Profile for comprehensive multi-signature verification reports for OASIS Digital Signature Services Version 2.0

Working Draft 01

DD Month YYYY

Technical Committee:

[OASIS Digital Signature Services eXtended (DSS-X) TC](https://www.oasis-open.org/committees/dss-x/)

Chairs:

Stefan Hagen ([stefan@hagen.link](mailto:stefan@hagen.link)), Individual

Editor:

Detlef Hühnlein, *Federal Office for Information Security, Germany* <[detlef.huehnlein@ecsec.de](mailto:detlef.huehnlein@ecsec.de)>

Andreas Kuehne ([kuehne@trustable.de](mailto:kuehne@trustable.de)), Individual

Stefan Hagen ([stefan@hagen.link](mailto:stefan@hagen.link)), Individual

Additional artifacts:

This prose specification is one component of a Work Product that also includes:

* JSON and XML schemas: <http://docs.oasis-open.org/dss-x/dss-core/v2.0/csd01/schemas/>

Related work:

This specification replaces or supersedes:

* Stefan Drees et al., Digital Signature Service Core Protocols, Elements, and Bindings, Version 1.0, OASIS Standard, 11 April 2007,  
  <http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-spec-v1.0-os.pdf>

This specification is related to:

* Related specifications (hyperlink, if available)

Declared XML namespaces:

* urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:[schema](http://www.oasis-open.org/committees/download.php/33059/VerificationReport-CD1.xsd)#

Abstract:

This document defines a protocol and processing profile of the DSS Verifying Protocol specified in **[DSSCore]**, which allows to return individual signature verification reports for each signature in a verification request and include detailed information of the different steps taken during verification.

Status:

This [Working Draft](https://www.oasis-open.org/policies-guidelines/tc-process) (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or [approved](https://www.oasis-open.org/policies-guidelines/tc-process) as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document [Approval Process](https://www.oasis-open.org/policies-guidelines/tc-process) begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

Any machine-readable content ([Computer Language Definitions](https://www.oasis-open.org/policies-guidelines/tc-process)) declared Normative for this Work Product must also be provided in separate plain text files. In the event of a discrepancy between such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

URI patterns:

Initial publication URI:  
<http://docs.oasis-open.org/dss-x/>???

Permanent “Latest version” URI:  
http://docs.oasis-open.org/dss-x/???

(Managed by OASIS TC Administration; please don’t modify.)

Copyright © OASIS Open 2017. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](https://www.oasis-open.org/policies-guidelines/ipr) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

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

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

Table of Contents

[1 Introduction 16](#_Toc509841811)

[1.1 Organization of DSS Core Protocols, Elements, and Bindings 16](#_Toc509841812)

[1.2 Terminology 16](#_Toc509841813)

[1.2.1 Terms and Definitions 16](#_Toc509841814)

[1.2.2 Abbreviated Terms 16](#_Toc509841815)

[1.3 Normative References 16](#_Toc509841816)

[1.4 Non-Normative References 17](#_Toc509841817)

[1.5 Typographical Conventions 17](#_Toc509841818)

[2 Design Considerations 19](#_Toc509841819)

[2.1 Construction Principles 19](#_Toc509841820)

[2.2 Domain Models 19](#_Toc509841821)

[2.2.1 Date and Time Model 19](#_Toc509841822)

[2.3 Schema Organization and Namespaces 19](#_Toc509841823)

[2.4 DSS Overview (Non-normative) 20](#_Toc509841824)

[2.5 Syntax variants 20](#_Toc509841825)

[3 Structure Models 21](#_Toc509841826)

[3.1 Structure Models defined in this document 21](#_Toc509841827)

[3.1.1 Component ReturnVerificationReport 21](#_Toc509841828)

[3.1.1.1 Semantics 21](#_Toc509841829)

[3.1.1.2 XML Syntax 21](#_Toc509841830)

[3.1.1.3 JSON Syntax 22](#_Toc509841831)

[3.1.2 Component VerificationReport 23](#_Toc509841832)

[3.1.2.1 Semantics 23](#_Toc509841833)

[3.1.2.2 XML Syntax 23](#_Toc509841834)

[3.1.2.3 JSON Syntax 23](#_Toc509841835)

[3.1.3 Component Identifier 24](#_Toc509841836)

[3.1.3.1 Semantics 24](#_Toc509841837)

[3.1.3.2 XML Syntax 25](#_Toc509841838)

[3.1.3.3 JSON Syntax 25](#_Toc509841839)

[3.1.4 Component ValidationProcess 26](#_Toc509841840)

[3.1.4.1 Semantics 26](#_Toc509841841)

[3.1.4.2 XML Syntax 26](#_Toc509841842)

[3.1.4.3 JSON Syntax 27](#_Toc509841843)

[3.1.5 Component ReportDetail 27](#_Toc509841844)

[3.1.5.1 Semantics 27](#_Toc509841845)

[3.1.5.2 XML Syntax 29](#_Toc509841846)

[3.1.5.3 JSON Syntax 30](#_Toc509841847)

[3.1.6 Component IndividualReport 33](#_Toc509841848)

[3.1.6.1 Semantics 33](#_Toc509841849)

[3.1.6.2 XML Syntax 33](#_Toc509841850)

[3.1.6.3 JSON Syntax 33](#_Toc509841851)

[3.1.7 Component OptionalOutput 34](#_Toc509841852)

[3.1.7.1 Semantics 34](#_Toc509841853)

[3.1.7.2 XML Syntax 34](#_Toc509841854)

[3.1.7.3 JSON Syntax 35](#_Toc509841855)

[3.1.8 Component OptionalInput 36](#_Toc509841856)

[3.1.8.1 Semantics 36](#_Toc509841857)

[3.1.8.2 XML Syntax 36](#_Toc509841858)

[3.1.8.3 JSON Syntax 37](#_Toc509841859)

[3.1.9 Component SignedObjectIdentifier 38](#_Toc509841860)

[3.1.9.1 Semantics 38](#_Toc509841861)

[3.1.9.2 XML Syntax 39](#_Toc509841862)

[3.1.9.3 JSON Syntax 40](#_Toc509841863)

[3.1.10 Component VerificationResult 41](#_Toc509841864)

[3.1.10.1 Semantics 41](#_Toc509841865)

[3.1.10.2 XML Syntax 41](#_Toc509841866)

[3.1.10.3 JSON Syntax 41](#_Toc509841867)

[3.1.11 Component DetailedSignatureReport 42](#_Toc509841868)

[3.1.11.1 Semantics 42](#_Toc509841869)

[3.1.11.2 XML Syntax 43](#_Toc509841870)

[3.1.11.3 JSON Syntax 43](#_Toc509841871)

[3.1.12 Component Properties 44](#_Toc509841872)

[3.1.12.1 Semantics 44](#_Toc509841873)

[3.1.12.2 XML Syntax 45](#_Toc509841874)

[3.1.12.3 JSON Syntax 45](#_Toc509841875)

[3.1.13 Component SignedProperties 46](#_Toc509841876)

[3.1.13.1 Semantics 46](#_Toc509841877)

[3.1.13.2 XML Syntax 46](#_Toc509841878)

[3.1.13.3 JSON Syntax 47](#_Toc509841879)

[3.1.14 Component SignedSignatureProperties 48](#_Toc509841880)

[3.1.14.1 Semantics 48](#_Toc509841881)

[3.1.14.2 XML Syntax 48](#_Toc509841882)

[3.1.14.3 JSON Syntax 49](#_Toc509841883)

[3.1.15 Component SignerRole 50](#_Toc509841884)

[3.1.15.1 Semantics 50](#_Toc509841885)

[3.1.15.2 XML Syntax 50](#_Toc509841886)

[3.1.15.3 JSON Syntax 50](#_Toc509841887)

[3.1.16 Component CertifiedRolesList 51](#_Toc509841888)

[3.1.16.1 Semantics 51](#_Toc509841889)

[3.1.16.2 XML Syntax 51](#_Toc509841890)

[3.1.16.3 JSON Syntax 51](#_Toc509841891)

[3.1.17 Component AttributeCertificateValidity 52](#_Toc509841892)

[3.1.17.1 Semantics 52](#_Toc509841893)

[3.1.17.2 XML Syntax 52](#_Toc509841894)

[3.1.17.3 JSON Syntax 53](#_Toc509841895)

[3.1.18 Component AttrCertID 54](#_Toc509841896)

[3.1.18.1 Semantics 54](#_Toc509841897)

[3.1.18.2 XML Syntax 54](#_Toc509841898)

[3.1.18.3 JSON Syntax 54](#_Toc509841899)

[3.1.19 Component Entity 55](#_Toc509841900)

[3.1.19.1 Semantics 55](#_Toc509841901)

[3.1.19.2 XML Syntax 55](#_Toc509841902)

[3.1.19.3 JSON Syntax 56](#_Toc509841903)

[3.1.20 Component Attribute 56](#_Toc509841904)

[3.1.20.1 Semantics 56](#_Toc509841905)

[3.1.20.2 XML Syntax 57](#_Toc509841906)

[3.1.20.3 JSON Syntax 57](#_Toc509841907)

[3.1.21 Component Extension 58](#_Toc509841908)

[3.1.21.1 Semantics 58](#_Toc509841909)

[3.1.21.2 XML Syntax 58](#_Toc509841910)

[3.1.21.3 JSON Syntax 58](#_Toc509841911)

[3.1.22 Component SignedDataObjectProperties 59](#_Toc509841912)

[3.1.22.1 Semantics 59](#_Toc509841913)

[3.1.22.2 XML Syntax 59](#_Toc509841914)

[3.1.22.3 JSON Syntax 60](#_Toc509841915)

[3.1.23 Component TimeStampValidity 61](#_Toc509841916)

[3.1.23.1 Semantics 61](#_Toc509841917)

[3.1.23.2 XML Syntax 61](#_Toc509841918)

[3.1.23.3 JSON Syntax 62](#_Toc509841919)

[3.1.24 Component SignatureValidity 63](#_Toc509841920)

[3.1.24.1 Semantics 63](#_Toc509841921)

[3.1.24.2 XML Syntax 63](#_Toc509841922)

[3.1.24.3 JSON Syntax 63](#_Toc509841923)

[3.1.25 Component AlgorithmValidity 64](#_Toc509841924)

[3.1.25.1 Semantics 64](#_Toc509841925)

[3.1.25.2 XML Syntax 64](#_Toc509841926)

[3.1.25.3 JSON Syntax 65](#_Toc509841927)

[3.1.26 Component CertificatePathValidity 65](#_Toc509841928)

[3.1.26.1 Semantics 65](#_Toc509841929)

[3.1.26.2 XML Syntax 66](#_Toc509841930)

[3.1.26.3 JSON Syntax 66](#_Toc509841931)

[3.1.27 Component CertificatePathValidityDetail 67](#_Toc509841932)

[3.1.27.1 Semantics 67](#_Toc509841933)

[3.1.27.2 XML Syntax 67](#_Toc509841934)

[3.1.27.3 JSON Syntax 68](#_Toc509841935)

[3.1.28 Component CertificateValidity 68](#_Toc509841936)

[3.1.28.1 Semantics 68](#_Toc509841937)

[3.1.28.2 XML Syntax 69](#_Toc509841938)

[3.1.28.3 JSON Syntax 70](#_Toc509841939)

[3.1.29 Component CertificateStatus 71](#_Toc509841940)

[3.1.29.1 Semantics 71](#_Toc509841941)

[3.1.29.2 XML Syntax 71](#_Toc509841942)

[3.1.29.3 JSON Syntax 72](#_Toc509841943)

[3.1.30 Component CRLValidity 74](#_Toc509841944)

[3.1.30.1 Semantics 74](#_Toc509841945)

[3.1.30.2 XML Syntax 74](#_Toc509841946)

[3.1.30.3 JSON Syntax 75](#_Toc509841947)

[3.1.31 Component OCSPValidity 75](#_Toc509841948)

[3.1.31.1 Semantics 75](#_Toc509841949)

[3.1.31.2 XML Syntax 76](#_Toc509841950)

[3.1.31.3 JSON Syntax 76](#_Toc509841951)

[3.1.32 Component UnsignedProperties 77](#_Toc509841952)

[3.1.32.1 Semantics 77](#_Toc509841953)

[3.1.32.2 XML Syntax 78](#_Toc509841954)

[3.1.32.3 JSON Syntax 78](#_Toc509841955)

[3.1.33 Component UnsignedSignatureProperties 79](#_Toc509841956)

[3.1.33.1 Semantics 79](#_Toc509841957)

[3.1.33.2 XML Syntax 81](#_Toc509841958)

[3.1.33.3 JSON Syntax 81](#_Toc509841959)

[3.1.34 Component RevocationValues 84](#_Toc509841960)

[3.1.34.1 Semantics 84](#_Toc509841961)

[3.1.34.2 XML Syntax 84](#_Toc509841962)

[3.1.34.3 JSON Syntax 85](#_Toc509841963)

[3.1.35 Component CertificateValues 86](#_Toc509841964)

[3.1.35.1 Semantics 86](#_Toc509841965)

[3.1.35.2 XML Syntax 86](#_Toc509841966)

[3.1.35.3 JSON Syntax 87](#_Toc509841967)

[3.1.36 Component EvidenceRecordValidity 87](#_Toc509841968)

[3.1.36.1 Semantics 87](#_Toc509841969)

[3.1.36.2 XML Syntax 88](#_Toc509841970)

[3.1.36.3 JSON Syntax 89](#_Toc509841971)

[3.1.37 Component ArchiveTimeStampValidity 91](#_Toc509841972)

[3.1.37.1 Semantics 91](#_Toc509841973)

[3.1.37.2 XML Syntax 92](#_Toc509841974)

[3.1.37.3 JSON Syntax 93](#_Toc509841975)

[3.1.38 Component HashValue 94](#_Toc509841976)

[3.1.38.1 Semantics 94](#_Toc509841977)

[3.1.38.2 XML Syntax 95](#_Toc509841978)

[3.1.38.3 JSON Syntax 95](#_Toc509841979)

[3.2 Referenced Structure Models from DSS-X base 96](#_Toc509841980)

[3.2.1 Component Base64Data 96](#_Toc509841981)

[3.2.1.1 Semantics 96](#_Toc509841982)

[3.2.1.2 XML Syntax 96](#_Toc509841983)

[3.2.1.3 JSON Syntax 96](#_Toc509841984)

[3.2.2 Component AttachmentReference 97](#_Toc509841985)

[3.2.2.1 Semantics 97](#_Toc509841986)

[3.2.2.2 XML Syntax 98](#_Toc509841987)

[3.2.2.3 JSON Syntax 98](#_Toc509841988)

[3.2.3 Component DigestInfo 99](#_Toc509841989)

[3.2.3.1 Semantics 99](#_Toc509841990)

[3.2.3.2 XML Syntax 99](#_Toc509841991)

[3.2.3.3 JSON Syntax 99](#_Toc509841992)

[3.2.4 Component NsPrefixMapping 100](#_Toc509841993)

[3.2.4.1 Semantics 100](#_Toc509841994)

[3.2.4.2 XML Syntax 100](#_Toc509841995)

[3.2.4.3 JSON Syntax 101](#_Toc509841996)

[3.2.5 Component Any 101](#_Toc509841997)

[3.2.5.1 Semantics 101](#_Toc509841998)

[3.2.5.2 XML Syntax 101](#_Toc509841999)

[3.2.5.3 JSON Syntax 102](#_Toc509842000)

[3.2.6 Component Result 102](#_Toc509842001)

[3.2.6.1 Semantics 102](#_Toc509842002)

[3.2.6.2 XML Syntax 103](#_Toc509842003)

[3.2.6.3 JSON Syntax 103](#_Toc509842004)

[3.2.7 Component InternationalString 104](#_Toc509842005)

[3.2.7.1 Semantics 104](#_Toc509842006)

[3.2.7.2 XML Syntax 105](#_Toc509842007)

[3.2.7.3 JSON Syntax 105](#_Toc509842008)

[3.2.8 Component ResponseBase 106](#_Toc509842009)

[3.2.8.1 Semantics 106](#_Toc509842010)

[3.2.8.2 XML Syntax 106](#_Toc509842011)

[3.2.8.3 JSON Syntax 106](#_Toc509842012)

[3.3 Referenced Structure Models from DSS-X core 107](#_Toc509842013)

[3.3.1 Component SignRequest 107](#_Toc509842014)

[3.3.1.1 Semantics 107](#_Toc509842015)

[3.3.1.2 XML Syntax 107](#_Toc509842016)

[3.3.1.3 JSON Syntax 108](#_Toc509842017)

[3.3.2 Component InputDocuments 109](#_Toc509842018)

[3.3.2.1 Semantics 109](#_Toc509842019)

[3.3.2.2 XML Syntax 109](#_Toc509842020)

[3.3.2.3 JSON Syntax 109](#_Toc509842021)

[3.3.3 Component Document 110](#_Toc509842022)

[3.3.3.1 Semantics 110](#_Toc509842023)

[3.3.3.2 XML Syntax 111](#_Toc509842024)

[3.3.3.3 JSON Syntax 111](#_Toc509842025)

[3.3.4 Component TransformedData 112](#_Toc509842026)

[3.3.4.1 Semantics 112](#_Toc509842027)

[3.3.4.2 XML Syntax 112](#_Toc509842028)

[3.3.4.3 JSON Syntax 113](#_Toc509842029)

[3.3.5 Component DocumentHash 114](#_Toc509842030)

[3.3.5.1 Semantics 114](#_Toc509842031)

[3.3.5.2 XML Syntax 114](#_Toc509842032)

[3.3.5.3 JSON Syntax 114](#_Toc509842033)

[3.3.6 Component OptionalInputsSign 115](#_Toc509842034)

[3.3.6.1 Semantics 115](#_Toc509842035)

[3.3.6.2 XML Syntax 116](#_Toc509842036)

[3.3.6.3 JSON Syntax 117](#_Toc509842037)

[3.3.7 Component ClaimedIdentity 119](#_Toc509842038)

[3.3.7.1 Semantics 119](#_Toc509842039)

[3.3.7.2 XML Syntax 119](#_Toc509842040)

[3.3.7.3 JSON Syntax 120](#_Toc509842041)

[3.3.8 Component Schemas 120](#_Toc509842042)

[3.3.8.1 Semantics 120](#_Toc509842043)

[3.3.8.2 XML Syntax 121](#_Toc509842044)

[3.3.8.3 JSON Syntax 121](#_Toc509842045)

[3.3.9 Component UpdateSignatureInstruction 122](#_Toc509842046)

[3.3.9.1 Semantics 122](#_Toc509842047)

[3.3.9.2 XML Syntax 122](#_Toc509842048)

[3.3.9.3 JSON Syntax 122](#_Toc509842049)

[3.3.10 Component IntendedAudience 123](#_Toc509842050)

[3.3.10.1 Semantics 123](#_Toc509842051)

[3.3.10.2 XML Syntax 123](#_Toc509842052)

[3.3.10.3 JSON Syntax 123](#_Toc509842053)

[3.3.11 Component KeySelector 124](#_Toc509842054)

[3.3.11.1 Semantics 124](#_Toc509842055)

[3.3.11.2 XML Syntax 124](#_Toc509842056)

[3.3.11.3 JSON Syntax 125](#_Toc509842057)

[3.3.12 Component X509Digest 125](#_Toc509842058)

[3.3.12.1 Semantics 125](#_Toc509842059)

[3.3.12.2 XML Syntax 126](#_Toc509842060)

[3.3.12.3 JSON Syntax 126](#_Toc509842061)

[3.3.13 Component PropertiesHolder 127](#_Toc509842062)

[3.3.13.1 Semantics 127](#_Toc509842063)

[3.3.13.2 XML Syntax 127](#_Toc509842064)

[3.3.13.3 JSON Syntax 127](#_Toc509842065)

[3.3.14 Component Properties 128](#_Toc509842066)

[3.3.14.1 Semantics 128](#_Toc509842067)

[3.3.14.2 XML Syntax 128](#_Toc509842068)

[3.3.14.3 JSON Syntax 128](#_Toc509842069)

[3.3.15 Component Property 129](#_Toc509842070)

[3.3.15.1 Semantics 129](#_Toc509842071)

[3.3.15.2 XML Syntax 129](#_Toc509842072)

[3.3.15.3 JSON Syntax 130](#_Toc509842073)

[3.3.16 Component IncludeObject 130](#_Toc509842074)

[3.3.16.1 Semantics 130](#_Toc509842075)

[3.3.16.2 XML Syntax 131](#_Toc509842076)

[3.3.16.3 JSON Syntax 131](#_Toc509842077)

[3.3.17 Component DocumentBase 132](#_Toc509842078)

[3.3.17.1 Semantics 132](#_Toc509842079)

[3.3.17.2 XML Syntax 132](#_Toc509842080)

[3.3.17.3 JSON Syntax 133](#_Toc509842081)

[3.3.18 Component SignaturePlacement 133](#_Toc509842082)

[3.3.18.1 Semantics 133](#_Toc509842083)

[3.3.18.2 XML Syntax 134](#_Toc509842084)

[3.3.18.3 JSON Syntax 134](#_Toc509842085)

[3.3.19 Component SignedReferences 135](#_Toc509842086)

[3.3.19.1 Semantics 135](#_Toc509842087)

[3.3.19.2 XML Syntax 136](#_Toc509842088)

[3.3.19.3 JSON Syntax 136](#_Toc509842089)

[3.3.20 Component SignedReference 137](#_Toc509842090)

[3.3.20.1 Semantics 137](#_Toc509842091)

[3.3.20.2 XML Syntax 137](#_Toc509842092)

[3.3.20.3 JSON Syntax 137](#_Toc509842093)

[3.3.21 Component SignResponse 138](#_Toc509842094)

[3.3.21.1 Semantics 138](#_Toc509842095)

[3.3.21.2 XML Syntax 138](#_Toc509842096)

[3.3.21.3 JSON Syntax 139](#_Toc509842097)

[3.3.22 Component OptionalOutputsSign 140](#_Toc509842098)

[3.3.22.1 Semantics 140](#_Toc509842099)

[3.3.22.2 XML Syntax 140](#_Toc509842100)

[3.3.22.3 JSON Syntax 140](#_Toc509842101)

[3.3.23 Component TransformedDocument 141](#_Toc509842102)

[3.3.23.1 Semantics 141](#_Toc509842103)

[3.3.23.2 XML Syntax 141](#_Toc509842104)

[3.3.23.3 JSON Syntax 142](#_Toc509842105)

[3.3.24 Component DocumentWithSignature 142](#_Toc509842106)

[3.3.24.1 Semantics 142](#_Toc509842107)

[3.3.24.2 XML Syntax 143](#_Toc509842108)

[3.3.24.3 JSON Syntax 143](#_Toc509842109)

[3.3.25 Component SignatureObject 144](#_Toc509842110)

[3.3.25.1 Semantics 144](#_Toc509842111)

[3.3.25.2 XML Syntax 144](#_Toc509842112)

[3.3.25.3 JSON Syntax 144](#_Toc509842113)

[3.3.26 Component SignaturePtr 145](#_Toc509842114)

[3.3.26.1 Semantics 145](#_Toc509842115)

[3.3.26.2 XML Syntax 145](#_Toc509842116)

[3.3.26.3 JSON Syntax 146](#_Toc509842117)

[3.3.27 Component VerifyRequest 146](#_Toc509842118)

[3.3.27.1 Semantics 146](#_Toc509842119)

[3.3.27.2 XML Syntax 147](#_Toc509842120)

[3.3.27.3 JSON Syntax 147](#_Toc509842121)

[3.3.28 Component OptionalInputsVerify 148](#_Toc509842122)

[3.3.28.1 Semantics 148](#_Toc509842123)

[3.3.28.2 XML Syntax 149](#_Toc509842124)

[3.3.28.3 JSON Syntax 150](#_Toc509842125)

[3.3.29 Component UseVerificationTime 152](#_Toc509842126)

[3.3.29.1 Semantics 152](#_Toc509842127)

[3.3.29.2 XML Syntax 153](#_Toc509842128)

[3.3.29.3 JSON Syntax 153](#_Toc509842129)

[3.3.30 Component AdditionalKeyInfo 154](#_Toc509842130)

[3.3.30.1 Semantics 154](#_Toc509842131)

[3.3.30.2 XML Syntax 154](#_Toc509842132)

[3.3.30.3 JSON Syntax 155](#_Toc509842133)

[3.3.31 Component ReturnTransformedDocument 156](#_Toc509842134)

[3.3.31.1 Semantics 156](#_Toc509842135)

[3.3.31.2 XML Syntax 156](#_Toc509842136)

[3.3.31.3 JSON Syntax 156](#_Toc509842137)

[3.3.32 Component VerifyResponse 157](#_Toc509842138)

[3.3.32.1 Semantics 157](#_Toc509842139)

[3.3.32.2 XML Syntax 157](#_Toc509842140)

[3.3.32.3 JSON Syntax 157](#_Toc509842141)

[3.3.33 Component OptionalOutputsVerify 158](#_Toc509842142)

[3.3.33.1 Semantics 158](#_Toc509842143)

[3.3.33.2 XML Syntax 159](#_Toc509842144)

[3.3.33.3 JSON Syntax 160](#_Toc509842145)

[3.3.34 Component VerifyManifestResults 161](#_Toc509842146)

[3.3.34.1 Semantics 161](#_Toc509842147)

[3.3.34.2 XML Syntax 162](#_Toc509842148)

[3.3.34.3 JSON Syntax 162](#_Toc509842149)

[3.3.35 Component ManifestResult 163](#_Toc509842150)

[3.3.35.1 Semantics 163](#_Toc509842151)

[3.3.35.2 XML Syntax 163](#_Toc509842152)

[3.3.35.3 JSON Syntax 163](#_Toc509842153)

[3.3.36 Component SigningTimeInfo 164](#_Toc509842154)

[3.3.36.1 Semantics 164](#_Toc509842155)

[3.3.36.2 XML Syntax 165](#_Toc509842156)

[3.3.36.3 JSON Syntax 165](#_Toc509842157)

[3.3.37 Component VerificationTimeInfo 166](#_Toc509842158)

[3.3.37.1 Semantics 166](#_Toc509842159)

[3.3.37.2 XML Syntax 166](#_Toc509842160)

[3.3.37.3 JSON Syntax 167](#_Toc509842161)

[3.3.38 Component AdditionalTimeInfo 168](#_Toc509842162)

[3.3.38.1 Semantics 168](#_Toc509842163)

[3.3.38.2 XML Syntax 168](#_Toc509842164)

[3.3.38.3 JSON Syntax 169](#_Toc509842165)

[3.3.39 Component ProcessingDetails 169](#_Toc509842166)

[3.3.39.1 Semantics 169](#_Toc509842167)

[3.3.39.2 XML Syntax 170](#_Toc509842168)

[3.3.39.3 JSON Syntax 170](#_Toc509842169)

[3.3.40 Component Detail 171](#_Toc509842170)

[3.3.40.1 Semantics 171](#_Toc509842171)

[3.3.40.2 XML Syntax 171](#_Toc509842172)

[3.3.40.3 JSON Syntax 172](#_Toc509842173)

[3.3.41 Component UpdatedSignature 173](#_Toc509842174)

[3.3.41.1 Semantics 173](#_Toc509842175)

[3.3.41.2 XML Syntax 173](#_Toc509842176)

[3.3.41.3 JSON Syntax 173](#_Toc509842177)

[3.4 Referenced Structure Models from other documents 174](#_Toc509842178)

[3.4.1 Component NameIdentifier 174](#_Toc509842179)

[3.4.1.1 Semantics 174](#_Toc509842180)

[3.4.1.2 XML Syntax 174](#_Toc509842181)

[3.4.1.3 JSON Syntax 175](#_Toc509842182)

[3.4.2 Component NameID 175](#_Toc509842183)

[3.4.2.1 Semantics 175](#_Toc509842184)

[3.4.2.2 XML Syntax 176](#_Toc509842185)

[3.4.2.3 JSON Syntax 176](#_Toc509842186)

[3.4.3 Component Transforms 177](#_Toc509842187)

[3.4.3.1 Semantics 177](#_Toc509842188)

[3.4.3.2 XML Syntax 177](#_Toc509842189)

[3.4.3.3 JSON Syntax 178](#_Toc509842190)

[3.4.4 Component Transform 178](#_Toc509842191)

[3.4.4.1 Semantics 178](#_Toc509842192)

[3.4.4.2 XML Syntax 179](#_Toc509842193)

[3.4.4.3 JSON Syntax 179](#_Toc509842194)

[3.4.5 Component X509Data 180](#_Toc509842195)

[3.4.5.1 Semantics 180](#_Toc509842196)

[3.4.5.2 XML Syntax 181](#_Toc509842197)

[3.4.5.3 JSON Syntax 181](#_Toc509842198)

[3.4.6 Component X509IssuerSerial 183](#_Toc509842199)

[3.4.6.1 Semantics 183](#_Toc509842200)

[3.4.6.2 XML Syntax 183](#_Toc509842201)

[3.4.6.3 JSON Syntax 183](#_Toc509842202)

[3.4.7 Component DigestMethod 184](#_Toc509842203)

[3.4.7.1 Semantics 184](#_Toc509842204)

[3.4.7.2 XML Syntax 184](#_Toc509842205)

[3.4.7.3 JSON Syntax 185](#_Toc509842206)

[3.4.8 Component CanonicalizationMethod 185](#_Toc509842207)

[3.4.8.1 Semantics 185](#_Toc509842208)

[3.4.8.2 XML Syntax 186](#_Toc509842209)

[3.4.8.3 JSON Syntax 186](#_Toc509842210)

[3.4.9 Component SignatureValue 187](#_Toc509842211)

[3.4.9.1 Semantics 187](#_Toc509842212)

[3.4.9.2 XML Syntax 188](#_Toc509842213)

[3.4.9.3 JSON Syntax 188](#_Toc509842214)

[3.4.10 Component ValidationConstraints 189](#_Toc509842215)

[3.4.10.1 Semantics 189](#_Toc509842216)

[3.4.10.2 XML Syntax 189](#_Toc509842217)

[3.4.10.3 JSON Syntax 189](#_Toc509842218)

[3.4.11 Component SingleValidationConstraint 190](#_Toc509842219)

[3.4.11.1 Semantics 190](#_Toc509842220)

[3.4.11.2 XML Syntax 190](#_Toc509842221)

[3.4.11.3 JSON Syntax 191](#_Toc509842222)

[3.4.12 Component ValidationConstraint 192](#_Toc509842223)

[3.4.12.1 Semantics 192](#_Toc509842224)

[3.4.12.2 XML Syntax 192](#_Toc509842225)

[3.4.12.3 JSON Syntax 192](#_Toc509842226)

[3.4.13 Component ValidationConstraintParameter 193](#_Toc509842227)

[3.4.13.1 Semantics 193](#_Toc509842228)

[3.4.13.2 XML Syntax 193](#_Toc509842229)

[3.4.13.3 JSON Syntax 194](#_Toc509842230)

[3.4.14 Component ConstraintStatus 194](#_Toc509842231)

[3.4.14.1 Semantics 194](#_Toc509842232)

[3.4.14.2 XML Syntax 195](#_Toc509842233)

[3.4.14.3 JSON Syntax 195](#_Toc509842234)

[3.4.15 Component SignersDocument 196](#_Toc509842235)

[3.4.15.1 Semantics 196](#_Toc509842236)

[3.4.15.2 XML Syntax 196](#_Toc509842237)

[3.4.15.3 JSON Syntax 196](#_Toc509842238)

[3.4.16 Component VOReference 197](#_Toc509842239)

[3.4.16.1 Semantics 197](#_Toc509842240)

[3.4.16.2 XML Syntax 197](#_Toc509842241)

[3.4.16.3 JSON Syntax 198](#_Toc509842242)

[3.4.17 Component ValidationObject 198](#_Toc509842243)

[3.4.17.1 Semantics 198](#_Toc509842244)

[3.4.17.2 XML Syntax 199](#_Toc509842245)

[3.4.17.3 JSON Syntax 200](#_Toc509842246)

[3.4.18 Component ValidationObjectRepresentation 201](#_Toc509842247)

[3.4.18.1 Semantics 201](#_Toc509842248)

[3.4.18.2 XML Syntax 202](#_Toc509842249)

[3.4.18.3 JSON Syntax 202](#_Toc509842250)

[3.4.19 Component PoE 203](#_Toc509842251)

[3.4.19.1 Semantics 203](#_Toc509842252)

[3.4.19.2 XML Syntax 203](#_Toc509842253)

[3.4.19.3 JSON Syntax 203](#_Toc509842254)

[3.4.20 Component SignerInformation 204](#_Toc509842255)

[3.4.20.1 Semantics 204](#_Toc509842256)

[3.4.20.2 XML Syntax 204](#_Toc509842257)

[3.4.20.3 JSON Syntax 205](#_Toc509842258)

[3.4.21 Component SignatureQualityList 206](#_Toc509842259)

[3.4.21.1 Semantics 206](#_Toc509842260)

[3.4.21.2 XML Syntax 206](#_Toc509842261)

[3.4.21.3 JSON Syntax 206](#_Toc509842262)

[3.4.22 Component SignatureValidationProcess 207](#_Toc509842263)

[3.4.22.1 Semantics 207](#_Toc509842264)

[3.4.22.2 XML Syntax 207](#_Toc509842265)

[3.4.22.3 JSON Syntax 208](#_Toc509842266)

[3.4.23 Component ValidationReportData 209](#_Toc509842267)

[3.4.23.1 Semantics 209](#_Toc509842268)

[3.4.23.2 XML Syntax 209](#_Toc509842269)

[3.4.23.3 JSON Syntax 210](#_Toc509842270)

[3.4.24 Component RevocationStatusInformation 211](#_Toc509842271)

[3.4.24.1 Semantics 211](#_Toc509842272)

[3.4.24.2 XML Syntax 211](#_Toc509842273)

[3.4.24.3 JSON Syntax 211](#_Toc509842274)

[3.4.25 Component CryptoInformation 212](#_Toc509842275)

[3.4.25.1 Semantics 212](#_Toc509842276)

[3.4.25.2 XML Syntax 213](#_Toc509842277)

[3.4.25.3 JSON Syntax 213](#_Toc509842278)

[3.4.26 Component AlgorithmParameter 214](#_Toc509842279)

[3.4.26.1 Semantics 214](#_Toc509842280)

[3.4.26.2 XML Syntax 214](#_Toc509842281)

[3.4.26.3 JSON Syntax 215](#_Toc509842282)

[3.4.27 Component AdditionalValidationReportData 215](#_Toc509842283)

[3.4.27.1 Semantics 215](#_Toc509842284)

[3.4.27.2 XML Syntax 216](#_Toc509842285)

[3.4.27.3 JSON Syntax 216](#_Toc509842286)

[3.4.28 Component ReportData 217](#_Toc509842287)

[3.4.28.1 Semantics 217](#_Toc509842288)

[3.4.28.2 XML Syntax 217](#_Toc509842289)

[3.4.28.3 JSON Syntax 217](#_Toc509842290)

[3.4.29 Component ValidationObjectList 218](#_Toc509842291)

[3.4.29.1 Semantics 218](#_Toc509842292)

[3.4.29.2 XML Syntax 218](#_Toc509842293)

[3.4.29.3 JSON Syntax 218](#_Toc509842294)

[3.4.30 Component AvailableSignatureValidationPolicies 219](#_Toc509842295)

[3.4.30.1 Semantics 219](#_Toc509842296)

[3.4.30.2 XML Syntax 219](#_Toc509842297)

[3.4.30.3 JSON Syntax 219](#_Toc509842298)

[3.4.31 Component ObjectIdentifier 220](#_Toc509842299)

[3.4.31.1 Semantics 220](#_Toc509842300)

[3.4.31.2 XML Syntax 220](#_Toc509842301)

