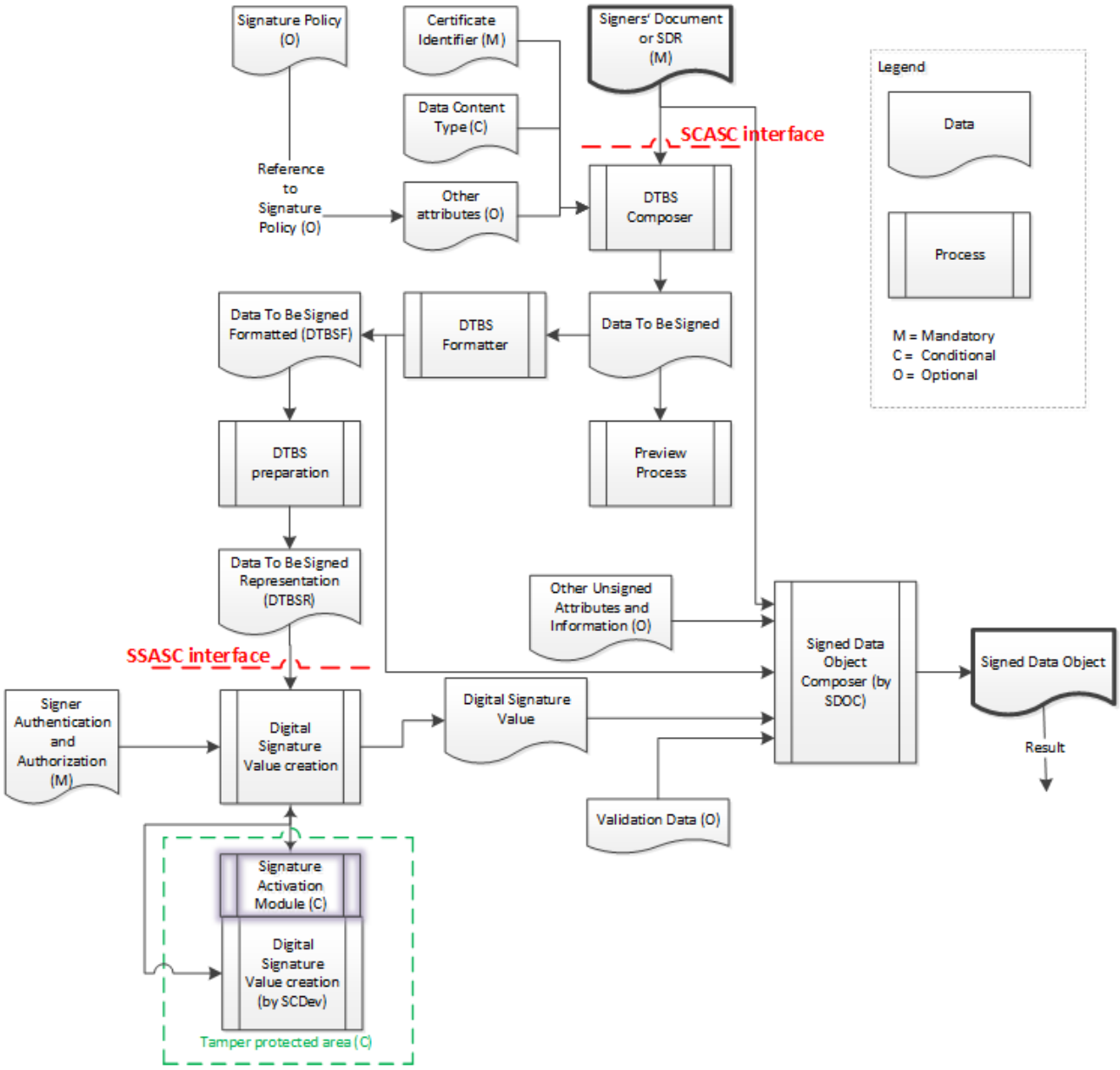

Figure 1: Process Steps and Data Elements in Signature Creation

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.

SCS denotes a TSP service implementing a Signature Creation Application (SCA) and/or a server signing application (SSA).

Some variants of these interfaces are possible depending on the functional split between the SCS 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 is transferred to the SCASC while in the latter case, the SD is 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 the present document 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 or their hash values, whose presence is needed in the DTBS, 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 the DTBSF. The SCASC prepares the entire DTBSF, calculates the hash, and sends the hash value (DTBSR) as input to an 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

4.4.1.1 Introduction

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

4.4.1.2 Signature activation

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

Two different levels of confidence of the control of the signing key, as defined in CEN EN 419241-1 [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 authorized signer's use of its key for signing is enforced by the SSASC which authenticates the signer. The activation of the signing key can remain for a given period and/or for a given number of signatures.

NOTE: 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 authorized 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.

4.4.1.3 Signature creation by SCDev

The signature creation process is performed by the SCDev. In the context of the present document, 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 can be used to generate 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

5.1 Overview

This clause describes the architectures of systems supporting remote server signing pointing out the fundamental interactions of SCASC and SSASC 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 (providing QES and/or AdES) includes a SCASC that is connected to a SSASC hosting the remote SCDev. 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 client application sends to the signature creation service provider requests to generate one or more AdESs;
- the signature client 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 SCASC and SSASC 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 are presented in the following clauses in which SCASC and SSASC 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 SCASC and/or SSASC 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 case of an SCS complying to SCAL2, the TSP protected environment includes a tamper protected device (e.g. cryptographic module conforming to CEN EN 419 221-5 [i.2]).

The signer's environment is local to the signer and its protection is under the 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 present document and are shown 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's 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 an SCASC or an SSASC depending on whether the SCSP is hosting an SCASC and/or an 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 the 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.

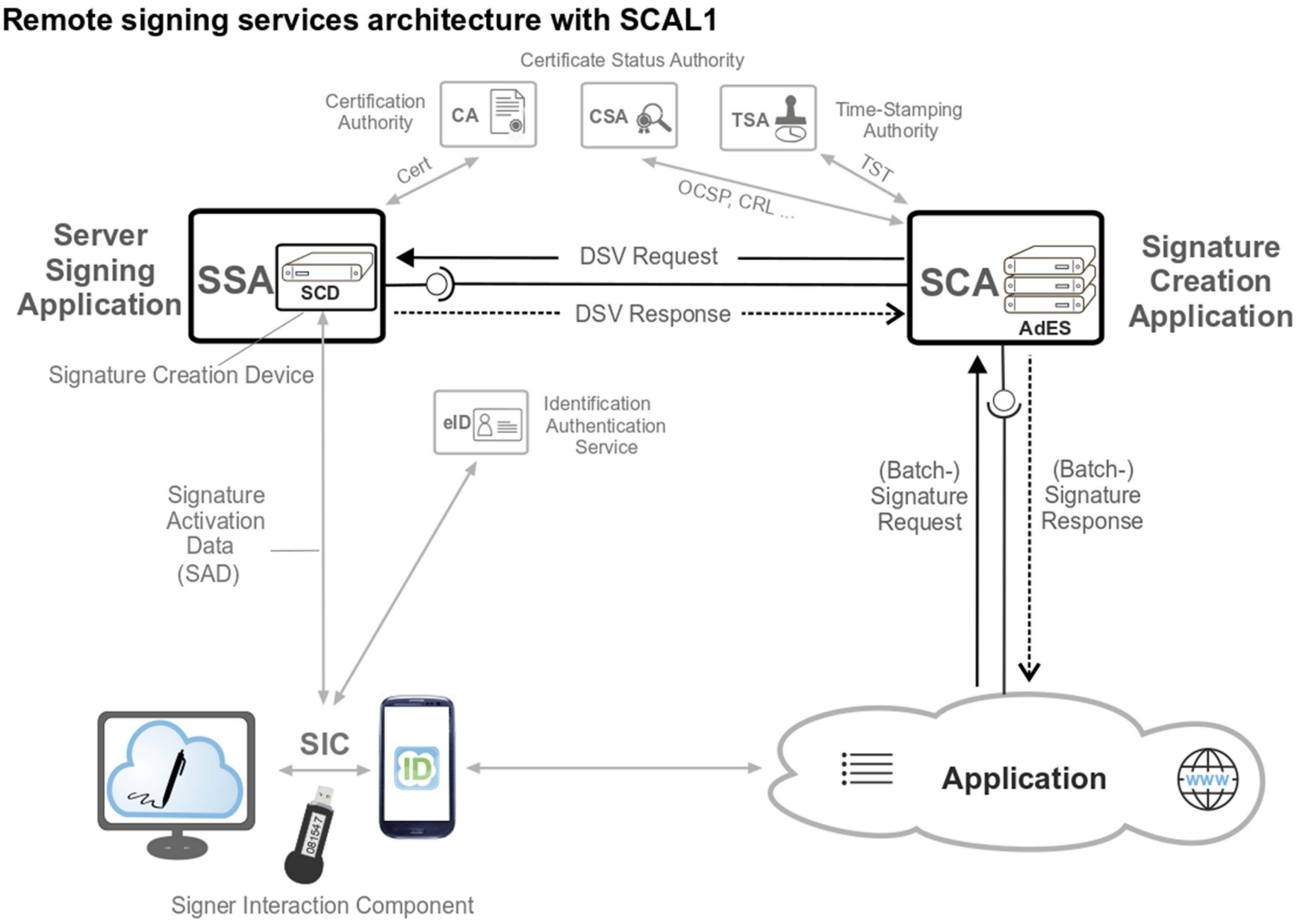
Remote signing services architecture with SCAL1
Certificate Status Authority
Certification Authority
CA
CSA
TSA
Time-Stamping Authority
Cert
OCSP, CRL ...
TST
Server Signing Application
SSA
SCD
DSV Request
DSV Response
SCA
AdES
Signature Creation Application
Signature Creation Device
eID
Identification Authentication Service
Signature Activation Data (SAD)
(Batch-) Signature Request
(Batch-) Signature Response
SIC
ID
Application
www
Signer Interaction Component

Figure 2

5.4 Remote signing services with SCAL2

In this model a third main environment is defined in addition to the signer's and the TSP's 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 SSASC. 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.