[3.4.31.3 JSON Syntax 221](#_Toc509842302)

[3.4.32 Component Identifier 221](#_Toc509842303)

[3.4.32.1 Semantics 221](#_Toc509842304)

[3.4.32.2 XML Syntax 222](#_Toc509842305)

[3.4.32.3 JSON Syntax 222](#_Toc509842306)

[3.4.33 Component DocumentationReferences 223](#_Toc509842307)

[3.4.33.1 Semantics 223](#_Toc509842308)

[3.4.33.2 XML Syntax 223](#_Toc509842309)

[3.4.33.3 JSON Syntax 223](#_Toc509842310)

[3.4.34 Component DigestAlgAndValue 224](#_Toc509842311)

[3.4.34.1 Semantics 224](#_Toc509842312)

[3.4.34.2 XML Syntax 224](#_Toc509842313)

[3.4.34.3 JSON Syntax 224](#_Toc509842314)

[3.4.35 Component CertIDList 225](#_Toc509842315)

[3.4.35.1 Semantics 225](#_Toc509842316)

[3.4.35.2 XML Syntax 225](#_Toc509842317)

[3.4.35.3 JSON Syntax 226](#_Toc509842318)

[3.4.36 Component CertID 226](#_Toc509842319)

[3.4.36.1 Semantics 226](#_Toc509842320)

[3.4.36.2 XML Syntax 226](#_Toc509842321)

[3.4.36.3 JSON Syntax 227](#_Toc509842322)

[3.4.37 Component SignaturePolicyIdentifier 227](#_Toc509842323)

[3.4.37.1 Semantics 227](#_Toc509842324)

[3.4.37.2 XML Syntax 228](#_Toc509842325)

[3.4.37.3 JSON Syntax 228](#_Toc509842326)

[3.4.38 Component SignaturePolicyId 229](#_Toc509842327)

[3.4.38.1 Semantics 229](#_Toc509842328)

[3.4.38.2 XML Syntax 229](#_Toc509842329)

[3.4.38.3 JSON Syntax 230](#_Toc509842330)

[3.4.39 Component SigPolicyQualifiersList 230](#_Toc509842331)

[3.4.39.1 Semantics 230](#_Toc509842332)

[3.4.39.2 XML Syntax 231](#_Toc509842333)

[3.4.39.3 JSON Syntax 231](#_Toc509842334)

[3.4.40 Component Any 232](#_Toc509842335)

[3.4.40.1 Semantics 232](#_Toc509842336)

[3.4.40.2 XML Syntax 232](#_Toc509842337)

[3.4.40.3 JSON Syntax 232](#_Toc509842338)

[3.4.41 Component SignatureProductionPlace 233](#_Toc509842339)

[3.4.41.1 Semantics 233](#_Toc509842340)

[3.4.41.2 XML Syntax 233](#_Toc509842341)

[3.4.41.3 JSON Syntax 233](#_Toc509842342)

[3.4.42 Component ClaimedRolesList 234](#_Toc509842343)

[3.4.42.1 Semantics 234](#_Toc509842344)

[3.4.42.2 XML Syntax 235](#_Toc509842345)

[3.4.42.3 JSON Syntax 235](#_Toc509842346)

[3.4.43 Component CRLIdentifier 236](#_Toc509842347)

[3.4.43.1 Semantics 236](#_Toc509842348)

[3.4.43.2 XML Syntax 236](#_Toc509842349)

[3.4.43.3 JSON Syntax 236](#_Toc509842350)

[3.4.44 Component OCSPIdentifier 237](#_Toc509842351)

[3.4.44.1 Semantics 237](#_Toc509842352)

[3.4.44.2 XML Syntax 237](#_Toc509842353)

[3.4.44.3 JSON Syntax 238](#_Toc509842354)

[3.4.45 Component ResponderID 238](#_Toc509842355)

[3.4.45.1 Semantics 238](#_Toc509842356)

[3.4.45.2 XML Syntax 239](#_Toc509842357)

[3.4.45.3 JSON Syntax 239](#_Toc509842358)

[3.4.46 Component DataObjectFormat 240](#_Toc509842359)

[3.4.46.1 Semantics 240](#_Toc509842360)

[3.4.46.2 XML Syntax 240](#_Toc509842361)

[3.4.46.3 JSON Syntax 240](#_Toc509842362)

[3.4.47 Component CommitmentTypeIndication 241](#_Toc509842363)

[3.4.47.1 Semantics 241](#_Toc509842364)

[3.4.47.2 XML Syntax 242](#_Toc509842365)

[3.4.47.3 JSON Syntax 242](#_Toc509842366)

[3.4.48 Component CommitmentTypeQualifiersList 243](#_Toc509842367)

[3.4.48.1 Semantics 243](#_Toc509842368)

[3.4.48.2 XML Syntax 243](#_Toc509842369)

[3.4.48.3 JSON Syntax 244](#_Toc509842370)

[3.4.49 Component CompleteCertificateRefs 244](#_Toc509842371)

[3.4.49.1 Semantics 244](#_Toc509842372)

[3.4.49.2 XML Syntax 245](#_Toc509842373)

[3.4.49.3 JSON Syntax 245](#_Toc509842374)

[3.4.50 Component CompleteRevocationRefs 246](#_Toc509842375)

[3.4.50.1 Semantics 246](#_Toc509842376)

[3.4.50.2 XML Syntax 246](#_Toc509842377)

[3.4.50.3 JSON Syntax 246](#_Toc509842378)

[3.4.51 Component CRLRefs 247](#_Toc509842379)

[3.4.51.1 Semantics 247](#_Toc509842380)

[3.4.51.2 XML Syntax 247](#_Toc509842381)

[3.4.51.3 JSON Syntax 248](#_Toc509842382)

[3.4.52 Component CRLRef 248](#_Toc509842383)

[3.4.52.1 Semantics 248](#_Toc509842384)

[3.4.52.2 XML Syntax 249](#_Toc509842385)

[3.4.52.3 JSON Syntax 249](#_Toc509842386)

[3.4.53 Component OCSPRefs 250](#_Toc509842387)

[3.4.53.1 Semantics 250](#_Toc509842388)

[3.4.53.2 XML Syntax 250](#_Toc509842389)

[3.4.53.3 JSON Syntax 250](#_Toc509842390)

[3.4.54 Component OCSPRef 251](#_Toc509842391)

[3.4.54.1 Semantics 251](#_Toc509842392)

[3.4.54.2 XML Syntax 251](#_Toc509842393)

[3.4.54.3 JSON Syntax 251](#_Toc509842394)

[3.4.55 Component OtherCertStatusRefs 252](#_Toc509842395)

[3.4.55.1 Semantics 252](#_Toc509842396)

[3.4.55.2 XML Syntax 252](#_Toc509842397)

[3.4.55.3 JSON Syntax 253](#_Toc509842398)

[3.4.56 Component UnsignedDataObjectProperties 253](#_Toc509842399)

[3.4.56.1 Semantics 253](#_Toc509842400)

[3.4.56.2 XML Syntax 254](#_Toc509842401)

[3.4.56.3 JSON Syntax 254](#_Toc509842402)

[4 Conformance 256](#_Toc509842403)

[4.1 Level 1 – “Basic” 256](#_Toc509842404)

[4.2 Level 2 – “Comprehensive” 256](#_Toc509842405)

[Appendix A. Index 257](#_Toc509842406)

[Appendix B. Revision History 258](#_Toc509842407)

# Introduction

## Organization of DSS Core Protocols, Elements, and Bindings

This document defines a protocol and processing profile of the DSS Verifying Protocol specified in **[DSSCore]**, which allows to support the verification of multiple signatures within a VerifyRequest component and include detailed information of the different steps taken during verification.

The following sections describe how to understand the rest of this document.

## Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

### Terms and Definitions

For the purposes of this document, the following applies:

**Term** — meaning and maybe ref

### Abbreviated Terms

**Acronym** — Spelled out

## Normative References

[RFC2119] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, BCP 14, RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>.

**[RFC 2396]** T. Berners-Lee et al. *Uniform Resource Identifiers (URI): Generic Syntax.* IETF RFC 2396, August 1998.   
<http://www.ietf.org/rfc/rfc2396.txt>.

[DSS2XSD] S. Hagen,. *DSS 2.0 Core Schema*. OASIS, ToDo.

[DSS2BASEXSD] S. Hagen,. *DSS 2.0 Base Schema*. OASIS, ToDo.

**[DSSVR-XSD]** D. Hühnlein, A. Kuehne, et. al.: “*DSS Verification**Report Schema”*, , ToDo.

**[RFC 2440]** J. Callas, L. Donnerhacke, H. Finney, R. Thayer. *OpenPGP Message Format*. IETF RFC 2440, November 1998.  
<http://www.ietf.org/rfc/rfc2440.txt>.

**[RFC 2616]** R. Fielding et al. *Hypertext Transfer Protocol – HTTP/1.1*. IETF RFC 2616, June 1999.  
[http://www.ietf.org/rfc/rfc2616.txt](http://www.ietf.org/rfc/rfc2440.txt).

**[RFC 2648]** R. Moats. *A URN Namespace for IETF Documents*. IETF RFC 2648, August 1999.   
<http://www.ietf.org/rfc/rfc2648.txt>.

**[RFC 2822]** P. Resnick. *Internet Message Format*. IETF RFC 2822, April 2001. <http://www.ietf.org/rfc/rfc2822.txt>

**[RFC 3161]** C. Adams, P. Cain, D. Pinkas, R. Zuccherato. *Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)*. IETF RFC 3161, August 2001.   
[http://www.ietf.org/rfc/rfc3161.txt](http://www.ietf.org/rfc/rfc3075.txt).

**[RFC 5280]** D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF RFC 5280, May 2008.   
<http://www.ietf.org/rfc/rfc5280.txt>.

**[RFC 5652]** R. Housley. *Cryptographic Message Syntax*. IETF RFC 5652, September 2009.  
<http://www.ietf.org/rfc/rfc5652.txt>.   
(Remark: As used in DSS, all implementations based upon RFC 5652 and previous releases of CMS will suffice. For the sake of simplicity the "urn:ietf:rfc:3369" is used throughout the document to indicate a CMS message as specified in RFC 5652 or RFC 3369 or any version (including PKCS #7).

**[****RFC7159]** T. Bray, Ed., Google, Inc., The JavaScript Object Notation (JSON) Data Interchange Format, ISSN: 2070-1721, March 2014.   
<https://tools.ietf.org/html/rfc7159>.

**[SAMLCore1.1]** E. Maler et al. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V 1.1. OASIS, November 2002.   
<http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf>

**[SAMLCore2.0]** “*Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0”,* S. Cantor, J. Kemp, R. Philpott, E. Maler, Editors, OASIS Standard, 15 March 2005.

**[XAdES]** ETSI: “*XML Advanced Electronic Signatures (XAdES)”*, ETSI TS 101 903, Version 1.3.2, March 2006

**[XMLDSIG]** *D. Eastlake et al. XML-Signature Syntax and Processing. W3C Recommendation, February 2002.*<http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/>

[XML] Extensible Markup Language (XML) 1.0 (Fifth Edition), T. Bray, J. Paoli, M. Sperberg-McQueen, E. Maler, F. Yergeau, Editors, W3C Recommendation, November 26, 2008, <http://www.w3.org/TR/2008/REC-xml-20081126/>.   
Latest version available at <http://www.w3.org/TR/xml>.

[XML-Schema-1] W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures, S. Gao, M. Sperberg-McQueen, H. Thompson, N. Mendelsohn, D. Beech, M. Maloney, Editors, W3C Recommendation, April 5, 2012,   
<http://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/>.   
Latest version available at <http://www.w3.org/TR/xmlschema11-1/>.

[XML-Schema-2] W3C XML Schema Definition Language (XSD) 1.1 Part 2: DatatypesW3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes, D. Peterson, S. Gao, A. Malhotra, M. Sperberg-McQueen, H. Thompson, Paul V. Biron, Editors, W3C Recommendation, April 5, 2012,   
<http://www.w3.org/TR/2012/REC-xmlschema11-2-20120405/>.   
Latest version available at <http://www.w3.org/TR/xmlschema11-2/>.

**[XPATH]** XML Path Language (XPath) Version 1.0. W3C Recommendation 16 November 1999 <http://www.w3.org/TR/xpath>

## Non-Normative References

[ISO8601] Data elements and interchange formats — Information interchange — Representation of dates and times, International Standard, ISO 8601:2004(E), December 1, 2004, <https://www.iso.org/standard/40874.html>.

## Typographical Conventions

Keywords defined by this specification use this monospaced font.

Normative source code uses this paragraph style.

Text following the special symbol («) – an opening Guillemet (or French quotation mark) – within this specification identifies conformance statements. Every conformance statement is separated from the following text with the special end symbol (») – a closing Guillemet, and has been assigned a reference that follows that end symbol in the format [dSS-section#-local#].

Some sections of this specification are illustrated with non-normative examples.

Example 1: text describing an example uses this paragraph style

Non-normative examples use this paragraph style.

All examples in this document are non-normative and informative only.

Representation-specific text is indented and marked with vertical lines.

Representation-Specific Headline

Normative representation-specific text

All other text is normative unless otherwise labeled e.g. like:

Non-normative Comment:

This is a pure informative comment that may be present, because the information conveyed is deemed useful advice or common pitfalls learned from implementer or operator experience and often given including the rationale.

# Design Considerations

Blurb

## Construction Principles

## Domain Models

### Date and Time Model

The specific concept of date and time used in this document is defined in this section and noted in subsequent usage as**:**

DateTime

« All date time values inside a DSS document MUST adhere to the ISO 8601 [[ISO8601](#refISO8601)] basic or extended Format (as given there in section 4.3.2 “Complete representations” and with the addition of decimal fractions for seconds, similar to ibid. section 4.2.2.4 “Representations with decimal fraction” but with the full stop (.) being the preferred separator for DSS). » [DSS-2.2.1-1].

## Schema Organization and Namespaces

The structures described in this specification are contained in the schema file **[Core2.0-XSD]**. All schema listings in the current document are excerpts from the schema file. In the case of a disagreement between the schema file and this document, the schema file takes precedence.

This schema is associated with the following XML namespace:

http://docs.oasis-open.org/dss/ns/verificationreport

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

Conventional XML namespace prefixes are used in the schema:

* The prefix dss2: stands for the DSS core namespace **[DSS2XSD]**.
* The prefix dsb: stands for the DSS base namespace **[DSS2XSD]**.
* The prefix xs: stands for the W3C XML Schema namespace **[Schema1]**.
* The prefix vr: (or no prefix) stands for this profiles namespace **[DSSVR-XSD]**.
* The prefix ds-rw: stands for a namespace of elements based on the W3C XML Signature **[XMLDSIG]**.
* The prefix saml-rw: stands for a namespace of elements based on the OASIS SAML Schema **[SAMLCore1.1]**.
* The prefix saml2-rw: stands for a namespace of elements based on the OASIS SAML 2 Schema namespace **[SAMLCore2.0]**.
* The prefix xades-rw: stands for a namespace of elements based on the ETSI XML Advanced Electronic Signatures (XAdES) document **[XAdES]**.

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

## DSS Overview (Non-normative)

While the DSS Verifying Protocol specified in **[DSSCore]** allows to verify digital signatures and time stamps, this protocol is fairly limited with respect to the verification of multiple signatures in a single request (cf. Section 4.3.1 of **[DSSCore]**).

In a similar manner it is possible to request and provide processing details (cf. Section 4.5.5 of **[DSSCore]**), but this simple mechanism does not support the verification of multiple signatures in a single request. And there are no defined structures yet, which reflect the necessary steps in the verification of a complex signature, like an advanced electronic signature according to the European Directive **[EC/1999/93]** for example.

Therefore, the present profile defines how

* individual verification results may be returned, if multiple signatures are part of a VerifyRequest component and
* detailed information gathered in the various steps taken during verification may be included in the response to form a comprehensive verification report.

The requester MAY request the activation of this profile by sending a ReturnVerificationReport element (cf. Section 3.1) in OptionalInputsVerify component. A responder, which conforms to the present profile SHALL return a VerificationReport element in OptionalOutputsVerify component.

## Syntax variants

This version of the DSS/X profile document handles the representation of requests and response elements according to the JSON and XML syntax. The general semantics of the elements is discussed in the element’s main section. Details of the JSON or XML formats are discussed in specific subsections

* JSON syntax
* XML syntax

# Structure Models

## Structure Models defined in this document

The XML elements of this section are defined in the XML namespace 'urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#'.

[namespace urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema# explanation]

### Component ReturnVerificationReport

#### Semantics

[component ReturnVerificationReport normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional IncludeVerifier element MUST contain a boolean. Its default value is 'true'. This option specifies, whether the identity of the verifier should be included into the report or not. This is especially useful when (possibly time stamped) reports are archived.
* The optional IncludeCertificateValues element MUST contain a boolean. Its default value is 'false'. With this option it is possible to include the certificate values, which are used to verify the signature into the report.
* The optional IncludeRevocationValues element MUST contain a boolean. Its default value is 'false'. This option specifies, whether the used revocation values (OCSP responses, CRLs and TSLs) should be included into the report or not.
* The optional ReportDetailLevel element MUST contain a URI. Its default value is 'urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:allDetails'. This option specifies the detail level of the verification report. The following options are defined: urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:noDetails  
  For every signature only the final result of the verification is reported. urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:noPathDetails  
  Additionally to the final result also the details of the signature verification including the result of the certificate path validation are reported. The details concerning the validation of individual certificates in the path are omitted however. urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:allDetails  
  For every signature, the certificate path details and details on the validation of individual certificates in the path are requested. For every signature, the certificate path and each individual certificate the details are reported.

Non-normative Comment:

[component ReturnVerificationReport non normative details]

#### XML Syntax

The XML type ReturnVerificationReportType SHALL implement the requirements defined in the ReturnVerificationReport component.

The ReturnVerificationReportType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ReturnVerificationReportType">

<sequence>

<element default="true" maxOccurs="1" minOccurs="0" name="IncludeVerifier" type="boolean"/>

<element default="false" maxOccurs="1" minOccurs="0" name="IncludeCertificateValues" type="boolean"/>

<element default="false" maxOccurs="1" minOccurs="0" name="IncludeRevocationValues" type="boolean"/>

<element default="urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:allDetails" maxOccurs="1" minOccurs="0" name="ReportDetailLevel" type="anyURI"/>

</sequence>

</xs:complexType>

Each child element of ReturnVerificationReportType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ReturnVerificationReport XML schema details]

#### JSON Syntax

The ReturnVerificationReportType JSON object SHALL implement in JSON syntax the requirements defined in the ReturnVerificationReport component.

The ReturnVerificationReportType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-ReturnVerificationReportType": {

"$xsd-full-type": "vr:ReturnVerificationReportType",

"type": "object",

"properties": {

"incVerifier": {

"type": "boolean",

"default": "true"

},

"incCertValues": {

"type": "boolean",

"default": "false"

},

"incRevocValues": {

"type": "boolean",

"default": "false"

},

"level": {

"type": "string",

"default": "urn:oasis:names:tc:dss:1.0:profiles:verificationreport:reportdetail:allDetails"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ReturnVerificationReport component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| IncludeVerifier | incVerifier | [] |
| IncludeCertificateValues | incCertValues | [] |
| IncludeRevocationValues | incRevocValues | [] |
| ReportDetailLevel | level | [] |

[component ReturnVerificationReport JSON schema details]

### Component VerificationReport

#### Semantics

If the element <ReturnVerificationReport> is provided as optional input in the request, the server MUST include in the response the component VerificationReport as optional output.

Below follows a list of the sub-components that MAY be present within this component:

* The optional VerificationTimeInfo element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerificationTimeInfo. This element MAY contain the verification time, which was used by the server and other relevant time instants.
* The optional VerifierIdentity element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Identifier. This element contains the identity of the verifier, if the report option <IncludeVerifier> was set to ‘true’
* The optional IndividualReport element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section IndividualReport. For each *independent* signed object (signature, time stamp, certificate, CRL, OCSP-response, evidence record etc.) that has been used in the signature verification process there will be one IndividualReport component in the verification report. The details of this element are specified in the following section.
* The optional VerificationProcess element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ValidationProcess. [sub component VerificationProcess details]

Non-normative Comment:

[component VerificationReport non normative details]

#### XML Syntax

The XML type VerificationReportType SHALL implement the requirements defined in the VerificationReport component.

The VerificationReportType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="VerificationReportType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="VerificationTimeInfo" type="dss2:VerificationTimeInfoType"/>

<element maxOccurs="1" minOccurs="0" name="VerifierIdentity" type="vr:IdentifierType"/>

<element maxOccurs="unbounded" minOccurs="0" name="IndividualReport" type="vr:IndividualReportType"/>

<xs:element maxOccurs="1" minOccurs="0" name="VerificationProcess" type="vr:ValidationProcessType"/>

</sequence>

</complexType>

Each child element of VerificationReportType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerificationReport XML schema details]

#### JSON Syntax

The VerificationReportType JSON object SHALL implement in JSON syntax the requirements defined in the VerificationReport component.

The VerificationReportType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-VerificationReportType": {

"$xsd-full-type": "vr:VerificationReportType",

"type": "object",

"properties": {

"verificationTimeInfo": {

"$ref": "#/definitions/dss2-VerificationTimeInfoType"

},

"verifierIdentity": {

"$ref": "#/definitions/vr-IdentifierType"

},

"individualReport": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-IndividualReportType"

}

},

"process": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of VerificationReport component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| VerificationTimeInfo | verificationTimeInfo | [] |
| VerifierIdentity | verifierIdentity | [] |
| IndividualReport | individualReport | [] |
| VerificationProcess | process | [] |

[component VerificationReport JSON schema details]

### Component Identifier

#### Semantics

[component Identifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional X509Data element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section X509Data. This element contains, if present, an X.509-certificate or certificate related information.
* The optional SAMLv1Identifier element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section NameIdentifier. This element contains, if present, an identifier of type saml:NameIdentifierType as defined in **[SAMLCore1.1]**
* The optional SAMLv2Identifier element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section NameID. This element contains, if present, an identifier of type saml2:NameIDType as defined in **[SAMLCore2.0]**.
* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain, if present, other identifying information.

Non-normative Comment:

[component Identifier non normative details]

#### XML Syntax

The XML type IdentifierType SHALL implement the requirements defined in the Identifier component.

The IdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="IdentifierType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="X509Data" type="ds-rw:X509DataType"/>

<element maxOccurs="1" minOccurs="0" name="SAMLv1Identifier" type="saml-rw:NameIdentifierType"/>

<element maxOccurs="1" minOccurs="0" name="SAMLv2Identifier" type="saml2-rw:NameIDType"/>

<element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</sequence>

</complexType>

Each child element of IdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Identifier XML schema details]

#### JSON Syntax

The IdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the Identifier component.

The IdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-IdentifierType": {

"$xsd-full-type": "vr:IdentifierType xades-rw:IdentifierType",

"type": "object",

"properties": {

"samlv1Identifier": {

"$ref": "#/definitions/samlrw-NameIdentifierType"

},

"samlv2Identifier": {

"$ref": "#/definitions/saml2rw-NameIDType"

},

"x509Data": {

"$ref": "#/definitions/dsigrw-X509DataType"

},

"samlV1Id": {

"$ref": "#/definitions/samlrw-NameIdentifierType"

},

"samlV2Id": {

"$ref": "#/definitions/saml2rw-NameIDType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Identifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| X509Data | x509Data | [] |
| SAMLv1Identifier | samlV1Id | [] |
| SAMLv2Identifier | samlV2Id | [] |
| Other | other | [] |

[component Identifier JSON schema details]

### Component ValidationProcess

#### Semantics

[component ValidationProcess normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a URI. [sub component value details]

Non-normative Comment:

[component ValidationProcess non normative details]

#### XML Syntax

The XML type ValidationProcessType SHALL implement the requirements defined in the ValidationProcess component.

The ValidationProcessType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:simpleType name="ValidationProcessType">

<xs:restriction base="xs:anyURI">

<xs:enumeration value="urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:BASIC"/>

<xs:enumeration value="urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:BASIC\_TIME"/>

<xs:enumeration value="urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:LONG\_TERM"/>

<xs:enumeration value="urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:LONG\_TERM\_AVAILABILITY\_INTEGRITY"/>

</xs:restriction>

</xs:simpleType>

Each child element of ValidationProcessType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationProcess XML schema details]

#### JSON Syntax

The ValidationProcessType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationProcess component.

The ValidationProcessType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-ValidationProcessType": {

"$xsd-full-type": "vr:ValidationProcessType",

"type": "string",

"enum": ["urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:BASIC", "urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:BASIC\_TIME", "urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:LONG\_TERM", "urn:oasis:names:tc:dss-x:1.0:profiles:verificationreport:schema#:Process:LONG\_TERM\_AVAILABILITY\_INTEGRITY"]

}

Properties in the JSON schema above SHALL implement sub-component of ValidationProcess component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |

[component ValidationProcess JSON schema details]

### Component ReportDetail

#### Semantics

[component ReportDetail normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional DetailedSignatureReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section DetailedSignatureReport. [sub component DetailedSignatureReport details]
* The optional IndividualTimeStampReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section TimeStampValidity. [sub component IndividualTimeStampReport details]
* The optional IndividualCertificateReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CertificateValidity. [sub component IndividualCertificateReport details]
* The optional IndividualAttributeCertificateReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section AttributeCertificateValidity. [sub component IndividualAttributeCertificateReport details]
* The optional IndividualCRLReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CRLValidity. [sub component IndividualCRLReport details]
* The optional IndividualOCSPReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section OCSPValidity. [sub component IndividualOCSPReport details]
* The optional EvidenceRecordReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section EvidenceRecordValidity. [sub component EvidenceRecordReport details]
* The optional ValidationConstraints element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ValidationConstraints. [sub component ValidationConstraints details]
* The optional SignatureValidationTimeInfo element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section VerificationTimeInfo. [sub component SignatureValidationTimeInfo details]
* The optional SignerDocument element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignersDocument. [sub component SignerDocument details]
* The optional SignerInformation element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignerInformation. [sub component SignerInformation details]
* The optional SignatureQualityList element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignatureQualityList. [sub component SignatureQualityList details]
* The optional SignatureValidationProcess element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignatureValidationProcess. [sub component SignatureValidationProcess details]
* The optional ValidationReportData element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ValidationReportData. [sub component ValidationReportData details]
* The optional ValidationReportSignature element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section SignatureObject. [sub component ValidationReportSignature details]
* The optional ValidationObjectList element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ValidationObjectList. [sub component ValidationObjectList details]
* The optional VerifyManifestResults element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerifyManifestResults. [sub component VerifyManifestResults details]
* The optional SigningTimeInfo element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section SigningTimeInfo. [sub component SigningTimeInfo details]
* The optional VerificationTimeInfo element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerificationTimeInfo. [sub component VerificationTimeInfo details]
* The optional ProcessingDetails element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section ProcessingDetails. [sub component ProcessingDetails details]
* The optional SignerIdentity element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section NameID. [sub component SignerIdentity details]
* The optional UpdatedSignature element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section UpdatedSignature. [sub component UpdatedSignature details]
* The optional TimestampedSignature element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section UpdatedSignature. [sub component TimestampedSignature details]

Non-normative Comment:

[component ReportDetail non normative details]

#### XML Syntax

The XML type ReportDetailType SHALL implement the requirements defined in the ReportDetail component.

The ReportDetailType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ReportDetailType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:DetailedSignatureReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:IndividualTimeStampReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:IndividualCertificateReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:IndividualAttributeCertificateReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:IndividualCRLReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:IndividualOCSPReport"/>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:EvidenceRecordReport"/>

<xs:element maxOccurs="1" minOccurs="0" name="ValidationConstraints" type="etsi-vr:ValidationConstraintsType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="SignatureValidationTimeInfo" type="dss2:VerificationTimeInfoType"/>

<xs:element minOccurs="0" name="SignerDocument" type="etsi-vr:SignersDocumentType"/>

<xs:element minOccurs="0" name="SignerInformation" type="etsi-vr:SignerInformationType"/>

<xs:element minOccurs="0" name="SignatureQualityList" type="etsi-vr:SignatureQualityListType"/>

<xs:element minOccurs="0" name="SignatureValidationProcess" type="etsi-vr:SignatureValidationProcessType"/>

<xs:element minOccurs="0" name="ValidationReportData" type="etsi-vr:ValidationReportDataType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ValidationReportSignature" type="dss2:SignatureObjectType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ValidationObjectList" type="etsi-vr:ValidationObjectListType"/>

<xs:element maxOccurs="1" minOccurs="0" name="VerifyManifestResults" type="dss2:VerifyManifestResultsType"/>

<xs:element maxOccurs="1" minOccurs="0" name="SigningTimeInfo" type="dss2:SigningTimeInfoType"/>

<xs:element maxOccurs="1" minOccurs="0" name="VerificationTimeInfo" type="dss2:VerificationTimeInfoType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ProcessingDetails" type="dss2:ProcessingDetailsType"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignerIdentity" type="saml2-rw:NameIDType"/>

<xs:element maxOccurs="1" minOccurs="0" name="UpdatedSignature" type="dss2:UpdatedSignatureType"/>

<xs:element maxOccurs="1" minOccurs="0" name="TimestampedSignature" type="dss2:UpdatedSignatureType"/>

</xs:sequence>

</xs:complexType>

Each child element of ReportDetailType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ReportDetail XML schema details]

#### JSON Syntax

The ReportDetailType JSON object SHALL implement in JSON syntax the requirements defined in the ReportDetail component.

The ReportDetailType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-ReportDetailType": {

"$xsd-full-type": "vr:ReportDetailType",

"type": "object",

"properties": {

"detailedSigReport": {

"$ref": "#/definitions/vr-DetailedSignatureReportType"

},

"individualTSReport": {

"$ref": "#/definitions/vr-TimeStampValidityType"

},

"individualCertReport": {

"$ref": "#/definitions/vr-CertificateValidityType"

},

"individualAttCertReport": {

"$ref": "#/definitions/vr-AttributeCertificateValidityType"

},

"individualCRLReport": {

"$ref": "#/definitions/vr-CRLValidityType"

},

"individualOCSPReport": {

"$ref": "#/definitions/vr-OCSPValidityType"

},

"erReport": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType"

},

"valConstraints": {

"$ref": "#/definitions/evr-ValidationConstraintsType"

},

"sigTime": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-VerificationTimeInfoType"

}

},

"sigDoc": {

"$ref": "#/definitions/evr-SignersDocumentType"

},

"sigInfo": {

"$ref": "#/definitions/evr-SignerInformationType"

},

"sigQualities": {

"$ref": "#/definitions/evr-SignatureQualityListType"

},

"sigValProcess": {

"$ref": "#/definitions/evr-SignatureValidationProcessType"

},

"repData": {

"$ref": "#/definitions/evr-ValidationReportDataType"

},

"repSignature": {

"$ref": "#/definitions/dss2-SignatureObjectType"

},

"valObjs": {

"$ref": "#/definitions/evr-ValidationObjectListType"

},

"result": {

"$ref": "#/definitions/dss2-VerifyManifestResultsType"

},

"signingTimeInfo": {

"$ref": "#/definitions/dss2-SigningTimeInfoType"

},

"verificationTimeInfo": {

"$ref": "#/definitions/dss2-VerificationTimeInfoType"

},

"procDetails": {

"$ref": "#/definitions/dss2-ProcessingDetailsType"

},

"signerIdentity": {

"$ref": "#/definitions/saml2rw-NameIDType"

},

"updSignature": {

"$ref": "#/definitions/dss2-UpdatedSignatureType"

},

"timestampedSignature": {

"$ref": "#/definitions/dss2-UpdatedSignatureType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ReportDetail component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DetailedSignatureReport | detailedSigReport | [] |
| IndividualTimeStampReport | individualTSReport | [] |
| IndividualCertificateReport | individualCertReport | [] |
| IndividualAttributeCertificateReport | individualAttCertReport | [] |
| IndividualCRLReport | individualCRLReport | [] |
| IndividualOCSPReport | individualOCSPReport | [] |
| EvidenceRecordReport | erReport | [] |
| ValidationConstraints | valConstraints | [] |
| SignatureValidationTimeInfo | sigTime | [] |
| SignerDocument | sigDoc | [] |
| SignerInformation | sigInfo | [] |
| SignatureQualityList | sigQualities | [] |
| SignatureValidationProcess | sigValProcess | [] |
| ValidationReportData | repData | [] |
| ValidationReportSignature | repSignature | [] |
| ValidationObjectList | valObjs | [] |
| VerifyManifestResults | result | [] |
| SigningTimeInfo | signingTimeInfo | [] |
| VerificationTimeInfo | verificationTimeInfo | [] |
| ProcessingDetails | procDetails | [] |
| SignerIdentity | signerIdentity | [] |
| UpdatedSignature | updSignature | [] |
| TimestampedSignature | timestampedSignature | [] |

[component ReportDetail JSON schema details]

### Component IndividualReport

#### Semantics

[component IndividualReport normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignedObjectIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignedObjectIdentifier. This component identifies the signature or validation data under consideration.
* The Result element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section Result. [sub component Result details]
* The optional Details element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ReportDetail. The Details component MAY contain a detailed report for the signature or validation data under consideration or any other signature-specific optional output.
* The optional ValidationBasedOnHash element MUST contain a boolean. Its default value is 'false'. [sub component ValidationBasedOnHash details]

Non-normative Comment:

[component IndividualReport non normative details]

#### XML Syntax

The XML type IndividualReportType SHALL implement the requirements defined in the IndividualReport component.

The IndividualReportType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="IndividualReportType">

<xs:sequence>

<xs:element name="SignedObjectIdentifier" type="vr:SignedObjectIdentifierType"/>

<xs:element name="Result" type="dsb:ResultType"/>

<xs:element maxOccurs="1" minOccurs="0" name="Details" type="vr:ReportDetailType"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="ValidationBasedOnHash" type="xs:boolean"/>

</xs:sequence>

</xs:complexType>

Each child element of IndividualReportType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component IndividualReport XML schema details]

#### JSON Syntax

The IndividualReportType JSON object SHALL implement in JSON syntax the requirements defined in the IndividualReport component.

The IndividualReportType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-IndividualReportType": {

"$xsd-full-type": "vr:IndividualReportType",

"type": "object",

"properties": {

"signedObjectIdentifier": {

"$ref": "#/definitions/vr-SignedObjectIdentifierType"

},

"result": {

"$ref": "#/definitions/dsb-ResultType"

},

"details": {

"$ref": "#/definitions/vr-ReportDetailType"

},

"basedOnHash": {

"type": "boolean",

"default": "false"

}

},

"required": ["signedObjectIdentifier", "result"]

}

Properties in the JSON schema above SHALL implement sub-component of IndividualReport component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignedObjectIdentifier | signedObjectIdentifier | [] |
| Result | result | [] |
| Details | details | [] |
| ValidationBasedOnHash | basedOnHash | [] |

[component IndividualReport JSON schema details]

### Component OptionalOutput

#### Semantics

[component OptionalOutput normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional VerificationReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section VerificationReport. [sub component VerificationReport details]
* The optional element MAY occur zero or more times containing a sub\_component . [sub component details]
* The optional SignedDetailedValidationReport element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section Any. [sub component SignedDetailedValidationReport details]

Non-normative Comment:

[component OptionalOutput non normative details]

#### XML Syntax

The XML type OptionalOutputType SHALL implement the requirements defined in the OptionalOutput component.

The OptionalOutputType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalOutputType">

<xs:sequence>

<xs:choice>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:VerificationReport"/>

<xs:element maxOccurs="unbounded" minOccurs="0" ref="etsival:AvailableSignatureValidationPolicies"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="SignedDetailedValidationReport" type="dsb:AnyType"/>

</xs:choice>

</xs:sequence>

</xs:complexType>

Each child element of OptionalOutputType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalOutput XML schema details]

#### JSON Syntax

The OptionalOutputType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalOutput component.

The OptionalOutputType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-OptionalOutputType": {

"$xsd-full-type": "vr:OptionalOutputType",

"type": "object",

"properties": {

"verificationReport": {

"$ref": "#/definitions/vr-VerificationReportType"

},

"availPolicies": {

"type": "array",

"items": {

"$ref": "#/definitions/evp-AvailableSignatureValidationPoliciesType"

}

},

"sigDetailedValReport": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalOutput component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| VerificationReport | verificationReport | [] |
| AvailableSignatureValidationPolicies | availPolicies | [] |
| SignedDetailedValidationReport | sigDetailedValReport | [] |

[component OptionalOutput JSON schema details]

### Component OptionalInput

#### Semantics

[component OptionalInput normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional ReturnVerificationReport element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ReturnVerificationReport. [sub component ReturnVerificationReport details]
* The optional EvidenceRecords element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section Any. [sub component EvidenceRecords details]
* The optional POE element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section Any. [sub component POE details]
* The optional AdditionalConstraint element MAY occur zero or more times containing a URI. [sub component AdditionalConstraint details]
* The optional VerificationPolicy element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section ObjectIdentifier. [sub component VerificationPolicy details]
* The optional ValidationProcess element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ValidationProcess. [sub component ValidationProcess details]
* The optional element MUST contain a sub\_component . [sub component details]

Non-normative Comment:

[component OptionalInput non normative details]

#### XML Syntax

The XML type OptionalInputType SHALL implement the requirements defined in the OptionalInput component.

The OptionalInputType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalInputType">

<xs:sequence>

<xs:choice>

<xs:element maxOccurs="1" minOccurs="0" ref="vr:ReturnVerificationReport"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="EvidenceRecords" type="dsb:AnyType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="POE" type="dsb:AnyType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AdditionalConstraint" type="xs:anyURI"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="VerificationPolicy" type="xades-rw:ObjectIdentifierType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ValidationProcess" type="vr:ValidationProcessType"/>

<xs:element maxOccurs="1" minOccurs="0" ref="etsival:SignVerificationReport"/>

</xs:choice>

</xs:sequence>

</xs:complexType>

Each child element of OptionalInputType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalInput XML schema details]

#### JSON Syntax

The OptionalInputType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalInput component.

The OptionalInputType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-OptionalInputType": {

"$xsd-full-type": "vr:OptionalInputType",

"type": "object",

"properties": {

"returnVerificationReport": {

"$ref": "#/definitions/vr-ReturnVerificationReportType"

},

"evidenceRecords": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"POE": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"addConstraint": {

"type": "array",

"items": {

"type": "string"

}

},

"verificationPolicy": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-ObjectIdentifierType"

}

},

"validationProcess": {

"type": "string"

},

"signReport": {

"type": "boolean"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalInput component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ReturnVerificationReport | returnVerificationReport | [] |
| EvidenceRecords | evidenceRecords | [] |
| POE | POE | [] |
| AdditionalConstraint | addConstraint | [] |
| VerificationPolicy | verificationPolicy | [] |
| ValidationProcess | validationProcess | [] |
| SignVerificationReport | signReport | [] |

[component OptionalInput JSON schema details]

### Component SignedObjectIdentifier

#### Semantics

The set of child elements of this component SHOULD be chosen to identify the signature or validation data in a given context in an unambiguous manner.

Below follows a list of the sub-components that MAY be present within this component:

* The optional DigestAlgAndValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section DigestAlgAndValue. This element contains, if present, the hash value of the signature or validation data under consideration, where the signed object itself (e.g. the <ds:Signature>-element in case of an XML-signature according to **[RFC3275]**, the SignedData-structure in case of a CMS-signature according to **[RFC3852]** or a time stamp according to **[RFC3161]**, the Certificate- or CertificateList-structure in case of an X.509-certificate or CRL according to **[RFC5280]** or the OCSPResponse structure in case of an OCSP-response according to **[RFC2560]** for example) serves as input for the hash-calculation. The structure of the DigestAlgAndValueType is defined in **[XAdES]**. This element SHOULD NOT be used if the unique identification can be guaranteed by other elements.
* The optional CanonicalizationMethod element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CanonicalizationMethod. This element indicates, if present, the canonicalization method to be used before hashing XML-formatted data. Please refer to **[RFC3275]** for details of this element. This element is only necessary if XML-based structures are subject to hashing.
* The optional SignedProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignedProperties. This element contains, if present, any number of signed properties, which may be useful to identify the signature under consideration. This MAY comprise information about the signatory and the signing time for example. In case of signatures according to **[RFC3275]** or **[RFC3852]** this element SHOULD be present.
* The optional SignatureValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignatureValue. This element specifies, if present, the binary signature value of the signature under consideration. This element SHOULD be present – particulary if the used signature algorithm is randomized and hence this element may serve as unique identifier.
* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain other elements, which (help to) identify a signature or related validation data in a unique manner.
* The optional WhichDocument element MUST contain one instance of a unique identifier reference. This element MAY specify the document which contains the signature under consideration. Note that this identifier is only unique with respect to a specific request message
* The optional XPath element MUST contain one instance of a string. This element MAY be used to point to a specific signature within an XML document.
* The optional Offset element MUST contain one instance of an integer. This element specifies the first byte of some signature and MAY be used to point to a specific signature within some binary document
* The optional FieldName element MUST contain one instance of a string. This element specifies the name of a signature field and MAY be used to point to a specific signature within some document format, in which there are field names such as PDF for example.

Non-normative Comment:

[component SignedObjectIdentifier non normative details]

#### XML Syntax

The XML type SignedObjectIdentifierType SHALL implement the requirements defined in the SignedObjectIdentifier component.

The SignedObjectIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignedObjectIdentifierType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="DigestAlgAndValue" type="xades-rw:DigestAlgAndValueType"/>

<element maxOccurs="1" minOccurs="0" name="CanonicalizationMethod" type="ds-rw:CanonicalizationMethodType"/>

<element maxOccurs="1" minOccurs="0" name="SignedProperties" type="vr:SignedPropertiesType"/>

<element maxOccurs="1" minOccurs="0" name="SignatureValue" type="ds-rw:SignatureValueType"/>

<element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</sequence>

<attribute name="WhichDocument" type="IDREF" use="optional"/>

<attribute name="XPath" type="string" use="optional"/>

<attribute name="Offset" type="integer" use="optional"/>

<attribute name="FieldName" type="string" use="optional"/>

</complexType>

Each child element of SignedObjectIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedObjectIdentifier XML schema details]

#### JSON Syntax

The SignedObjectIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the SignedObjectIdentifier component.

The SignedObjectIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignedObjectIdentifierType": {

"$xsd-full-type": "vr:SignedObjectIdentifierType",

"type": "object",

"properties": {

"xpath": {

"type": "string"

},

"digestAlgAndValue": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

},

"canonMethod": {

"$ref": "#/definitions/dsigrw-CanonicalizationMethodType"

},

"signedProps": {

"$ref": "#/definitions/vr-SignedPropertiesType"

},

"sigValue": {

"$ref": "#/definitions/dsigrw-SignatureValueType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

},

"whichDoc": {

"$ref": "#/definitions/dss2-DocumentBaseType"

},

"xPath": {

"type": "string"

},

"offset": {

"type": "integer"

},

"fieldName": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignedObjectIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestAlgAndValue | digestAlgAndValue | [] |
| CanonicalizationMethod | canonMethod | [] |
| SignedProperties | signedProps | [] |
| SignatureValue | sigValue | [] |
| Other | other | [] |
| WhichDocument | whichDoc | [] |
| XPath | xPath | [] |
| Offset | offset | [] |
| FieldName | fieldName | [] |

[component SignedObjectIdentifier JSON schema details]

### Component VerificationResult

#### Semantics

[component VerificationResult normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component Other details]

A set of sub-components is inherited from component Result and is not repeated here.

Non-normative Comment:

[component VerificationResult non normative details]

#### XML Syntax

The XML type VerificationResultType SHALL implement the requirements defined in the VerificationResult component.

The VerificationResultType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VerificationResultType">

<xs:complexContent>

<xs:extension base="dsb:ResultType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of VerificationResultType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerificationResult XML schema details]

#### JSON Syntax

The VerificationResultType JSON object SHALL implement in JSON syntax the requirements defined in the VerificationResult component.

The VerificationResultType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-VerificationResultType": {

"$xsd-full-type": "vr:VerificationResultType",

"type": "object",

"properties": {

"maj": {

"type": "string"

},

"min": {

"type": "string"

},

"msg": {

"$ref": "#/definitions/dsb-InternationalStringType"

},

"pRef": {

"type": "string"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of VerificationResult component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Other | other | [] |

[component VerificationResult JSON schema details]

### Component DetailedSignatureReport

#### Semantics

The DetailedSignatureReport component MAY appear in the Details component within the IndividualReport component.

Below follows a list of the sub-components that MAY be present within this component:

* The FormatOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates whether the format of the signature is ok or not.
* The optional Properties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Properties. This element contains information gathered during the verification of signed or unsigned properties.
* The optional VerifyManifestResults element MUST contain a sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerifyManifestResults. This element is present, if a manifest verification has been performed.
* The optional SignatureHasVisibleContent element MUST contain a boolean. This element is only present if the FieldName-attribute is present and indicates whether the signature under consideration has visual signature content.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. This element contains information about the mathematical validity of the digital signature under consideration.
* The CertificatePathValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificatePathValidity. This element contains the results of the certificate path validation.

Non-normative Comment:

[component DetailedSignatureReport non normative details]

#### XML Syntax

The XML type DetailedSignatureReportType SHALL implement the requirements defined in the DetailedSignatureReport component.

The DetailedSignatureReportType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="DetailedSignatureReportType">

<sequence>

<element name="FormatOK" type="vr:VerificationResultType"/>

<xs:element maxOccurs="1" minOccurs="0" name="Properties" type="vr:PropertiesType"/>

<element maxOccurs="1" minOccurs="0" name="VerifyManifestResults" type="dss2:VerifyManifestResultsType"/>

<element maxOccurs="1" minOccurs="0" name="SignatureHasVisibleContent" type="boolean"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificatePathValidity" type="vr:CertificatePathValidityType"/>

</sequence>

</complexType>

Each child element of DetailedSignatureReportType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DetailedSignatureReport XML schema details]

#### JSON Syntax

The DetailedSignatureReportType JSON object SHALL implement in JSON syntax the requirements defined in the DetailedSignatureReport component.

The DetailedSignatureReportType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-DetailedSignatureReportType": {

"$xsd-full-type": "vr:DetailedSignatureReportType",

"type": "object",

"properties": {

"formatOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"props": {

"$ref": "#/definitions/vr-PropertiesType"

},

"manifestResults": {

"$ref": "#/definitions/dss2-VerifyManifestResultsType"

},

"sigHasVisibleContent": {

"type": "boolean"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"certPathValidity": {

"$ref": "#/definitions/vr-CertificatePathValidityType"

}

},

"required": ["formatOK", "sigOK", "certPathValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of DetailedSignatureReport component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| FormatOK | formatOK | [] |
| Properties | props | [] |
| VerifyManifestResults | manifestResults | [] |
| SignatureHasVisibleContent | sigHasVisibleContent | [] |
| SignatureOK | sigOK | [] |
| CertificatePathValidity | certPathValidity | [] |

[component DetailedSignatureReport JSON schema details]

### Component Properties

#### Semantics

[component Properties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignedProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignedProperties. This element contains information gathered during the verification of signed properties.
* The optional UnsignedProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section UnsignedProperties. This element contains information gathered during the verification of unsigned properties.
* The optional Target element MUST contain one instance of a URI. [sub component Target details]
* The optional Id element MUST contain one instance of a unique identifier. contains, if present, an optional identifier for this component.

Non-normative Comment:

[component Properties non normative details]

#### XML Syntax

The XML type PropertiesType SHALL implement the requirements defined in the Properties component.

The PropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="PropertiesType">

<sequence>

<element minOccurs="0" name="SignedProperties" type="vr:SignedPropertiesType"/>

<element minOccurs="0" name="UnsignedProperties" type="vr:UnsignedPropertiesType"/>

</sequence>

<attribute name="Target" type="anyURI" use="optional"/>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of PropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Properties XML schema details]

#### JSON Syntax

The PropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the Properties component.

The PropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-PropertiesType": {

"$xsd-full-type": "vr:PropertiesType dss2:PropertiesType",

"type": "object",

"properties": {

"signedProps": {

"$ref": "#/definitions/vr-SignedPropertiesType"

},

"unsignedProps": {

"$ref": "#/definitions/vr-UnsignedPropertiesType"

},

"target": {

"type": "string"

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Properties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignedProperties | signedProps | [] |
| UnsignedProperties | unsignedProps | [] |
| Target | target | [] |
| Id | id | [] |

[component Properties JSON schema details]

### Component SignedProperties

#### Semantics

The SignedProperties component is aligned to the corresponding **[XAdES]** structure

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignedSignatureProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignedSignatureProperties. This element contains information gathered during the verification of signed properties related to the signature itself.
* The optional SignedDataObjectProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignedDataObjectProperties. This element contains information gathered during the verification of signed properties related to the signed data object.
* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element contains, if present, information about other signed properties.
* The optional Id element MUST contain one instance of a unique identifier. This element contains an optional identifier for this component.

Non-normative Comment:

[component SignedProperties non normative details]

#### XML Syntax

The XML type SignedPropertiesType SHALL implement the requirements defined in the SignedProperties component.

The SignedPropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignedPropertiesType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="SignedSignatureProperties" type="vr:SignedSignaturePropertiesType"/>

<element minOccurs="0" name="SignedDataObjectProperties" type="vr:SignedDataObjectPropertiesType"/>

<element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of SignedPropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedProperties XML schema details]

#### JSON Syntax

The SignedPropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the SignedProperties component.

The SignedPropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignedPropertiesType": {

"$xsd-full-type": "vr:SignedPropertiesType xades-rw:SignedPropertiesType",

"type": "object",

"properties": {

"signedSigProps": {

"$ref": "#/definitions/vr-SignedSignaturePropertiesType"

},

"signedDataObjProps": {

"$ref": "#/definitions/vr-SignedDataObjectPropertiesType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignedProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignedSignatureProperties | signedSigProps | [] |
| SignedDataObjectProperties | signedDataObjProps | [] |
| Other | other | [] |
| Id | id | [] |

[component SignedProperties JSON schema details]

### Component SignedSignatureProperties

#### Semantics

The SignedSignatureProperties component is aligned to **[RFC3275]**

Below follows a list of the sub-components that MAY be present within this component:

* The optional SigningTime element MUST contain a date/time value. This element contains, if present, the signing time (see Section 5.2.1 of
* The optional SigningCertificate element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CertIDList. This element contains, if present, a reference to the certificate upon which the signature is based (see Section 5.2.2 of **[XAdES]**).
* The optional SignaturePolicyIdentifier element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignaturePolicyIdentifier. This element references, if present, the policy under which the signature was produced (see Section 5.2.3 of **[XAdES]**).
* The optional SignatureProductionPlace element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureProductionPlace. This element contains, if present, information about the place where the signature was generated (see Section 5.2.7 of **[XAdES]**). This element SHOULD be used in case of a XAdES- or CAdES-based signature.
* The optional Location element MUST contain one instance of a string. This element contains, if present, information about the place where the signature was generated (see Section 5.2.7 of **[XAdES]**). This element SHOULD be used in case of a PDF-based signature.
* The optional SignerRole element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section SignerRole. This element contains, if present, information about the role of the signer (see Section 5.2.8 of **[XAdES]**).

Non-normative Comment:

[component SignedSignatureProperties non normative details]

#### XML Syntax

The XML type SignedSignaturePropertiesType SHALL implement the requirements defined in the SignedSignatureProperties component.

The SignedSignaturePropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignedSignaturePropertiesType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="SigningTime" type="dateTime"/>

<element maxOccurs="1" minOccurs="0" name="SigningCertificate" type="xades-rw:CertIDListType"/>

<element maxOccurs="1" minOccurs="0" name="SignaturePolicyIdentifier" type="xades-rw:SignaturePolicyIdentifierType"/>

<choice maxOccurs="1" minOccurs="0">

<element name="SignatureProductionPlace" type="xades-rw:SignatureProductionPlaceType"/>

<element name="Location" type="string"/>

</choice>

<element minOccurs="0" name="SignerRole" type="vr:SignerRoleType"/>

</sequence>

</complexType>

Each child element of SignedSignaturePropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedSignatureProperties XML schema details]

#### JSON Syntax

The SignedSignaturePropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the SignedSignatureProperties component.

The SignedSignaturePropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignedSignaturePropertiesType": {

"$xsd-full-type": "vr:SignedSignaturePropertiesType xades-rw:SignedSignaturePropertiesType",

"type": "object",

"properties": {

"sigTime": {

"type": "integer",

"format": "utc-millisec"

},

"sigCert": {

"$ref": "#/definitions/xadesrw-CertIDListType"

},

"sigPolicyId": {

"$ref": "#/definitions/xadesrw-SignaturePolicyIdentifierType"

},

"sigProdPlace": {

"$ref": "#/definitions/xadesrw-SignatureProductionPlaceType"

},

"loc": {

"type": "string"

},

"signerRole": {

"$ref": "#/definitions/vr-SignerRoleType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignedSignatureProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigningTime | sigTime | [] |
| SigningCertificate | sigCert | [] |
| SignaturePolicyIdentifier | sigPolicyId | [] |
| SignatureProductionPlace | sigProdPlace | [] |
| Location | loc | [] |
| SignerRole | signerRole | [] |

[component SignedSignatureProperties JSON schema details]

### Component SignerRole

#### Semantics

[component SignerRole normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional ClaimedRoles element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section ClaimedRolesList. This element contains information about the claimed roles of the signer. The information is directly extracted from the signature.
* The optional CertifiedRoles element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CertifiedRolesList. This element contains information gathered during the verification of attribute certificates.

Non-normative Comment:

[component SignerRole non normative details]

#### XML Syntax

The XML type SignerRoleType SHALL implement the requirements defined in the SignerRole component.

The SignerRoleType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignerRoleType">

<sequence>

<element minOccurs="0" name="ClaimedRoles" type="xades-rw:ClaimedRolesListType"/>

<element minOccurs="0" name="CertifiedRoles" type="vr:CertifiedRolesListType"/>

</sequence>

</complexType>

Each child element of SignerRoleType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignerRole XML schema details]

#### JSON Syntax

The SignerRoleType JSON object SHALL implement in JSON syntax the requirements defined in the SignerRole component.

The SignerRoleType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignerRoleType": {

"$xsd-full-type": "vr:SignerRoleType xades-rw:SignerRoleType",

"type": "object",

"properties": {

"claimedRoles": {

"$ref": "#/definitions/xadesrw-ClaimedRolesListType"

},

"certifiedRoles": {

"$ref": "#/definitions/vr-CertifiedRolesListType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignerRole component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ClaimedRoles | claimedRoles | [] |
| CertifiedRoles | certifiedRoles | [] |

[component SignerRole JSON schema details]

### Component CertifiedRolesList

#### Semantics

[component CertifiedRolesList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The AttributeCertificateValidity element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section AttributeCertificateValidity. It contains information about the content and validity of an attribute certificate according to **[RFC3281]**.

Non-normative Comment:

[component CertifiedRolesList non normative details]

#### XML Syntax

The XML type CertifiedRolesListType SHALL implement the requirements defined in the CertifiedRolesList component.

The CertifiedRolesListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertifiedRolesListType">

<sequence>

<element maxOccurs="unbounded" name="AttributeCertificateValidity" type="vr:AttributeCertificateValidityType"/>

</sequence>

</complexType>

Each child element of CertifiedRolesListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertifiedRolesList XML schema details]

#### JSON Syntax

The CertifiedRolesListType JSON object SHALL implement in JSON syntax the requirements defined in the CertifiedRolesList component.

The CertifiedRolesListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertifiedRolesListType": {

"$xsd-full-type": "vr:CertifiedRolesListType xades-rw:CertifiedRolesListType",

"type": "object",

"properties": {

"attCertValidity": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-AttributeCertificateValidityType"

}

}

},

"required": ["attCertValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of CertifiedRolesList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| AttributeCertificateValidity | attCertValidity | [] |

[component CertifiedRolesList JSON schema details]

### Component AttributeCertificateValidity

#### Semantics

[component AttributeCertificateValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The AttributeCertificateIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section AttrCertID. This element MAY refer to an X.509v3 attribute certificate according to **[RFC3281]**.
* The optional AttributeCertificateValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Base64Data. This element MAY contain the certificate in binary form (coded in ASN.1), if the report option IncludeCertificateValues is set to ‘true’.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. This element indicates whether the digital signature is mathematically valid or not.
* The CertificatePathValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificatePathValidity. This element contains the result of the validation of the certificate path of the certificate which has been used to sign the attribute certificate.

Non-normative Comment:

[component AttributeCertificateValidity non normative details]

#### XML Syntax

The XML type AttributeCertificateValidityType SHALL implement the requirements defined in the AttributeCertificateValidity component.

The AttributeCertificateValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="AttributeCertificateValidityType">

<sequence>

<element maxOccurs="1" minOccurs="1" name="AttributeCertificateIdentifier" type="vr:AttrCertIDType"/>

<xs:element maxOccurs="1" minOccurs="0" name="AttributeCertificateValue" type="dsb:Base64DataType"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificatePathValidity" type="vr:CertificatePathValidityType"/>

</sequence>

</complexType>

Each child element of AttributeCertificateValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AttributeCertificateValidity XML schema details]

#### JSON Syntax

The AttributeCertificateValidityType JSON object SHALL implement in JSON syntax the requirements defined in the AttributeCertificateValidity component.

The AttributeCertificateValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-AttributeCertificateValidityType": {

"$xsd-full-type": "vr:AttributeCertificateValidityType",

"type": "object",

"properties": {

"attCertId": {

"$ref": "#/definitions/vr-AttrCertIDType"

},

"attCertValue": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"certPathValidity": {

"$ref": "#/definitions/vr-CertificatePathValidityType"

}

},

"required": ["attCertId", "sigOK", "certPathValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of AttributeCertificateValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| AttributeCertificateIdentifier | attCertId | [] |
| AttributeCertificateValue | attCertValue | [] |
| SignatureOK | sigOK | [] |
| CertificatePathValidity | certPathValidity | [] |

[component AttributeCertificateValidity JSON schema details]

### Component AttrCertID

#### Semantics

[component AttrCertID normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Holder element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Entity. This element contains, if present, information about the holder of the certificate.
* The Issuer element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section Entity. This element contains information about the issuer of the attribute certificate.
* The SerialNumber element MUST contain one instance of a string. This element contains the serial number of the attribute certificate, which (together with the information provided in the Issuer element) uniquely identifies the attribute certificate.

Non-normative Comment:

[component AttrCertID non normative details]

#### XML Syntax

The XML type AttrCertIDType SHALL implement the requirements defined in the AttrCertID component.

The AttrCertIDType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="AttrCertIDType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="Holder" type="vr:EntityType"/>

<element name="Issuer" type="vr:EntityType"/>

<element name="SerialNumber" type="string"/>

</sequence>

</complexType>

Each child element of AttrCertIDType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AttrCertID XML schema details]

#### JSON Syntax

The AttrCertIDType JSON object SHALL implement in JSON syntax the requirements defined in the AttrCertID component.

The AttrCertIDType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-AttrCertIDType": {

"$xsd-full-type": "vr:AttrCertIDType",

"type": "object",

"properties": {

"holder": {

"$ref": "#/definitions/vr-EntityType"

},

"iss": {

"$ref": "#/definitions/vr-EntityType"

},

"serial": {

"type": "string"

}

},

"required": ["iss", "serial"]

}

Properties in the JSON schema above SHALL implement sub-component of AttrCertID component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Holder | holder | [] |
| Issuer | iss | [] |
| SerialNumber | serial | [] |

[component AttrCertID JSON schema details]

### Component Entity

#### Semantics

The Entity component is aligned to the structure of Holder and V2Form in **[RFC3281]**.

Below follows a list of the sub-components that MAY be present within this component:

* The optional BaseCertificateID element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section X509IssuerSerial. This element identifies, if present, the public-key certificate of the entity. The structure of the element is defined in **[RFC3275]**.
* The optional Name element MUST contain a string. This element contains, if present, the name of the entity.
* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain other information which is used to identify the entity.

Non-normative Comment:

[component Entity non normative details]

#### XML Syntax

The XML type EntityType SHALL implement the requirements defined in the Entity component.

The EntityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="EntityType">

<sequence>

<element maxOccurs="1" minOccurs="0" name="BaseCertificateID" type="ds-rw:X509IssuerSerialType"/>

<element maxOccurs="1" minOccurs="0" name="Name" type="string"/>

<element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</sequence>

</complexType>

Each child element of EntityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Entity XML schema details]

#### JSON Syntax

The EntityType JSON object SHALL implement in JSON syntax the requirements defined in the Entity component.

The EntityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-EntityType": {

"$xsd-full-type": "vr:EntityType",

"type": "object",

"properties": {

"baseCertID": {

"$ref": "#/definitions/dsigrw-X509IssuerSerialType"

},

"name": {

"type": "string"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Entity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| BaseCertificateID | baseCertID | [] |
| Name | name | [] |
| Other | other | [] |

[component Entity JSON schema details]

### Component Attribute

#### Semantics

[component Attribute normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Type element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. [sub component Type details]
* The optional Value element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section Any. [sub component Value details]

Non-normative Comment:

[component Attribute non normative details]

#### XML Syntax

The XML type AttributeType SHALL implement the requirements defined in the Attribute component.

The AttributeType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="AttributeType">

<sequence>

<element name="Type" type="vr:VerificationResultType"/>

<element maxOccurs="unbounded" minOccurs="0" name="Value" type="dsb:AnyType"/>

</sequence>

</complexType>

Each child element of AttributeType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Attribute XML schema details]

#### JSON Syntax

The AttributeType JSON object SHALL implement in JSON syntax the requirements defined in the Attribute component.

The AttributeType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-AttributeType": {

"$xsd-full-type": "vr:AttributeType saml-rw:AttributeType",

"type": "object",

"properties": {

"type": {

"$ref": "#/definitions/vr-VerificationResultType",

"format": "uri"

},

"value": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

}

},

"required": ["type"]

}

Properties in the JSON schema above SHALL implement sub-component of Attribute component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Type | type | [] |
| Value | value | [] |

[component Attribute JSON schema details]

### Component Extension

#### Semantics

[component Extension normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ExtnId element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section ObjectIdentifier. This element MUST contain the identifier of the extension as urn:oid: … in the Identifier element and MAY contain further information in the Description and DocumentationReferences elements. Please refer to **[XAdES]** for more information on the XAdES:ObjectIdentifierType.
* The Critical element MUST contain one instance of a boolean. It specifies whether the extension is critical or not.
* The optional ExtnValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element SHOULD contain the value of the extension.
* The ExtensionOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. It contains information about the validity of the specific extension within the given context of the certificate.

Non-normative Comment:

[component Extension non normative details]

#### XML Syntax

The XML type ExtensionType SHALL implement the requirements defined in the Extension component.

The ExtensionType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="ExtensionType">

<sequence>

<element name="ExtnId" type="xades-rw:ObjectIdentifierType"/>

<element name="Critical" type="boolean"/>

<element maxOccurs="1" minOccurs="0" name="ExtnValue" type="dsb:AnyType"/>

<element name="ExtensionOK" type="vr:VerificationResultType"/>

</sequence>

</complexType>

Each child element of ExtensionType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Extension XML schema details]

#### JSON Syntax

The component Extension is not used as JSON object directly.

[component Extension JSON schema details]

### Component SignedDataObjectProperties

#### Semantics

[component SignedDataObjectProperties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional DataObjectFormat element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section DataObjectFormat. It contains information about the format of the signed data object (see Section 5.2.5 of **[XAdES]**). This information is extracted directly from the signature.
* The optional CommitmentTypeIndication element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CommitmentTypeIndication. It contains, if present, an indication of the type of commitment implied by the signature (see Section 5.2.6 of **[XAdES]**). This element SHOULD be used in case of a XAdES- or CAdES-based signature.
* The optional Reason element MUST contain one instance of a string. This element contains, if present, a description of the reason of the signature generation. This element is only relevant in case of a PDF-based signature identified by a FieldName attribute (cf. Section 3.3).
* The optional AllDataObjectsTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. It contains the verification results for time stamps covering all data objects (see Section 5.2.6 of **[XAdES]**).
* The optional IndividualDataObjectsTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. It element contains the verification results for time stamps covering only certain data objects (see Section 5.2.10 of **[XAdES]**).
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the component.

Non-normative Comment:

[component SignedDataObjectProperties non normative details]

#### XML Syntax

The XML type SignedDataObjectPropertiesType SHALL implement the requirements defined in the SignedDataObjectProperties component.

The SignedDataObjectPropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignedDataObjectPropertiesType">

<sequence>

<element maxOccurs="unbounded" minOccurs="0" name="DataObjectFormat" type="xades-rw:DataObjectFormatType"/>

<choice maxOccurs="1" minOccurs="0">

<element name="CommitmentTypeIndication" type="xades-rw:CommitmentTypeIndicationType"/>

<element name="Reason" type="string"/>

</choice>

<element maxOccurs="unbounded" minOccurs="0" name="AllDataObjectsTimeStamp" type="vr:TimeStampValidityType"/>

<element maxOccurs="unbounded" minOccurs="0" name="IndividualDataObjectsTimeStamp" type="vr:TimeStampValidityType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of SignedDataObjectPropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedDataObjectProperties XML schema details]

#### JSON Syntax

The SignedDataObjectPropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the SignedDataObjectProperties component.

The SignedDataObjectPropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignedDataObjectPropertiesType": {

"$xsd-full-type": "vr:SignedDataObjectPropertiesType xades-rw:SignedDataObjectPropertiesType",

"type": "object",

"properties": {

"dataObjFormat": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-DataObjectFormatType"

}

},

"commitmentTypeIndication": {

"$ref": "#/definitions/xadesrw-CommitmentTypeIndicationType"

},

"reason": {

"type": "string"

},

"allDataObjectsTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"individualDataObjectsTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignedDataObjectProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DataObjectFormat | dataObjFormat | [] |
| CommitmentTypeIndication | commitmentTypeIndication | [] |
| Reason | reason | [] |
| AllDataObjectsTimeStamp | allDataObjectsTimeStamp | [] |
| IndividualDataObjectsTimeStamp | individualDataObjectsTimeStamp | [] |
| Id | id | [] |

[component SignedDataObjectProperties JSON schema details]

### Component TimeStampValidity

#### Semantics

[component TimeStampValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The FormatOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates whether the format of the time stamp is ok or not.
* The optional MessageHashAlgorithm element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section AlgorithmValidity. This element contains, if present, information about the message hash algorithm and its suitability.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. This element indicates, whether the digital signature is mathematically valid or not.
* The CertificatePathValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificatePathValidity. This element contains the result of the validity check of the certificate.
* The optional Id element MUST contain one instance of a unique identifier. This element contains an optional identifier of the component.

Non-normative Comment:

[component TimeStampValidity non normative details]

#### XML Syntax

The XML type TimeStampValidityType SHALL implement the requirements defined in the TimeStampValidity component.

The TimeStampValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="TimeStampValidityType">

<sequence>

<element name="FormatOK" type="vr:VerificationResultType"/>

<element maxOccurs="1" minOccurs="0" name="MessageHashAlgorithm" type="vr:AlgorithmValidityType"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificatePathValidity" type="vr:CertificatePathValidityType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of TimeStampValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component TimeStampValidity XML schema details]

#### JSON Syntax

The TimeStampValidityType JSON object SHALL implement in JSON syntax the requirements defined in the TimeStampValidity component.

The TimeStampValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-TimeStampValidityType": {

"$xsd-full-type": "vr:TimeStampValidityType",

"type": "object",

"properties": {

"formatOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"messageHashAlgo": {

"$ref": "#/definitions/vr-AlgorithmValidityType"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"certPathValidity": {

"$ref": "#/definitions/vr-CertificatePathValidityType"

},

"id": {

"type": "string"

}

},

"required": ["formatOK", "sigOK", "certPathValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of TimeStampValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| FormatOK | formatOK | [] |
| MessageHashAlgorithm | messageHashAlgo | [] |
| SignatureOK | sigOK | [] |
| CertificatePathValidity | certPathValidity | [] |
| Id | id | [] |

[component TimeStampValidity JSON schema details]

### Component SignatureValidity

#### Semantics

[component SignatureValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SigMathOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. It contains information about the mathematical validity of the digital signature under consideration.
* The optional SignatureAlgorithm element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section AlgorithmValidity. This element MAY contain information about the applied signature algorithm.

Non-normative Comment:

[component SignatureValidity non normative details]

#### XML Syntax

The XML type SignatureValidityType SHALL implement the requirements defined in the SignatureValidity component.

The SignatureValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignatureValidityType">

<sequence>

<element name="SigMathOK" type="vr:VerificationResultType"/>

<element maxOccurs="1" minOccurs="0" name="SignatureAlgorithm" type="vr:AlgorithmValidityType"/>

</sequence>

</complexType>

Each child element of SignatureValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignatureValidity XML schema details]

#### JSON Syntax

The SignatureValidityType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureValidity component.

The SignatureValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-SignatureValidityType": {

"$xsd-full-type": "vr:SignatureValidityType",

"type": "object",

"properties": {

"sigMathOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"sigAlgo": {

"$ref": "#/definitions/vr-AlgorithmValidityType"

}

},

"required": ["sigMathOK"]

}

Properties in the JSON schema above SHALL implement sub-component of SignatureValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigMathOK | sigMathOK | [] |
| SignatureAlgorithm | sigAlgo | [] |

[component SignatureValidity JSON schema details]

### Component AlgorithmValidity

#### Semantics

[component AlgorithmValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Algorithm element MUST contain one instance of a URI. This element contains the URI for the algorithm.
* The optional Parameters element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain further parameters for the cryptographic algorithm.
* The optional Suitability element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section VerificationResult. This element MAY contain the information about the suitability of the algorithm under consideration.

Non-normative Comment:

Note that it MAY depend on the policy of the specific signature and/or the policy under which the DSS server is operated, whether the suitability of the algorithms is verified and what kind of algorithms are considered appropriate under given circumstances and which are not.

#### XML Syntax

The XML type AlgorithmValidityType SHALL implement the requirements defined in the AlgorithmValidity component.

The AlgorithmValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="AlgorithmValidityType">

<sequence>

<element name="Algorithm" type="anyURI"/>

<element maxOccurs="1" minOccurs="0" name="Parameters" type="dsb:AnyType"/>

<element maxOccurs="1" minOccurs="0" name="Suitability" type="vr:VerificationResultType"/>

</sequence>

</complexType>

Each child element of AlgorithmValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AlgorithmValidity XML schema details]

#### JSON Syntax

The AlgorithmValidityType JSON object SHALL implement in JSON syntax the requirements defined in the AlgorithmValidity component.

The AlgorithmValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-AlgorithmValidityType": {

"$xsd-full-type": "vr:AlgorithmValidityType",

"type": "object",

"properties": {

"algo": {

"type": "string"

},

"params": {

"$ref": "#/definitions/dsb-AnyType"

},

"suitability": {

"$ref": "#/definitions/vr-VerificationResultType"

}

},

"required": ["algo"]

}

Properties in the JSON schema above SHALL implement sub-component of AlgorithmValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Algorithm | algo | [] |
| Parameters | params | [] |
| Suitability | suitability | [] |

[component AlgorithmValidity JSON schema details]

### Component CertificatePathValidity

#### Semantics

[component CertificatePathValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The PathValiditySummary element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. It contains a summary of the result of the certificate path validation.
* The CertificateIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section X509IssuerSerial. It contains a unique reference to the certificate whose path has been checked.
* The optional PathValidityDetail element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section CertificatePathValidityDetail. It contains detailed results of the certificate path validation, if the element ReportDetailLevel in the report options was set to urn:oasis:names:tc:dss:1.0: profiles:verificationreport:reportdetail:allDetails and the detailed validity information has not been included elsewhere in the verification report.

Non-normative Comment:

[component CertificatePathValidity non normative details]

#### XML Syntax

The XML type CertificatePathValidityType SHALL implement the requirements defined in the CertificatePathValidity component.

The CertificatePathValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertificatePathValidityType">

<sequence>

<element name="PathValiditySummary" type="vr:VerificationResultType"/>

<element name="CertificateIdentifier" type="ds-rw:X509IssuerSerialType"/>

<element maxOccurs="1" minOccurs="0" name="PathValidityDetail" type="vr:CertificatePathValidityDetailType"/>

</sequence>

</complexType>

Each child element of CertificatePathValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertificatePathValidity XML schema details]

#### JSON Syntax

The CertificatePathValidityType JSON object SHALL implement in JSON syntax the requirements defined in the CertificatePathValidity component.

The CertificatePathValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertificatePathValidityType": {

"$xsd-full-type": "vr:CertificatePathValidityType",

"type": "object",

"properties": {

"pathValiditySummary": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"certIdentifier": {

"$ref": "#/definitions/dsigrw-X509IssuerSerialType"

},

"pathValidityDetail": {

"$ref": "#/definitions/vr-CertificatePathValidityDetailType"

}

},

"required": ["pathValiditySummary", "certIdentifier"]

}

Properties in the JSON schema above SHALL implement sub-component of CertificatePathValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| PathValiditySummary | pathValiditySummary | [] |
| CertificateIdentifier | certIdentifier | [] |
| PathValidityDetail | pathValidityDetail | [] |

[component CertificatePathValidity JSON schema details]

### Component CertificatePathValidityDetail

#### Semantics

[component CertificatePathValidityDetail normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional CertificateValidity element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CertificateValidity. For every certificate in the certificate path there will be a CertificateValidity element, which provides information about the validity of the specific certificate.
* The optional TSLValidity element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain information about a Trust-service Status List and its validity.
* The TrustAnchor element MUST contain one instance of a URI. This element indicates how the trusted root certificate, which is used as trust anchor within the verification process, is stored. The following URIs are defined for this purpose: urn:oasis:names:tc:dss:1.0:profiles:verificationreport:trustanchor:SSCD – indicates that the trusted root certificate is stored within a secure signature creation device according to **[EC/1999/93]**. urn:oasis:names:tc:dss:1.0:profiles:verificationreport:trustanchor:otherCard – indicates that the trusted root certificate is stored within some other hardware token. urn:oasis:names:tc:dss:1.0:profiles:verificationreport:trustanchor:certDataBase – indicates that the trusted root certificate is stored within some certificate data base. urn:oasis:names:tc:dss:1.0:profiles:verificationreport:trustanchor:other – indicates that the trusted root certificate is stored using other means.

Non-normative Comment:

[component CertificatePathValidityDetail non normative details]

#### XML Syntax

The XML type CertificatePathValidityDetailType SHALL implement the requirements defined in the CertificatePathValidityDetail component.

The CertificatePathValidityDetailType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertificatePathValidityDetailType">

<sequence>

<element maxOccurs="unbounded" minOccurs="0" name="CertificateValidity" type="vr:CertificateValidityType"/>

<element maxOccurs="1" minOccurs="0" name="TSLValidity" type="dsb:AnyType"/>

<xs:element name="TrustAnchor" type="anyURI"/>

</sequence>

</complexType>

Each child element of CertificatePathValidityDetailType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertificatePathValidityDetail XML schema details]

#### JSON Syntax

The CertificatePathValidityDetailType JSON object SHALL implement in JSON syntax the requirements defined in the CertificatePathValidityDetail component.

The CertificatePathValidityDetailType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertificatePathValidityDetailType": {

"$xsd-full-type": "vr:CertificatePathValidityDetailType",

"type": "object",

"properties": {

"tslvalidity": {

"$ref": "#/definitions/dsb-AnyType"

},

"certValidity": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-CertificateValidityType"

}

},

"tslValidity": {

"$ref": "#/definitions/dsb-AnyType"

},

"trustAnchor": {

"type": "string"

}

},

"required": ["trustAnchor"]

}

Properties in the JSON schema above SHALL implement sub-component of CertificatePathValidityDetail component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CertificateValidity | certValidity | [] |
| TSLValidity | tslValidity | [] |
| TrustAnchor | trustAnchor | [] |

[component CertificatePathValidityDetail JSON schema details]

### Component CertificateValidity

#### Semantics

[component CertificateValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CertificateIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section X509IssuerSerial. This element identifies the certificate under consideration
* The Subject element MUST contain one instance of a string. This element contains the subject of the certificate, where the string representation of distinguished names defined in **[RFC4514]** MUST be used and hence an example of a <Subject>-element may be CN=John Doe,O=Foo Inc.,OU=Sales etc.
* The ChainingOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. If present, this element indicates whether the chaining to a previous certificate in the certificate path is ok or not. If the certificate under consideration is the first certificate in the certificate path, this element SHOULD be omitted.
* The ValidityPeriodOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates whether the reference point in time is within the validity period of the certificate.
* The ExtensionsOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates, whether the certificate extensions are correct.
* The optional CertificateValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Base64Data. It contains the certificate in binary form (coded in ASN.1), if the report option IncludeCertificateValues is set to ‘true’ and if the certificate is not already included in the verification report.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. This element indicates whether the digital signature of the certificate is mathematically correct.
* The CertificateStatus element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificateStatus. It contains information about the result of the certificate revocation check.

Non-normative Comment:

[component CertificateValidity non normative details]

#### XML Syntax

The XML type CertificateValidityType SHALL implement the requirements defined in the CertificateValidity component.

The CertificateValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertificateValidityType">

<sequence>

<element name="CertificateIdentifier" type="ds-rw:X509IssuerSerialType"/>

<element name="Subject" type="string"/>

<element name="ChainingOK" type="vr:VerificationResultType"/>

<element name="ValidityPeriodOK" type="vr:VerificationResultType"/>

<element name="ExtensionsOK" type="vr:VerificationResultType"/>

<xs:element maxOccurs="1" minOccurs="0" name="CertificateValue" type="dsb:Base64DataType"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificateStatus" type="vr:CertificateStatusType"/>

</sequence>

</complexType>

Each child element of CertificateValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertificateValidity XML schema details]

#### JSON Syntax

The CertificateValidityType JSON object SHALL implement in JSON syntax the requirements defined in the CertificateValidity component.

The CertificateValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertificateValidityType": {

"$xsd-full-type": "vr:CertificateValidityType",

"type": "object",

"properties": {

"certIdentifier": {

"$ref": "#/definitions/dsigrw-X509IssuerSerialType"

},

"subject": {

"type": "string"

},

"chainingOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"validityPeriodOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"extensionsOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"value": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"status": {

"$ref": "#/definitions/vr-CertificateStatusType"

}

},

"required": ["certIdentifier", "subject", "chainingOK", "validityPeriodOK", "extensionsOK", "sigOK", "status"]

}

Properties in the JSON schema above SHALL implement sub-component of CertificateValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CertificateIdentifier | certIdentifier | [] |
| Subject | subject | [] |
| ChainingOK | chainingOK | [] |
| ValidityPeriodOK | validityPeriodOK | [] |
| ExtensionsOK | extensionsOK | [] |
| CertificateValue | value | [] |
| SignatureOK | sigOK | [] |
| CertificateStatus | status | [] |

[component CertificateValidity JSON schema details]

### Component CertificateStatus

#### Semantics

[component CertificateStatus normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CertStatusOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. It contain the status of the certificate.
* The optional RevocationInfo element MUST contain sub-components. [sub component RevocationInfo details]
* The RevocationDate element MUST contain one instance of a date/time value. It contains the date and time of revocation.
* The RevocationReason element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. It contains the reason for revocation. Following the definition of CRLReason in **[RFC5280]** there are the following URIs to specify the revocation reason: urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:unspecified urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:keyCompromise urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:cACompromise urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:affiliationChanged urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:superseded urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:cessationOfOperation urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:certificateHold urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:removeFromCRL urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:privilegeWithdrawn urn:oasis:names:tc:dss:1.0:profiles:verificationreport:revocationreason:aACompromise
* The optional RevocationEvidence element MUST contain sub-components. [sub component RevocationEvidence details]
* The CRLValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CRLValidity. It contains information about the used CRL and its validity.
* The CRLReference element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CRLIdentifier. It contains a reference to the CRL in case it is already included elsewhere in the present verification report.
* The OCSPValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section OCSPValidity. It contains information about the used OCSP response and its validity.
* The OCSPReference element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section OCSPIdentifier. It contains a reference to the used OCSP response, if it is already included elsewhere in the present verification report.
* The Other element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section Any. This element MAY contain information about alternative sources of revocation information.

Non-normative Comment:

[component CertificateStatus non normative details]

#### XML Syntax

The XML type CertificateStatusType SHALL implement the requirements defined in the CertificateStatus component.

The CertificateStatusType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertificateStatusType">

<sequence>

<element name="CertStatusOK" type="vr:VerificationResultType"/>

<element maxOccurs="1" minOccurs="0" name="RevocationInfo">

<complexType>

<sequence>

<element name="RevocationDate" type="dateTime"/>

<element name="RevocationReason" type="vr:VerificationResultType"/>

</sequence>

</complexType>

</element>

<element maxOccurs="1" minOccurs="0" name="RevocationEvidence">

<complexType>

<choice>

<element name="CRLValidity" type="vr:CRLValidityType"/>

<element name="CRLReference" type="xades-rw:CRLIdentifierType"/>

<element name="OCSPValidity" type="vr:OCSPValidityType"/>

<element name="OCSPReference" type="xades-rw:OCSPIdentifierType"/>

<element name="Other" type="dsb:AnyType"/>

</choice>

</complexType>

</element>

</sequence>

</complexType>

Each child element of CertificateStatusType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertificateStatus XML schema details]

#### JSON Syntax

The CertificateStatusType JSON object SHALL implement in JSON syntax the requirements defined in the CertificateStatus component.

The CertificateStatusType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertificateStatusType": {

"$xsd-full-type": "vr:CertificateStatusType",

"type": "object",

"properties": {

"certOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"revInfo": {

"$ref": "#/definitions/vr-CertificateStatusType:RevocationInfo"

},

"revEvidence": {

"$ref": "#/definitions/vr-CertificateStatusType:RevocationEvidence"

}

},

"required": ["certOK"]

}

"vr-CertificateStatusType:RevocationInfo": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"revDate": {

"type": "integer",

"format": "utc-millisec"

},

"reason": {

"$ref": "#/definitions/vr-VerificationResultType"

}

},

"required": ["revDate", "reason"]

}

"vr-CertificateStatusType:RevocationEvidence": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"crlvalidity": {

"$ref": "#/definitions/vr-CRLValidityType"

},

"ocspreference": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"ocspvalidity": {

"$ref": "#/definitions/vr-OCSPValidityType"

},

"crlreference": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

},

"crlValidity": {

"$ref": "#/definitions/vr-CRLValidityType"

},

"crlRef": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

},

"ocspValidity": {

"$ref": "#/definitions/vr-OCSPValidityType"

},

"ocspRef": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of CertificateStatus component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CertStatusOK | certOK | [] |
| RevocationInfo | revInfo | [] |
| RevocationDate | revDate | [] |
| RevocationReason | reason | [] |
| RevocationEvidence | revEvidence | [] |
| CRLValidity | crlValidity | [] |
| CRLReference | crlRef | [] |
| OCSPValidity | ocspValidity | [] |
| OCSPReference | ocspRef | [] |
| Other | other | [] |

[component CertificateStatus JSON schema details]

### Component CRLValidity

#### Semantics

[component CRLValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CRLIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CRLIdentifier. This element refers to an X.509v2 CRL according to **[RFC5280]**.
* The optional CRLValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Base64Data. It contains the CRL (encoded in ASN.1) if the report element IncludeRevocationValues is set to ‘true’.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. This element indicates, whether the digital signature of the CRL is mathematically correct or not.
* The CertificatePathValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificatePathValidity. This element contains the result of the validation of the certificate path of the certificate which has been used to sign the CRL.
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the element.

Non-normative Comment:

[component CRLValidity non normative details]

#### XML Syntax

The XML type CRLValidityType SHALL implement the requirements defined in the CRLValidity component.

The CRLValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CRLValidityType">

<sequence>

<element maxOccurs="1" minOccurs="1" name="CRLIdentifier" type="xades-rw:CRLIdentifierType"/>

<xs:element maxOccurs="1" minOccurs="0" name="CRLValue" type="dsb:Base64DataType"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificatePathValidity" type="vr:CertificatePathValidityType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of CRLValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CRLValidity XML schema details]

#### JSON Syntax

The CRLValidityType JSON object SHALL implement in JSON syntax the requirements defined in the CRLValidity component.

The CRLValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CRLValidityType": {

"$xsd-full-type": "vr:CRLValidityType",

"type": "object",

"properties": {

"crlvalue": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"crlidentifier": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

},

"crlId": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

},

"crlValue": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"certPathValidity": {

"$ref": "#/definitions/vr-CertificatePathValidityType"

},

"id": {

"type": "string"

}

},

"required": ["crlId", "sigOK", "certPathValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of CRLValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CRLIdentifier | crlId | [] |
| CRLValue | crlValue | [] |
| SignatureOK | sigOK | [] |
| CertificatePathValidity | certPathValidity | [] |
| Id | id | [] |

[component CRLValidity JSON schema details]

### Component OCSPValidity

#### Semantics

[component OCSPValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The OCSPIdentifier element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section OCSPIdentifier. It refers to an OCSP response according to **[RFC2560]**.
* The optional OCSPValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Base64Data. This element contains the OCSP response (encoded in ASN.1) if the report element IncludeRevocationValues has been set to ‘true’.
* The SignatureOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section SignatureValidity. It indicates whether the digital signature of the OCSP-response is mathematically correct or not.
* The CertificatePathValidity element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section CertificatePathValidity. It contains the result of the validation of the certificate path of the certificate which has been used to sign the OCSP-response.
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the element.

Non-normative Comment:

[component OCSPValidity non normative details]

#### XML Syntax

The XML type OCSPValidityType SHALL implement the requirements defined in the OCSPValidity component.

The OCSPValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="OCSPValidityType">

<sequence>

<element name="OCSPIdentifier" type="xades-rw:OCSPIdentifierType"/>

<xs:element maxOccurs="1" minOccurs="0" name="OCSPValue" type="dsb:Base64DataType"/>

<element name="SignatureOK" type="vr:SignatureValidityType"/>

<element name="CertificatePathValidity" type="vr:CertificatePathValidityType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of OCSPValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OCSPValidity XML schema details]

#### JSON Syntax

The OCSPValidityType JSON object SHALL implement in JSON syntax the requirements defined in the OCSPValidity component.

The OCSPValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-OCSPValidityType": {

"$xsd-full-type": "vr:OCSPValidityType",

"type": "object",

"properties": {

"ocspidentifier": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"ocspvalue": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"ocspId": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"ocspValue": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"sigOK": {

"$ref": "#/definitions/vr-SignatureValidityType"

},

"certPathValidity": {

"$ref": "#/definitions/vr-CertificatePathValidityType"

},

"id": {

"type": "string"

}

},

"required": ["ocspId", "sigOK", "certPathValidity"]

}

Properties in the JSON schema above SHALL implement sub-component of OCSPValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OCSPIdentifier | ocspId | [] |
| OCSPValue | ocspValue | [] |
| SignatureOK | sigOK | [] |
| CertificatePathValidity | certPathValidity | [] |
| Id | id | [] |

[component OCSPValidity JSON schema details]

### Component UnsignedProperties

#### Semantics

[component UnsignedProperties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional UnsignedSignatureProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section UnsignedSignatureProperties. This element contains information gathered during the verification of the unsigned properties related to the signature itself.
* The optional UnsignedDataObjectProperties element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section UnsignedDataObjectProperties. This element contains unsigned properties referring to the signed data objects. These properties are directly extracted from the signature.
* The optional Other element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain information about other unsigned properties.
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the element.

Non-normative Comment:

[component UnsignedProperties non normative details]

#### XML Syntax

The XML type UnsignedPropertiesType SHALL implement the requirements defined in the UnsignedProperties component.

The UnsignedPropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="UnsignedPropertiesType">

<sequence>

<element minOccurs="0" name="UnsignedSignatureProperties" type="vr:UnsignedSignaturePropertiesType"/>

<element maxOccurs="1" minOccurs="0" name="UnsignedDataObjectProperties" type="xades-rw:UnsignedDataObjectPropertiesType"/>

<element maxOccurs="1" minOccurs="0" name="Other" type="dsb:AnyType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of UnsignedPropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UnsignedProperties XML schema details]

#### JSON Syntax

The UnsignedPropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the UnsignedProperties component.

The UnsignedPropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-UnsignedPropertiesType": {

"$xsd-full-type": "vr:UnsignedPropertiesType xades-rw:UnsignedPropertiesType",

"type": "object",

"properties": {

"unsignedSigProperties": {

"$ref": "#/definitions/vr-UnsignedSignaturePropertiesType"

},

"unsignedDataObjectProperties": {

"$ref": "#/definitions/xadesrw-UnsignedDataObjectPropertiesType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of UnsignedProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| UnsignedSignatureProperties | unsignedSigProperties | [] |
| UnsignedDataObjectProperties | unsignedDataObjectProperties | [] |
| Other | other | [] |
| Id | id | [] |

[component UnsignedProperties JSON schema details]

### Component UnsignedSignatureProperties

#### Semantics

[component UnsignedSignatureProperties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional CounterSignature element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section SignatureValidity. This element contains the results of the verification of a counter signature (see Section 7.2.4 of **[XAdES]**).
* The optional SignatureTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. This element contains verification results of a time stamp of the signature (see Section 7.3 of **[XAdES]**).
* The optional CompleteCertificateRefs element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CompleteCertificateRefs. This element contains references to the certificates used during verification of the signature (see Section 7.4.1 of **[XAdES]**).
* The optional CompleteRevocationRefs element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CompleteRevocationRefs. It contains references to the revocation data used for the verification of the signature (see Section 7.4.2 of **[XAdES]**).
* The optional AttributeCertificateRefs element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CompleteCertificateRefs. This element contains the references to the full set of attribute authorities certificates that have been used to validate the attribute certificate (see section 7.4.3 of **[XAdES]**). This information is extracted directly from the signature.
* The optional AttributeRevocationRefs element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CompleteRevocationRefs. It contains the references to the full set of revocation data that have been used in the validation of the attribute certificate(s) present in the signature (see section 7.4.4 of **[XAdES]**).
* The optional SigAndRefsTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. It contains verification results for a time stamp referring to the signature and references on certificates and revocation data (see section 7.5.1 of **[XAdES]**).
* The optional RefsOnlyTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. It contains verification results for a time stamp referring only to references on certificates and revocation data (see section 7.5.2 of **[XAdES]**).
* The optional CertificateValues element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CertificateValues. It contains verification results for the certificates, which were used in the verification of the signature (see section 7.6.1 of **[XAdES]**).
* The optional RevocationValues element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section RevocationValues. It contains verification results of the revocation data used in the verification of the signature (see section 7.6.2 of **[XAdES]**).
* The optional AttrAuthoritiesCertValues element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CertificateValues. It contains verification results of the certificates of Attribute Authorities that have been used to validate the attribute certificates, which are contained in the signature (see section 7.6.3 of **[XAdES]**).
* The optional AttributeRevocationValues element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section RevocationValues. It contains verification results of the revocation data that have been used to validate the attribute certificate when present in the signature (see section 7.6.4 of **[XAdES]**).
* The optional ArchiveTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section TimeStampValidity. It contains verification results for a time stamp covering the complete signature including all attributes (see section 7.7 of **[XAdES]**).
* The optional Id element MUST contain one instance of a unique identifier. [sub component Id details]

Non-normative Comment:

[component UnsignedSignatureProperties non normative details]

#### XML Syntax

The XML type UnsignedSignaturePropertiesType SHALL implement the requirements defined in the UnsignedSignatureProperties component.

The UnsignedSignaturePropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="UnsignedSignaturePropertiesType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="CounterSignature" type="vr:SignatureValidityType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="SignatureTimeStamp" type="vr:TimeStampValidityType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" ref="xades-rw:CompleteCertificateRefs"/>

<xs:element maxOccurs="unbounded" minOccurs="0" ref="xades-rw:CompleteRevocationRefs"/>

<xs:element maxOccurs="unbounded" minOccurs="0" ref="xades-rw:AttributeCertificateRefs"/>

<xs:element maxOccurs="unbounded" minOccurs="0" ref="xades-rw:AttributeRevocationRefs"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="SigAndRefsTimeStamp" type="vr:TimeStampValidityType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="RefsOnlyTimeStamp" type="vr:TimeStampValidityType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="CertificateValues" type="vr:CertificateValuesType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="RevocationValues" type="vr:RevocationValuesType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AttrAuthoritiesCertValues" type="vr:CertificateValuesType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AttributeRevocationValues" type="vr:RevocationValuesType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ArchiveTimeStamp" type="vr:TimeStampValidityType"/>

</xs:sequence>

<xs:attribute name="Id" type="ID" use="optional"/>

</xs:complexType>

Each child element of UnsignedSignaturePropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UnsignedSignatureProperties XML schema details]

#### JSON Syntax

The UnsignedSignaturePropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the UnsignedSignatureProperties component.

The UnsignedSignaturePropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-UnsignedSignaturePropertiesType": {

"$xsd-full-type": "vr:UnsignedSignaturePropertiesType xades-rw:UnsignedSignaturePropertiesType",

"type": "object",

"properties": {

"counterSig": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-SignatureValidityType"

}

},

"sigTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"completeCertRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CompleteCertificateRefsType"

}

},

"completeRevRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CompleteRevocationRefsType"

}

},

"attCertRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CompleteCertificateRefsType"

}

},

"attRevRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CompleteRevocationRefsType"

}

},

"sigAndRefsTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"refsOnlyTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"certValues": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-CertificateValuesType"

}

},

"revValues": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-RevocationValuesType"

}

},

"attrAuthoritiesCertValues": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-CertificateValuesType"

}

},

"attRevValues": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-RevocationValuesType"

}

},

"archiveTimeStamp": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-TimeStampValidityType"

}

},

"id": {

"type": "string"

}

},

"minProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of UnsignedSignatureProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CounterSignature | counterSig | [] |
| SignatureTimeStamp | sigTimeStamp | [] |
| CompleteCertificateRefs | completeCertRefs | [] |
| CompleteRevocationRefs | completeRevRefs | [] |
| AttributeCertificateRefs | attCertRefs | [] |
| AttributeRevocationRefs | attRevRefs | [] |
| SigAndRefsTimeStamp | sigAndRefsTimeStamp | [] |
| RefsOnlyTimeStamp | refsOnlyTimeStamp | [] |
| CertificateValues | certValues | [] |
| RevocationValues | revValues | [] |
| AttrAuthoritiesCertValues | attrAuthoritiesCertValues | [] |
| AttributeRevocationValues | attRevValues | [] |
| ArchiveTimeStamp | archiveTimeStamp | [] |
| Id | id | [] |

[component UnsignedSignatureProperties JSON schema details]

### Component RevocationValues

#### Semantics

[component RevocationValues normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional CRLValues element MUST contain sub-components. It contains the verification results for all CRLs included in a signature.
* The VerifiedCRL element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section CRLValidity. [sub component VerifiedCRL details]
* The optional OCSPValues element MUST contain sub-components. It contains the verification results for all OCSP responses included in a signature.
* The VerifiedOCSPResponse element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section OCSPValidity. [sub component VerifiedOCSPResponse details]
* The optional OtherValues element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. This element MAY contain verification results for other revocation data included in the signature. If other revocation data with unknown format is included in the signature, a warning (error urn:oasis:names:tc:dss:1.0:resultminor:improperRevocationInformation) SHOULD be returned.
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the element.

Non-normative Comment:

[component RevocationValues non normative details]

#### XML Syntax

The XML type RevocationValuesType SHALL implement the requirements defined in the RevocationValues component.

The RevocationValuesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="RevocationValuesType">

<sequence>

<element minOccurs="0" name="CRLValues">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="1">

<element name="VerifiedCRL" type="vr:CRLValidityType"/>

</sequence>

</complexType>

</element>

<element minOccurs="0" name="OCSPValues">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="1">

<element name="VerifiedOCSPResponse" type="vr:OCSPValidityType"/>

</sequence>

</complexType>

</element>

<element minOccurs="0" name="OtherValues" type="dsb:AnyType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of RevocationValuesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component RevocationValues XML schema details]

#### JSON Syntax

The RevocationValuesType JSON object SHALL implement in JSON syntax the requirements defined in the RevocationValues component.

The RevocationValuesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-RevocationValuesType": {

"$xsd-full-type": "vr:RevocationValuesType xades-rw:RevocationValuesType",

"type": "object",

"properties": {

"crlvalues": {

"$ref": "#/definitions/vr-RevocationValuesType:CRLValues"

},

"ocspvalues": {

"$ref": "#/definitions/vr-RevocationValuesType:OCSPValues"

},

"crlValues": {

"$ref": "#/definitions/vr-RevocationValuesType:CRLValues"

},

"ocspValues": {

"$ref": "#/definitions/vr-RevocationValuesType:OCSPValues"

},

"otherValues": {

"$ref": "#/definitions/dsb-AnyType"

},

"id": {

"type": "string"

}

}

}

"vr-RevocationValuesType:CRLValues": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"verifiedCRL": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-CRLValidityType"

}

}

}

}

"vr-RevocationValuesType:OCSPValues": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"verifiedOCSPResponse": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-OCSPValidityType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of RevocationValues component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CRLValues | crlValues | [] |
| VerifiedCRL | verifiedCRL | [] |
| OCSPValues | ocspValues | [] |
| VerifiedOCSPResponse | verifiedOCSPResponse | [] |
| OtherValues | otherValues | [] |
| Id | id | [] |

[component RevocationValues JSON schema details]

### Component CertificateValues

#### Semantics

[component CertificateValues normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The EncapsulatedX509Certificate element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section CertificateValidity. It contains verification results for an X.509 certificate included in the signature.
* The OtherCertificate element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section Any. This element contains verification results for other certificates included in the signature. If a certificate with unknown format is included in the signature, a warning (error code urn:oasis:names:tc:dss:1.0:resultminor:certificateFormatNotCorrectWarning) SHOULD be returned.
* The optional Id element MUST contain one instance of a unique identifier. contains an optional identifier for the element.

Non-normative Comment:

[component CertificateValues non normative details]

#### XML Syntax

The XML type CertificateValuesType SHALL implement the requirements defined in the CertificateValues component.

The CertificateValuesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="CertificateValuesType">

<xs:sequence maxOccurs="unbounded" minOccurs="0">

<xs:element name="EncapsulatedX509Certificate" type="vr:CertificateValidityType"/>

<xs:element name="OtherCertificate" type="dsb:AnyType"/>

</xs:sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of CertificateValuesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertificateValues XML schema details]

#### JSON Syntax

The CertificateValuesType JSON object SHALL implement in JSON syntax the requirements defined in the CertificateValues component.

The CertificateValuesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-CertificateValuesType": {

"$xsd-full-type": "vr:CertificateValuesType xades-rw:CertificateValuesType",

"type": "object",

"properties": {

"encapsulatedX509CertificateAndOtherCertificate": {

"type": "array"

},

"id": {

"type": "string"

}

},

"minProperties": 0

}

Properties in the JSON schema above SHALL implement sub-component of CertificateValues component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| EncapsulatedX509Certificate | encX509Cert | [] |
| OtherCertificate | otherCert | [] |
| Id | id | [] |

[component CertificateValues JSON schema details]

### Component EvidenceRecordValidity

#### Semantics

[component EvidenceRecordValidity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The FormatOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates whether the format of the evidence record according to **[RFC4998]** is ok or not.
* The optional Version element MUST contain an integer. This element contains, if present, the version of the Evidence Record Syntax.
* The optional DigestAlgorithm element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section AlgorithmValidity. This element appears for each hash algorithm used to produce the evidence record and contains information about the hash algorithm and possibly its suitability.
* The optional CryptoInfos element MUST contain sub-components. This element MAY contain further data useful in the validation of the ArchiveTimeStampSequence element. As explained in **[RFC4998]** this MAY include possible Trust Anchors, certificates, revocation information, or the information concerning the suitability of cryptographic algorithms.
* The Attribute element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section Attribute. [sub component Attribute details]
* The optional EncryptionInfo element MUST contain sub-components. This element MAY contain the necessary information to support encrypted content (cf. **[RFC4998]**, Section 6.1).
* The EncryptionInfoType element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section AlgorithmValidity. [sub component EncryptionInfoType details]
* The optional EncryptionInfoValue element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component EncryptionInfoValue details]
* The ArchiveTimeStampSequence element MUST contain one instance of sub-components. This element is required contain a sequence of ArchiveTimeStampChain elements (cf. **[RFC4998]**, Section 5), which in turn MAY contain a sequence of ArchiveTimeStamp elements.
* The ArchiveTimeStampChain element MAY occur zero or more times containing sub-components. [sub component ArchiveTimeStampChain details]
* The ArchiveTimeStamp element MAY occur zero or more times containing a sub-component. If present each instance MUST satisfy the requirements specified in section ArchiveTimeStampValidity. [sub component ArchiveTimeStamp details]
* The optional Id element MUST contain one instance of a unique identifier. It contains an optional identifier for the component.

Non-normative Comment:

[component EvidenceRecordValidity non normative details]

#### XML Syntax

The XML type EvidenceRecordValidityType SHALL implement the requirements defined in the EvidenceRecordValidity component.

The EvidenceRecordValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="EvidenceRecordValidityType">

<sequence>

<element name="FormatOK" type="vr:VerificationResultType"/>

<element maxOccurs="1" minOccurs="0" name="Version" type="integer"/>

<element maxOccurs="unbounded" minOccurs="0" name="DigestAlgorithm" type="vr:AlgorithmValidityType"/>

<element maxOccurs="1" minOccurs="0" name="CryptoInfos">

<complexType>

<sequence>

<element maxOccurs="unbounded" minOccurs="1" name="Attribute" type="vr:AttributeType"/>

</sequence>

</complexType>

</element>

<element maxOccurs="1" minOccurs="0" name="EncryptionInfo">

<complexType>

<sequence>

<element name="EncryptionInfoType" type="vr:AlgorithmValidityType"/>

<element minOccurs="0" name="EncryptionInfoValue" type="dsb:AnyType"/>

</sequence>

</complexType>

</element>

<element maxOccurs="1" minOccurs="1" name="ArchiveTimeStampSequence">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="0">

<element name="ArchiveTimeStampChain">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="0">

<element name="ArchiveTimeStamp" type="vr:ArchiveTimeStampValidityType"/>

</sequence>

</complexType>

</element>

</sequence>

</complexType>

</element>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of EvidenceRecordValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component EvidenceRecordValidity XML schema details]

#### JSON Syntax

The EvidenceRecordValidityType JSON object SHALL implement in JSON syntax the requirements defined in the EvidenceRecordValidity component.

The EvidenceRecordValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-EvidenceRecordValidityType": {

"$xsd-full-type": "vr:EvidenceRecordValidityType",

"type": "object",

"properties": {

"formatOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"version": {

"type": "integer"

},

"digestAlgo": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-AlgorithmValidityType"

}

},

"cryptoInfos": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType:CryptoInfos"

},

"encryptionInfos": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType:EncryptionInfo"

},

"archiveTimeStampSeq": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType:ArchiveTimeStampSequence"

},

"id": {

"type": "string"

}

},

"required": ["formatOK", "archiveTimeStampSeq"]

}

"vr-EvidenceRecordValidityType:CryptoInfos": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"attrs": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-AttributeType"

}

}

},

"required": ["attrs"]

}

"vr-EvidenceRecordValidityType:EncryptionInfo": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"type": {

"$ref": "#/definitions/vr-AlgorithmValidityType",

"format": "uri"

},

"value": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"required": ["type"]

}

"vr-EvidenceRecordValidityType:ArchiveTimeStampSequence": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"archiveTimeStampChain": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType:ArchiveTimeStampSequence:ArchiveTimeStampChain"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of EvidenceRecordValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| FormatOK | formatOK | [] |
| Version | version | [] |
| DigestAlgorithm | digestAlgo | [] |
| CryptoInfos | cryptoInfos | [] |
| Attribute | attrs | [] |
| EncryptionInfo | encryptionInfos | [] |
| EncryptionInfoType | type | [] |
| EncryptionInfoValue | value | [] |
| ArchiveTimeStampSequence | archiveTimeStampSeq | [] |
| ArchiveTimeStampChain | archiveTimeStampChain | [] |
| ArchiveTimeStamp | archiveTimeStamp | [] |
| Id | id | [] |

[component EvidenceRecordValidity JSON schema details]

### Component ArchiveTimeStampValidity

#### Semantics

This component is based on the definition of the ArchiveTimeStamp-element in **[RFC4998]**

Below follows a list of the sub-components that MAY be present within this component:

* The FormatOK element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section VerificationResult. This element indicates, whether the format of the evidence record according to **[RFC4998]** is ok or not.
* The optional DigestAlgorithm element MUST contain a sub-component. A given element MUST satisfy the requirements specified in section AlgorithmValidity. This element contains, if present, information about the hash algorithm and possibly its suitability.
* The optional Attributes element MUST contain sub-components. This element contains, if present, information about further attributes related to the archive time stamp.
* The Attribute element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section Attribute. [sub component Attribute details]
* The optional ReducedHashTree element MUST contain sub-components. This element MAY contain a sequence of PartialHashTree elements, which in turn contain a list of HashValue elements
* The PartialHashTree element MUST occur 1 or more times containing sub-components. [sub component PartialHashTree details]
* The HashValue element MUST occur 1 or more times containing a sub-component. Each instance MUST satisfy the requirements specified in section HashValue. [sub component HashValue details]
* The TimeStamp element MUST contain one instance of a sub-component. This element MUST satisfy the requirements specified in section TimeStampValidity. contains information about the validity of the conventional time stamp, which is included in the present archive timestamp.
* The optional Id element MUST contain one instance of a unique identifier. contains an optional identifier for the component.

Non-normative Comment:

[component ArchiveTimeStampValidity non normative details]

#### XML Syntax

The XML type ArchiveTimeStampValidityType SHALL implement the requirements defined in the ArchiveTimeStampValidity component.

The ArchiveTimeStampValidityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="ArchiveTimeStampValidityType">

<sequence>

<element name="FormatOK" type="vr:VerificationResultType"/>

<element maxOccurs="1" minOccurs="0" name="DigestAlgorithm" type="vr:AlgorithmValidityType"/>

<element maxOccurs="1" minOccurs="0" name="Attributes">

<complexType>

<sequence>

<element maxOccurs="unbounded" minOccurs="1" name="Attribute" type="vr:AttributeType"/>

</sequence>

</complexType>

</element>

<element maxOccurs="1" minOccurs="0" name="ReducedHashTree">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="1">

<element name="PartialHashTree">

<complexType>

<sequence maxOccurs="unbounded" minOccurs="1">

<element name="HashValue" type="vr:HashValueType"/>

</sequence>

</complexType>

</element>

</sequence>

</complexType>

</element>

<element name="TimeStamp" type="vr:TimeStampValidityType"/>

</sequence>

<attribute name="Id" type="ID" use="optional"/>

</complexType>

Each child element of ArchiveTimeStampValidityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ArchiveTimeStampValidity XML schema details]

#### JSON Syntax

The ArchiveTimeStampValidityType JSON object SHALL implement in JSON syntax the requirements defined in the ArchiveTimeStampValidity component.

The ArchiveTimeStampValidityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-ArchiveTimeStampValidityType": {

"$xsd-full-type": "vr:ArchiveTimeStampValidityType",

"type": "object",

"properties": {

"formatOK": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"digestAlgo": {

"$ref": "#/definitions/vr-AlgorithmValidityType"

},

"attrs": {

"$ref": "#/definitions/vr-ArchiveTimeStampValidityType:Attributes"

},

"reducedHashTree": {

"$ref": "#/definitions/vr-ArchiveTimeStampValidityType:ReducedHashTree"

},

"timeStamp": {

"$ref": "#/definitions/vr-TimeStampValidityType"

},

"id": {

"type": "string"

}

},

"required": ["formatOK", "timeStamp"]

}

"vr-ArchiveTimeStampValidityType:Attributes": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"attr": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-AttributeType"

}

}

},

"required": ["attr"]

}

"vr-ArchiveTimeStampValidityType:ReducedHashTree": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"partialHashTree": {

"type": "array",

"items": {

"$ref": "#/definitions/vr-ArchiveTimeStampValidityType:ReducedHashTree:PartialHashTree"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ArchiveTimeStampValidity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| FormatOK | formatOK | [] |
| DigestAlgorithm | digestAlgo | [] |
| Attributes | attrs | [] |
| Attribute | attr | [] |
| ReducedHashTree | reducedHashTree | [] |
| PartialHashTree | partialHashTree | [] |
| HashValue | hashes | [] |
| TimeStamp | timeStamp | [] |
| Id | id | [] |

[component ArchiveTimeStampValidity JSON schema details]

### Component HashValue

#### Semantics

[component HashValue normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The HashValue element MUST contain one instance of hex encoded binary data. This element contains the hash value produced by applying the hash algorithm specified by the DigestAlgorithm or TimeStamp element to the data specified by the HashedObject element.
* The optional HashedObject element MUST contain one instance of a unique identifier reference. It MAY be used to point to the object, which served as pre-image of the hash value.

Non-normative Comment:

[component HashValue non normative details]

#### XML Syntax

The XML type HashValueType SHALL implement the requirements defined in the HashValue component.

The HashValueType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="HashValueType">

<sequence>

<element name="HashValue" type="hexBinary"/>

</sequence>

<attribute name="HashedObject" type="IDREF" use="optional"/>

</complexType>

Each child element of HashValueType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component HashValue XML schema details]

#### JSON Syntax

The HashValueType JSON object SHALL implement in JSON syntax the requirements defined in the HashValue component.

The HashValueType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"vr-HashValueType": {

"$xsd-full-type": "vr:HashValueType",

"type": "object",

"properties": {

"hashValue": {

"type": "string"

},

"hashedObject": {

"$ref": "#/definitions/dss2-DocumentBaseType"

}

},

"required": ["hashValue"]

}

Properties in the JSON schema above SHALL implement sub-component of HashValue component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| HashValue | hashValue | [] |
| HashedObject | hashedObject | [] |

[component HashValue JSON schema details]

## Referenced Structure Models from DSS-X base

### Component Base64Data

#### Semantics

[component Base64Data normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Value element MUST contain one instance of base64 encoded binary data. [sub component Value details]
* The optional AttRef element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section AttachmentReference. [sub component AttRef details]
* The optional MimeType element MUST contain one instance of a string. [sub component MimeType details]
* The optional ID element MUST contain one instance of a unique identifier. [sub component ID details]
* The optional IDREF element MUST contain one instance of a unique identifier reference. [sub component IDREF details]

Non-normative Comment:

[component Base64Data non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type Base64DataType SHALL implement the requirements defined in the Base64Data component.

The Base64DataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="Base64DataType">

<xs:choice minOccurs="0">

<xs:element name="Value" type="xs:base64Binary"/>

<xs:element name="AttRef" type="dsb:AttachmentReferenceType"/>

</xs:choice>

<xs:attribute name="MimeType" type="xs:string" use="optional"/>

<xs:attribute name="ID" type="xs:ID" use="optional"/>

<xs:attribute name="IDREF" type="xs:IDREF" use="optional"/>

</xs:complexType>

Each child element of Base64DataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Base64Data XML schema details]

#### JSON Syntax

The Base64DataType JSON object SHALL implement in JSON syntax the requirements defined in the Base64Data component.

The Base64DataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-Base64DataType": {

"$xsd-full-type": "dsb:Base64DataType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"value": {

"type": "string"

},

"attRef": {

"$ref": "#/definitions/dsb-AttachmentReferenceType"

},

"mimeType": {

"type": "string"

},

"IDREF": {

"type": "string"

}

},

"minProperties": 0

}

Properties in the JSON schema above SHALL implement sub-component of Base64Data component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Value | value | [] |
| AttRef | attRef | [] |
| MimeType | mimeType | [] |
| ID | ID | [] |
| IDREF | IDREF | [] |

[component Base64Data JSON schema details]

### Component AttachmentReference

#### Semantics

[component AttachmentReference normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional DigestInfo element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section DigestInfo. [sub component DigestInfo details]
* The AttRefURI element MUST contain one instance of a URI. [sub component AttRefURI details]

Non-normative Comment:

[component AttachmentReference non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type AttachmentReferenceType SHALL implement the requirements defined in the AttachmentReference component.

The AttachmentReferenceType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AttachmentReferenceType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="DigestInfo" type="dsb:DigestInfoType"/>

</xs:sequence>

<xs:attribute name="AttRefURI" type="xs:anyURI" use="required"/>

</xs:complexType>

Each child element of AttachmentReferenceType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AttachmentReference XML schema details]

#### JSON Syntax

The AttachmentReferenceType JSON object SHALL implement in JSON syntax the requirements defined in the AttachmentReference component.

The AttachmentReferenceType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-AttachmentReferenceType": {

"$xsd-full-type": "dsb:AttachmentReferenceType",

"type": "object",

"properties": {

"di": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-DigestInfoType"

}

},

"attURI": {

"type": "string"

}

},

"required": ["attURI"]

}

Properties in the JSON schema above SHALL implement sub-component of AttachmentReference component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestInfo | di | [] |
| AttRefURI | attURI | [] |

[component AttachmentReference JSON schema details]

### Component DigestInfo

#### Semantics

[component DigestInfo normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The DigestMethod element MUST contain one instance of a string. [sub component DigestMethod details]
* The DigestValue element MUST contain one instance of base64 encoded binary data. [sub component DigestValue details]

Non-normative Comment:

[component DigestInfo non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type DigestInfoType SHALL implement the requirements defined in the DigestInfo component.

The DigestInfoType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DigestInfoType">

<xs:sequence>

<xs:element name="DigestMethod" type="xs:string"/>

<xs:element name="DigestValue" type="xs:base64Binary"/>

</xs:sequence>

</xs:complexType>

Each child element of DigestInfoType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DigestInfo XML schema details]

#### JSON Syntax

The DigestInfoType JSON object SHALL implement in JSON syntax the requirements defined in the DigestInfo component.

The DigestInfoType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-DigestInfoType": {

"$xsd-full-type": "dsb:DigestInfoType",

"type": "object",

"properties": {

"alg": {

"type": "string"

},

"value": {

"type": "string"

}

},

"required": ["alg", "value"]

}

Properties in the JSON schema above SHALL implement sub-component of DigestInfo component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestMethod | alg | [] |
| DigestValue | value | [] |

[component DigestInfo JSON schema details]

### Component NsPrefixMapping

#### Semantics

[component NsPrefixMapping normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The NamespaceURI element MUST contain one instance of a URI. [sub component NamespaceURI details]
* The NamespacePrefix element MUST contain one instance of a string. [sub component NamespacePrefix details]

Non-normative Comment:

[component NsPrefixMapping non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type NsPrefixMappingType SHALL implement the requirements defined in the NsPrefixMapping component.

The NsPrefixMappingType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="NsPrefixMappingType">

<xs:sequence>

<xs:element name="NamespaceURI" type="xs:anyURI"/>

<xs:element name="NamespacePrefix" type="xs:string"/>

</xs:sequence>

</xs:complexType>

Each child element of NsPrefixMappingType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component NsPrefixMapping XML schema details]

#### JSON Syntax

The NsPrefixMappingType JSON object SHALL implement in JSON syntax the requirements defined in the NsPrefixMapping component.

The NsPrefixMappingType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-NsPrefixMappingType": {

"$xsd-full-type": "dsb:NsPrefixMappingType",

"type": "object",

"properties": {

"uri": {

"type": "string"

},

"pre": {

"type": "string"

}

},

"required": ["uri", "pre"]

}

Properties in the JSON schema above SHALL implement sub-component of NsPrefixMapping component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| NamespaceURI | uri | [] |
| NamespacePrefix | pre | [] |

[component NsPrefixMapping JSON schema details]

### Component Any

#### Semantics

[component Any normative details]

Below follows a list of the sub-components that MAY be present within this component:

A set of sub-components is inherited from component Base64Data and is not repeated here.

Non-normative Comment:

[component Any non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type AnyType SHALL implement the requirements defined in the Any component.

The AnyType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AnyType">

<xs:complexContent>

<xs:extension base="dsb:Base64DataType"/>

</xs:complexContent>

</xs:complexType>

Each child element of AnyType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Any XML schema details]

#### JSON Syntax

The AnyType JSON object SHALL implement in JSON syntax the requirements defined in the Any component.

The AnyType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-AnyType": {

"$xsd-full-type": "dsb:AnyType xades-rw:AnyType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"value": {

"type": "string"

},

"attRef": {

"$ref": "#/definitions/dsb-AttachmentReferenceType"

},

"mimeType": {

"type": "string"

},

"IDREF": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Any component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |

[component Any JSON schema details]

### Component Result

#### Semantics

[component Result normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ResultMajor element MUST contain one instance of a URI. Its value is limited to an item of the following set:  
  urn:oasis:names:tc:dss:1.0:resultmajor:Success  
  urn:oasis:names:tc:dss:1.0:resultmajor:RequesterError  
  urn:oasis:names:tc:dss:1.0:resultmajor:ResponderError  
  urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation  
  [sub component ResultMajor details]
* The optional ResultMinor element MUST contain a URI. [sub component ResultMinor details]
* The optional ResultMessage element MUST contain sub-component. A given element MUST satisfy the requirements specified in section InternationalString. [sub component ResultMessage details]
* The optional ProblemReference element MUST contain a string. [sub component ProblemReference details]

Non-normative Comment:

[component Result non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type ResultType SHALL implement the requirements defined in the Result component.

The ResultType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ResultType">

<xs:sequence>

<xs:element name="ResultMajor">

<xs:simpleType>

<xs:restriction base="xs:anyURI">

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:resultmajor:Success"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:resultmajor:RequesterError"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:resultmajor:ResponderError"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element minOccurs="0" name="ResultMinor" type="xs:anyURI"/>

<xs:element minOccurs="0" name="ResultMessage" type="dsb:InternationalStringType"/>

<xs:element minOccurs="0" name="ProblemReference" type="xs:string"/>

</xs:sequence>

</xs:complexType>

Each child element of ResultType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Result XML schema details]

#### JSON Syntax

The ResultType JSON object SHALL implement in JSON syntax the requirements defined in the Result component.

The ResultType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-ResultType": {

"$xsd-full-type": "dsb:ResultType",

"type": "object",

"properties": {

"maj": {

"type": "string",

"enum": ["urn:oasis:names:tc:dss:1.0:resultmajor:Success", "urn:oasis:names:tc:dss:1.0:resultmajor:RequesterError", "urn:oasis:names:tc:dss:1.0:resultmajor:ResponderError", "urn:oasis:names:tc:dss:1.0:resultmajor:InsufficientInformation"]

},

"min": {

"type": "string"

},

"msg": {

"$ref": "#/definitions/dsb-InternationalStringType"

},

"pRef": {

"type": "string"

}

},

"required": ["maj"]

}

Properties in the JSON schema above SHALL implement sub-component of Result component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ResultMajor | maj | [] |
| ResultMinor | min | [] |
| ResultMessage | msg | [] |
| ProblemReference | pRef | [] |

[component Result JSON schema details]

### Component InternationalString

#### Semantics

[component InternationalString normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a string. [sub component value details]
* The lang element MUST contain one instance of a ISO language descriptor. [sub component lang details]

Non-normative Comment:

[component InternationalString non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type InternationalStringType SHALL implement the requirements defined in the InternationalString component.

The InternationalStringType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="InternationalStringType">

<xs:simpleContent>

<xs:extension base="xs:string">

<xs:attribute ref="xml:lang" use="required"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

Each child element of InternationalStringType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' is represented by the component's XML tag text content. [component InternationalString XML schema details]

#### JSON Syntax

The InternationalStringType JSON object SHALL implement in JSON syntax the requirements defined in the InternationalString component.

The InternationalStringType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-InternationalStringType": {

"$xsd-full-type": "dsb:InternationalStringType",

"type": "object",

"properties": {

"value": {

"type": "string"

},

"lang": {

"type": "string"

}

},

"required": ["lang"]

}

Properties in the JSON schema above SHALL implement sub-component of InternationalString component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| lang | lang | [] |

[component InternationalString JSON schema details]

### Component ResponseBase

#### Semantics

[component ResponseBase normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Result element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Result. [sub component Result details]
* The optional AppliedProfile element MAY occur zero or more times containing a URI. [sub component AppliedProfile details]
* The optional RequestID element MUST contain one instance of a string. [sub component RequestID details]

Non-normative Comment:

[component ResponseBase non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/base' .The XML type ResponseBaseType SHALL implement the requirements defined in the ResponseBase component.

The ResponseBaseType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType abstract="true" name="ResponseBaseType">

<xs:sequence>

<xs:element name="Result" type="dsb:ResultType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AppliedProfile" type="xs:anyURI"/>

</xs:sequence>

<xs:attribute name="RequestID" type="xs:string" use="optional"/>

</xs:complexType>

Each child element of ResponseBaseType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ResponseBase XML schema details]

#### JSON Syntax

The ResponseBaseType JSON object SHALL implement in JSON syntax the requirements defined in the ResponseBase component.

The ResponseBaseType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-ResponseBaseType": {

"$xsd-full-type": "dsb:ResponseBaseType",

"type": "object",

"properties": {

"result": {

"$ref": "#/definitions/dsb-ResultType"

},

"profile": {

"type": "array",

"items": {

"type": "string"

}

},

"reqID": {

"type": "string"

}

},

"required": ["result"]

}

Properties in the JSON schema above SHALL implement sub-component of ResponseBase component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Result | result | [] |
| AppliedProfile | profile | [] |
| RequestID | reqID | [] |

[component ResponseBase JSON schema details]

## Referenced Structure Models from DSS-X core

### Component SignRequest

#### Semantics

[component SignRequest normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional InputDocuments element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section InputDocuments. [sub component InputDocuments details]
* The optional OptionalInputs element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section OptionalInputsSign. [sub component OptionalInputs details]

A set of sub-components is inherited from component RequestBase and is not repeated here.

Non-normative Comment:

[component SignRequest non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignRequestType SHALL implement the requirements defined in the SignRequest component.

The SignRequestType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignRequestType">

<xs:complexContent>

<xs:extension base="dsb:RequestBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="InputDocuments" type="dss2:InputDocumentsType"/>

<xs:element minOccurs="0" name="OptionalInputs" type="dss2:OptionalInputsSignType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of SignRequestType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignRequest XML schema details]

#### JSON Syntax

The SignRequestType JSON object SHALL implement in JSON syntax the requirements defined in the SignRequest component.

The SignRequestType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignRequestType": {

"$xsd-full-type": "dss2:SignRequestType",

"type": "object",

"properties": {

"profile": {

"type": "array",

"items": {

"type": "string"

}

},

"reqID": {

"type": "string"

},

"inDocs": {

"$ref": "#/definitions/dss2-InputDocumentsType"

},

"optInp": {

"$ref": "#/definitions/dss2-OptionalInputsSignType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignRequest component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| InputDocuments | inDocs | [] |
| OptionalInputs | optInp | [] |

[component SignRequest JSON schema details]

### Component InputDocuments

#### Semantics

[component InputDocuments normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Document element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section Document. [sub component Document details]
* The TransformedData element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section TransformedData. [sub component TransformedData details]
* The DocumentHash element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section DocumentHash. [sub component DocumentHash details]

Non-normative Comment:

[component InputDocuments non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type InputDocumentsType SHALL implement the requirements defined in the InputDocuments component.

The InputDocumentsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="InputDocumentsType">

<xs:choice>

<xs:sequence maxOccurs="unbounded">

<xs:element name="Document" type="dss2:DocumentType"/>

</xs:sequence>

<xs:sequence maxOccurs="unbounded">

<xs:element name="TransformedData" type="dss2:TransformedDataType"/>

</xs:sequence>

<xs:sequence maxOccurs="unbounded">

<xs:element name="DocumentHash" type="dss2:DocumentHashType"/>

</xs:sequence>

</xs:choice>

</xs:complexType>

Each child element of InputDocumentsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component InputDocuments XML schema details]

#### JSON Syntax

The InputDocumentsType JSON object SHALL implement in JSON syntax the requirements defined in the InputDocuments component.

The InputDocumentsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-InputDocumentsType": {

"$xsd-full-type": "dss2:InputDocumentsType",

"type": "object",

"properties": {

"doc": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

},

"transformed": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-TransformedDataType"

}

},

"docHash": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentHashType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of InputDocuments component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Document | doc | [] |
| TransformedData | transformed | [] |
| DocumentHash | docHash | [] |

[component InputDocuments JSON schema details]

### Component Document

#### Semantics

[component Document normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Base64Data element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Base64Data. [sub component Base64Data details]

A set of sub-components is inherited from component DocumentBase and is not repeated here.

Non-normative Comment:

[component Document non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type DocumentType SHALL implement the requirements defined in the Document component.

The DocumentType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DocumentType">

<xs:complexContent>

<xs:extension base="dss2:DocumentBaseType">

<xs:choice>

<xs:element name="Base64Data" type="dsb:Base64DataType"/>

</xs:choice>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of DocumentType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Document XML schema details]

#### JSON Syntax

The DocumentType JSON object SHALL implement in JSON syntax the requirements defined in the Document component.

The DocumentType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-DocumentType": {

"$xsd-full-type": "dss2:DocumentType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"refURI": {

"type": "string"

},

"refType": {

"type": "string"

},

"schemaRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

},

"b64Data": {

"$ref": "#/definitions/dsb-Base64DataType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Document component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Base64Data | b64Data | [] |

[component Document JSON schema details]

### Component TransformedData

#### Semantics

[component TransformedData normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Transforms element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Transforms. [sub component Transforms details]
* The Base64Data element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Base64Data. [sub component Base64Data details]
* The optional WhichReference element MUST contain one instance of an integer. [sub component WhichReference details]

A set of sub-components is inherited from component DocumentBase and is not repeated here.

Non-normative Comment:

[component TransformedData non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type TransformedDataType SHALL implement the requirements defined in the TransformedData component.

The TransformedDataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="TransformedDataType">

<xs:complexContent>

<xs:extension base="dss2:DocumentBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="Transforms" type="ds-rw:TransformsType"/>

<xs:element name="Base64Data" type="dsb:Base64DataType"/>

</xs:sequence>

<xs:attribute name="WhichReference" type="xs:integer" use="optional"/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of TransformedDataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component TransformedData XML schema details]

#### JSON Syntax

The TransformedDataType JSON object SHALL implement in JSON syntax the requirements defined in the TransformedData component.

The TransformedDataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-TransformedDataType": {

"$xsd-full-type": "dss2:TransformedDataType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"refURI": {

"type": "string"

},

"refType": {

"type": "string"

},

"schemaRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

},

"transforms": {

"$ref": "#/definitions/dsigrw-TransformsType"

},

"b64Data": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"whichRef": {

"type": "integer"

}

},

"required": ["b64Data"]

}

Properties in the JSON schema above SHALL implement sub-component of TransformedData component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Transforms | transforms | [] |
| Base64Data | b64Data | [] |
| WhichReference | whichRef | [] |

[component TransformedData JSON schema details]

### Component DocumentHash

#### Semantics

[component DocumentHash normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Transforms element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Transforms. [sub component Transforms details]
* The DigestInfos element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section DigestInfo. [sub component DigestInfos details]
* The optional WhichReference element MUST contain one instance of an integer. [sub component WhichReference details]

A set of sub-components is inherited from component DocumentBase and is not repeated here.

Non-normative Comment:

[component DocumentHash non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type DocumentHashType SHALL implement the requirements defined in the DocumentHash component.

The DocumentHashType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DocumentHashType">

<xs:complexContent>

<xs:extension base="dss2:DocumentBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="Transforms" type="ds-rw:TransformsType"/>

<xs:element maxOccurs="unbounded" minOccurs="1" name="DigestInfos" type="dsb:DigestInfoType"/>

</xs:sequence>

<xs:attribute name="WhichReference" type="xs:integer" use="optional"/>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of DocumentHashType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DocumentHash XML schema details]

#### JSON Syntax

The DocumentHashType JSON object SHALL implement in JSON syntax the requirements defined in the DocumentHash component.

The DocumentHashType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-DocumentHashType": {

"$xsd-full-type": "dss2:DocumentHashType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"refURI": {

"type": "string"

},

"refType": {

"type": "string"

},

"schemaRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

},

"transforms": {

"$ref": "#/definitions/dsigrw-TransformsType"

},

"di": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-DigestInfoType"

}

},

"whichRef": {

"type": "integer"

}

},

"required": ["di"]

}

Properties in the JSON schema above SHALL implement sub-component of DocumentHash component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Transforms | transforms | [] |
| DigestInfos | di | [] |
| WhichReference | whichRef | [] |

[component DocumentHash JSON schema details]

### Component OptionalInputsSign

#### Semantics

[component OptionalInputsSign normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignatureType element MUST contain a URI. [sub component SignatureType details]
* The optional IntendedAudience element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section IntendedAudience. [sub component IntendedAudience details]
* The optional KeySelector element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section KeySelector. [sub component KeySelector details]
* The optional Properties element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section PropertiesHolder. [sub component Properties details]
* The optional IncludeObject element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section IncludeObject. [sub component IncludeObject details]
* The optional IncludeEContent element MUST contain a boolean. Its default value is 'false'. [sub component IncludeEContent details]
* The optional SignaturePlacement element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section SignaturePlacement. [sub component SignaturePlacement details]
* The optional SignedReferences element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section SignedReferences. [sub component SignedReferences details]
* The optional Nonce element MUST contain an integer. [sub component Nonce details]
* The optional SignatureAlgorithm element MUST contain a string. [sub component SignatureAlgorithm details]
* The optional SignatureActivationData element MUST contain a string. [sub component SignatureActivationData details]
* The JWSProfileInputs element MUST contain one instance of a sub\_component . This element MUST satisfy the requirements specified in section . [sub component JWSProfileInputs details]

A set of sub-components is inherited from component OptionalInputsBase and is not repeated here.

Non-normative Comment:

[component OptionalInputsSign non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type OptionalInputsSignType SHALL implement the requirements defined in the OptionalInputsSign component.

The OptionalInputsSignType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalInputsSignType">

<xs:complexContent>

<xs:extension base="dss2:OptionalInputsBaseType">

<xs:sequence>

<xs:choice>

<xs:element maxOccurs="1" minOccurs="0" name="SignatureType" type="xs:anyURI"/>

<xs:element maxOccurs="1" minOccurs="0" name="IntendedAudience" type="dss2:IntendedAudienceType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="KeySelector" type="dss2:KeySelectorType"/>

<xs:element maxOccurs="1" minOccurs="0" name="Properties" type="dss2:PropertiesHolderType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="IncludeObject" type="dss2:IncludeObjectType"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="IncludeEContent" type="xs:boolean"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignaturePlacement" type="dss2:SignaturePlacementType"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignedReferences" type="dss2:SignedReferencesType"/>

<xs:element maxOccurs="1" minOccurs="0" name="Nonce" type="xs:integer"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignatureAlgorithm" type="xs:string"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignatureActivationData" type="xs:string"/>

<xs:element name="JWSProfileInputs" type="jws:OptionalInputSignType"/>

</xs:choice>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of OptionalInputsSignType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalInputsSign XML schema details]

#### JSON Syntax

The OptionalInputsSignType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalInputsSign component.

The OptionalInputsSignType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-OptionalInputsSignType": {

"$xsd-full-type": "dss2:OptionalInputsSignType",

"type": "object",

"properties": {

"jwsprofileInputs": {

"$ref": "#/definitions/async-OptionalInputSignType"

},

"policy": {

"type": "array",

"items": {

"type": "string"

}

},

"lang": {

"type": "string"

},

"other": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"claimedIdentity": {

"$ref": "#/definitions/dss2-ClaimedIdentityType"

},

"schemas": {

"$ref": "#/definitions/dss2-SchemasType"

},

"addTimestamp": {

"$ref": "#/definitions/dss2-UpdateSignatureInstructionType"

},

"sigType": {

"type": "string"

},

"aud": {

"$ref": "#/definitions/dss2-IntendedAudienceType"

},

"keySel": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-KeySelectorType"

}

},

"props": {

"$ref": "#/definitions/dss2-PropertiesHolderType"

},

"incObj": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-IncludeObjectType"

}

},

"incContent": {

"type": "boolean",

"default": "false"

},

"sigPlacement": {

"$ref": "#/definitions/dss2-SignaturePlacementType"

},

"sigRefs": {

"$ref": "#/definitions/dss2-SignedReferencesType"

},

"nonce": {

"type": "integer"

},

"sigAlgo": {

"type": "string"

},

"sad": {

"type": "string"

},

"optJws": {

"$ref": "#/definitions/async-OptionalInputSignType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalInputsSign component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignatureType | sigType | [] |
| IntendedAudience | aud | [] |
| KeySelector | keySel | [] |
| Properties | props | [] |
| IncludeObject | incObj | [] |
| IncludeEContent | incContent | [] |
| SignaturePlacement | sigPlacement | [] |
| SignedReferences | sigRefs | [] |
| Nonce | nonce | [] |
| SignatureAlgorithm | sigAlgo | [] |
| SignatureActivationData | sad | [] |
| JWSProfileInputs | optJws | [] |

[component OptionalInputsSign JSON schema details]

### Component ClaimedIdentity

#### Semantics

[component ClaimedIdentity normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Name element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section NameID. [sub component Name details]
* The optional SupportingInfo element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component SupportingInfo details]

Non-normative Comment:

[component ClaimedIdentity non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type ClaimedIdentityType SHALL implement the requirements defined in the ClaimedIdentity component.

The ClaimedIdentityType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ClaimedIdentityType">

<xs:sequence>

<xs:element name="Name" type="saml2-rw:NameIDType"/>

<xs:element minOccurs="0" name="SupportingInfo" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of ClaimedIdentityType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ClaimedIdentity XML schema details]

#### JSON Syntax

The ClaimedIdentityType JSON object SHALL implement in JSON syntax the requirements defined in the ClaimedIdentity component.

The ClaimedIdentityType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-ClaimedIdentityType": {

"$xsd-full-type": "dss2:ClaimedIdentityType",

"type": "object",

"properties": {

"name": {

"$ref": "#/definitions/saml2rw-NameIDType"

},

"suppInfo": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"required": ["name"]

}

Properties in the JSON schema above SHALL implement sub-component of ClaimedIdentity component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Name | name | [] |
| SupportingInfo | suppInfo | [] |

[component ClaimedIdentity JSON schema details]

### Component Schemas

#### Semantics

[component Schemas normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Schema element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in the core specification in section Document. [sub component Schema details]

Non-normative Comment:

[component Schemas non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SchemasType SHALL implement the requirements defined in the Schemas component.

The SchemasType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SchemasType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="Schema" type="dss2:DocumentType"/>

</xs:sequence>

</xs:complexType>

Each child element of SchemasType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Schemas XML schema details]

#### JSON Syntax

The SchemasType JSON object SHALL implement in JSON syntax the requirements defined in the Schemas component.

The SchemasType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SchemasType": {

"$xsd-full-type": "dss2:SchemasType",

"type": "object",

"properties": {

"schema": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

}

},

"required": ["schema"]

}

Properties in the JSON schema above SHALL implement sub-component of Schemas component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Schema | schema | [] |

[component Schemas JSON schema details]

### Component UpdateSignatureInstruction

#### Semantics

[component UpdateSignatureInstruction normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Type element MUST contain one instance of a URI. [sub component Type details]

Non-normative Comment:

[component UpdateSignatureInstruction non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type UpdateSignatureInstructionType SHALL implement the requirements defined in the UpdateSignatureInstruction component.

The UpdateSignatureInstructionType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="UpdateSignatureInstructionType">

<xs:attribute name="Type" type="xs:anyURI" use="optional"/>

</xs:complexType>

Each child element of UpdateSignatureInstructionType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UpdateSignatureInstruction XML schema details]

#### JSON Syntax

The UpdateSignatureInstructionType JSON object SHALL implement in JSON syntax the requirements defined in the UpdateSignatureInstruction component.

The UpdateSignatureInstructionType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-UpdateSignatureInstructionType": {

"$xsd-full-type": "dss2:UpdateSignatureInstructionType",

"type": "object",

"properties": {

"type": {

"type": "string",

"format": "uri"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of UpdateSignatureInstruction component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Type | type | [] |

[component UpdateSignatureInstruction JSON schema details]

### Component IntendedAudience

#### Semantics

[component IntendedAudience normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Recipient element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section NameID. [sub component Recipient details]

Non-normative Comment:

[component IntendedAudience non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type IntendedAudienceType SHALL implement the requirements defined in the IntendedAudience component.

The IntendedAudienceType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="IntendedAudienceType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="Recipient" type="saml2-rw:NameIDType"/>

</xs:sequence>

</xs:complexType>

Each child element of IntendedAudienceType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component IntendedAudience XML schema details]

#### JSON Syntax

The IntendedAudienceType JSON object SHALL implement in JSON syntax the requirements defined in the IntendedAudience component.

The IntendedAudienceType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-IntendedAudienceType": {

"$xsd-full-type": "dss2:IntendedAudienceType",

"type": "object",

"properties": {

"recipient": {

"type": "array",

"items": {

"$ref": "#/definitions/saml2rw-NameIDType"

}

}

},

"required": ["recipient"]

}

Properties in the JSON schema above SHALL implement sub-component of IntendedAudience component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Recipient | recipient | [] |

[component IntendedAudience JSON schema details]

### Component KeySelector

#### Semantics

[component KeySelector normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The X509Digest element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section X509Digest. [sub component X509Digest details]
* The X509SubjectName element MUST contain one instance of a string. [sub component X509SubjectName details]
* The X509SKI element MUST contain one instance of base64 encoded binary data. [sub component X509SKI details]
* The X509Certificate element MUST contain one instance of base64 encoded binary data. [sub component X509Certificate details]
* The KeyName element MUST contain one instance of a string. [sub component KeyName details]

Non-normative Comment:

[component KeySelector non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type KeySelectorType SHALL implement the requirements defined in the KeySelector component.

The KeySelectorType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="KeySelectorType">

<xs:choice>

<xs:element name="X509Digest" type="dss2:X509DigestType"/>

<xs:element name="X509SubjectName" type="xs:string"/>

<xs:element name="X509SKI" type="xs:base64Binary"/>

<xs:element name="X509Certificate" type="xs:base64Binary"/>

<xs:element name="KeyName" type="xs:string"/>

</xs:choice>

</xs:complexType>

Each child element of KeySelectorType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component KeySelector XML schema details]

#### JSON Syntax

The KeySelectorType JSON object SHALL implement in JSON syntax the requirements defined in the KeySelector component.

The KeySelectorType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-KeySelectorType": {

"$xsd-full-type": "dss2:KeySelectorType",

"type": "object",

"properties": {

"x509Digest": {

"$ref": "#/definitions/dss2-X509DigestType"

},

"subject": {

"type": "string"

},

"ski": {

"type": "string"

},

"cert": {

"type": "string"

},

"name": {

"type": "string"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of KeySelector component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| X509Digest | x509Digest | [] |
| X509SubjectName | subject | [] |
| X509SKI | ski | [] |
| X509Certificate | cert | [] |
| KeyName | name | [] |

[component KeySelector JSON schema details]

### Component X509Digest

#### Semantics

[component X509Digest normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of base64 encoded binary data. [sub component value details]
* The Algorithm element MUST contain one instance of a string. [sub component Algorithm details]

Non-normative Comment:

[component X509Digest non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type X509DigestType SHALL implement the requirements defined in the X509Digest component.

The X509DigestType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="X509DigestType">

<xs:simpleContent>

<xs:extension base="xs:base64Binary">

<xs:attribute name="Algorithm" type="xs:string" use="required"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

Each child element of X509DigestType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' holding the base64 content is represented by the component's XML tag text content. [component X509Digest XML schema details]

#### JSON Syntax

The X509DigestType JSON object SHALL implement in JSON syntax the requirements defined in the X509Digest component.

The X509DigestType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-X509DigestType": {

"$xsd-full-type": "dss2:X509DigestType",

"type": "object",

"properties": {

"value": {

"type": "string"

},

"algo": {

"type": "string"

}

},

"required": ["algo"]

}

Properties in the JSON schema above SHALL implement sub-component of X509Digest component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Algorithm | algo | [] |

[component X509Digest JSON schema details]

### Component PropertiesHolder

#### Semantics

[component PropertiesHolder normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignedProperties element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section Properties. [sub component SignedProperties details]
* The optional UnsignedProperties element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section Properties. [sub component UnsignedProperties details]

Non-normative Comment:

[component PropertiesHolder non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type PropertiesHolderType SHALL implement the requirements defined in the PropertiesHolder component.

The PropertiesHolderType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="PropertiesHolderType">

<xs:sequence>

<xs:element minOccurs="0" name="SignedProperties" type="dss2:PropertiesType"/>

<xs:element minOccurs="0" name="UnsignedProperties" type="dss2:PropertiesType"/>

</xs:sequence>

</xs:complexType>

Each child element of PropertiesHolderType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component PropertiesHolder XML schema details]

#### JSON Syntax

The PropertiesHolderType JSON object SHALL implement in JSON syntax the requirements defined in the PropertiesHolder component.

The PropertiesHolderType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-PropertiesHolderType": {

"$xsd-full-type": "dss2:PropertiesHolderType",

"type": "object",

"properties": {

"signedProps": {

"$ref": "#/definitions/dss2-PropertiesType"

},

"unsignedProps": {

"$ref": "#/definitions/dss2-PropertiesType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of PropertiesHolder component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignedProperties | signedProps | [] |
| UnsignedProperties | unsignedProps | [] |

[component PropertiesHolder JSON schema details]

### Component Properties

#### Semantics

[component Properties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Property element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in the core specification in section Property. [sub component Property details]

Non-normative Comment:

[component Properties non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type PropertiesType SHALL implement the requirements defined in the Properties component.

The PropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="PropertiesType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="Property" type="dss2:PropertyType"/>

</xs:sequence>

</xs:complexType>

Each child element of PropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Properties XML schema details]

#### JSON Syntax

The PropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the Properties component.

The PropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-PropertiesType": {

"$xsd-full-type": "vr:PropertiesType dss2:PropertiesType",

"type": "object",

"properties": {

"prop": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-PropertyType"

}

}

},

"required": ["prop"]

}

Properties in the JSON schema above SHALL implement sub-component of Properties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Property | prop | [] |

[component Properties JSON schema details]

### Component Property

#### Semantics

[component Property normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Identifier element MUST contain one instance of a string. [sub component Identifier details]
* The optional Value element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component Value details]

Non-normative Comment:

[component Property non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type PropertyType SHALL implement the requirements defined in the Property component.

The PropertyType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="PropertyType">

<xs:sequence>

<xs:element name="Identifier" type="xs:string"/>

<xs:element minOccurs="0" name="Value" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of PropertyType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Property XML schema details]

#### JSON Syntax

The PropertyType JSON object SHALL implement in JSON syntax the requirements defined in the Property component.

The PropertyType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-PropertyType": {

"$xsd-full-type": "dss2:PropertyType",

"type": "object",

"properties": {

"id": {

"type": "string"

},

"value": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"required": ["id"]

}

Properties in the JSON schema above SHALL implement sub-component of Property component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Identifier | id | [] |
| Value | value | [] |

[component Property JSON schema details]

### Component IncludeObject

#### Semantics

[component IncludeObject normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional WhichDocument element MUST contain one instance of a unique identifier reference. [sub component WhichDocument details]
* The optional hasObjectTagsAndAttributesSet element MUST contain one instance of a boolean. Its default value is 'false'. [sub component hasObjectTagsAndAttributesSet details]
* The optional ObjId element MUST contain one instance of a string. [sub component ObjId details]
* The optional createReference element MUST contain one instance of a boolean. Its default value is 'true'. [sub component createReference details]

Non-normative Comment:

[component IncludeObject non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type IncludeObjectType SHALL implement the requirements defined in the IncludeObject component.

The IncludeObjectType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="IncludeObjectType">

<xs:attribute name="WhichDocument" type="xs:IDREF"/>

<xs:attribute default="false" name="hasObjectTagsAndAttributesSet" type="xs:boolean"/>

<xs:attribute name="ObjId" type="xs:string" use="optional"/>

<xs:attribute default="true" name="createReference" type="xs:boolean" use="optional"/>

</xs:complexType>

Each child element of IncludeObjectType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component IncludeObject XML schema details]

#### JSON Syntax

The IncludeObjectType JSON object SHALL implement in JSON syntax the requirements defined in the IncludeObject component.

The IncludeObjectType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-IncludeObjectType": {

"$xsd-full-type": "dss2:IncludeObjectType",

"type": "object",

"properties": {

"whichDoc": {

"$ref": "#/definitions/dss2-DocumentBaseType"

},

"hasObjectTagsAndAttributesSet": {

"type": "boolean",

"default": "false"

},

"objId": {

"type": "string"

},

"createRef": {

"type": "boolean",

"default": "true"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of IncludeObject component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| WhichDocument | whichDoc | [] |
| hasObjectTagsAndAttributesSet | hasObjectTagsAndAttributesSet | [] |
| ObjId | objId | [] |
| createReference | createRef | [] |

[component IncludeObject JSON schema details]

### Component DocumentBase

#### Semantics

[component DocumentBase normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional ID element MUST contain one instance of a unique identifier. [sub component ID details]
* The optional RefURI element MUST contain one instance of a URI. [sub component RefURI details]
* The optional RefType element MUST contain one instance of a URI. [sub component RefType details]
* The optional SchemaRefs element MUST contain one instance of a unique identifier reference. [sub component SchemaRefs details]

Non-normative Comment:

[component DocumentBase non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type DocumentBaseType SHALL implement the requirements defined in the DocumentBase component.

The DocumentBaseType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType abstract="true" name="DocumentBaseType">

<xs:attribute name="ID" type="xs:ID" use="optional"/>

<xs:attribute name="RefURI" type="xs:anyURI" use="optional"/>

<xs:attribute name="RefType" type="xs:anyURI" use="optional"/>

<xs:attribute name="SchemaRefs" type="xs:IDREFS" use="optional"/>

</xs:complexType>

Each child element of DocumentBaseType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DocumentBase XML schema details]

#### JSON Syntax

The DocumentBaseType JSON object SHALL implement in JSON syntax the requirements defined in the DocumentBase component.

The DocumentBaseType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-DocumentBaseType": {

"$xsd-full-type": "dss2:DocumentBaseType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"refURI": {

"type": "string"

},

"refType": {

"type": "string"

},

"schemaRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of DocumentBase component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ID | ID | [] |
| RefURI | refURI | [] |
| RefType | refType | [] |
| SchemaRefs | schemaRefs | [] |

[component DocumentBase JSON schema details]

### Component SignaturePlacement

#### Semantics

[component SignaturePlacement normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The XPathAfter element MUST contain one instance of a string. [sub component XPathAfter details]
* The XPathFirstChildOf element MUST contain one instance of a string. [sub component XPathFirstChildOf details]
* The optional NsPrefixMapping element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section NsPrefixMapping. [sub component NsPrefixMapping details]
* The optional WhichDocument element MUST contain one instance of a unique identifier reference. [sub component WhichDocument details]
* The optional CreateEnvelopedSignature element MUST contain one instance of a boolean. Its default value is 'true'. [sub component CreateEnvelopedSignature details]

Non-normative Comment:

[component SignaturePlacement non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignaturePlacementType SHALL implement the requirements defined in the SignaturePlacement component.

The SignaturePlacementType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignaturePlacementType">

<xs:sequence>

<xs:choice>

<xs:element name="XPathAfter" type="xs:string"/>

<xs:element name="XPathFirstChildOf" type="xs:string"/>

</xs:choice>

<xs:element maxOccurs="unbounded" minOccurs="0" name="NsPrefixMapping" type="dsb:NsPrefixMappingType"/>

</xs:sequence>

<xs:attribute name="WhichDocument" type="xs:IDREF"/>

<xs:attribute default="true" name="CreateEnvelopedSignature" type="xs:boolean"/>

</xs:complexType>

Each child element of SignaturePlacementType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignaturePlacement XML schema details]

#### JSON Syntax

The SignaturePlacementType JSON object SHALL implement in JSON syntax the requirements defined in the SignaturePlacement component.

The SignaturePlacementType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignaturePlacementType": {

"$xsd-full-type": "dss2:SignaturePlacementType",

"type": "object",

"properties": {

"xpathFirstChildOf": {

"type": "string"

},

"xpathAfter": {

"type": "string"

},

"xPathAfter": {

"type": "string"

},

"xPathFirstChildOf": {

"type": "string"

},

"nsDecl": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-NsPrefixMappingType"

}

},

"whichDoc": {

"$ref": "#/definitions/dss2-DocumentBaseType"

},

"createEnvelopedSignature": {

"type": "boolean",

"default": "true"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignaturePlacement component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| XPathAfter | xPathAfter | [] |
| XPathFirstChildOf | xPathFirstChildOf | [] |
| NsPrefixMapping | nsDecl | [] |
| WhichDocument | whichDoc | [] |
| CreateEnvelopedSignature | createEnvelopedSignature | [] |

[component SignaturePlacement JSON schema details]

### Component SignedReferences

#### Semantics

[component SignedReferences normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignedReference element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in the core specification in section SignedReference. [sub component SignedReference details]

Non-normative Comment:

[component SignedReferences non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignedReferencesType SHALL implement the requirements defined in the SignedReferences component.

The SignedReferencesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignedReferencesType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="SignedReference" type="dss2:SignedReferenceType"/>

</xs:sequence>

</xs:complexType>

Each child element of SignedReferencesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedReferences XML schema details]

#### JSON Syntax

The SignedReferencesType JSON object SHALL implement in JSON syntax the requirements defined in the SignedReferences component.

The SignedReferencesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignedReferencesType": {

"$xsd-full-type": "dss2:SignedReferencesType",

"type": "object",

"properties": {

"signedRef": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-SignedReferenceType"

}

}

},

"required": ["signedRef"]

}

Properties in the JSON schema above SHALL implement sub-component of SignedReferences component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignedReference | signedRef | [] |

[component SignedReferences JSON schema details]

### Component SignedReference

#### Semantics

[component SignedReference normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Transforms element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Transforms. [sub component Transforms details]
* The WhichDocument element MUST contain one instance of a unique identifier reference. [sub component WhichDocument details]
* The optional RefURI element MUST contain one instance of a URI. [sub component RefURI details]
* The optional RefId element MUST contain one instance of a string. [sub component RefId details]

Non-normative Comment:

[component SignedReference non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignedReferenceType SHALL implement the requirements defined in the SignedReference component.

The SignedReferenceType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignedReferenceType">

<xs:sequence>

<xs:element minOccurs="0" name="Transforms" type="ds-rw:TransformsType"/>

</xs:sequence>

<xs:attribute name="WhichDocument" type="xs:IDREF" use="required"/>

<xs:attribute name="RefURI" type="xs:anyURI" use="optional"/>

<xs:attribute name="RefId" type="xs:string" use="optional"/>

</xs:complexType>

Each child element of SignedReferenceType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignedReference XML schema details]

#### JSON Syntax

The SignedReferenceType JSON object SHALL implement in JSON syntax the requirements defined in the SignedReference component.

The SignedReferenceType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignedReferenceType": {

"$xsd-full-type": "dss2:SignedReferenceType",

"type": "object",

"properties": {

"transforms": {

"$ref": "#/definitions/dsigrw-TransformsType"

},

"whichDoc": {

"$ref": "#/definitions/dss2-DocumentBaseType"

},

"refURI": {

"type": "string"

},

"refId": {

"type": "string"

}

},

"required": ["whichDoc"]

}

Properties in the JSON schema above SHALL implement sub-component of SignedReference component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Transforms | transforms | [] |
| WhichDocument | whichDoc | [] |
| RefURI | refURI | [] |
| RefId | refId | [] |

[component SignedReference JSON schema details]

### Component SignResponse

#### Semantics

[component SignResponse normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional OptionalOutputs element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section OptionalOutputsSign. [sub component OptionalOutputs details]
* The optional SignatureObject element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section SignatureObject. [sub component SignatureObject details]

A set of sub-components is inherited from component ResponseBase and is not repeated here.

Non-normative Comment:

[component SignResponse non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignResponseType SHALL implement the requirements defined in the SignResponse component.

The SignResponseType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignResponseType">

<xs:complexContent>

<xs:extension base="dsb:ResponseBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="OptionalOutputs" type="dss2:OptionalOutputsSignType"/>

<xs:element minOccurs="0" name="SignatureObject" type="dss2:SignatureObjectType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of SignResponseType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignResponse XML schema details]

#### JSON Syntax

The SignResponseType JSON object SHALL implement in JSON syntax the requirements defined in the SignResponse component.

The SignResponseType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignResponseType": {

"$xsd-full-type": "dss2:SignResponseType",

"type": "object",

"properties": {

"result": {

"$ref": "#/definitions/dsb-ResultType"

},

"profile": {

"type": "array",

"items": {

"type": "string"

}

},

"reqID": {

"type": "string"

},

"optOutp": {

"$ref": "#/definitions/dss2-OptionalOutputsSignType"

},

"sigObj": {

"$ref": "#/definitions/dss2-SignatureObjectType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignResponse component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OptionalOutputs | optOutp | [] |
| SignatureObject | sigObj | [] |

[component SignResponse JSON schema details]

### Component OptionalOutputsSign

#### Semantics

[component OptionalOutputsSign normative details]

Below follows a list of the sub-components that MAY be present within this component:

A set of sub-components is inherited from component OptionalOutputsBase and is not repeated here.

Non-normative Comment:

[component OptionalOutputsSign non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type OptionalOutputsSignType SHALL implement the requirements defined in the OptionalOutputsSign component.

The OptionalOutputsSignType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalOutputsSignType">

<xs:complexContent>

<xs:extension base="dss2:OptionalOutputsBaseType"/>

</xs:complexContent>

</xs:complexType>

Each child element of OptionalOutputsSignType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalOutputsSign XML schema details]

#### JSON Syntax

The OptionalOutputsSignType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalOutputsSign component.

The OptionalOutputsSignType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-OptionalOutputsSignType": {

"$xsd-full-type": "dss2:OptionalOutputsSignType",

"type": "object",

"properties": {

"policy": {

"type": "array",

"items": {

"type": "string"

}

},

"other": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"transformed": {

"$ref": "#/definitions/dss2-TransformedDocumentType"

},

"schemas": {

"$ref": "#/definitions/dss2-SchemasType"

},

"docWithSignature": {

"$ref": "#/definitions/dss2-DocumentWithSignatureType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalOutputsSign component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |

[component OptionalOutputsSign JSON schema details]

### Component TransformedDocument

#### Semantics

[component TransformedDocument normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Document element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section Document. [sub component Document details]
* The WhichReference element MUST contain one instance of an integer. [sub component WhichReference details]

Non-normative Comment:

[component TransformedDocument non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type TransformedDocumentType SHALL implement the requirements defined in the TransformedDocument component.

The TransformedDocumentType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="TransformedDocumentType">

<xs:sequence>

<xs:element name="Document" type="dss2:DocumentType"/>

</xs:sequence>

<xs:attribute name="WhichReference" type="xs:integer" use="required"/>

</xs:complexType>

Each child element of TransformedDocumentType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component TransformedDocument XML schema details]

#### JSON Syntax

The TransformedDocumentType JSON object SHALL implement in JSON syntax the requirements defined in the TransformedDocument component.

The TransformedDocumentType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-TransformedDocumentType": {

"$xsd-full-type": "dss2:TransformedDocumentType",

"type": "object",

"properties": {

"doc": {

"$ref": "#/definitions/dss2-DocumentType"

},

"whichRef": {

"type": "integer"

}

},

"required": ["doc", "whichRef"]

}

Properties in the JSON schema above SHALL implement sub-component of TransformedDocument component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Document | doc | [] |
| WhichReference | whichRef | [] |

[component TransformedDocument JSON schema details]

### Component DocumentWithSignature

#### Semantics

[component DocumentWithSignature normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Document element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section Document. [sub component Document details]

Non-normative Comment:

[component DocumentWithSignature non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type DocumentWithSignatureType SHALL implement the requirements defined in the DocumentWithSignature component.

The DocumentWithSignatureType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DocumentWithSignatureType">

<xs:sequence>

<xs:element name="Document" type="dss2:DocumentType"/>

</xs:sequence>

</xs:complexType>

Each child element of DocumentWithSignatureType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DocumentWithSignature XML schema details]

#### JSON Syntax

The DocumentWithSignatureType JSON object SHALL implement in JSON syntax the requirements defined in the DocumentWithSignature component.

The DocumentWithSignatureType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-DocumentWithSignatureType": {

"$xsd-full-type": "dss2:DocumentWithSignatureType",

"type": "object",

"properties": {

"doc": {

"$ref": "#/definitions/dss2-DocumentType"

}

},

"required": ["doc"]

}

Properties in the JSON schema above SHALL implement sub-component of DocumentWithSignature component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Document | doc | [] |

[component DocumentWithSignature JSON schema details]

### Component SignatureObject

#### Semantics

[component SignatureObject normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Base64Signature element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Base64Data. [sub component Base64Signature details]
* The SignaturePtr element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section SignaturePtr. [sub component SignaturePtr details]
* The optional SchemaRefs element MUST contain one instance of a unique identifier reference. [sub component SchemaRefs details]

Non-normative Comment:

[component SignatureObject non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignatureObjectType SHALL implement the requirements defined in the SignatureObject component.

The SignatureObjectType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignatureObjectType">

<xs:sequence>

<xs:choice>

<xs:element name="Base64Signature" type="dsb:Base64DataType"/>

<xs:element name="SignaturePtr" type="dss2:SignaturePtrType"/>

</xs:choice>

</xs:sequence>

<xs:attribute name="SchemaRefs" type="xs:IDREFS" use="optional"/>

</xs:complexType>

Each child element of SignatureObjectType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignatureObject XML schema details]

#### JSON Syntax

The SignatureObjectType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureObject component.

The SignatureObjectType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignatureObjectType": {

"$xsd-full-type": "dss2:SignatureObjectType",

"type": "object",

"properties": {

"b64Sig": {

"$ref": "#/definitions/dsb-Base64DataType"

},

"sigPtr": {

"$ref": "#/definitions/dss2-SignaturePtrType"

},

"schemaRefs": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DocumentBaseType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignatureObject component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Base64Signature | b64Sig | [] |
| SignaturePtr | sigPtr | [] |
| SchemaRefs | schemaRefs | [] |

[component SignatureObject JSON schema details]

### Component SignaturePtr

#### Semantics

[component SignaturePtr normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional NsPrefixMapping element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section NsPrefixMapping. [sub component NsPrefixMapping details]
* The WhichDocument element MUST contain one instance of a unique identifier reference. [sub component WhichDocument details]
* The optional XPath element MUST contain one instance of a string. [sub component XPath details]

Non-normative Comment:

[component SignaturePtr non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SignaturePtrType SHALL implement the requirements defined in the SignaturePtr component.

The SignaturePtrType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignaturePtrType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="NsPrefixMapping" type="dsb:NsPrefixMappingType"/>

</xs:sequence>

<xs:attribute name="WhichDocument" type="xs:IDREF" use="required"/>

<xs:attribute name="XPath" type="xs:string" use="optional"/>

</xs:complexType>

Each child element of SignaturePtrType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignaturePtr XML schema details]

#### JSON Syntax

The SignaturePtrType JSON object SHALL implement in JSON syntax the requirements defined in the SignaturePtr component.

The SignaturePtrType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SignaturePtrType": {

"$xsd-full-type": "dss2:SignaturePtrType",

"type": "object",

"properties": {

"xpath": {

"type": "string"

},

"nsDecl": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-NsPrefixMappingType"

}

},

"whichDoc": {

"$ref": "#/definitions/dss2-DocumentBaseType"

},

"xPath": {

"type": "string"

}

},

"required": ["whichDoc"]

}

Properties in the JSON schema above SHALL implement sub-component of SignaturePtr component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| NsPrefixMapping | nsDecl | [] |
| WhichDocument | whichDoc | [] |
| XPath | xPath | [] |

[component SignaturePtr JSON schema details]

### Component VerifyRequest

#### Semantics

[component VerifyRequest normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional InputDocuments element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section InputDocuments. [sub component InputDocuments details]
* The optional OptionalInputs element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section OptionalInputsVerify. [sub component OptionalInputs details]
* The optional SignatureObject element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section SignatureObject. [sub component SignatureObject details]

A set of sub-components is inherited from component RequestBase and is not repeated here.

Non-normative Comment:

[component VerifyRequest non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type VerifyRequestType SHALL implement the requirements defined in the VerifyRequest component.

The VerifyRequestType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VerifyRequestType">

<xs:complexContent>

<xs:extension base="dsb:RequestBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="InputDocuments" type="dss2:InputDocumentsType"/>

<xs:element minOccurs="0" name="OptionalInputs" type="dss2:OptionalInputsVerifyType"/>

<xs:element minOccurs="0" name="SignatureObject" type="dss2:SignatureObjectType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of VerifyRequestType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerifyRequest XML schema details]

#### JSON Syntax

The VerifyRequestType JSON object SHALL implement in JSON syntax the requirements defined in the VerifyRequest component.

The VerifyRequestType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-VerifyRequestType": {

"$xsd-full-type": "dss2:VerifyRequestType",

"type": "object",

"properties": {

"profile": {

"type": "array",

"items": {

"type": "string"

}

},

"reqID": {

"type": "string"

},

"inDocs": {

"$ref": "#/definitions/dss2-InputDocumentsType"

},

"optInp": {

"$ref": "#/definitions/dss2-OptionalInputsVerifyType"

},

"sigObj": {

"$ref": "#/definitions/dss2-SignatureObjectType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of VerifyRequest component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| InputDocuments | inDocs | [] |
| OptionalInputs | optInp | [] |
| SignatureObject | sigObj | [] |

[component VerifyRequest JSON schema details]

### Component OptionalInputsVerify

#### Semantics

[component OptionalInputsVerify normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional UseVerificationTime element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section UseVerificationTime. [sub component UseVerificationTime details]
* The optional ReturnVerificationTimeInfo element MUST contain a boolean. Its default value is 'false'. [sub component ReturnVerificationTimeInfo details]
* The optional AdditionalKeyInfo element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section AdditionalKeyInfo. [sub component AdditionalKeyInfo details]
* The optional ReturnProcessingDetails element MUST contain a boolean. Its default value is 'false'. [sub component ReturnProcessingDetails details]
* The optional ReturnSigningTimeInfo element MUST contain a boolean. Its default value is 'false'. [sub component ReturnSigningTimeInfo details]
* The optional ReturnSignerIdentity element MUST contain a boolean. Its default value is 'false'. [sub component ReturnSignerIdentity details]
* The optional ReturnUpdatedSignature element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section UpdateSignatureInstruction. [sub component ReturnUpdatedSignature details]
* The optional ReturnTransformedDocument element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section ReturnTransformedDocument. [sub component ReturnTransformedDocument details]
* The optional ReturnTimestampedSignature element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section UpdateSignatureInstruction. [sub component ReturnTimestampedSignature details]
* The optional VerifyManifests element MUST contain a boolean. Its default value is 'false'. [sub component VerifyManifests details]
* The VRProfileOptionalInput element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section OptionalInput. [sub component VRProfileOptionalInput details]
* The JWSProfileInputs element MUST contain one instance of a sub\_component . This element MUST satisfy the requirements specified in section . [sub component JWSProfileInputs details]

A set of sub-components is inherited from component OptionalInputsBase and is not repeated here.

Non-normative Comment:

[component OptionalInputsVerify non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type OptionalInputsVerifyType SHALL implement the requirements defined in the OptionalInputsVerify component.

The OptionalInputsVerifyType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalInputsVerifyType">

<xs:complexContent>

<xs:extension base="dss2:OptionalInputsBaseType">

<xs:sequence>

<xs:choice>

<xs:element maxOccurs="1" minOccurs="0" name="UseVerificationTime" type="dss2:UseVerificationTimeType"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="ReturnVerificationTimeInfo" type="xs:boolean"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AdditionalKeyInfo" type="dss2:AdditionalKeyInfoType"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="ReturnProcessingDetails" type="xs:boolean"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="ReturnSigningTimeInfo" type="xs:boolean"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="ReturnSignerIdentity" type="xs:boolean"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ReturnUpdatedSignature" type="dss2:UpdateSignatureInstructionType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ReturnTransformedDocument" type="dss2:ReturnTransformedDocumentType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ReturnTimestampedSignature" type="dss2:UpdateSignatureInstructionType"/>

<xs:element default="false" maxOccurs="1" minOccurs="0" name="VerifyManifests" type="xs:boolean"/>

<xs:element name="VRProfileOptionalInput" type="vr:OptionalInputType"/>

<xs:element name="JWSProfileInputs" type="jws:OptionalInputVerifyType"/>

</xs:choice>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of OptionalInputsVerifyType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalInputsVerify XML schema details]

#### JSON Syntax

The OptionalInputsVerifyType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalInputsVerify component.

The OptionalInputsVerifyType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-OptionalInputsVerifyType": {

"$xsd-full-type": "dss2:OptionalInputsVerifyType",

"type": "object",

"properties": {

"jwsprofileInputs": {

"$ref": "#/definitions/async-OptionalInputVerifyType"

},

"vrprofileOptionalInput": {

"$ref": "#/definitions/vr-OptionalInputType"

},

"policy": {

"type": "array",

"items": {

"type": "string"

}

},

"lang": {

"type": "string"

},

"other": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"claimedIdentity": {

"$ref": "#/definitions/dss2-ClaimedIdentityType"

},

"schemas": {

"$ref": "#/definitions/dss2-SchemasType"

},

"addTimestamp": {

"$ref": "#/definitions/dss2-UpdateSignatureInstructionType"

},

"useVerificationTime": {

"$ref": "#/definitions/dss2-UseVerificationTimeType"

},

"returnVerificationTime": {

"type": "boolean",

"default": "false"

},

"addKeyInfo": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-AdditionalKeyInfoType"

}

},

"returnProcDetails": {

"type": "boolean",

"default": "false"

},

"returnSigningTime": {

"type": "boolean",

"default": "false"

},

"returnSigner": {

"type": "boolean",

"default": "false"

},

"returnUpdated": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-UpdateSignatureInstructionType"

}

},

"returnTransformed": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-ReturnTransformedDocumentType"

}

},

"returnTimestamped": {

"$ref": "#/definitions/dss2-UpdateSignatureInstructionType"

},

"verifyManifests": {

"type": "boolean",

"default": "false"

},

"optVR": {

"$ref": "#/definitions/vr-OptionalInputType"

},

"optJws": {

"$ref": "#/definitions/async-OptionalInputVerifyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalInputsVerify component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| UseVerificationTime | useVerificationTime | [] |
| ReturnVerificationTimeInfo | returnVerificationTime | [] |
| AdditionalKeyInfo | addKeyInfo | [] |
| ReturnProcessingDetails | returnProcDetails | [] |
| ReturnSigningTimeInfo | returnSigningTime | [] |
| ReturnSignerIdentity | returnSigner | [] |
| ReturnUpdatedSignature | returnUpdated | [] |
| ReturnTransformedDocument | returnTransformed | [] |
| ReturnTimestampedSignature | returnTimestamped | [] |
| VerifyManifests | verifyManifests | [] |
| VRProfileOptionalInput | optVR | [] |
| JWSProfileInputs | optJws | [] |

[component OptionalInputsVerify JSON schema details]

### Component UseVerificationTime

#### Semantics

[component UseVerificationTime normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CurrentTime element MUST contain one instance of a boolean. Its default value is 'false'. [sub component CurrentTime details]
* The SpecificTime element MUST contain one instance of a date/time value. [sub component SpecificTime details]
* The optional Base64Content element MUST contain base64 encoded binary data. [sub component Base64Content details]

Non-normative Comment:

[component UseVerificationTime non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type UseVerificationTimeType SHALL implement the requirements defined in the UseVerificationTime component.

The UseVerificationTimeType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="UseVerificationTimeType">

<xs:choice>

<xs:element default="false" name="CurrentTime" type="xs:boolean"/>

<xs:element name="SpecificTime" type="xs:dateTime"/>

<xs:element maxOccurs="1" minOccurs="0" name="Base64Content" type="xs:base64Binary"/>

</xs:choice>

</xs:complexType>

Each child element of UseVerificationTimeType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UseVerificationTime XML schema details]

#### JSON Syntax

The UseVerificationTimeType JSON object SHALL implement in JSON syntax the requirements defined in the UseVerificationTime component.

The UseVerificationTimeType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-UseVerificationTimeType": {

"$xsd-full-type": "dss2:UseVerificationTimeType",

"type": "object",

"properties": {

"currTime": {

"type": "boolean",

"default": "false"

},

"specTime": {

"type": "integer",

"format": "utc-millisec"

},

"b64Content": {

"type": "string"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of UseVerificationTime component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CurrentTime | currTime | [] |
| SpecificTime | specTime | [] |
| Base64Content | b64Content | [] |

[component UseVerificationTime JSON schema details]

### Component AdditionalKeyInfo

#### Semantics

[component AdditionalKeyInfo normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The X509Digest element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section X509Digest. [sub component X509Digest details]
* The X509SubjectName element MUST contain one instance of a string. [sub component X509SubjectName details]
* The X509SKI element MUST contain one instance of base64 encoded binary data. [sub component X509SKI details]
* The X509Certificate element MUST contain one instance of base64 encoded binary data. [sub component X509Certificate details]
* The KeyName element MUST contain one instance of a string. [sub component KeyName details]
* The X509CRL element MUST contain one instance of base64 encoded binary data. [sub component X509CRL details]

Non-normative Comment:

[component AdditionalKeyInfo non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type AdditionalKeyInfoType SHALL implement the requirements defined in the AdditionalKeyInfo component.

The AdditionalKeyInfoType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AdditionalKeyInfoType">

<xs:choice>

<xs:element name="X509Digest" type="dss2:X509DigestType"/>

<xs:element name="X509SubjectName" type="xs:string"/>

<xs:element name="X509SKI" type="xs:base64Binary"/>

<xs:element name="X509Certificate" type="xs:base64Binary"/>

<xs:element name="KeyName" type="xs:string"/>

<xs:element name="X509CRL" type="xs:base64Binary"/>

</xs:choice>

</xs:complexType>

Each child element of AdditionalKeyInfoType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AdditionalKeyInfo XML schema details]

#### JSON Syntax

The AdditionalKeyInfoType JSON object SHALL implement in JSON syntax the requirements defined in the AdditionalKeyInfo component.

The AdditionalKeyInfoType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-AdditionalKeyInfoType": {

"$xsd-full-type": "dss2:AdditionalKeyInfoType",

"type": "object",

"properties": {

"x509Digest": {

"$ref": "#/definitions/dss2-X509DigestType"

},

"subject": {

"type": "string"

},

"ski": {

"type": "string"

},

"cert": {

"type": "string"

},

"name": {

"type": "string"

},

"X509CRL": {

"type": "string"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of AdditionalKeyInfo component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| X509Digest | x509Digest | [] |
| X509SubjectName | subject | [] |
| X509SKI | ski | [] |
| X509Certificate | cert | [] |
| KeyName | name | [] |
| X509CRL | X509CRL | [] |

[component AdditionalKeyInfo JSON schema details]

### Component ReturnTransformedDocument

#### Semantics

[component ReturnTransformedDocument normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The WhichReference element MUST contain one instance of an integer. [sub component WhichReference details]

Non-normative Comment:

[component ReturnTransformedDocument non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type ReturnTransformedDocumentType SHALL implement the requirements defined in the ReturnTransformedDocument component.

The ReturnTransformedDocumentType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ReturnTransformedDocumentType">

<xs:attribute name="WhichReference" type="xs:integer" use="required"/>

</xs:complexType>

Each child element of ReturnTransformedDocumentType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ReturnTransformedDocument XML schema details]

#### JSON Syntax

The ReturnTransformedDocumentType JSON object SHALL implement in JSON syntax the requirements defined in the ReturnTransformedDocument component.

The ReturnTransformedDocumentType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-ReturnTransformedDocumentType": {

"$xsd-full-type": "dss2:ReturnTransformedDocumentType",

"type": "object",

"properties": {

"whichRef": {

"type": "integer"

}

},

"required": ["whichRef"]

}

Properties in the JSON schema above SHALL implement sub-component of ReturnTransformedDocument component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| WhichReference | whichRef | [] |

[component ReturnTransformedDocument JSON schema details]

### Component VerifyResponse

#### Semantics

[component VerifyResponse normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional OptionalOutputs element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section OptionalOutputsVerify. [sub component OptionalOutputs details]

A set of sub-components is inherited from component ResponseBase and is not repeated here.

Non-normative Comment:

[component VerifyResponse non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type VerifyResponseType SHALL implement the requirements defined in the VerifyResponse component.

The VerifyResponseType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VerifyResponseType">

<xs:complexContent>

<xs:extension base="dsb:ResponseBaseType">

<xs:sequence>

<xs:element minOccurs="0" name="OptionalOutputs" type="dss2:OptionalOutputsVerifyType"/>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of VerifyResponseType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerifyResponse XML schema details]

#### JSON Syntax

The VerifyResponseType JSON object SHALL implement in JSON syntax the requirements defined in the VerifyResponse component.

The VerifyResponseType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-VerifyResponseType": {

"$xsd-full-type": "dss2:VerifyResponseType",

"type": "object",

"properties": {

"result": {

"$ref": "#/definitions/dsb-ResultType"

},

"profile": {

"type": "array",

"items": {

"type": "string"

}

},

"reqID": {

"type": "string"

},

"optOutp": {

"$ref": "#/definitions/dss2-OptionalOutputsVerifyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of VerifyResponse component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OptionalOutputs | optOutp | [] |

[component VerifyResponse JSON schema details]

### Component OptionalOutputsVerify

#### Semantics

[component OptionalOutputsVerify normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional VerifyManifestResults element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerifyManifestResults. [sub component VerifyManifestResults details]
* The optional SigningTimeInfo element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section SigningTimeInfo. [sub component SigningTimeInfo details]
* The optional VerificationTimeInfo element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section VerificationTimeInfo. [sub component VerificationTimeInfo details]
* The optional ProcessingDetails element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section ProcessingDetails. [sub component ProcessingDetails details]
* The optional SignerIdentity element MUST contain sub-component. A given element MUST satisfy the requirements specified in section NameID. [sub component SignerIdentity details]
* The optional UpdatedSignature element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section UpdatedSignature. [sub component UpdatedSignature details]
* The optional TimestampedSignature element MUST contain sub-component. A given element MUST satisfy the requirements specified in the core specification in section UpdatedSignature. [sub component TimestampedSignature details]
* The VRProfileOptionalOutput element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section OptionalOutput. [sub component VRProfileOptionalOutput details]

A set of sub-components is inherited from component OptionalOutputsBase and is not repeated here.

Non-normative Comment:

[component OptionalOutputsVerify non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type OptionalOutputsVerifyType SHALL implement the requirements defined in the OptionalOutputsVerify component.

The OptionalOutputsVerifyType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="OptionalOutputsVerifyType">

<xs:complexContent>

<xs:extension base="dss2:OptionalOutputsBaseType">

<xs:sequence>

<xs:choice>

<xs:element maxOccurs="1" minOccurs="0" name="VerifyManifestResults" type="dss2:VerifyManifestResultsType"/>

<xs:element maxOccurs="1" minOccurs="0" name="SigningTimeInfo" type="dss2:SigningTimeInfoType"/>

<xs:element maxOccurs="1" minOccurs="0" name="VerificationTimeInfo" type="dss2:VerificationTimeInfoType"/>

<xs:element maxOccurs="1" minOccurs="0" name="ProcessingDetails" type="dss2:ProcessingDetailsType"/>

<xs:element maxOccurs="1" minOccurs="0" name="SignerIdentity" type="saml2-rw:NameIDType"/>

<xs:element maxOccurs="1" minOccurs="0" name="UpdatedSignature" type="dss2:UpdatedSignatureType"/>

<xs:element maxOccurs="1" minOccurs="0" name="TimestampedSignature" type="dss2:UpdatedSignatureType"/>

<xs:element name="VRProfileOptionalOutput" type="vr:OptionalOutputType"/>

</xs:choice>

</xs:sequence>

</xs:extension>

</xs:complexContent>

</xs:complexType>

Each child element of OptionalOutputsVerifyType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OptionalOutputsVerify XML schema details]

#### JSON Syntax

The OptionalOutputsVerifyType JSON object SHALL implement in JSON syntax the requirements defined in the OptionalOutputsVerify component.

The OptionalOutputsVerifyType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-OptionalOutputsVerifyType": {

"$xsd-full-type": "dss2:OptionalOutputsVerifyType",

"type": "object",

"properties": {

"vrprofileOptionalOutput": {

"$ref": "#/definitions/vr-OptionalOutputType"

},

"policy": {

"type": "array",

"items": {

"type": "string"

}

},

"other": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-AnyType"

}

},

"transformed": {

"$ref": "#/definitions/dss2-TransformedDocumentType"

},

"schemas": {

"$ref": "#/definitions/dss2-SchemasType"

},

"docWithSignature": {

"$ref": "#/definitions/dss2-DocumentWithSignatureType"

},

"result": {

"$ref": "#/definitions/dss2-VerifyManifestResultsType"

},

"signingTimeInfo": {

"$ref": "#/definitions/dss2-SigningTimeInfoType"

},

"verificationTimeInfo": {

"$ref": "#/definitions/dss2-VerificationTimeInfoType"

},

"procDetails": {

"$ref": "#/definitions/dss2-ProcessingDetailsType"

},

"signerIdentity": {

"$ref": "#/definitions/saml2rw-NameIDType"

},

"updSignature": {

"$ref": "#/definitions/dss2-UpdatedSignatureType"

},

"timestampedSignature": {

"$ref": "#/definitions/dss2-UpdatedSignatureType"

},

"optVR": {

"$ref": "#/definitions/vr-OptionalOutputType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OptionalOutputsVerify component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| VerifyManifestResults | result | [] |
| SigningTimeInfo | signingTimeInfo | [] |
| VerificationTimeInfo | verificationTimeInfo | [] |
| ProcessingDetails | procDetails | [] |
| SignerIdentity | signerIdentity | [] |
| UpdatedSignature | updSignature | [] |
| TimestampedSignature | timestampedSignature | [] |
| VRProfileOptionalOutput | optVR | [] |

[component OptionalOutputsVerify JSON schema details]

### Component VerificationTimeInfo

#### Semantics

[component VerificationTimeInfo normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The VerificationTime element MUST contain one instance of a date/time value. [sub component VerificationTime details]
* The optional AdditionalTimeInfo element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section AdditionalTimeInfo. [sub component AdditionalTimeInfo details]

Non-normative Comment:

[component VerificationTimeInfo non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type VerificationTimeInfoType SHALL implement the requirements defined in the VerificationTimeInfo component.