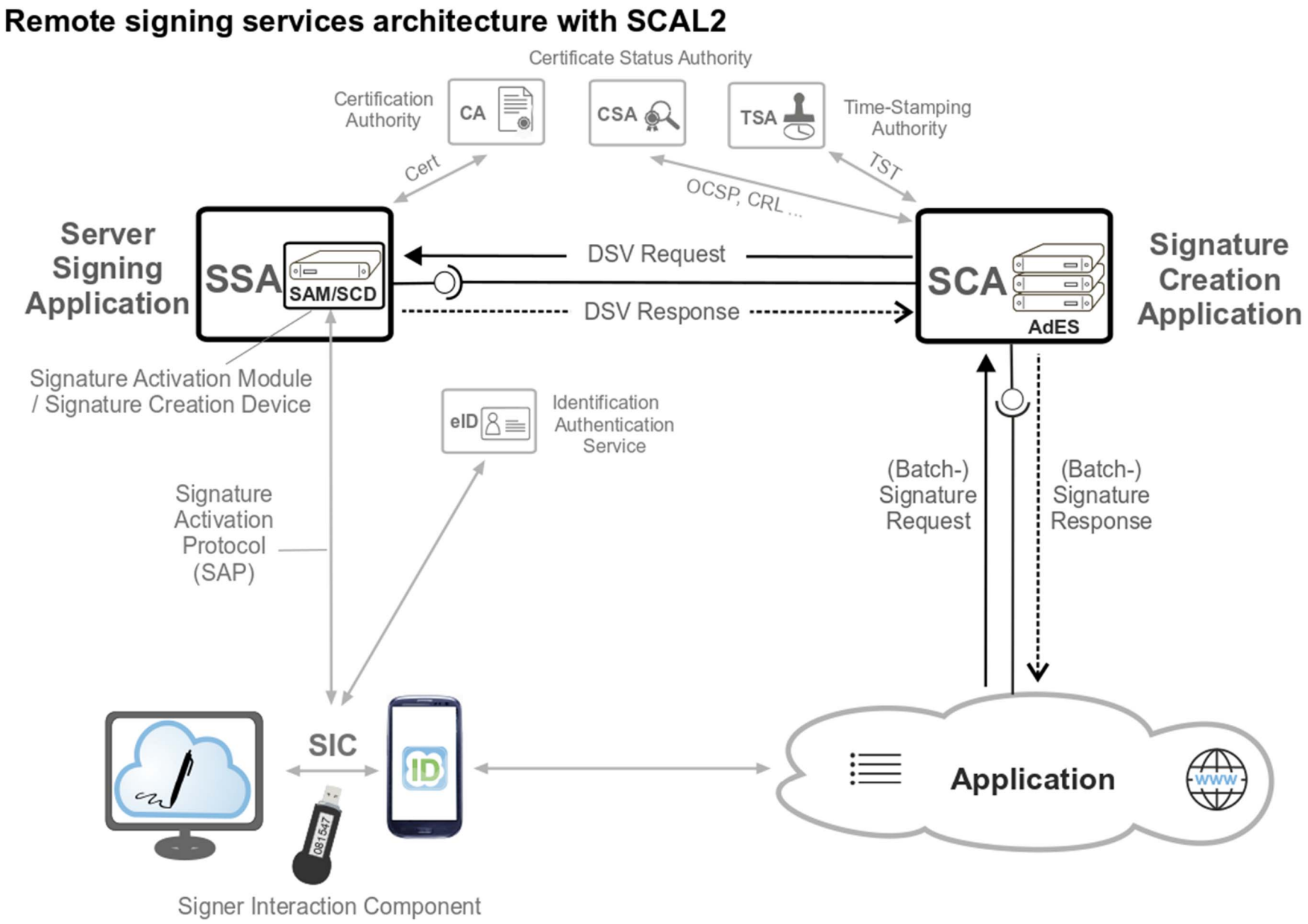
Remote signing services architecture with SCAL2
Certificate Status Authority
Certification Authority
CA
CSA
TSA
Time-Stamping Authority
Cert
OCSP, CRL ...
TST
Server Signing Application
SSA
SAM/SCD
DSV Request
DSV Response
SCA
AdES
Signature Creation Application
Signature Activation Module / Signature Creation Device
eID
Identification Authentication Service
Signature Activation Protocol (SAP)
(Batch-) Signature Request
(Batch-) Signature Response
SIC
Application
Signer Interaction Component

Figure 3

5.5 Security, integrity and confidentiality

ETSI TS 119 431-2 [i.13] provides requirements for TSPs operating an SCASC supporting AdES digital signature creation.

ETSI TS 119 431-1 [i.12] provides requirements for TSPs operating an SSASC supporting digital signature value creation.

6 Protocol profiles specification

6.1 Introduction

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

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

As far as it has been possible and suitable, the profiles specified in the present document have taken as starting point a number of OASIS DSS-X Technical Committees' specifications, namely OASIS DSS-v2.0 [2], OASIS AdES profiles for DSS-v2.0 [3] and the CSC standard [1].

The rest of the present document is organized as follows:

- Clauses 6.2 and 6.3 provide general remarks on the XML and JSON 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 messages that client applications and SCSs can exchange by using the components defined in clause 7.

For each component of the protocols mentioned above, 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. These requirements are defined in clauses named "XML related component".
- Defines requirements for the JSON. These requirements are defined in clauses named "JSON related component".

6.2 OASIS DSS-X XML related protocol

The structures described in the present document are contained in the schema files [DSS_Core_XSD], [XMLDSIG_XSD] and the four xml schema files equally identified with [XSDSIGCREATIONPROT] throughout the present document and whose location is defined in annex A. 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#.

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

Table 1

URI value of the XML Namespace	Prefix
http://uri.etsi.org/19432/v1.1.1#	etsisig
http://www.w3.org/2000/09/xmldsig#	ds
http://docs.oasis-open.org/dss-x/ns/base	dsb
http://docs.oasis-open.org/dss-x/ns/core	dss2

The present document references components in the schema files mentioned above and further profiles some of them.

In the absence of any further requirement defined in the present document, the requirements defined in the schema files mentioned above for each element present in the present document shall apply.

In addition the present document specifies elements that are not specified in the schema files mentioned above. For these elements, the present document also defines the processing model for the server. 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 the present document are contained in the schema files [ETSI_SIG_CORE_JSCHEMA].

In addition, the present document specifies elements that are not specified in the CSC standard [1]. For these elements the present document also defines the processing model for the server. 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

Clause 7 defines the protocol components 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

The term synchronous operation mode means that the client application, after sending any request to the SCS, will wait for it to finish before moving on to another task. In this operation mode the SCS will perform the requested operation and return the corresponding outcome(s) to the client application. The term asynchronous operation mode means that the client application, after sending any request to the SCS, can move on to another task before the requested operation finishes. In this operation mode the SCS will accept the request and return a notification informing the client application about the acceptance of its request. The client will be able to request the corresponding outcome(s) to the SCS later on.

This component shall be used by a client application to request a synchronous or asynchronous operation mode from the SCS.

NOTE 1: When this component is not specified by the client application, the SCS may take one of the following behaviours to perform the requested operation, according to the selected service policy:

- process the request only in synchronous mode;
- process the request only in asynchronous mode;
- process the request in synchronous mode and decide under particular conditions to process the request in asynchronous mode.

NOTE 2: When the asynchronous processing is initiated by the SCS it returns a response including:

- a result code informing the client application about the requested operation processing in asynchronous mode (see clause 7.24); and
- a unique response identifier that the client application will use to obtain the outcome(s) of the requested operation from the SCS (see clause 7.26).

Details on the asynchronous processing mode are described in clause 8.4.

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 table 2.

Table 2

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

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.

The `OperationMode` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, 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>
```

7.2.4 Processing model

When this component is not provided by the client application, the SCS shall take one of the following behaviours to perform the requested operation, according to the selected service policy:

- process the request only in synchronous mode; or
- process the request only in asynchronous mode; or
- process the request in synchronous mode and decide under particular conditions to process the request in asynchronous mode.

When the synchronous processing is adopted by the SCS it shall return:

- the response defined in clause 8.2.2 in case of AdES signatures creation requests;
- the response defined in clause 8.3.2 in case of DSVs creation requests.

When the asynchronous processing is adopted by the SCS it shall return an initial response including:

- a result code informing the client application about the requested operation processing in asynchronous mode (see clause 7.24); and

- a unique response identifier that the client application will use to obtain the outcome(s) of the requested operation from the SCS (see clause 7.26).

The asynchronous processing mode is defined in clause 8.4.

7.3 Component for identification of the request

7.3.1 Component semantics

This component contains a string value included by the client application and is used to correlate requests with subsequent corresponding responses or to poll asynchronous requests outcome(s).

7.3.2 JSON related component

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

- `requestID` string type parameter.

Specified according to table 3.

Table 3

Parameter	Type	Description
requestID	String	Data from the client application generally used to handle a signature transaction identifier.

7.3.3 XML related component

The element for correlating requests with subsequent responses shall be the `RequestID` attribute of the `dss2:SignRequest` root element and the `RequestId` attribute of the `dss2:SignResponse` root element defined in OASIS DSS-v2.0 [2], clause 4.3.1.

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 table 4.

Table 4

Parameter	Type	Description
SAD	*String*	Authentication data used to authorize the use of the signing key.

The "SAD" element is defined in clause 11.6 of the CSC standard [1].

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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

```
<xs:element name="SignatureActivationData" type="etsisig:SignatureActivationDataType" />
<xs:complexType name="SignatureActivationDataType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="type" type="xs:string" use="optional" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

7.5 Component for defining optional data to be returned

7.5.1 Component semantics

This component defines a set of additional outputs associated with the processing of a signature or certificate information retrieval request and shall be used by the client application to define optional data to be returned from the SCS. This element shall contain a list of element names of the optional data requested.

Below follows a list of the sub-components that constitute this component:

- The optional `ReturnSigningCertificateInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain various information concerning the signing certificate/chain/key used by the SCS to perform the signature or identified in the certificate information retrieval request.
- The component defined in clause 7.10 shall be used in order to specify to the SCS which information concerning the signing certificate/chain/key shall be returned.
- The optional `ReturnSupportMultiSignatureInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain the information whether the signing key supports the creation of multiple signatures with a single authorization request.
- The optional `ReturnServicePolicyInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain the name of the service policy used by the server to perform the requested operation.
- The optional `ReturnSignatureCreationPolicyInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain the name of the signature creation policy used by the server to perform the requested signature creation operation.
- The optional `ReturnCredentialAuthorizationModeInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain the authorization mode required by the signing key identified in the certificate information retrieval request.
- The optional `ReturnSoleControlAssuranceLevelInfo` element, if present, shall contain a Boolean. Its default value is 'false'. This element can be used by the client application to obtain the sole control assurance level required by the signing key identified in the certificate information retrieval request.

7.5.2 JSON related component

The component for specifying the optional data to be returned from the SCS shall be represented by the following parameters:

- `ReturnSigningCertificateInfo`
- `ReturnSupportMultiSignatureInfo`
- `ReturnServicePolicyInfo`
- `ReturnSignatureCreationPolicyInfo`
- `ReturnCredentialAuthorizationModeInfo`
- `ReturnSoleControlAssuranceLevelInfo`

Specified according to table 5.

Table 5

Parameter	Presence	Type
`ReturnSigningCertificateInfo`	Optional	Boolean
`ReturnSupportMultiSignatureInfo`	Optional	Boolean
`ReturnServicePolicyInfo`	Optional	Boolean
`ReturnSignatureCreationPolicyInfo`	Optional	Boolean
`ReturnCredentialAuthorizationModeInfo`	Optional	Boolean
`ReturnSoleControlAssuranceLevelInfo`	Optional	Boolean

The "ReturnSigningCertificateInfo", "ReturnSupportMultiSignatureInfo", "ReturnServicePolicyInfo", "ReturnSignatureCreationPolicyInfo", "ReturnCredentialAuthorizationModeInfo" and "ReturnSoleControlAssuranceLevelInfo" elements are not defined in CSC standard [1].

7.5.3 XML related component

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

The `ReturnOptionalData` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

```
<xs:element name="ReturnOptionalData" type="etsisig:ReturnOptionalDataType"/>
<xs:complexType name="ReturnOptionalDataType">
    <xs:sequence>
        <xs:element ref="etsisig:SigningCertificateInfo" minOccurs="0"/>
        <xs:element ref="etsisig:SupportMultiSignatureInfo" minOccurs="0"/>
        <xs:element ref="etsisig:ServicePolicyInfo" minOccurs="0"/>
        <xs:element ref="etsisig:SignatureCreationPolicyInfo" minOccurs="0"/>
        <xs:element ref="etsisig:CredentialAuthorizationModeInfo" minOccurs="0"/>
        <xs:element ref="etsisig:SoleControlAssuranceLevelInfo" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="SigningCertificateInfo" type="xs:Boolean"/>
<xs:element name="SupportMultiSignatureInfo" type="xs:Boolean"/>
<xs:element name="ServicePolicyInfo" type="xs:Boolean"/>
<xs:element name="SignatureCreationPolicyInfo" type="xs:Boolean"/>
<xs:element name="CredentialAuthorizationModeInfo" type="xs:Boolean"/>
<xs:element name="SoleControlAssuranceLevelInfo" type="xs:Boolean"/>
```

7.5.4 Processing model

If the SCS does not recognize or cannot handle any optional data to be returned, it shall reject the request and return an error.

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 until which the SCS shall keep the request outcome(s) available for the client application retrieval. The validity period shall be calculated starting from the moment of the request acceptance. The SCS needs not to return any outcome(s) after the maximum period of time expiration and may destroy any outcome(s) already produced.

7.6.2 JSON related component

The component for requesting the maximum period of time setting shall be represented by the following:

- `validity_period` number type parameter.

Specified according to table 6.

Table 6

Parameter	Type	Description
validity_period	Integer	Maximum period of time expressed in milliseconds.

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 can be completed.

The `ValidityPeriod` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

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

7.6.4 Processing model

If the component is present, the SCS shall perform the requested operations and keep available the corresponding outcome(s) within the period of time specified in this component. If the SCS does not complete the requested operations or the client application does not request the available outcome(s) within the specified period of time, the processing of the client application request shall not be completed successfully.

7.7 Component for the client application authentication

7.7.1 Component semantics

This component shall contain information to authenticate the client application to access to the SCASC or the SSASC.

NOTE 1: The way a client application authenticates to the SCASC or SSASC is out of scope of the present document.

NOTE 2: The SCSP can define other ways in which a client application can authenticate to the SCASC or SSASC in addition or alternatively to this component usage.

7.7.2 JSON related component

The authorization component shall be included into the Authorization HTTP header of every call.

7.7.3 XML related component

The element for the client application authentication shall be `dss2:ClaimedIdentity`, child element of the `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.3.9. The component semantics are defined as being the identification of the requesting service. The precise authentication mechanism (i.e. the usage of SupportingInfo, etc.) is out of scope for this document and will be defined by the implementing service.

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 table 7.

Table 7

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

The "credentialID" element is defined in clause 11.6 of the CSC standard [1].

7.8.3 XML related component

The element for identifying signature credentials shall be the `dss2:KeySelector`, child element of `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.3.12. The contents of `KeySelector` shall be processed in the same way as it is specified in the document mentioned above.

7.9 Component for language 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 IETF RFC 5646 [7].

The service should 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 the SCS default language.

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 table 8.

Table 8

Parameter	Type	Description
Lang	*String*	requested response language specified as in IETF RFC 5646 [7].

The "Lang" element is defined in clause 11.1 of the CSC standard [1].

7.9.3 XML related component

The element for language and culture selection shall be `dsb:Language`, child element of the `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.1.9.

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 is not defined the SCS shall return only the end-entity certificate.

7.10.2 JSON related component

The component for specifying which contents of the certificate chain shall be returned by the SCS shall be represented by the following:

- `certificates` string type parameter.
- `authInfo` Boolean type parameter.
- `certInfo` Boolean type parameter.

Specified according to table 9.

Table 9

Parameter	Presence	Type	Description
`certificates`	*Optional*	*String*	*Specifies which certificates from the certificates chain shall be returned in the SCS 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".*
`certInfo`	*Optional conditional*	*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. This element shall carry a value only if the parameter certificates contains a value different from "none".
`authInfo`	*Optional conditional*	*Boolean*	Specifies if the information on the authorization mechanisms supported by this credential shall be returned. The default value is "false", so if the parameter is omitted then the information will not be returned. This element shall carry a value only if the parameter certificates contains a value different from "none".

The "certificates", "certInfo" and "authInfo" elements are defined in clause 11.5 of the CSC standard [1].

7.10.3 XML related component

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

The `ReturnSigningCertificate` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

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

```
<xs:complexType name="ReturnSigningCertificateType">
    <xs:attribute name="ReturnCertificates" type="etsisig:ReturnCertificatesType" use="optional" />
    <xs:attribute name="CertificateInfo" type="xs:Boolean" use="optional" />
    <xs:attribute name="AuthorizationInfo" type="xs:Boolean" use="optional" />
</xs:complexType>

<xs:simpleType name="ReturnCertificatesType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="None" />
        <xs:enumeration value="Single" />
        <xs:enumeration value="Chain" />
    </xs:restriction>
</xs:simpleType>
```

7.10.4 Processing model

If the component is present, the SCS shall check the values passed in the component parameters.

If the parameter indicating which certificates shall be returned is present the SCS shall return in its response no certificate, only the signing certificate or the full signing certificate chain if the parameter value is respectively "none", "single" or "chain".

If the parameter indicating if signing certificate information shall be returned in the form of strings is valued "true" the SCS shall return in its response the signing certificate information as specified in clause 7.22.

If the parameter indicating if signing key and certificate authorization information shall be returned is valued "true" the SCS shall return in its response the signing key and certificate authorization information as specified in clause 7.28.

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 7.3 that will be handled in order to create a transaction identifier uniquely within the SCS.

EXAMPLE 1: A working transaction in which a signer needs to put in a document a certain number of PAdES signatures in different parts in order to acknowledge different contents in the document. In such a case, a new DTBS is computed before every new signature. Using a working transaction to complete all the signatures allows the signer to have a better control of the whole signature process.