The VerificationTimeInfoType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VerificationTimeInfoType">

<xs:sequence>

<xs:element name="VerificationTime" type="xs:dateTime"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AdditionalTimeInfo" type="dss2:AdditionalTimeInfoType"/>

</xs:sequence>

</xs:complexType>

Each child element of VerificationTimeInfoType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerificationTimeInfo XML schema details]

#### JSON Syntax

The VerificationTimeInfoType JSON object SHALL implement in JSON syntax the requirements defined in the VerificationTimeInfo component.

The VerificationTimeInfoType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-VerificationTimeInfoType": {

"$xsd-full-type": "dss2:VerificationTimeInfoType",

"type": "object",

"properties": {

"verificationTime": {

"type": "integer",

"format": "utc-millisec"

},

"additionalTimeInfo": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-AdditionalTimeInfoType"

}

}

},

"required": ["verificationTime"]

}

Properties in the JSON schema above SHALL implement sub-component of VerificationTimeInfo component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| VerificationTime | verificationTime | [] |
| AdditionalTimeInfo | additionalTimeInfo | [] |

[component VerificationTimeInfo JSON schema details]

### Component AdditionalTimeInfo

#### Semantics

[component AdditionalTimeInfo normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a date/time value. [sub component value details]
* The Type element MUST contain one instance of a URI. Its value is limited to an item of the following set:  
  urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signatureTimestamp  
  urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signatureTimemark  
  urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signedObjectTimestamp  
  urn:oasis:names:tc:dss:1.0:additionaltimeinfo:claimedSigningTime  
  [sub component Type details]
* The optional Ref element MUST contain one instance of a string. [sub component Ref details]

Non-normative Comment:

[component AdditionalTimeInfo non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type AdditionalTimeInfoType SHALL implement the requirements defined in the AdditionalTimeInfo component.

The AdditionalTimeInfoType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AdditionalTimeInfoType">

<xs:simpleContent>

<xs:extension base="xs:dateTime">

<xs:attribute name="Type" use="required">

<xs:simpleType>

<xs:restriction base="xs:anyURI">

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signatureTimestamp"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signatureTimemark"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:additionaltimeinfo:signedObjectTimestamp"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:additionaltimeinfo:claimedSigningTime"/>

</xs:restriction>

</xs:simpleType>

</xs:attribute>

<xs:attribute name="Ref" type="xs:string" use="optional"/>

</xs:extension>

</xs:simpleContent>

</xs:complexType>

Each child element of AdditionalTimeInfoType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' holding a date/time is represented by the component's XML tag text content. [component AdditionalTimeInfo XML schema details]

#### JSON Syntax

The AdditionalTimeInfoType JSON object SHALL implement in JSON syntax the requirements defined in the AdditionalTimeInfo component.

The AdditionalTimeInfoType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-AdditionalTimeInfoType": {

"$xsd-full-type": "dss2:AdditionalTimeInfoType",

"type": "object",

"properties": {

"value": {

"type": "integer",

"format": "utc-millisec"

},

"type": {

"type": "string",

"format": "uri"

},

"ref": {

"type": "string"

}

},

"required": ["type"]

}

Properties in the JSON schema above SHALL implement sub-component of AdditionalTimeInfo component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Type | type | [] |
| Ref | ref | [] |

[component AdditionalTimeInfo JSON schema details]

### Component VerifyManifestResults

#### Semantics

[component VerifyManifestResults normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ManifestResult element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in the core specification in section ManifestResult. [sub component ManifestResult details]

Non-normative Comment:

[component VerifyManifestResults non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type VerifyManifestResultsType SHALL implement the requirements defined in the VerifyManifestResults component.

The VerifyManifestResultsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VerifyManifestResultsType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="ManifestResult" type="dss2:ManifestResultType"/>

</xs:sequence>

</xs:complexType>

Each child element of VerifyManifestResultsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VerifyManifestResults XML schema details]

#### JSON Syntax

The VerifyManifestResultsType JSON object SHALL implement in JSON syntax the requirements defined in the VerifyManifestResults component.

The VerifyManifestResultsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-VerifyManifestResultsType": {

"$xsd-full-type": "dss2:VerifyManifestResultsType",

"type": "object",

"properties": {

"signedRef": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-ManifestResultType"

}

}

},

"required": ["signedRef"]

}

Properties in the JSON schema above SHALL implement sub-component of VerifyManifestResults component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ManifestResult | signedRef | [] |

[component VerifyManifestResults JSON schema details]

### Component ManifestResult

#### Semantics

[component ManifestResult normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ReferenceXpath element MUST contain one instance of a string. [sub component ReferenceXpath details]
* The Status element MUST contain one instance of a URI. Its value is limited to an item of the following set:  
  urn:oasis:names:tc:dss:1.0:manifeststatus:Valid  
  urn:oasis:names:tc:dss:1.0:manifeststatus:Invalid  
  [sub component Status details]
* The optional NsPrefixMapping element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section NsPrefixMapping. [sub component NsPrefixMapping details]

Non-normative Comment:

[component ManifestResult non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type ManifestResultType SHALL implement the requirements defined in the ManifestResult component.

The ManifestResultType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ManifestResultType">

<xs:sequence>

<xs:element name="ReferenceXpath" type="xs:string"/>

<xs:element name="Status">

<xs:simpleType>

<xs:restriction base="xs:anyURI">

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:manifeststatus:Valid"/>

<xs:enumeration value="urn:oasis:names:tc:dss:1.0:manifeststatus:Invalid"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

<xs:element maxOccurs="unbounded" minOccurs="0" name="NsPrefixMapping" type="dsb:NsPrefixMappingType"/>

</xs:sequence>

</xs:complexType>

Each child element of ManifestResultType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ManifestResult XML schema details]

#### JSON Syntax

The ManifestResultType JSON object SHALL implement in JSON syntax the requirements defined in the ManifestResult component.

The ManifestResultType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-ManifestResultType": {

"$xsd-full-type": "dss2:ManifestResultType",

"type": "object",

"properties": {

"xPath": {

"type": "string"

},

"status": {

"type": "string",

"enum": ["urn:oasis:names:tc:dss:1.0:manifeststatus:Valid", "urn:oasis:names:tc:dss:1.0:manifeststatus:Invalid"]

},

"nsDecl": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-NsPrefixMappingType"

}

}

},

"required": ["xPath", "status"]

}

Properties in the JSON schema above SHALL implement sub-component of ManifestResult component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ReferenceXpath | xPath | [] |
| Status | status | [] |
| NsPrefixMapping | nsDecl | [] |

[component ManifestResult JSON schema details]

### Component SigningTimeInfo

#### Semantics

[component SigningTimeInfo normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SigningTime element MUST contain one instance of a date/time value. [sub component SigningTime details]
* The optional SigningTimeBoundaries element MUST contain sub-components. [sub component SigningTimeBoundaries details]
* The optional LowerBoundary element MUST contain a date/time value. [sub component LowerBoundary details]
* The optional UpperBoundary element MUST contain a date/time value. [sub component UpperBoundary details]

Non-normative Comment:

[component SigningTimeInfo non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type SigningTimeInfoType SHALL implement the requirements defined in the SigningTimeInfo component.

The SigningTimeInfoType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SigningTimeInfoType">

<xs:sequence>

<xs:element name="SigningTime" type="xs:dateTime"/>

<xs:element minOccurs="0" name="SigningTimeBoundaries">

<xs:complexType>

<xs:sequence>

<xs:element minOccurs="0" name="LowerBoundary" type="xs:dateTime"/>

<xs:element minOccurs="0" name="UpperBoundary" type="xs:dateTime"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

Each child element of SigningTimeInfoType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SigningTimeInfo XML schema details]

#### JSON Syntax

The SigningTimeInfoType JSON object SHALL implement in JSON syntax the requirements defined in the SigningTimeInfo component.

The SigningTimeInfoType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-SigningTimeInfoType": {

"$xsd-full-type": "dss2:SigningTimeInfoType",

"type": "object",

"properties": {

"signingTime": {

"type": "integer",

"format": "utc-millisec"

},

"signingTimeBounds": {

"$ref": "#/definitions/dss2-SigningTimeInfoType:SigningTimeBoundaries"

}

},

"required": ["signingTime"]

}

"dss2-SigningTimeInfoType:SigningTimeBoundaries": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"lowerBound": {

"type": "integer",

"format": "utc-millisec"

},

"upperBound": {

"type": "integer",

"format": "utc-millisec"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SigningTimeInfo component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigningTime | signingTime | [] |
| SigningTimeBoundaries | signingTimeBounds | [] |
| LowerBoundary | lowerBound | [] |
| UpperBoundary | upperBound | [] |

[component SigningTimeInfo JSON schema details]

### Component ProcessingDetails

#### Semantics

[component ProcessingDetails normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional ValidDetail element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section Detail. [sub component ValidDetail details]
* The optional IndeterminateDetail element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section Detail. [sub component IndeterminateDetail details]
* The optional InvalidDetail element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in the core specification in section Detail. [sub component InvalidDetail details]

Non-normative Comment:

[component ProcessingDetails non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type ProcessingDetailsType SHALL implement the requirements defined in the ProcessingDetails component.

The ProcessingDetailsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ProcessingDetailsType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ValidDetail" type="dss2:DetailType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="IndeterminateDetail" type="dss2:DetailType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="InvalidDetail" type="dss2:DetailType"/>

</xs:sequence>

</xs:complexType>

Each child element of ProcessingDetailsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ProcessingDetails XML schema details]

#### JSON Syntax

The ProcessingDetailsType JSON object SHALL implement in JSON syntax the requirements defined in the ProcessingDetails component.

The ProcessingDetailsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-ProcessingDetailsType": {

"$xsd-full-type": "dss2:ProcessingDetailsType",

"type": "object",

"properties": {

"valid": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DetailType"

}

},

"indeterminate": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DetailType"

}

},

"invalid": {

"type": "array",

"items": {

"$ref": "#/definitions/dss2-DetailType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ProcessingDetails component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidDetail | valid | [] |
| IndeterminateDetail | indeterminate | [] |
| InvalidDetail | invalid | [] |

[component ProcessingDetails JSON schema details]

### Component Detail

#### Semantics

[component Detail normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Code element MUST contain a URI. [sub component Code details]
* The optional Message element MUST contain sub-component. A given element MUST satisfy the requirements specified in section InternationalString. [sub component Message details]
* The optional Base64Content element MUST contain base64 encoded binary data. [sub component Base64Content details]
* The Type element MUST contain one instance of a URI. [sub component Type details]

Non-normative Comment:

[component Detail non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type DetailType SHALL implement the requirements defined in the Detail component.

The DetailType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DetailType">

<xs:sequence>

<xs:element minOccurs="0" name="Code" type="xs:anyURI"/>

<xs:element minOccurs="0" name="Message" type="dsb:InternationalStringType"/>

<xs:element maxOccurs="1" minOccurs="0" name="Base64Content" type="xs:base64Binary"/>

</xs:sequence>

<xs:attribute name="Type" type="xs:anyURI" use="required"/>

</xs:complexType>

Each child element of DetailType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Detail XML schema details]

#### JSON Syntax

The DetailType JSON object SHALL implement in JSON syntax the requirements defined in the Detail component.

The DetailType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-DetailType": {

"$xsd-full-type": "dss2:DetailType",

"type": "object",

"properties": {

"code": {

"type": "string"

},

"msg": {

"$ref": "#/definitions/dsb-InternationalStringType"

},

"b64Content": {

"type": "string"

},

"type": {

"type": "string",

"format": "uri"

}

},

"required": ["type"]

}

Properties in the JSON schema above SHALL implement sub-component of Detail component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Code | code | [] |
| Message | msg | [] |
| Base64Content | b64Content | [] |
| Type | type | [] |

[component Detail JSON schema details]

### Component UpdatedSignature

#### Semantics

[component UpdatedSignature normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignatureObject element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in the core specification in section SignatureObject. [sub component SignatureObject details]
* The optional Type element MUST contain one instance of a URI. [sub component Type details]

Non-normative Comment:

[component UpdatedSignature non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/core' .The XML type UpdatedSignatureType SHALL implement the requirements defined in the UpdatedSignature component.

The UpdatedSignatureType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="UpdatedSignatureType">

<xs:sequence>

<xs:element name="SignatureObject" type="dss2:SignatureObjectType"/>

</xs:sequence>

<xs:attribute name="Type" type="xs:anyURI" use="optional"/>

</xs:complexType>

Each child element of UpdatedSignatureType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UpdatedSignature XML schema details]

#### JSON Syntax

The UpdatedSignatureType JSON object SHALL implement in JSON syntax the requirements defined in the UpdatedSignature component.

The UpdatedSignatureType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dss2-UpdatedSignatureType": {

"$xsd-full-type": "dss2:UpdatedSignatureType",

"type": "object",

"properties": {

"sigObj": {

"$ref": "#/definitions/dss2-SignatureObjectType"

},

"type": {

"type": "string",

"format": "uri"

}

},

"required": ["sigObj"]

}

Properties in the JSON schema above SHALL implement sub-component of UpdatedSignature component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignatureObject | sigObj | [] |
| Type | type | [] |

[component UpdatedSignature JSON schema details]

## Referenced Structure Models from other documents

### Component NameIdentifier

#### Semantics

[component NameIdentifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a string. [sub component value details]
* The optional NameQualifier element MUST contain one instance of a string. [sub component NameQualifier details]
* The optional Format element MUST contain one instance of a URI. [sub component Format details]

Non-normative Comment:

[component NameIdentifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/SAML\_1.0/assertion/rewritten' .The XML type NameIdentifierType SHALL implement the requirements defined in the NameIdentifier component.

The NameIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="NameIdentifierType">

<simpleContent>

<extension base="string">

<attribute name="NameQualifier" type="string" use="optional"/>

<attribute name="Format" type="anyURI" use="optional"/>

</extension>

</simpleContent>

</complexType>

Each child element of NameIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' is represented by the component's XML tag text content. [component NameIdentifier XML schema details]

#### JSON Syntax

The NameIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the NameIdentifier component.

The NameIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"samlrw-NameIdentifierType": {

"$xsd-full-type": "saml-rw:NameIdentifierType",

"type": "object",

"properties": {

"value": {

"type": "string"

},

"NameQualifier": {

"type": "string"

},

"Format": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of NameIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| NameQualifier |  | [] |
| Format |  | [] |

[component NameIdentifier JSON schema details]

### Component NameID

#### Semantics

[component NameID normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a string. [sub component value details]
* The optional Format element MUST contain one instance of a URI. [sub component Format details]
* The optional SPProvidedID element MUST contain one instance of a string. [sub component SPProvidedID details]
* The optional NameQualifier element MUST contain one instance of a string. [sub component NameQualifier details]
* The optional SPNameQualifier element MUST contain one instance of a string. [sub component SPNameQualifier details]

Non-normative Comment:

[component NameID non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/saml2/rewritten' .The XML type NameIDType SHALL implement the requirements defined in the NameID component.

The NameIDType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="NameIDType">

<simpleContent>

<extension base="string">

<attributeGroup ref="saml2-rw:IDNameQualifiers"/>

<xs:attribute name="Format" type="anyURI" use="optional"/>

<xs:attribute name="SPProvidedID" type="string" use="optional"/>

</extension>

</simpleContent>

</complexType>

Each child element of NameIDType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' is represented by the component's XML tag text content. [component NameID XML schema details]

#### JSON Syntax

The NameIDType JSON object SHALL implement in JSON syntax the requirements defined in the NameID component.

The NameIDType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"saml2rw-NameIDType": {

"$xsd-full-type": "saml2-rw:NameIDType",

"type": "object",

"properties": {

"spnameQualifier": {

"type": "string"

},

"spprovidedID": {

"type": "string"

},

"value": {

"type": "string"

},

"format": {

"type": "string"

},

"provId": {

"type": "string"

},

"nameQual": {

"type": "string"

},

"spNameQual": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of NameID component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Format | format | [] |
| SPProvidedID | provId | [] |
| NameQualifier | nameQual | [] |
| SPNameQualifier | spNameQual | [] |

[component NameID JSON schema details]

### Component Transforms

#### Semantics

[component Transforms normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Transform element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section Transform. [sub component Transform details]

Non-normative Comment:

[component Transforms non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' .The XML type TransformsType SHALL implement the requirements defined in the Transforms component.

The TransformsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="TransformsType">

<sequence>

<xs:element maxOccurs="unbounded" ref="ds-rw:Transform"/>

</sequence>

</complexType>

Each child element of TransformsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Transforms XML schema details]

#### JSON Syntax

The TransformsType JSON object SHALL implement in JSON syntax the requirements defined in the Transforms component.

The TransformsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-TransformsType": {

"$xsd-full-type": "ds-rw:TransformsType",

"type": "object",

"properties": {

"transform": {

"type": "array",

"items": {

"$ref": "#/definitions/dsigrw-TransformType"

}

}

},

"required": ["transform"]

}

Properties in the JSON schema above SHALL implement sub-component of Transforms component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Transform | transform | [] |

[component Transforms JSON schema details]

### Component Transform

#### Semantics

[component Transform normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional value element MUST contain a string. [sub component value details]
* The optional Base64Content element MUST contain base64 encoded binary data. [sub component Base64Content details]
* The optional XPath element MAY occur zero or more times containing a string. [sub component XPath details]
* The optional NsPrefixMapping element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section NsPrefixMapping. [sub component NsPrefixMapping details]
* The Algorithm element MUST contain one instance of a URI. [sub component Algorithm details]

Non-normative Comment:

[component Transform non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' . The original definition of this element uses the 'mixed' content attribute. To support non-XML syntax using a common object model the attribute is dropped and a 'value' component is introduced.The XML type TransformType SHALL implement the requirements defined in the Transform component.

The TransformType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="TransformType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="value" type="string"/>

<xs:element maxOccurs="1" minOccurs="0" name="Base64Content" type="xs:base64Binary"/>

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

<xs:element maxOccurs="unbounded" minOccurs="0" name="NsPrefixMapping" type="dsb:NsPrefixMappingType"/>

</xs:sequence>

<xs:attribute name="Algorithm" type="xs:anyURI" use="required"/>

</xs:complexType>

Each child element of TransformType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Transform XML schema details]

#### JSON Syntax

The TransformType JSON object SHALL implement in JSON syntax the requirements defined in the Transform component.

The TransformType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-TransformType": {

"$xsd-full-type": "ds-rw:TransformType",

"type": "object",

"properties": {

"xpath": {

"type": "array",

"items": {

"type": "string"

}

},

"value": {

"type": "string"

},

"b64Content": {

"type": "string"

},

"nsDecl": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-NsPrefixMappingType"

}

},

"algo": {

"type": "string"

}

},

"required": ["algo"]

}

Properties in the JSON schema above SHALL implement sub-component of Transform component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Base64Content | b64Content | [] |
| XPath | xpath | [] |
| NsPrefixMapping | nsDecl | [] |
| Algorithm | algo | [] |

[component Transform JSON schema details]

### Component X509Data

#### Semantics

[component X509Data normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional X509IssuerSerial element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section X509IssuerSerial. [sub component X509IssuerSerial details]
* The optional X509SKI element MAY occur zero or more times containing base64 encoded binary data. [sub component X509SKI details]
* The optional X509SubjectName element MAY occur zero or more times containing a string. [sub component X509SubjectName details]
* The optional X509Certificate element MAY occur zero or more times containing base64 encoded binary data. [sub component X509Certificate details]
* The optional X509CRL element MAY occur zero or more times containing base64 encoded binary data. [sub component X509CRL details]

Non-normative Comment:

[component X509Data non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' .The XML type X509DataType SHALL implement the requirements defined in the X509Data component.

The X509DataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="X509DataType">

<xs:sequence>

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="X509IssuerSerial" type="ds-rw:X509IssuerSerialType"/>

</xs:sequence>

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="X509SKI" type="base64Binary"/>

</xs:sequence>

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="X509SubjectName" type="string"/>

</xs:sequence>

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="X509Certificate" type="base64Binary"/>

</xs:sequence>

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="X509CRL" type="base64Binary"/>

</xs:sequence>

</xs:sequence>

</xs:complexType>

Each child element of X509DataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component X509Data XML schema details]

#### JSON Syntax

The X509DataType JSON object SHALL implement in JSON syntax the requirements defined in the X509Data component.

The X509DataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-X509DataType": {

"$xsd-full-type": "ds-rw:X509DataType",

"type": "object",

"properties": {

"issSerial": {

"type": "array",

"items": {

"$ref": "#/definitions/dsigrw-X509IssuerSerialType"

}

},

"ski": {

"type": "array",

"items": {

"type": "string"

}

},

"sub": {

"type": "array",

"items": {

"type": "string"

}

},

"cert": {

"type": "array",

"items": {

"type": "string"

}

},

"crl": {

"type": "array",

"items": {

"type": "string"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of X509Data component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| X509IssuerSerial | issSerial | [] |
| X509SKI | ski | [] |
| X509SubjectName | sub | [] |
| X509Certificate | cert | [] |
| X509CRL | crl | [] |

[component X509Data JSON schema details]

### Component X509IssuerSerial

#### Semantics

[component X509IssuerSerial normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The X509IssuerName element MUST contain one instance of a string. [sub component X509IssuerName details]
* The X509SerialNumber element MUST contain one instance of an integer. [sub component X509SerialNumber details]

Non-normative Comment:

[component X509IssuerSerial non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' .The XML type X509IssuerSerialType SHALL implement the requirements defined in the X509IssuerSerial component.

The X509IssuerSerialType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="X509IssuerSerialType">

<sequence>

<element name="X509IssuerName" type="string"/>

<element name="X509SerialNumber" type="integer"/>

</sequence>

</complexType>

Each child element of X509IssuerSerialType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component X509IssuerSerial XML schema details]

#### JSON Syntax

The X509IssuerSerialType JSON object SHALL implement in JSON syntax the requirements defined in the X509IssuerSerial component.

The X509IssuerSerialType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-X509IssuerSerialType": {

"$xsd-full-type": "ds-rw:X509IssuerSerialType",

"type": "object",

"properties": {

"name": {

"type": "string"

},

"serial": {

"type": "integer"

}

},

"required": ["name", "serial"]

}

Properties in the JSON schema above SHALL implement sub-component of X509IssuerSerial component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| X509IssuerName | name | [] |
| X509SerialNumber | serial | [] |

[component X509IssuerSerial JSON schema details]

### Component DigestMethod

#### Semantics

[component DigestMethod normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional value element MUST contain a string. [sub component value details]
* The optional Base64Content element MUST contain base64 encoded binary data. [sub component Base64Content details]
* The Algorithm element MUST contain one instance of a URI. [sub component Algorithm details]

Non-normative Comment:

[component DigestMethod non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' . The original definition of this element uses the 'mixed' content attribute. To support non-XML syntax using a common object model the attribute is dropped and a 'value' component is introduced.The XML type DigestMethodType SHALL implement the requirements defined in the DigestMethod component.

The DigestMethodType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="DigestMethodType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="value" type="string"/>

<xs:sequence maxOccurs="unbounded" minOccurs="0">

<xs:element maxOccurs="1" minOccurs="0" name="Base64Content" type="xs:base64Binary"/>

</xs:sequence>

</xs:sequence>

<xs:attribute name="Algorithm" type="anyURI" use="required"/>

</xs:complexType>

Each child element of DigestMethodType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DigestMethod XML schema details]

#### JSON Syntax

The DigestMethodType JSON object SHALL implement in JSON syntax the requirements defined in the DigestMethod component.

The DigestMethodType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-DigestMethodType": {

"$xsd-full-type": "ds-rw:DigestMethodType",

"type": "object",

"properties": {

"val": {

"type": "string"

},

"content": {

"type": "array",

"items": {

"type": "string"

}

},

"alg": {

"type": "string"

}

},

"required": ["alg"]

}

Properties in the JSON schema above SHALL implement sub-component of DigestMethod component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | val | [] |
| Base64Content | content | [] |
| Algorithm | alg | [] |

[component DigestMethod JSON schema details]

### Component CanonicalizationMethod

#### Semantics

[component CanonicalizationMethod normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional value element MUST contain a string. [sub component value details]
* The optional Base64Content element MUST contain base64 encoded binary data. [sub component Base64Content details]
* The optional XPath element MAY occur zero or more times containing a string. [sub component XPath details]
* The optional NsPrefixMapping element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section NsPrefixMapping. [sub component NsPrefixMapping details]
* The Algorithm element MUST contain one instance of a URI. [sub component Algorithm details]

Non-normative Comment:

[component CanonicalizationMethod non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' . The original definition of this element uses the 'mixed' content attribute. To support non-XML syntax using a common object model the attribute is dropped and a 'value' component is introduced.The XML type CanonicalizationMethodType SHALL implement the requirements defined in the CanonicalizationMethod component.

The CanonicalizationMethodType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="CanonicalizationMethodType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="value" type="string"/>

<xs:element maxOccurs="1" minOccurs="0" name="Base64Content" type="xs:base64Binary"/>

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

<xs:element maxOccurs="unbounded" minOccurs="0" name="NsPrefixMapping" type="dsb:NsPrefixMappingType"/>

</xs:sequence>

<xs:attribute name="Algorithm" type="xs:anyURI" use="required"/>

</xs:complexType>

Each child element of CanonicalizationMethodType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CanonicalizationMethod XML schema details]

#### JSON Syntax

The CanonicalizationMethodType JSON object SHALL implement in JSON syntax the requirements defined in the CanonicalizationMethod component.

The CanonicalizationMethodType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-CanonicalizationMethodType": {

"$xsd-full-type": "ds-rw:CanonicalizationMethodType",

"type": "object",

"properties": {

"xpath": {

"type": "array",

"items": {

"type": "string"

}

},

"value": {

"type": "string"

},

"b64Content": {

"type": "string"

},

"nsDecl": {

"type": "array",

"items": {

"$ref": "#/definitions/dsb-NsPrefixMappingType"

}

},

"algo": {

"type": "string"

}

},

"required": ["algo"]

}

Properties in the JSON schema above SHALL implement sub-component of CanonicalizationMethod component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Base64Content | b64Content | [] |
| XPath | xpath | [] |
| NsPrefixMapping | nsDecl | [] |
| Algorithm | algo | [] |

[component CanonicalizationMethod JSON schema details]

### Component SignatureValue

#### Semantics

[component SignatureValue normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of base64 encoded binary data. [sub component value details]
* The optional Id element MUST contain one instance of a unique identifier. [sub component Id details]

Non-normative Comment:

[component SignatureValue non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://docs.oasis-open.org/dss/ns/xmldsig/rewritten' .The XML type SignatureValueType SHALL implement the requirements defined in the SignatureValue component.

The SignatureValueType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<complexType name="SignatureValueType">

<simpleContent>

<extension base="base64Binary">

<attribute name="Id" type="ID" use="optional"/>

</extension>

</simpleContent>

</complexType>

Each child element of SignatureValueType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' holding the base64 content is represented by the component's XML tag text content. [component SignatureValue XML schema details]

#### JSON Syntax

The SignatureValueType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureValue component.

The SignatureValueType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsigrw-SignatureValueType": {

"$xsd-full-type": "ds-rw:SignatureValueType",

"type": "object",

"properties": {

"value": {

"type": "string"

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignatureValue component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Id | id | [] |

[component SignatureValue JSON schema details]

### Component ValidationConstraints

#### Semantics

[component ValidationConstraints normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignaturePolicyIdentifier element MUST contain sub-component. A given element MUST satisfy the requirements specified in section SignaturePolicyIdentifier. [sub component SignaturePolicyIdentifier details]
* The optional ValidationConstraint element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section SingleValidationConstraint. [sub component ValidationConstraint details]

Non-normative Comment:

[component ValidationConstraints non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationConstraintsType SHALL implement the requirements defined in the ValidationConstraints component.

The ValidationConstraintsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationConstraintsType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="SignaturePolicyIdentifier" type="xades-rw:SignaturePolicyIdentifierType"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ValidationConstraint" type="etsi-vr:SingleValidationConstraintType"/>

</xs:sequence>

</xs:complexType>

Each child element of ValidationConstraintsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationConstraints XML schema details]

#### JSON Syntax

The ValidationConstraintsType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationConstraints component.

The ValidationConstraintsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationConstraintsType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"sigPolId": {

"$ref": "#/definitions/xadesrw-SignaturePolicyIdentifierType"

},

"valConstraint": {

"type": "array",

"items": {

"$ref": "#/definitions/evr-SingleValidationConstraintType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationConstraints component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignaturePolicyIdentifier | sigPolId | [] |
| ValidationConstraint | valConstraint | [] |

[component ValidationConstraints JSON schema details]

### Component SingleValidationConstraint

#### Semantics

[component SingleValidationConstraint normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ValidationConstraint element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ValidationConstraint. [sub component ValidationConstraint details]
* The ConstraintStatus element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ConstraintStatus. [sub component ConstraintStatus details]
* The optional VerificationResult element MUST contain sub-component. A given element MUST satisfy the requirements specified in section VerificationResult. [sub component VerificationResult details]
* The optional Hint element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component Hint details]

Non-normative Comment:

[component SingleValidationConstraint non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type SingleValidationConstraintType SHALL implement the requirements defined in the SingleValidationConstraint component.

The SingleValidationConstraintType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SingleValidationConstraintType">

<xs:sequence>

<xs:element minOccurs="1" name="ValidationConstraint" type="etsi-vr:ValidationConstraintType"/>

<xs:element maxOccurs="1" minOccurs="1" name="ConstraintStatus" type="etsi-vr:ConstraintStatusType"/>

<xs:element minOccurs="0" name="VerificationResult" type="vr:VerificationResultType"/>

<xs:element minOccurs="0" name="Hint" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of SingleValidationConstraintType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SingleValidationConstraint XML schema details]

#### JSON Syntax

The SingleValidationConstraintType JSON object SHALL implement in JSON syntax the requirements defined in the SingleValidationConstraint component.

The SingleValidationConstraintType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-SingleValidationConstraintType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"valConstraint": {

"$ref": "#/definitions/evr-ValidationConstraintType"

},

"constraintStatus": {

"$ref": "#/definitions/evr-ConstraintStatusType"

},

"verResult": {

"$ref": "#/definitions/vr-VerificationResultType"

},

"hint": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SingleValidationConstraint component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidationConstraint | valConstraint | [] |
| ConstraintStatus | constraintStatus | [] |
| VerificationResult | verResult | [] |
| Hint | hint | [] |

[component SingleValidationConstraint JSON schema details]

### Component ValidationConstraint

#### Semantics

[component ValidationConstraint normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ValidationConstraintIdentifier element MUST contain one instance of a URI. [sub component ValidationConstraintIdentifier details]
* The optional ValidationConstraintParameter element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section ValidationConstraintParameter. [sub component ValidationConstraintParameter details]

Non-normative Comment:

[component ValidationConstraint non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationConstraintType SHALL implement the requirements defined in the ValidationConstraint component.

The ValidationConstraintType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationConstraintType">

<xs:sequence>

<xs:element minOccurs="1" name="ValidationConstraintIdentifier" type="xs:anyURI"/>

<xs:element maxOccurs="unbounded" minOccurs="0" name="ValidationConstraintParameter" type="etsi-vr:ValidationConstraintParameterType"/>

</xs:sequence>

</xs:complexType>

Each child element of ValidationConstraintType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationConstraint XML schema details]

#### JSON Syntax

The ValidationConstraintType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationConstraint component.

The ValidationConstraintType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationConstraintType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"valConstraintId": {

"type": "string"

},

"valConstraintParam": {

"type": "array",

"items": {

"$ref": "#/definitions/evr-ValidationConstraintParameterType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationConstraint component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidationConstraintIdentifier | valConstraintId | [] |
| ValidationConstraintParameter | valConstraintParam | [] |

[component ValidationConstraint JSON schema details]

### Component ValidationConstraintParameter

#### Semantics

[component ValidationConstraintParameter normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ParameterType element MUST contain one instance of a URI. [sub component ParameterType details]
* The ParameterValue element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Any. [sub component ParameterValue details]

Non-normative Comment:

[component ValidationConstraintParameter non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationConstraintParameterType SHALL implement the requirements defined in the ValidationConstraintParameter component.

The ValidationConstraintParameterType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationConstraintParameterType">

<xs:sequence>

<xs:element name="ParameterType" type="xs:anyURI"/>

<xs:element name="ParameterValue" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of ValidationConstraintParameterType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationConstraintParameter XML schema details]

#### JSON Syntax

The ValidationConstraintParameterType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationConstraintParameter component.

The ValidationConstraintParameterType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationConstraintParameterType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"type": {

"type": "string",

"format": "uri"

},

"value": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationConstraintParameter component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ParameterType | type | [] |
| ParameterValue | value | [] |

[component ValidationConstraintParameter JSON schema details]

### Component ConstraintStatus

#### Semantics

[component ConstraintStatus normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Status element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ConstraintApplication. [sub component Status details]
* The optional OverriddenBy element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component OverriddenBy details]
* The optional Other element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component Other details]

Non-normative Comment:

[component ConstraintStatus non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ConstraintStatusType SHALL implement the requirements defined in the ConstraintStatus component.

The ConstraintStatusType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ConstraintStatusType">

<xs:sequence>

<xs:element minOccurs="1" name="Status" type="etsi-vr:ConstraintApplicationType"/>

<xs:element minOccurs="0" name="OverriddenBy" type="dsb:AnyType"/>

<xs:element minOccurs="0" name="Other" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of ConstraintStatusType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ConstraintStatus XML schema details]

#### JSON Syntax

The ConstraintStatusType JSON object SHALL implement in JSON syntax the requirements defined in the ConstraintStatus component.

The ConstraintStatusType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ConstraintStatusType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"status": {

"type": "string"

},

"overriddenBy": {

"$ref": "#/definitions/dsb-AnyType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ConstraintStatus component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Status | status | [] |
| OverriddenBy | overriddenBy | [] |
| Other | other | [] |

[component ConstraintStatus JSON schema details]

### Component SignersDocument

#### Semantics

[component SignersDocument normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional DigestAlgAndValue element MUST contain sub-component. A given element MUST satisfy the requirements specified in section DigestAlgAndValue. [sub component DigestAlgAndValue details]
* The SignersDocument element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component SignersDocument details]

Non-normative Comment:

[component SignersDocument non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type SignersDocumentType SHALL implement the requirements defined in the SignersDocument component.

The SignersDocumentType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignersDocumentType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="0" name="DigestAlgAndValue" type="xades-rw:DigestAlgAndValueType"/>

<xs:element minOccurs="1" name="SignersDocument" type="etsi-vr:VOReferenceType"/>

</xs:sequence>

</xs:complexType>

Each child element of SignersDocumentType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignersDocument XML schema details]

#### JSON Syntax

The SignersDocumentType JSON object SHALL implement in JSON syntax the requirements defined in the SignersDocument component.