EXAMPLE 2: 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 signer action. The session key would be used to be sure that subsequent documents belong to the same transaction.

NOTE: 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:

- `numSignatures` integer parameter.

Specified according to table 10.

Table 10

Parameter	Type	Description
numSignatures	*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.

The "numSignatures" element is defined in clause 11.6 of the CSC standard [1].

7.11.3 XML related component

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

The `etsisig:NumberOfSignatures` element shall be defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information.

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

7.11.4 Processing model

If the component is present, the SCS shall check and store the value passed in the component parameter. The SCS response shall include a unique identifier in the component for the identification of response as defined in clause 7.26. Any further signature request that contains this unique identifier in the component for request identification, shall be considered being part of the signature transaction. The response to any signature request being part of the transaction shall include the same value in the component for response identification. After the SCS will have received the number of signature requests expected in the transaction any further requests including the same request identification value shall be refused by the SCS that shall return an unsuccessful response.

7.12 Component for service policy selection

7.12.1 Component semantics

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

7.12.2 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 table 11.

Table 11

Parameter	Type	Description
policy	String	The element that identifies a particular service 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.

7.12.3 XML related component

The element for service policy selection shall be `dsb:ServicePolicy`, child element of the `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.1.9.

7.13 Component for signature creation policy selection

7.13.1 Component semantics

This component shall contain the signature creation policy that shall be used while signing the DTBSR(s) and 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 OID mentioned above as specified in IETF RFC 3061 [9].

Alternatively to an explicit identification of the signature creation policy the information specifying the signature algorithm to be used may be provided.

7.13.2 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;
- `signAlgo` string type parameter;
- `signAlgoParams` string type parameter.

Specified according to table 12.

Table 12

Parameter	Presence	Type	Description
signaturePolicyID	Required Conditional	*String*	The element that identifies a particular signature creation policy associated with the SCS. This element shall carry a value if the signAlgo parameter is not specified.
signAlgo	*Required* *Conditional*	*String*	The element specifies the algorithm OID used for signing. This element shall carry a value if the signaturePolicyID parameter is not specified.
signAlgoParams	*Optional* *Conditional*	*String*	The element specifies the Base64-encoding of the DER-encoded ASN.1 signature parameters. This element may carry a value only if required and/or allowed by the signature algorithm like, for example, the RSA-PSS cryptographic signature scheme (IETF RFC 8017 [i.8]).

The "signaturePolicyID" element is not defined in CSC standard [1]. The "signAlgo" and "signAlgoParams" elements are defined in clause 11.9 of the CSC standard [1].

7.13.3 XML related component

The element for signature algorithm selection shall be `dss2:SignatureAlgorithm`, child element of the `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.3.4.

The element for signature policy selection shall be `etsisig:SignaturePolicyId`, child element of the `dss2:OptionalInputs` element.

The element for signature algorithm parameters shall be `etsisig:SignatureAlgorithmParameters`, child element of the `dss2:OptionalInputs` element.

7.14 Component for optional signature attributes/properties selection

7.14.1 Component semantics

The request may include this component if there are certain signed attributes/properties that the SCASC is requested to include in the signature.

Below follows a list of the attributes/properties names that can be referenced in this component in order to request the inclusion of the corresponding signed attributes/properties in the signature. As an alternative to the attributes/properties names it is also possible using the corresponding attributes/properties oids:

- The `commitment-type-indication` element, if present, shall contain the Base64-encoding of the attribute commitment-type-indication defined in clause 5.2.3 of ETSI EN 319 122-1 [11].
- The `content-hints` element, if present, shall contain the Base64-encoding of the attribute content-hints defined in clause 5.2.4.1 of ETSI EN 319 122-1 [11].
- The `mime-type` element, if present, shall contain the Base64-encoding of the attribute mime-type defined in clause 5.4.2.2 of ETSI EN 319 122-1 [11].
- The `signer-location` element, if present, shall contain the Base64-encoding of the attribute signer-location defined in clause 5.2.5 of ETSI EN 319 122-1 [11].
- The `signer-attributes-v2` element, if present, shall contain the Base64-encoding of the attribute signer-attributes-v2 defined in clause 5.2.6.1 of ETSI EN 319 122-1 [11].
- The `content-time-stamp` element, if present, shall contain the Base64-encoding of the attribute content-time-stamp defined in clause 5.2.8 of ETSI EN 319 122-1 [11].
- The `signature-policy-identifier` element, if present, shall contain the Base64-encoding of the attribute signature-policy-identifier defined in clause 5.2.9 of ETSI EN 319 122-1 [11].
- The `content-reference` element, if present, shall contain the Base64-encoding of the attribute content-reference defined in clause 5.2.11 of ETSI EN 319 122-1 [11].
- The `content-identifier` element, if present, shall contain the Base64-encoding of the attribute content-identifier defined in clause 5.2.12 of ETSI EN 319 122-1 [11].
- The `Location` element, if present, shall contain the Base64-encoding of the attribute Location defined in clause 5.3 of ETSI EN 319 142-1 [13].
- The `Reason` element, if present, shall contain the Base64-encoding of the attribute Reason defined in clause 5.3 of ETSI EN 319 142-1 [13].
- The `Name` element, if present, shall contain the Base64-encoding of the attribute Name defined in clause 5.3 of ETSI EN 319 142-1 [13].
- The `ContactInfo` element, if present, shall contain the Base64-encoding of the attribute ContactInfo defined in clause 5.3 of ETSI EN 319 142-1 [13].
- The `SignerRoleV2` element, if present, shall contain the Base64-encoding of the attribute SignerRoleV2 defined in clause 5.2.6 of ETSI EN 319 132-1 [12].
- The `CommitmentTypeIndication` element, if present, shall contain the Base64-encoding of the attribute CommitmentTypeIndication defined in clause 5.2.3 of ETSI EN 319 132-1 [12].
- The `SignatureProductionPlaceV2` element, if present, shall contain the Base64-encoding of the attribute SignatureProductionPlaceV2 defined in clause 5.2.5 of ETSI EN 319 132-1 [12].
- The `AllDataObjectsTimeStamp` element, if present, shall contain the Base64-encoding of the attribute AllDataObjectsTimeStamp defined in clause 5.2.8.1 of ETSI EN 319 132-1 [12].

- The `IndividualDataObjectsTimeStamp` element, if present, shall contain the Base64-encoding of the attribute IndividualDataObjectsTimeStamp defined in clause 5.2.8.2 of ETSI EN 319 132-1 [12].
- The `SignaturePolicyIdentifier` element, if present, shall contain the Base64-encoding of the attribute SignaturePolicyIdentifier defined in clause 5.2.8.2 of ETSI EN 319 132-1 [12].

7.14.2 JSON related component

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

- `signed_props` containing the list of attributes to be added to the signatures' signed attributes.

Specified according to table 13.

Table 13

Parameter	Type	Description
signed_props	Array of Attribute Object	List of signed attributes. The attributes that may be included depend on the signature format and the signature creation policy.

The 'Attribute Object' is a JSON Object composed by the following attributes:

- `attribute_name` string type parameter.
- `attribute_value` array of string type parameter.

Specified according to table 14.

Table 14

Parameter	Presence	Type	Description
attribute_name	Required	String	Name or OID of the attribute/property to be included in the signature. The attributes and/or properties names defined in clause 7.14.1 shall be used. Other attributes and/or properties whose names are defined in the table in clause 6.3 of ETSI EN 319 122-1 [11], ETSI EN 319 132-1 [12], ETSI EN 319 142-1 [13] documents may be supported by the SCS.
attribute_value	Conditional	String	Depending on the attribute/property specified in the attribute_name parameter, this parameter contains the value to be used for such attribute/property to be included in the signature. When some element of this parameter is not defined the SCASC shall calculate it, if needed.

7.14.3 XML related component

The element for optional signature attributes/properties shall be `dss2:Properties`, child element to the `dss2:OptionalInputs` element, defined in OASIS DSS-v2.0 [2], clause 4.3.15.

7.14.4 Processing model

The client application 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 may include additional attributes/properties in the signature, even if these ones are not explicitly requested by the client application (i.e. because such attributes/properties are mandated by the signature profiles).

7.15 Component for protocol identifier

7.15.1 Component semantics

This component may be used by the client application to tell 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 protocols defined by the present document.

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

- `http://uri.etsi.org/19432/v1.1.1#/creationprofile#`

The request may contain additional components whose values are identifiers of other protocols 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 protocols defined by the present document.

7.15.2 JSON related component

The component to indicate the protocol used by the client application shall be represented by the following:

- `profile` string type parameter.

Specified according to table 15.

Table 15

Parameter	Type	Description
profile	String	String that identifies the protocol being used by the client application to communicate to SCS.

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`, defined in OASIS DSS-v2.0 [2], clause 4.1.11.

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, defined in ETSI EN 319 122-1 [11], ETSI EN 319 132-1 [12], ETSI EN 319 142-1 [13], should be used. The conformance levels of the "baseline profiles" standards, defined in ETSI TS 103 171 [i.10], ETSI TS 103 172 [i.11], and ETSI TS 103 173 [i.9], may be used.

Table 16

Signature format	Conformance level	URI
CAdES/PAdES/XAdES	AdES-B-B	http://www.etsi.org/ades/191x2/level/baseline/B-B#
CAdES/PAdES/XAdES	AdES-B-T	http://www.etsi.org/ades/191x2/level/baseline/B-T#
CAdES/PAdES/XAdES	AdES-B-LT	http://www.etsi.org/ades/191x2/level/baseline/B-LT#
CAdES/PAdES/XAdES	AdES-B-LTA	http://www.etsi.org/ades/191x2/level/baseline/B-LTA#
CAdES/PAdES/XAdES	AdES-B	http://www.etsi.org/ades/etsits/level/baseline/B-B#
CAdES/PAdES/XAdES	AdES-T	http://www.etsi.org/ades/etsits/level/baseline/B-T#
CAdES/PAdES/XAdES	AdES-LT	http://www.etsi.org/ades/etsits/level/baseline/B-LT#
CAdES/PAdES/XAdES	AdES-LTA	http://www.etsi.org/ades/etsits/level/baseline/B-LTA#

According to the type of signature selected a client may also specify the following signature properties.

Table 17

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

7.16.2 JSON related component

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

- `signature_format` string type parameter;
- `conformance_level` string type parameter;
- `signed_envelope_property` string type parameter.

Specified according to table 18.

Table 18

Parameter	Presence	Type	Description
signature_format	Required	String	The required signature format: • "C" shall be used to request the creation of a CAdES signature; • "X" shall be used to request the creation of a XAdES signature; • "P" shall be used to request the creation of a PAdES signature.
conformance_level	Optional	String	• "AdES-B-B" shall be used to request the creation of a baseline 191x2 level B signature. • "AdES-B-T" shall be used to request the creation of a baseline 191x2 level T signature. • "AdES-B-LT" shall be used to request the creation of a baseline 191x2 level LT signature. • "AdES-B-LTA" shall be used to request the creation of a baseline 191x2 level LTA signature. • "AdES-B" shall be used to request the creation of a baseline etsits level B signature. • "AdES-T" shall be used to request the creation of a baseline etsits level T signature. • "AdES-LT" shall be used to request the creation of a baseline etsits level LT signature. • "AdES-LTA" shall be used to request the creation of a baseline etsits level LTA signature. The parameter is optional. The default baseline level is AdES-B-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 whose possible values depend on the value of the signature_format parameter.

7.16.3 XML related component

The element for requesting a specific signature format shall be the `etsisig:SignatureFormat`, child element of the `dss2:OptionalInputs` element. It shall contain a URI reference to the predefined signature type the server shall produce., with values as specified in OASIS DSS-v2.0 [2].

The element for requesting a specific signature conformance level shall be `etsisig:ConformanceLevel`, child element of the `dss2:OptionalInputs` element. It shall contain a URI reference to the predefined conformance level at which the server shall produce the signature.

The element for specifying the envelope property shall be `etsisig:SignedEnvelopeProperty`. Its possible values depend on the specified envelope type and are defined in 7.16.1.

```
<xs:element name="ConformanceLevel" type="xs:anyURI"/>
<xs:element name="SignatureType" type="xs:anyURI"/>
<xs:element name="SignedEnvelopeProperty" type="xs:anyURI"/>
```

The table defined in clause 7.16.1 lists URIs for the levels specified for AdES signatures in ETSI EN 319 122-1 [11], ETSI EN 319 132-1 [12], ETSI EN 319 142-1 [13], ETSI TS 103 171 [i.10], ETSI TS 103 172 [i.11], and ETSI TS 103 173 [i.9].

7.17 Component for signer identification

7.17.1 Component semantics

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

A request shall contain a component for signer identification.

NOTE: This component can be accepted or rejected by the server based on the current service authentication (i.e. in order to avoid to obtain the list of credentials associated to a signer not bound to the current service authentication).

7.17.2 JSON related component

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

- `SignerIdentity` string type parameter.

Specified according to table 19.

Table 19

Parameter	Type	Description
SignerIdentity	String	The identifier associated to the signer identity.

7.17.3 XML related component

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

The `etsisig:SignerIdentity` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

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

7.18 Component for specifying response URL

7.18.1 Component semantics

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

This component shall contain an absolute URL.

7.18.2 JSON related component

The component for specifying the response URL where the client application requests to receive the notification of the completed operation shall be represented by the following:

- `response_uri` string type parameter.

Specified according to table 20.

Table 20

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

7.18.3 XML related component

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

The `ResponseURL` element shall be defined as in XML schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2 and is copied below for information:

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

7.18.4 Processing model

If this component is present and the SCS performs the requested operation in asynchronous operation mode, the SCS shall invoke the location specified by this component after the completion of the requested operation. The SCS shall provide as input parameter the component specified in clause 7.26 that the client application can include in the subsequent request, polling the pending signature results.

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

7.19.1 Component semantics

This component shall be used to pass to the SCS the list of document(s) or the list of hash(es) for which the signature(s) generation is requested.

Document(s) and hash(es) shall be provided either in:

- a container containing the base-64 encoded content(s) of the document(s) to be signed; or in
- a container containing the base-64 encoded hash(es) of the document(s) to be signed and the digest algorithm used to calculate such hash(es).

Therefore an instance of this component shall not contain both a list of documents and a list of hashes.

Table 20 contains a list of hashing algorithms recommended in ETSI TS 119 312 [i.4].

Table 21

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`

7.19.2 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:` JSON objects containing hash(es) to be signed and the digest algorithm OID used to calculate the hash(es) specified according to table 22.

Table 22

Parameter	Presence	Type	Description
documents	Required Conditional	Array of String	Base64-encoded document(s) content(s), to be signed. This parameter shall not be used if the parameter `documentDigests` is passed.
documentDigests	Required Conditional	JSONObject	JSON Object containing hash(es) to be signed and digest algorithm OID used to calculate di hash(es). This parameter shall not be used if the parameter `documents` is passed.

The `documentDigests` element shall be a JSONObject containing the following parameters:

- `hashes` array of string type parameter.
- `hashAlgorithmOID` string type parameter.

Specified according to table 23.

Table 23

Parameter	Presence	Type	Description
hashes	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).

7.19.3 XML related component

The element for submitting documents and hashes shall be `dss2:InputDocuments`, with values as specified in OASIS DSS-v2.0 [2].

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 supported, i.e. the URI `http://uri.etsi.org/19432/v1.1.1/creationprofile#` specifies the support of the profile specified in the present document;
6) a list of the versions of the protocol supported;
7) a list of supported languages, the first language in the list being the SCS default language;

8) a list of signature policies implemented by the SCS;

9) a list of service policies implemented by the SCS;

10) a list of accepted operation modes;

11) a list of supported authentication modes;

12) a list of signature formats, conformance levels and signed envelope properties;

13) a list of names of all the API methods implemented and supported by the SCS.

Some of the above items may be empty or absent indicating a feature not being 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) `signatureFormats` an array of signature format, conformance level and signed envelope property values defining which values may be passed in the component defined in clause 7.16;

13) `methods` an array of strings result.

Specified according to table 24.

Table 24

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 [14] alpha-2 country code where the service is operating.
protocol	*Required*	*String*	The URI identifyingthe 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 IETF RFC 5646 [7].
signaturePolicies	Required	Array of String	List of the supported signature policies.
servicePolicies	Required	Array of String	List of the supported service policies.
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.
signatureFormats	Required	Array of JSONObject	List of JSONObject specifying the signature formats supported by the SCS and containing the following parameters: • signature_format string type parameter; • conformance_level string type parameter; • signed_envelope_property string type parameter; as specified in clause 7.16.2.
methods	*Required*	*Array of String*	List of the supported API methods names.

The "signatureFormats", "signaturePolicies", "servicePolicies" and "operationModes" elements are not defined in CSC standard [1]. The other elements are defined in clause 11.1 of the CSC standard [1].

7.20.3 XML related component

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

The `ServiceInformation` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

```
<xs:element name="ServiceInformation" type="etsisig: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="SupportedSignatureFormat" type="etsisig:SupportedSignatureFormatType"
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>
<xs:complexType name="SupportedSignatureFormatType">
    <xs:sequence>
        <xs:element name="SupportedSignatureType" type="xs:anyURI" minOccurs="0"/>
        <xs:element name="SupportedConformanceLevel" type="xs:anyURI" minOccurs="0"/>
        <xs:element name="SupportedEnvelope" type="xs:string" minOccurs="0"/>
    </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.

The protocol supports either a 1-to-1 relation between input documents and signatures or a many-to-1 relation where all input documents are covered by one signature.

7.21.2 JSON related component

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

- `DocumentWithSignature: JSON` array of string type result;
- `SignatureObject: JSON` array of string type result;

Specified according to table 25.

Table 25

Parameter	Presence	Type	Description
DocumentWithSignature	Required Conditional	Array of String	Base64-encoded signatures enveloped within the documents. This element shall carry a value only if the client application requested the creation of signature(s) enveloped within the signed document(s).
SignatureObject	Required Conditional	Array of String	Base64-encoded signatures detached from the documents. This element shall carry a value only if the client application requested the creation of not enveloped signature(s).

7.21.3 XML related component

The element for returning signatures shall be `dss2:SignatureObject`, with values as specified in OASIS DSS-v2.0 [2], clause 4.4.6. In the case of signatures enveloped in documents, the `dss2:DocumentWithSignature` element, defined in OASIS DSS-v2.0 [2] clause 4.3.19, shall be used.

7.22 Component for returning signing credential information

7.22.1 Component semantics

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

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

This component shall contain the following data:

1) the signing X.509 certificate/chain;

2) information about the signing key:

   - status of the key;

- algorithms supported;
- length of the key;
- curve;

3) the signer certificate attribute details:
   - status;
   - validFrom;
   - validTo;
   - issuerDN;
   - serialNumber;
   - subjectDN;

4) the support of multiple signatures creation with a single authorization request specification by the credential

The inclusion of this component in the response may be requested by the client application using the "Component for defining optional data to be returned" defined in clause 7.5. The information about signing X.509 certificate/chain and/or signing key to be included in this component may be specified by the client application using the "Component for specifying the contents from certificate info to be returned" defined in clause 7.10.

7.22.2 JSON related component

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

- `cert/certificates`: array of string type result;
- `key/status`: string type result;
- `key/algo:` array of string type result;
- `key/len:` integer type result;
- `key/curve`: string type result;
- `cert/status`: string type result;
- `cert/validFrom`: string type result;
- `cert/validTo`: string type result;
- `cert/IssuerDN:` string type result;
- `cert/serialNumber:` string type result;
- `cert/subjectDN`: string type result;
- `multisign`: Boolean type result.

Specified according to table 26.

Table 26

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 clause 7.10.1.2 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, 1.2.840.113549.1.1.10 = RSASSA-PSS, 1.2.840.10045.4.3.4 = ECDSA with SHA512.*
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 otherwise this parameter shall not be included in the response. The "valid" status shall be returned when the certificate is in the validity period, is generated by a CA trusted by the SCS and is not revoked or suspended. Therefore the SCS shall not include this parameter in the response if it cannot check if the signing certificate is revoked, suspended or not.
cert/validFrom	*Required*	*String*	*The validity start date from the X.509v3 signing certificate in printable string format, encoded as GeneralizedTime format (IETF RFC 5280 [4]) (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 (IETF RFC 5280 [4]) (e.g. "YYYYMMDDHHMMSSZ"). This parameter shall be returned when certInfo defined in clause 7.10 is "true".*
cert/IssuerDN	*Required Conditional*	*String*	*The Issuer Distinguished Name from the X.509v3 end entity certificate in printable string format, UTF-8-encoded according to IETF RFC 4514 [8]. 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 Subject Distinguished Name from the X.509v3 certificate in printable string format, UTF-8-encoded according to IETF RFC 4514 [8]. 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.

The above elements are defined in clause 11.5 of the CSC standard [1].

7.22.3 XML related component

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

The `X509Details` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, and is copied below for information:

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

<xs:complexType name="X509DetailsType">
    <xs:sequence>
        <xs:element ref="etsisig:Status"/>
        <xs:element ref="etsisig:NotBefore"/>
        <xs:element ref="etsisig:NotAfter"/>
    </xs:sequence>
</xs:complexType>

<xs:element name="NotBefore" type="xs:dateTime"/>
<xs:element name="NotAfter" type="xs:dateTime"/>

<xs:element name="Status" type="etsisig:CertificateStatusType"/>
<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>

<xs:element name="Enabled" type="xs:Boolean"/>
<xs:element name="Algorithm" type="xs:anyURI"/>
<xs:element name="Length" type="xs:int"/>
<xs:element name="Curve" type="xs:anyURI"/>

<xs:element name="KeyDetails" type="etsisig:KeyDetailsType"/>
<xs:complexType name="KeyDetailsType">
    <xs:sequence>
        <xs:element ref="etsisig:Enabled"/>
        <xs:element ref="etsisig:Algorithm"/>
        <xs:element ref="etsisig:Length"/>
        <xs:element ref="etsisig:Curve"/>
    </xs:sequence>
</xs:complexType>
<xs:element name="MultipleSignaturesEnabled" type="xs:Boolean"/>
```

EXAMPLE:

```
<ds:KeyInfo>
    <ds:X509Data>
        <etsisig:X509Details>
            <etsisig:Status>Valid</etsisig:Status>
            <etsisig:NotBefore>2002-05-30T09:00:00</etsisig:NotBefore>
            <etsisig:NotAfter>2022-05-30T09:00:00</etsisig:NotAfter>
        </etsisig:X509Details>
        <etsisig:KeyDetails>
            <etsisig:Enabled>true</etsisig:Enabled>
            <etsisig:Algorithm>1.2.840.10045.4.3.2</etsisig:Algorithm>
            <etsisig:Length>4096</etsisig:Length>
            <etsisig:Curve>1.2.840.10045.3.1.1.7</etsisig:Curve>
        </etsisig:KeyDetails>
        <etsisig:MultipleSignaturesEnabled>false</etsisig:MultipleSignaturesEnabled>
    </ds:X509Data>
</ds:KeyInfo>
```

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 table 27.

Table 27

Parameter	Presence	Type	Description
credentialIDs	Required	Array of String	One or more credentialIDs associated with the provided or implicit signer identification.
certificates	Conditional	Array of String	Each String shall contains one or more certificates from the signing certificate chain, each certificate textually encoded as specified in clause 5 of IETF RFC 7468 [10]. If the `certificates` parameter defined in clause 7.10.1.2 is "chain", each string shall contain the entire certificate chain with the end entity certificate at the beginning of the string and more additional CAs certificates following. If the `certificates` parameter is "single", each string shall contain only the end entity certificate. If the `certificates` 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 `ds:KeyInfo`, child element of the `etsisig:OptionalOutputs` root element.

If the ReturnCertificates parameter defined in clause 7.10.3 clause is "Chain", the entire certificate chain shall be returned with the end entity certificate as the first ds:Keyinfo element. If the ReturnCertificates parameter is "Single", only the end entity certificate shall be returned. If the ReturnCertificates parameter is "None", no ds:Keyinfo element shall be returned.

7.24 Component for notifying operation result(s)

7.24.1 Component semantics

This component shall contain information representing the outcome(s) of the request.

NOTE: When the requesting mode is synchronous this component will return the outcome(s) of the processing. When the requesting mode is asynchronous this component will return the confirmation or not of the acceptance of the request.

7.24.2 JSON related component

When the HTTP status of the request is different from 200 OK, the outcome(s) 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 value;

- `error_description` string type value.

Specified according to table 28.

Table 28

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.

The status codes and error messages defined in clause 10.1 of CSC standard [1] shall be used if applicable.

7.24.3 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 OASIS DSS-v2.0 [2], clause 4.1.8.

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 may be requested 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 table 29.

Table 29

Parameter	Type	Description
policy	String	The element that identifies a particular service 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, defined in OASIS DSS-v2.0 [2], clause 4.1.10.

7.26 Component for identification of the response

7.26.1 Component semantics

This component contains a string value generated by the SCS uniquely identifying the response originated from the SCS itself. This component is mainly used in asynchronous operation mode where the client application shall provide the responseID value received with the initial response included in the requestID component of any subsequent request polling the pending signature results.

7.26.2 JSON related component

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

- `responseID` string type parameter.

Specified according to table 30.

Table 30

Parameter	Type	Description
responseID	String	Arbitrary string value generated by the SCS uniquely identifying the response originated from the SCS itself.

7.26.3 XML related component

The element for identifying the signature request whose results are requested shall be the `responseID` attribute of the element `dss2:SignResponse` root element, defined in OASIS DSS-v2.0 [2], clause 4.2.2.

7.27 Component for signature creation policy identification

7.27.1 Component semantics

This component shall contain information 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 may be requested by the client using the "Component for defining optional data to be returned" defined in clause 7.5.

NOTE: The signature creation policy may be a part of the service policy. The signature creation policy may therefore be implicitly identified by the applied service policy.

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 table 31.

Table 31

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 OID mentioned above as specified in IETF RFC 3061 [9].
signaturePolicyLocations	Optional	Array of String	Every string element shall have as an URI value the value of one location where the signature creation policy document can be accessed.

7.27.3 XML related component

The element for the signature creation policy identification shall be `dsb:AppliedPolicy`, child element of the `dss2:OptionalOutputs` element, defined in OASIS DSS-v2.0 [2], clause 4.1.10.

7.28 Component for returning credential authorization mode

7.28.1 Component semantics

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

EXAMPLE: The SCSP can specify the supported authorization modes in the service policy or in the terms and conditions of the service.

Returned values may be:

1) "implicit": the authorization process is managed by the SCS autonomously. In such mode the SCS will deal directly with the user in order to perform the authorization process without any involvement of the client application;

2) "explicit": the client application provides the needed factors of security elements to the SCS for the SCS to perform the authorization process;

3) "authorizationCode": the authorization process is managed by the SCS using a mechanism based on authorization code, for example an OAuth 2.0 mechanism based on authorization code as described in IETF RFC 6749 [i.5];

4) "identificationToken": the authorization process is managed by the SCS using a mechanism based on security tokens containing user profile information (like the user's name, email, and so forth), represented in the form of claims (for example a JWT as described in IETF RFC 7519 [i.6] or a SAML assertion); or

5) other different values according to the authorization modes supported and specified by the SCSP.

7.28.2 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 table 32.

Table 32

Parameter	Type	Description
authMode	*String*	Specifies the authorization mode of the signature credential.

The "authMode" element is defined in clause 11.5 of the CSC standard [1]. The value "identificationToken", mentioned in clause 7.28.1, is not defined in CSC standard [1].

7.28.3 XML related component

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

The `etsisig:AuthorizationMode` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, 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>
```

7.29 Component for returning digital signature value(s)

7.29.1 Component semantics

This component shall contain a list of base64 encoded signature values corresponding to the DTBSR(s) passed in the `documentDigests` parameter specified in clause 7.19.

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

The values returned in this component may 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(s).

2) When one or more of the requested signatures fail, the corresponding DSV(s) are returned as empty values.

7.29.2 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 table 33.

Table 33

Parameter	Type	Description
signatures	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 as the corresponding hashes provided as an input parameter.

7.29.3 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. The optional attribute `WhichDoc` shall contain the identifier of the related document from the signature creation request.

7.30 Component for returning sole control assurance level required

7.30.1 Component semantics

This component specifies the sole control assurance level required by the identified signature credential, as defined in CEN EN 419 241-1 [6]. Only two values of sole control assurance level shall be supported: "SCAL1" and "SCAL2". Such values identify two different levels of confidence of the control of the signing key, as specified in clause 4.4.1.1.

NOTE: With regards to this document, the "SCAL2" value indicates that the signature activation data, used in order to ensure the authorized signer's use of its signing key, is linked to the document or the documents to be signed and that a two-factor authorization is needed to authorize a signature.

7.30.2 JSON related component

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

- SCAL string type result.

Specified according to table 34.

Table 34

Parameter	Type	Description
SCAL	*String*	Specifies the Sole Control Assurance Level required by the credential: • "1": the sole control assurance level 1 is required by the identified signature credential. • "2": the sole control assurance level 2 is required by the identified signature credential.

The "SCAL" element is defined in clause 11.5 of the CSC standard [1].

7.30.3 XML related component

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

The `SoleControlAssuranceLevel` element is defined in XML Schema file "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2, 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>
```

8 Remote signature creation messages

8.1 Introduction

In the present document the messages represent the way in which any client application can interact with any SCS conforming to this specification. The following clauses define the messages that a client application and an SCS can exchange. For any request and response pair 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 tables 36 to 47 has the meaning defined in table 35.

Table 35

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 AdES signatures creation messages

8.2.1 Request message (A)

8.2.1.1 Component for requesting AdES signatures creation

The message for requesting the creation of AdES signatures to an SCASC shall contain components for:

1) Submitting either the documents to be signed or the SDR (signer's document representation, for example the hash value of the document).

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) Providing credential authorization in the form of signature activation data.

3) Identifying one or more protocols and/or profiles with which the request message is compliant. The first one of such components shall have the following URI as value, identifying the request message as one that has been built using the "AdES signature creation" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/signaturecreationprotocol#`

NOTE 2: The protocol 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 the present document.

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 some of these optional components and contain references to clauses that specify semantic requirements for each component.

This message includes the following components.

Table 36

Ref.	Component for	Presence
7.2	asynchronous/synchronous operation mode selection	**O**
7.3	identification of the request	**O**
7.4	credential authorization	**M**
7.5	defining optional data to be returned	**O**
7.6	defining the validity period for asynchronous requests	**O**
7.7	service authentication	**O**
7.8	identifying signature credentials	**C**
7.9	language selection	**O**
7.10	contents from certificate info to be returned	**O**
7.11	managing digital signatures transactions	**O**
7.12	service policy selection	**O**
7.13	signature creation policy selection	**O**
7.14	optional signature attributes/properties selection	**O**
7.15	protocol identifier	**M**
7.16	requesting specific signature formats	**O**
7.18	specifying response URL	**O**
7.19	submitting document(s) or hash(es) to be signed	**M**

The component for identifying signature credential may not be included in the message if there is only one signing credential for the user which has been previously authenticated.

8.2.1.2 JSON related component

The method `signatures/signDoc` shall be invoked in order to request the signature of one or more documents or document representations. This method calculate the signature of the documents or document representations provided in input. The elements to request the signature of a document or a document representation shall be the set of required and optional parameters, specified in clause 8.2.1.1, 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 8.2.1.1 of the present document.

8.2.1.3 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 OASIS DSS-v2.0 [2].

Processing model

The server shall process the components in the `dss:SignRequest` as indicated in the corresponding clauses of OASIS DSS-v2.0 [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.

8.2.2 Response message (B)

8.2.2.1 Component for responding to AdES signatures creation requests

The AdES signature creation response message resulting from one request of AdES signatures creation, shall include the component for notifying the global result of the signature operation requested by the client application.

The AdES signature creation response message may include one or more signed document or signature elements as defined in clause 7.21.

This message includes the following components.

Table 37

Ref	Component for	Presence
7.21	returning signed documents or signatures	**O**
7.22	returning signing certificate information	**O**
7.24	notifying operation result(s)	**M**
7.25	service policy identification	**O**
7.26	response identification	**C**
7.27	signature creation policy identification	**O**

The response identification may be included in the response message. The response identification shall be included in the response message if one of the following conditions occurs:

1) The AdES signatures creation request includes the component indicating an asynchronous operation mode to the SCASC, as specified in clause 7.2.

2) The SCASC chooses to process the request in asynchronous mode.

8.2.2.2 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 8.2.2.1.

8.2.2.3 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 OASIS DSS-v2.0 [2].

8.3 DSVs creation messages

8.3.1 Request message (C)

8.3.1.1 Component for requesting DSVs creation

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

The message for requesting the creation of DSVs to an SSASC shall contain components for:

1) Submitting the DTBSRs.

NOTE 1: The message may contain one or more data to be signed representations for DSVs creation. A DSV will be created for each of these input components.

2) Providing credential authorization in the form of signature activation data.

3) Identifying one or more protocols and/or profiles with which the request message is compliant. The first one of such components shall have the following URI as value, identifying the request message as one that has been built using the "digital signature value creation" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/dsvcreationprotocol#`

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

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

The message for requesting the creation of DSVs to the SSASC may contain other components for requesting additional features. Clause 7.5 lists some of these optional components and contain references to clauses that specify semantic requirements for each component.

This message includes the following components.

Table 38

Ref.	Component for	Presence
7.2	asynchronous/synchronous operation mode selection	**O**
7.3	identification of the request	**O**
7.4	credential authorization	**M**
7.5	defining optional data to be returned	**O**
7.6	defining the validity period for asynchronous requests	**O**
7.7	service authentication	**O**
7.8	identifying signature credentials	**C**
7.9	language selection	**O**
7.10	contents from certificate info to be returned	**O**
7.11	managing digital signatures transactions	**O**
7.12	service policy selection	**O**
7.13	signature creation policy selection	**O**
7.15	protocol identifier	**M**
7.18	specifying response URL	**O**
7.19	submitting hash(es) to be signed	**M**

The component for identifying signature credential may not be included in the message if there is only one signing credential for the user which has been previously authenticated.

8.3.1.2 JSON related component

The element to request the signature on a DTBSR shall be the set of required parameters to invoke the signature by means of the `signatures/signHash` method as specified in CSC standard [1]. The signature creation policy element may be used in order to specify the signature algorithm and eventual signature algorithm parameters, alternatively to the parameters signAlgo and signAlgoParams defined in CSC standard [1].

Processing model

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

The server shall process each component as indicated in the clause 8.3.1.1 of the present document if the child is not specified in the `signatures/signHash` description of the CSC standard [1].

8.3.1.3 XML related component

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

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 DSS-v2.0 [2].

8.3.2 Response message (D)

8.3.2.1 Component for responding to DSVs creation requests

The DSVs signature creation response message resulting from one request of DSVs creation, shall include the component for notifying the global result of the signature operation requested by the client application.

The DSVs creation response message may include one or more DSV elements as defined in clause 7.29.

This message includes the following components.

Table 39

Ref	Component for	Presence
7.22	returning signing certificate information	**O**
7.24	notifying operation result(s)	**M**
7.25	service policy identification	**O**
7.26	response identification	**C**
7.27	signature creation policy identification	**O**
7.29	returning DSV	**O**

The response identification may be included in the response message. The response identification shall be included in the response message if one of the following conditions occurs:

1) The DSVs creation request includes the component indicating an asynchronous operation mode to the SCASC, as specified in clause 7.2.

2) The SSASC chooses to process the request in asynchronous mode.

8.3.2.2 JSON related component

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

8.3.2.3 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 OASIS DSS-v2.0 [2] specifications.

8.4 Messages for asynchronous processing (E)

8.4.1 Component for managing pending-requests

This component shall manage the messages for requesting to the SCS to return the responses corresponding to previously sent (initial) signatures or DSVs creation requests, defined in clauses 8.2.1 and 8.3.1, when processed in asynchronous mode by the SCS itself. Requests of this type are named pending-requests hereinafter.

In asynchronous processing one client application sends an initial signatures or DSVs creation request to the server. The initial request may contain, among other things, a component by which an asynchronous operation mode is requested to the SCS, as specified in clause 7.2 and a request identifier generated by the client application, as specified in clause 7.3.

The server can return a response indicating that the signature creation request has been accepted but it has not yet been completed, either because the client application has requested an asynchronous operation mode to the SCS or because the SCS has chosen to perform the requested operation in asynchronous mode. Within this initial response, the SCS shall convey a response identifier, as specified in clause 7.26. If the initial request contains the request identifier, both the client application and the SCS can correlate this response identifier to the initial request identifier.

Under this processing model the client application, after the initial request, can send a pending-request to the SCS. This pending-request shall include in the component for request identification, as defined in clause 7.3, the value of the response identifier previously returned by the SCS in its response to the initial signature creation request. This response identifier value allows the SCS to correlate this pending-request to the initial signature creation request. The SCS shall return the signature creation result or again an indication of "not yet finished".

If the latter is the case, the client application can send subsequent requests until the SCS returns a response with the signature creation result..

The pending-requests message shall contain components for:

1) Identifying the request as a pending-request associated to an initial signatures or DSVs creation request. The component specified in clause 7.3 shall be used to include the response identifier identifying the pending-request.

2) Identifying one or more protocols and/or profiles with which the pending-request message is compliant. The first one of such components shall have the following URI as value, identifying the pending-request message as one that has been built using the "asynchronous processing" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/asynchronousprotocol#`

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

The response to a pending-request has the format of the response to the initial signatures or DSVs creation request. A pending-request message includes the following components.

Table 40

Ref	Component for	Presence
7.3	identification of the request/polling signature results	**M**
7.7	service authentication	**O**
7.9	language selection	**O**
7.15	protocol identifier	**M**

8.4.2 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 8.4.1 to invoke the `signatures/signPolling` method.

Processing model

The server shall process each component as indicated in the clause 8.4.1.

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).

8.4.3 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 OASIS DSS-v2.0 [2].

8.5 Signing certificates list messages

8.5.1 Request message (F)

8.5.1.1 Component for requesting signing certificates list

The message for requesting the list of the signing certificates of a signer shall contain the component for:

1) Identifying one or more protocols and/or profiles with which the request message is compliant. The first one of such components shall have the following URI as value, identifying the request message as one that has been built using the "signing certificates list" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/certificateslistprotocol#`

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

A signer may have one or multiple credentials associated within a single signer identifier.

This message includes the following components.

Table 41

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

The component for the signer identification may not be included in the message if it is already implicit in the service authentication performed.

8.5.1.2 JSON related component

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

Processing model

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

The server shall process each component as indicated in the clause 8.5.1.1 of the present document if the child is not specified in any of the referenced CSC standard [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC standard [1].

8.5.1.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The `etsisig:SignerIdentity`, child element of the `etsisig: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.

8.5.2 Response message (G)

8.5.2.1 Component for responding to certificates list requests

The certificates list response message resulting from one request of certificates list, shall include the component for notifying the global result of the operation requested by the client application.

The certificates list creation response message may include one or more certificates elements as defined in clause 7.23.

This message includes the following components.

Table 42

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

The SCS shall return an error in the case of a request to obtain the list of certificates associated to a signer other than the one authenticated to the service.

8.5.2.2 JSON related component

The response to retrieve user's credentials list request shall consist of the set of results specified in clause 8.5.2.1 as defined in clauses 7.23.2 and 7.24.2.

8.5.2.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The `ds:KeyInfo`, child element of the `etsisig:OptionalOutputs` element shall return the requested signing certificates list.

8.6 Credential information retrieval messages

8.6.1 Request message (H)

8.6.1.1 Component for requesting credential information

The message for requesting information about a signing credential to the SCS shall contain components for:

1) Identifying the signing credential.

2) Identifying one or more protocols and/or profiles with which the request message is compliant. The first one of such components shall have the following URI as value, identifying the request message as one that has been built using the "credential information" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/credentialinfoprotocol#`

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

The message for requesting information about a signing credential to the SCS may contain other components for requesting additional features. Clause 7.5 lists some of these optional components and contain references to clauses that specify semantic requirements for each component.

This message includes the following components.

Table 43

Ref	Component for	Presence
7.3	identification of the request	**O**
7.5	defining optional data to be returned	**O**
7.7	service authentication	**O**
7.8	identifying signature credentials	**C**
7.9	language selection	**O**
7.10	contents from certificate info to be returned	**O**
7.15	protocol identifier	**M**

The component for identifying signature credential may not be included in the message if there is only one signing credential for the user which has been previously authenticated.

8.6.1.2 JSON related component

The element to request credentials information shall be the set of parameters specified in clause 8.6.1.1.

Processing model

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

The server shall process each component as indicated in the clause 8.6.1.1 of the present document if the child is not specified in any of the referenced CSC standard [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC standard [1].

8.6.1.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

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

8.6.2 Response message (I)

8.6.2.1 Component for responding to credential information requests

The credential information response message resulting from one request of credential information, shall include the component for notifying the global result of the operation requested by the client application.

The credential information response message may include one or more credential information elements as defined in clause 7.22.

This message includes the following components.

Table 44

Ref	Component for	Presence
7.22	returning signing credential information	**O**
7.24	notifying operation result(s)	**M**
7.28	returning credential authorization mode	**O**
7.30	returning SCAL level required	**O**

8.6.2.2 JSON related component

The response to the credential information retrieval request shall consist of the set of results specified in clause 8.6.2.1 and returned as in the response of the `credentials/info` method as specified in CSC standard [1].

8.6.2.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The `ds:KeyInfo`, child element of the `etsisig: InformationResponse` element will return the requested signing certificate and key information.

8.7 Service information messages (J)

8.7.1 Request message (J)

8.7.1.1 Component for requesting service information

The message for requesting the service information shall contain the component for:

1) Identifying one or more protocols and/or profiles with which the request message is compliant. The first one of such components shall have the following URI as value, identifying the request message as one that has been built using the "service information" protocol specified in the present document:

   - `http://uri.etsi.org/19432/v1.1.1/serviceinformationprotocol#`

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

This message includes the following components.

Table 45

Ref	Component for	Presence
7.9	language selection	**O**
7.15	protocol identifier	**M**

8.7.1.2 JSON related component

The element to request service information shall be the set of parameters specified in clause 8.7.1.1.

Processing model

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

The server shall process each component as indicated in the clause 8.7.1.1 of the present document if the child is not specified in any of the referenced CSC standard [1]. Otherwise, the server shall follow the processing model defined in the corresponding CSC standard [1].

8.7.1.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

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

8.7.2 Response message (K)

8.7.2.1 Component for responding to service information requests

This profile includes the following components.

Table 46

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

8.7.2.2 JSON related component

The response to service information request shall consist of the set of parameters specified in clause 8.7.2.1 and returned as in the response of the `info` method as specified in CSC standard [1].

8.7.2.3 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 "[XSDSIGCREATIONPROT]", whose location is detailed in clause A.2.

The `etsisig:ServiceInformation`, child element of the `etsisig:OptionalOutputs` element will return the requested service information.

8.8 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

Table 47

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	identification of the request	**O**		**O**		**M**	**O**		**O**			
7.4	credential authorization	**O**		**O**								
7.5	defining optional data to be returned	**O**		**O**					**O**			
7.6	defining the validity period for asynchronous requests	**O**		**O**								
7.7	service authentication	**O**		**O**		**O**	**O**		**O**			
7.8	identifying signature credentials	**C**		**C**					**C**			
7.9	language selection	**O**		**O**		**O**	**O**		**O**		**O**	
7.10	contents from certificate info to be returned	**O**		**O**					**O**			
7.11	managing digital signatures transactions	**O**		**O**								
7.12	service policy selection	**O**		**O**								
7.13	signature creation policy selection	**O**		**O**								
7.14	optional signature attributes/properties selection	**O**										
7.15	protocol identifier	**M**		**M**		**M**	**M**		**M**		**M**	
7.16	requesting specific signature formats	**O**										
7.17	signer identification						**C**					
7.18	specifying response URL	**O**		**O**								
7.19	submitting document(s) or hash(es) to be signed	**M**		**M**								
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	notifying operation result(s)		**M**		**M**			**M**		**M**		
7.25	service policy identification		**O**		**O**							
7.26	identification of the response		**C**		**C**							
7.27	signature creation policy identification		**O**		**O**							
7.28	returning credential authorization mode									**O**		
7.29	returning DSV				**O**							
7.30	returning SCAL level required									**O**		

Annex A (normative): XML and JSON Schema files

A.1 JSON Schema file location for "$schema" "http://uri.etsi.org/19432/v1.1.1/json#"

The file 19432-schema.json in attachment and available at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/19432-schema.json contains the definitions of elements and types defined within the JSON schema whose "$schema" value is "http://uri.etsi.org/19432/v1.1.1/json#".

A.2 XML Schema file location for namespace http://uri.etsi.org/19432/v1.1.1#

The files etsi-org-19432-xmlSchema-common.xsd at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/etsi-org-19432-xmlSchema-common.xsd, etsi-org-19432-xmlSchema-dsv-creation.xsd at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/etsi-org-19432-xmlSchema-dsv-creation.xsd, etsi-org-19432-xmlSchema-service-information.xsd at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/etsi-org-19432-xmlSchema-service-information.xsd and etsi-org-19432-xmlSchema-signature-creation.xsd at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/etsi-org-19432-xmlSchema-signature-creation.xsd contain the definitions of elements and types defined within the namespace whose URI value is http://uri.etsi.org/19432/v1.1.1#. The four files are also available in attachment.

Annex B (informative): OpenAPI description file

An OpenAPI 3.0 description file (19432-openapi.yaml), as defined by the OpenAPI Initiative (OAI) (https://www.openapis.org), is provided in attachment and is available at https://forge.etsi.org/rep/esi/x19_432_sign_creation_protocol/raw/v1.1.1/19432-openapi.yaml. It contains the JSON Schema definitions for every component and the description of every message defined within the protocol specified in the present document.

Annex C (informative): Bibliography

- ETSI TS 119 441: "Electronic Signatures and Infrastructures (ESI); Policy requirements for TSP providing signature validation services".
- ETSI TS 119 442: "Electronic signatures and infrastructures (ESI); Protocol profiles for trust service providers providing AdES digital signature validation services".
- ETSI EN 319 122-2: "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures; Part 2: Extended CAdES signatures".
- ETSI EN 319 132-2: "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures; Part 2: Extended XAdES signatures".
- ETSI EN 319 142-2: "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures; Part 2: Additional PAdES signatures profiles".

History

Document history		
V1.1.1	March 2019	Publication