The SignersDocumentType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-SignersDocumentType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"digAlgVal": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

},

"sigDoc": {

"$ref": "#/definitions/evr-VOReferenceType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignersDocument component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestAlgAndValue | digAlgVal | [] |
| SignersDocument | sigDoc | [] |

[component SignersDocument JSON schema details]

### Component VOReference

#### Semantics

[component VOReference normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The VOReference element MUST contain one instance of a unique identifier reference. [sub component VOReference details]

Non-normative Comment:

[component VOReference non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type VOReferenceType SHALL implement the requirements defined in the VOReference component.

The VOReferenceType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="VOReferenceType">

<xs:attribute name="VOReference" type="xs:IDREFS" use="required"/>

</xs:complexType>

Each child element of VOReferenceType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component VOReference XML schema details]

#### JSON Syntax

The VOReferenceType JSON object SHALL implement in JSON syntax the requirements defined in the VOReference component.

The VOReferenceType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-VOReferenceType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"voreference": {

"type": "array",

"items": {

"$ref": "#/definitions/evr-ValidationObjectType"

}

},

"voRef": {

"type": "array",

"items": {

"$ref": "#/definitions/evr-ValidationObjectType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of VOReference component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| VOReference | voRef | [] |

[component VOReference JSON schema details]

### Component ValidationObject

#### Semantics

[component ValidationObject normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ObjectType element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Object. [sub component ObjectType details]
* The ValidationObject element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ValidationObjectRepresentation. [sub component ValidationObject details]
* The optional PoE element MUST contain sub-component. A given element MUST satisfy the requirements specified in section PoE. [sub component PoE details]
* The optional ValidationReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section IndividualReport. [sub component ValidationReport details]
* The optional IndividualTimeStampReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section TimeStampValidity. [sub component IndividualTimeStampReport details]
* The optional IndividualCertificateReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section CertificateValidity. [sub component IndividualCertificateReport details]
* The optional IndividualAttributeCertificateReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section AttributeCertificateValidity. [sub component IndividualAttributeCertificateReport details]
* The optional IndividualCRLReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section CRLValidity. [sub component IndividualCRLReport details]
* The optional IndividualOCSPReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section OCSPValidity. [sub component IndividualOCSPReport details]
* The optional EvidenceRecordReport element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section EvidenceRecordValidity. [sub component EvidenceRecordReport details]
* The id element MUST contain one instance of a unique identifier. [sub component id details]

Non-normative Comment:

[component ValidationObject non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationObjectType SHALL implement the requirements defined in the ValidationObject component.

The ValidationObjectType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationObjectType">

<xs:sequence>

<xs:element name="ObjectType" type="etsi-vr:ObjectType"/>

<xs:element name="ValidationObject" type="etsi-vr:ValidationObjectRepresentationType"/>

<xs:element minOccurs="0" name="PoE" type="etsi-vr:PoEType"/>

<xs:choice maxOccurs="1" minOccurs="0">

<xs:element name="ValidationReport" type="vr:IndividualReportType"/>

<xs:element ref="vr:IndividualTimeStampReport"/>

<xs:element ref="vr:IndividualCertificateReport"/>

<xs:element ref="vr:IndividualAttributeCertificateReport"/>

<xs:element ref="vr:IndividualCRLReport"/>

<xs:element ref="vr:IndividualOCSPReport"/>

<xs:element ref="vr:EvidenceRecordReport"/>

</xs:choice>

</xs:sequence>

<xs:attribute name="id" type="xs:ID" use="required"/>

</xs:complexType>

Each child element of ValidationObjectType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationObject XML schema details]

#### JSON Syntax

The ValidationObjectType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationObject component.

The ValidationObjectType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationObjectType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"objType": {

"type": "string"

},

"valObj": {

"$ref": "#/definitions/evr-ValidationObjectRepresentationType"

},

"poe": {

"$ref": "#/definitions/evr-PoEType"

},

"valReport": {

"$ref": "#/definitions/vr-IndividualReportType"

},

"tsReport": {

"$ref": "#/definitions/vr-TimeStampValidityType"

},

"certReport": {

"$ref": "#/definitions/vr-CertificateValidityType"

},

"attCertReport": {

"$ref": "#/definitions/vr-AttributeCertificateValidityType"

},

"crlReport": {

"$ref": "#/definitions/vr-CRLValidityType"

},

"ocspReport": {

"$ref": "#/definitions/vr-OCSPValidityType"

},

"erReport": {

"$ref": "#/definitions/vr-EvidenceRecordValidityType"

},

"id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationObject component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ObjectType | objType | [] |
| ValidationObject | valObj | [] |
| PoE | poe | [] |
| ValidationReport | valReport | [] |
| IndividualTimeStampReport | tsReport | [] |
| IndividualCertificateReport | certReport | [] |
| IndividualAttributeCertificateReport | attCertReport | [] |
| IndividualCRLReport | crlReport | [] |
| IndividualOCSPReport | ocspReport | [] |
| EvidenceRecordReport | erReport | [] |
| id | id | [] |

[component ValidationObject JSON schema details]

### Component ValidationObjectRepresentation

#### Semantics

[component ValidationObjectRepresentation normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The direct element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Any. [sub component direct details]
* The base64 element MUST contain one instance of base64 encoded binary data. [sub component base64 details]
* The URI element MUST contain one instance of a URI. [sub component URI details]

Non-normative Comment:

[component ValidationObjectRepresentation non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationObjectRepresentationType SHALL implement the requirements defined in the ValidationObjectRepresentation component.

The ValidationObjectRepresentationType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationObjectRepresentationType">

<xs:choice>

<xs:element name="direct" type="dsb:AnyType"/>

<xs:element name="base64" type="xs:base64Binary"/>

<xs:element name="URI" type="xs:anyURI"/>

</xs:choice>

</xs:complexType>

Each child element of ValidationObjectRepresentationType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationObjectRepresentation XML schema details]

#### JSON Syntax

The ValidationObjectRepresentationType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationObjectRepresentation component.

The ValidationObjectRepresentationType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationObjectRepresentationType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"dir": {

"$ref": "#/definitions/dsb-AnyType"

},

"b64Cnt": {

"type": "string"

},

"uri": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationObjectRepresentation component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| direct | dir | [] |
| base64 | b64Cnt | [] |
| URI | uri | [] |

[component ValidationObjectRepresentation JSON schema details]

### Component PoE

#### Semantics

[component PoE normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The PoETime element MUST contain one instance of a date/time value. [sub component PoETime details]
* The optional PoEObject element MUST contain sub-component. A given element MUST satisfy the requirements specified in section VOReference. [sub component PoEObject details]

Non-normative Comment:

[component PoE non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type PoEType SHALL implement the requirements defined in the PoE component.

The PoEType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="PoEType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="1" name="PoETime" type="xs:dateTime"/>

<xs:element maxOccurs="1" minOccurs="0" name="PoEObject" type="etsi-vr:VOReferenceType"/>

</xs:sequence>

</xs:complexType>

Each child element of PoEType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component PoE XML schema details]

#### JSON Syntax

The PoEType JSON object SHALL implement in JSON syntax the requirements defined in the PoE component.

The PoEType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-PoEType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"poeTime": {

"type": "integer",

"format": "utc-millisec"

},

"poeObj": {

"$ref": "#/definitions/evr-VOReferenceType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of PoE component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| PoETime | poeTime | [] |
| PoEObject | poeObj | [] |

[component PoE JSON schema details]

### Component SignerInformation

#### Semantics

[component SignerInformation normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignerCertificate element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component SignerCertificate details]
* The Signer element MUST contain one instance of a string. [sub component Signer details]
* The optional SignerInfo element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component SignerInfo details]
* The optional Pseudonym element MUST contain one instance of a boolean. [sub component Pseudonym details]

Non-normative Comment:

[component SignerInformation non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type SignerInformationType SHALL implement the requirements defined in the SignerInformation component.

The SignerInformationType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignerInformationType">

<xs:sequence>

<xs:element minOccurs="1" name="SignerCertificate" type="etsi-vr:VOReferenceType"/>

<xs:element name="Signer" type="xs:string"/>

<xs:element minOccurs="0" name="SignerInfo" type="dsb:AnyType"/>

</xs:sequence>

<xs:attribute name="Pseudonym" type="xs:boolean" use="optional"/>

</xs:complexType>

Each child element of SignerInformationType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignerInformation XML schema details]

#### JSON Syntax

The SignerInformationType JSON object SHALL implement in JSON syntax the requirements defined in the SignerInformation component.

The SignerInformationType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-SignerInformationType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"signerCert": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"signer": {

"type": "string"

},

"signerInfo": {

"$ref": "#/definitions/dsb-AnyType"

},

"pseudo": {

"type": "boolean"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignerInformation component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignerCertificate | signerCert | [] |
| Signer | signer | [] |
| SignerInfo | signerInfo | [] |
| Pseudonym | pseudo | [] |

[component SignerInformation JSON schema details]

### Component SignatureQualityList

#### Semantics

[component SignatureQualityList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional SignatureQualityInformation element MAY occur zero or more times containing a URI. [sub component SignatureQualityInformation details]

Non-normative Comment:

[component SignatureQualityList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type SignatureQualityListType SHALL implement the requirements defined in the SignatureQualityList component.

The SignatureQualityListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignatureQualityListType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="SignatureQualityInformation" type="xs:anyURI"/>

</xs:sequence>

</xs:complexType>

Each child element of SignatureQualityListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignatureQualityList XML schema details]

#### JSON Syntax

The SignatureQualityListType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureQualityList component.

The SignatureQualityListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-SignatureQualityListType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"sigQuality": {

"type": "array",

"items": {

"type": "string"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignatureQualityList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignatureQualityInformation | sigQuality | [] |

[component SignatureQualityList JSON schema details]

### Component SignatureValidationProcess

#### Semantics

[component SignatureValidationProcess normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignatureValidationProcessID element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section SignatureValidationProcessID. [sub component SignatureValidationProcessID details]
* The optional SignatureValidationServicePolicy element MUST contain a URI. [sub component SignatureValidationServicePolicy details]
* The optional SignatureValidationPracticeStatement element MUST contain a URI. [sub component SignatureValidationPracticeStatement details]
* The optional AugmentationInfo element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component AugmentationInfo details]
* The optional Other element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Any. [sub component Other details]

Non-normative Comment:

[component SignatureValidationProcess non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type SignatureValidationProcessType SHALL implement the requirements defined in the SignatureValidationProcess component.

The SignatureValidationProcessType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="SignatureValidationProcessType">

<xs:sequence>

<xs:element name="SignatureValidationProcessID" type="etsi-vr:SignatureValidationProcessIDType"/>

<xs:element minOccurs="0" name="SignatureValidationServicePolicy" type="xs:anyURI"/>

<xs:element minOccurs="0" name="SignatureValidationPracticeStatement" type="xs:anyURI"/>

<xs:element minOccurs="0" name="AugmentationInfo" type="dsb:AnyType"/>

<xs:element minOccurs="0" name="Other" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of SignatureValidationProcessType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignatureValidationProcess XML schema details]

#### JSON Syntax

The SignatureValidationProcessType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureValidationProcess component.

The SignatureValidationProcessType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-SignatureValidationProcessType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"sigValProcId": {

"type": "string"

},

"sigValPol": {

"type": "string"

},

"sigValPS": {

"type": "string"

},

"augInfo": {

"$ref": "#/definitions/dsb-AnyType"

},

"other": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignatureValidationProcess component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignatureValidationProcessID | sigValProcId | [] |
| SignatureValidationServicePolicy | sigValPol | [] |
| SignatureValidationPracticeStatement | sigValPS | [] |
| AugmentationInfo | augInfo | [] |
| Other | other | [] |

[component SignatureValidationProcess JSON schema details]

### Component ValidationReportData

#### Semantics

[component ValidationReportData normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SigningCertificate element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component SigningCertificate details]
* The CertificateChain element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component CertificateChain details]
* The SignedDataObjects element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component SignedDataObjects details]
* The RevocationStatusInformation element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section RevocationStatusInformation. [sub component RevocationStatusInformation details]
* The CryptoInformation element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section CryptoInformation. [sub component CryptoInformation details]
* The AdditionalValidationReportData element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section AdditionalValidationReportData. [sub component AdditionalValidationReportData details]

Non-normative Comment:

[component ValidationReportData non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationReportDataType SHALL implement the requirements defined in the ValidationReportData component.

The ValidationReportDataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationReportDataType">

<xs:sequence>

<xs:element maxOccurs="1" name="SigningCertificate" type="etsi-vr:VOReferenceType"/>

<xs:element maxOccurs="1" name="CertificateChain" type="etsi-vr:VOReferenceType"/>

<xs:element maxOccurs="1" name="SignedDataObjects" type="etsi-vr:VOReferenceType"/>

<xs:element name="RevocationStatusInformation" type="etsi-vr:RevocationStatusInformationType"/>

<xs:element name="CryptoInformation" type="etsi-vr:CryptoInformationType"/>

<xs:element name="AdditionalValidationReportData" type="etsi-vr:AdditionalValidationReportDataType"/>

</xs:sequence>

</xs:complexType>

Each child element of ValidationReportDataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationReportData XML schema details]

#### JSON Syntax

The ValidationReportDataType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationReportData component.

The ValidationReportDataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationReportDataType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"sigCert": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"certChain": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"sigData": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"revStat": {

"$ref": "#/definitions/evr-RevocationStatusInformationType"

},

"cryptInfo": {

"$ref": "#/definitions/evr-CryptoInformationType"

},

"addValReport": {

"$ref": "#/definitions/evr-AdditionalValidationReportDataType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationReportData component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigningCertificate | sigCert | [] |
| CertificateChain | certChain | [] |
| SignedDataObjects | sigData | [] |
| RevocationStatusInformation | revStat | [] |
| CryptoInformation | cryptInfo | [] |
| AdditionalValidationReportData | addValReport | [] |

[component ValidationReportData JSON schema details]

### Component RevocationStatusInformation

#### Semantics

[component RevocationStatusInformation normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ValidationObjectId element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component ValidationObjectId details]
* The RevocationTime element MUST contain one instance of a date/time value. [sub component RevocationTime details]
* The optional RevocationReason element MUST contain a URI. [sub component RevocationReason details]
* The optional RevocationObject element MUST contain sub-component. A given element MUST satisfy the requirements specified in section VOReference. [sub component RevocationObject details]

Non-normative Comment:

[component RevocationStatusInformation non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type RevocationStatusInformationType SHALL implement the requirements defined in the RevocationStatusInformation component.

The RevocationStatusInformationType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="RevocationStatusInformationType">

<xs:sequence>

<xs:element minOccurs="1" name="ValidationObjectId" type="etsi-vr:VOReferenceType"/>

<xs:element name="RevocationTime" type="xs:dateTime"/>

<xs:element maxOccurs="1" minOccurs="0" name="RevocationReason" type="xs:anyURI"/>

<xs:element minOccurs="0" name="RevocationObject" type="etsi-vr:VOReferenceType"/>

</xs:sequence>

</xs:complexType>

Each child element of RevocationStatusInformationType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component RevocationStatusInformation XML schema details]

#### JSON Syntax

The RevocationStatusInformationType JSON object SHALL implement in JSON syntax the requirements defined in the RevocationStatusInformation component.

The RevocationStatusInformationType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-RevocationStatusInformationType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"valObjID": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"revTime": {

"type": "integer",

"format": "utc-millisec"

},

"revReason": {

"type": "string"

},

"revObj": {

"$ref": "#/definitions/evr-VOReferenceType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of RevocationStatusInformation component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidationObjectId | valObjID | [] |
| RevocationTime | revTime | [] |
| RevocationReason | revReason | [] |
| RevocationObject | revObj | [] |

[component RevocationStatusInformation JSON schema details]

### Component CryptoInformation

#### Semantics

[component CryptoInformation normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ValidationObjectId element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section VOReference. [sub component ValidationObjectId details]
* The Algorithm element MUST contain one instance of a URI. [sub component Algorithm details]
* The optional AlgorithmParameters element MUST contain sub-component. A given element MUST satisfy the requirements specified in section AlgorithmParameter. [sub component AlgorithmParameters details]
* The optional NotAfter element MUST contain a date/time value. [sub component NotAfter details]

Non-normative Comment:

[component CryptoInformation non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type CryptoInformationType SHALL implement the requirements defined in the CryptoInformation component.

The CryptoInformationType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="CryptoInformationType">

<xs:sequence>

<xs:element minOccurs="1" name="ValidationObjectId" type="etsi-vr:VOReferenceType"/>

<xs:element minOccurs="1" name="Algorithm" type="xs:anyURI"/>

<xs:element minOccurs="0" name="AlgorithmParameters" type="etsi-vr:AlgorithmParameterType"/>

<xs:element minOccurs="0" name="NotAfter" type="xs:dateTime"/>

</xs:sequence>

</xs:complexType>

Each child element of CryptoInformationType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CryptoInformation XML schema details]

#### JSON Syntax

The CryptoInformationType JSON object SHALL implement in JSON syntax the requirements defined in the CryptoInformation component.

The CryptoInformationType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-CryptoInformationType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"valObjId": {

"$ref": "#/definitions/evr-VOReferenceType"

},

"alg": {

"type": "string"

},

"algParam": {

"$ref": "#/definitions/evr-AlgorithmParameterType"

},

"notAfter": {

"type": "integer",

"format": "utc-millisec"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CryptoInformation component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidationObjectId | valObjId | [] |
| Algorithm | alg | [] |
| AlgorithmParameters | algParam | [] |
| NotAfter | notAfter | [] |

[component CryptoInformation JSON schema details]

### Component AlgorithmParameter

#### Semantics

[component AlgorithmParameter normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ParameterID element MUST contain one instance of a URI. [sub component ParameterID details]
* The Value element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Any. [sub component Value details]

Non-normative Comment:

[component AlgorithmParameter non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type AlgorithmParameterType SHALL implement the requirements defined in the AlgorithmParameter component.

The AlgorithmParameterType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AlgorithmParameterType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="1" name="ParameterID" type="xs:anyURI"/>

<xs:element maxOccurs="1" minOccurs="1" name="Value" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of AlgorithmParameterType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AlgorithmParameter XML schema details]

#### JSON Syntax

The AlgorithmParameterType JSON object SHALL implement in JSON syntax the requirements defined in the AlgorithmParameter component.

The AlgorithmParameterType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-AlgorithmParameterType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"paramId": {

"type": "string"

},

"val": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of AlgorithmParameter component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ParameterID | paramId | [] |
| Value | val | [] |

[component AlgorithmParameter JSON schema details]

### Component AdditionalValidationReportData

#### Semantics

[component AdditionalValidationReportData normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ReportData element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ReportData. [sub component ReportData details]

Non-normative Comment:

[component AdditionalValidationReportData non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type AdditionalValidationReportDataType SHALL implement the requirements defined in the AdditionalValidationReportData component.

The AdditionalValidationReportDataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AdditionalValidationReportDataType">

<xs:sequence>

<xs:element minOccurs="1" name="ReportData" type="etsi-vr:ReportDataType"/>

</xs:sequence>

</xs:complexType>

Each child element of AdditionalValidationReportDataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AdditionalValidationReportData XML schema details]

#### JSON Syntax

The AdditionalValidationReportDataType JSON object SHALL implement in JSON syntax the requirements defined in the AdditionalValidationReportData component.

The AdditionalValidationReportDataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-AdditionalValidationReportDataType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"repData": {

"$ref": "#/definitions/evr-ReportDataType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of AdditionalValidationReportData component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ReportData | repData | [] |

[component AdditionalValidationReportData JSON schema details]

### Component ReportData

#### Semantics

[component ReportData normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The InfoType element MUST contain one instance of a URI. [sub component InfoType details]
* The InfoData element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Any. [sub component InfoData details]

Non-normative Comment:

[component ReportData non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ReportDataType SHALL implement the requirements defined in the ReportData component.

The ReportDataType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ReportDataType">

<xs:sequence>

<xs:element maxOccurs="1" minOccurs="1" name="InfoType" type="xs:anyURI"/>

<xs:element maxOccurs="1" minOccurs="1" name="InfoData" type="dsb:AnyType"/>

</xs:sequence>

</xs:complexType>

Each child element of ReportDataType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ReportData XML schema details]

#### JSON Syntax

The ReportDataType JSON object SHALL implement in JSON syntax the requirements defined in the ReportData component.

The ReportDataType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ReportDataType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"infoType": {

"type": "string"

},

"infoData": {

"$ref": "#/definitions/dsb-AnyType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ReportData component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| InfoType | infoType | [] |
| InfoData | infoData | [] |

[component ReportData JSON schema details]

### Component ValidationObjectList

#### Semantics

[component ValidationObjectList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ValidationObject element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section ValidationObject. [sub component ValidationObject details]

Non-normative Comment:

[component ValidationObjectList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/1191022/v0.1.2' .The XML type ValidationObjectListType SHALL implement the requirements defined in the ValidationObjectList component.

The ValidationObjectListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="ValidationObjectListType">

<xs:sequence>

<xs:element maxOccurs="unbounded" name="ValidationObject" type="etsi-vr:ValidationObjectType"/>

</xs:sequence>

</xs:complexType>

Each child element of ValidationObjectListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ValidationObjectList XML schema details]

#### JSON Syntax

The ValidationObjectListType JSON object SHALL implement in JSON syntax the requirements defined in the ValidationObjectList component.

The ValidationObjectListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evr-ValidationObjectListType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"valObj": {

"type": "array",

"items": {

"$ref": "#/definitions/evr-ValidationObjectType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ValidationObjectList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ValidationObject | valObj | [] |

[component ValidationObjectList JSON schema details]

### Component AvailableSignatureValidationPolicies

#### Semantics

[component AvailableSignatureValidationPolicies normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional AvailableSignatureValidationPolicyID element MAY occur zero or more times containing a URI. [sub component AvailableSignatureValidationPolicyID details]

Non-normative Comment:

[component AvailableSignatureValidationPolicies non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'http://uri.etsi.org/19442/v1.1.1#' .The XML type AvailableSignatureValidationPoliciesType SHALL implement the requirements defined in the AvailableSignatureValidationPolicies component.

The AvailableSignatureValidationPoliciesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AvailableSignatureValidationPoliciesType">

<xs:sequence>

<xs:element maxOccurs="unbounded" minOccurs="0" name="AvailableSignatureValidationPolicyID" type="xs:anyURI"/>

</xs:sequence>

</xs:complexType>

Each child element of AvailableSignatureValidationPoliciesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component AvailableSignatureValidationPolicies XML schema details]

#### JSON Syntax

The AvailableSignatureValidationPoliciesType JSON object SHALL implement in JSON syntax the requirements defined in the AvailableSignatureValidationPolicies component.

The AvailableSignatureValidationPoliciesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"evp-AvailableSignatureValidationPoliciesType": {

"$xsd-full-type": "",

"type": "object",

"properties": {

"availableSignatureValidationPolicyID": {

"type": "array",

"items": {

"type": "string"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of AvailableSignatureValidationPolicies component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| AvailableSignatureValidationPolicyID |  | [] |

[component AvailableSignatureValidationPolicies JSON schema details]

### Component ObjectIdentifier

#### Semantics

[component ObjectIdentifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Identifier element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Identifier. [sub component Identifier details]
* The optional Description element MUST contain a string. [sub component Description details]
* The optional DocumentationReferences element MUST contain sub-component. A given element MUST satisfy the requirements specified in section DocumentationReferences. [sub component DocumentationReferences details]

Non-normative Comment:

[component ObjectIdentifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type ObjectIdentifierType SHALL implement the requirements defined in the ObjectIdentifier component.

The ObjectIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="ObjectIdentifierType">

<xsd:sequence>

<xsd:element name="Identifier" type="xades-rw:IdentifierType"/>

<xsd:element minOccurs="0" name="Description" type="xsd:string"/>

<xsd:element minOccurs="0" name="DocumentationReferences" type="xades-rw:DocumentationReferencesType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of ObjectIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ObjectIdentifier XML schema details]

#### JSON Syntax

The ObjectIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the ObjectIdentifier component.

The ObjectIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-ObjectIdentifierType": {

"$xsd-full-type": "xades-rw:ObjectIdentifierType",

"type": "object",

"properties": {

"id": {

"$ref": "#/definitions/xadesrw-IdentifierType"

},

"Description": {

"type": "string"

},

"DocumentationReferences": {

"$ref": "#/definitions/xadesrw-DocumentationReferencesType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ObjectIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Identifier | idType | [] |
| Description | desc | [] |
| DocumentationReferences | docRefType | [] |

[component ObjectIdentifier JSON schema details]

### Component Identifier

#### Semantics

[component Identifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The value element MUST contain one instance of a URI. [sub component value details]
* The optional Qualifier element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section Qualifier. [sub component Qualifier details]

Non-normative Comment:

[component Identifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type IdentifierType SHALL implement the requirements defined in the Identifier component.

The IdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="IdentifierType">

<xsd:simpleContent>

<xsd:extension base="xsd:anyURI">

<xsd:attribute name="Qualifier" type="xades-rw:QualifierType" use="optional"/>

</xsd:extension>

</xsd:simpleContent>

</xsd:complexType>

Each child element of IdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. The element 'value' holding a URI is represented by the component's XML tag text content. [component Identifier XML schema details]

#### JSON Syntax

The IdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the Identifier component.

The IdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-IdentifierType": {

"$xsd-full-type": "vr:IdentifierType xades-rw:IdentifierType",

"type": "object",

"properties": {

"value": {

"type": "string"

},

"Qualifier": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Identifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| value | value | [] |
| Qualifier | qualType | [] |

[component Identifier JSON schema details]

### Component DocumentationReferences

#### Semantics

[component DocumentationReferences normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The DocumentationReference element MUST occur 1 or more times containing a URI. [sub component DocumentationReference details]

Non-normative Comment:

[component DocumentationReferences non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type DocumentationReferencesType SHALL implement the requirements defined in the DocumentationReferences component.

The DocumentationReferencesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="DocumentationReferencesType">

<xsd:sequence maxOccurs="unbounded">

<xsd:element name="DocumentationReference" type="xsd:anyURI"/>

</xsd:sequence>

</xsd:complexType>

Each child element of DocumentationReferencesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DocumentationReferences XML schema details]

#### JSON Syntax

The DocumentationReferencesType JSON object SHALL implement in JSON syntax the requirements defined in the DocumentationReferences component.

The DocumentationReferencesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-DocumentationReferencesType": {

"$xsd-full-type": "xades-rw:DocumentationReferencesType",

"type": "object",

"properties": {

"DocumentationReference": {

"type": "array",

"items": {

"type": "string"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of DocumentationReferences component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DocumentationReference | docRef | [] |

[component DocumentationReferences JSON schema details]

### Component DigestAlgAndValue

#### Semantics

[component DigestAlgAndValue normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The DigestMethod element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section DigestMethod. [sub component DigestMethod details]
* The DigestValue element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section DigestValue. [sub component DigestValue details]

Non-normative Comment:

[component DigestAlgAndValue non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type DigestAlgAndValueType SHALL implement the requirements defined in the DigestAlgAndValue component.

The DigestAlgAndValueType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="DigestAlgAndValueType">

<xsd:sequence>

<xsd:element name="DigestMethod" type="ds-rw:DigestMethodType"/>

<xsd:element name="DigestValue" type="ds-rw:DigestValueType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of DigestAlgAndValueType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DigestAlgAndValue XML schema details]

#### JSON Syntax

The DigestAlgAndValueType JSON object SHALL implement in JSON syntax the requirements defined in the DigestAlgAndValue component.

The DigestAlgAndValueType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-DigestAlgAndValueType": {

"$xsd-full-type": "xades-rw:DigestAlgAndValueType",

"type": "object",

"properties": {

"DigestMethod": {

"$ref": "#/definitions/dsigrw-DigestMethodType"

},

"DigestValue": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of DigestAlgAndValue component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestMethod | alg | [] |
| DigestValue | val | [] |

[component DigestAlgAndValue JSON schema details]

### Component CertIDList

#### Semantics

[component CertIDList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Cert element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section CertID. [sub component Cert details]

Non-normative Comment:

[component CertIDList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CertIDListType SHALL implement the requirements defined in the CertIDList component.

The CertIDListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CertIDListType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="Cert" type="xades-rw:CertIDType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of CertIDListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertIDList XML schema details]

#### JSON Syntax

The CertIDListType JSON object SHALL implement in JSON syntax the requirements defined in the CertIDList component.

The CertIDListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CertIDListType": {

"$xsd-full-type": "xades-rw:CertIDListType",

"type": "object",

"properties": {

"Cert": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CertIDType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CertIDList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Cert | cert | [] |

[component CertIDList JSON schema details]

### Component CertID

#### Semantics

[component CertID normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CertDigest element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section DigestAlgAndValue. [sub component CertDigest details]
* The IssuerSerial element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section X509IssuerSerial. [sub component IssuerSerial details]
* The optional URI element MUST contain one instance of a URI. [sub component URI details]

Non-normative Comment:

[component CertID non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CertIDType SHALL implement the requirements defined in the CertID component.

The CertIDType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CertIDType">

<xsd:sequence>

<xsd:element name="CertDigest" type="xades-rw:DigestAlgAndValueType"/>

<xsd:element name="IssuerSerial" type="ds-rw:X509IssuerSerialType"/>

</xsd:sequence>

<xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>

</xsd:complexType>

Each child element of CertIDType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CertID XML schema details]

#### JSON Syntax

The CertIDType JSON object SHALL implement in JSON syntax the requirements defined in the CertID component.

The CertIDType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CertIDType": {

"$xsd-full-type": "xades-rw:CertIDType",

"type": "object",

"properties": {

"CertDigest": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

},

"IssuerSerial": {

"$ref": "#/definitions/dsigrw-X509IssuerSerialType"

},

"URI": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CertID component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CertDigest | certDigest | [] |
| IssuerSerial | issSerial | [] |
| URI | uri | [] |

[component CertID JSON schema details]

### Component SignaturePolicyIdentifier

#### Semantics

[component SignaturePolicyIdentifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SignaturePolicyId element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section SignaturePolicyId. [sub component SignaturePolicyId details]
* The SignaturePolicyImplied element MUST contain one instance of a boolean. [sub component SignaturePolicyImplied details]

Non-normative Comment:

[component SignaturePolicyIdentifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type SignaturePolicyIdentifierType SHALL implement the requirements defined in the SignaturePolicyIdentifier component.

The SignaturePolicyIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="SignaturePolicyIdentifierType">

<xsd:choice>

<xsd:element name="SignaturePolicyId" type="xades-rw:SignaturePolicyIdType"/>

<xsd:element name="SignaturePolicyImplied" type="xs:boolean"/>

</xsd:choice>

</xsd:complexType>

Each child element of SignaturePolicyIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignaturePolicyIdentifier XML schema details]

#### JSON Syntax

The SignaturePolicyIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the SignaturePolicyIdentifier component.

The SignaturePolicyIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-SignaturePolicyIdentifierType": {

"$xsd-full-type": "xades-rw:SignaturePolicyIdentifierType",

"type": "object",

"properties": {

"sigPolicyId": {

"$ref": "#/definitions/xadesrw-SignaturePolicyIdType"

},

"sigPolicyImplied": {

"type": "boolean"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of SignaturePolicyIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SignaturePolicyId | sigPolicyId | [] |
| SignaturePolicyImplied | sigPolicyImplied | [] |

[component SignaturePolicyIdentifier JSON schema details]

### Component SignaturePolicyId

#### Semantics

[component SignaturePolicyId normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SigPolicyId element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ObjectIdentifier. [sub component SigPolicyId details]
* The optional Transforms element MUST contain sub-component. A given element MUST satisfy the requirements specified in section Transforms. [sub component Transforms details]
* The SigPolicyHash element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section DigestAlgAndValue. [sub component SigPolicyHash details]
* The optional SigPolicyQualifiers element MUST contain sub-component. A given element MUST satisfy the requirements specified in section SigPolicyQualifiersList. [sub component SigPolicyQualifiers details]

Non-normative Comment:

[component SignaturePolicyId non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type SignaturePolicyIdType SHALL implement the requirements defined in the SignaturePolicyId component.

The SignaturePolicyIdType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="SignaturePolicyIdType">

<xsd:sequence>

<xsd:element name="SigPolicyId" type="xades-rw:ObjectIdentifierType"/>

<xsd:element minOccurs="0" name="Transforms" type="ds-rw:TransformsType"/>

<xsd:element name="SigPolicyHash" type="xades-rw:DigestAlgAndValueType"/>

<xsd:element minOccurs="0" name="SigPolicyQualifiers" type="xades-rw:SigPolicyQualifiersListType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of SignaturePolicyIdType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignaturePolicyId XML schema details]

#### JSON Syntax

The SignaturePolicyIdType JSON object SHALL implement in JSON syntax the requirements defined in the SignaturePolicyId component.

The SignaturePolicyIdType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-SignaturePolicyIdType": {

"$xsd-full-type": "xades-rw:SignaturePolicyIdType",

"type": "object",

"properties": {

"sigPolicyId": {

"$ref": "#/definitions/xadesrw-ObjectIdentifierType"

},

"transforms": {

"$ref": "#/definitions/dsigrw-TransformsType"

},

"sigPolicyHash": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

},

"sigPolicyQualifiers": {

"$ref": "#/definitions/xadesrw-SigPolicyQualifiersListType"

}

},

"required": ["sigPolicyId", "sigPolicyHash"]

}

Properties in the JSON schema above SHALL implement sub-component of SignaturePolicyId component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigPolicyId | sigPolicyId | [] |
| Transforms | transforms | [] |
| SigPolicyHash | sigPolicyHash | [] |
| SigPolicyQualifiers | sigPolicyQuals | [] |

[component SignaturePolicyId JSON schema details]

### Component SigPolicyQualifiersList

#### Semantics

[component SigPolicyQualifiersList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The SigPolicyQualifier element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section Any. [sub component SigPolicyQualifier details]

Non-normative Comment:

[component SigPolicyQualifiersList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type SigPolicyQualifiersListType SHALL implement the requirements defined in the SigPolicyQualifiersList component.

The SigPolicyQualifiersListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="SigPolicyQualifiersListType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="SigPolicyQualifier" type="xades-rw:AnyType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of SigPolicyQualifiersListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SigPolicyQualifiersList XML schema details]

#### JSON Syntax

The SigPolicyQualifiersListType JSON object SHALL implement in JSON syntax the requirements defined in the SigPolicyQualifiersList component.

The SigPolicyQualifiersListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-SigPolicyQualifiersListType": {

"$xsd-full-type": "xades-rw:SigPolicyQualifiersListType",

"type": "object",

"properties": {

"sigPolicyQualifier": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-AnyType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SigPolicyQualifiersList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| SigPolicyQualifier | sigPolicyQual | [] |

[component SigPolicyQualifiersList JSON schema details]

### Component Any

#### Semantics

[component Any normative details]

Below follows a list of the sub-components that MAY be present within this component:

A set of sub-components is inherited from component Base64Data and is not repeated here.

Non-normative Comment:

[component Any non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type AnyType SHALL implement the requirements defined in the Any component.

The AnyType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xs:complexType name="AnyType">

<xs:complexContent>

<xs:extension base="dsb:Base64DataType"/>

</xs:complexContent>

</xs:complexType>

Each child element of AnyType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component Any XML schema details]

#### JSON Syntax

The AnyType JSON object SHALL implement in JSON syntax the requirements defined in the Any component.

The AnyType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"dsb-AnyType": {

"$xsd-full-type": "dsb:AnyType xades-rw:AnyType",

"type": "object",

"properties": {

"ID": {

"type": "string"

},

"value": {

"type": "string"

},

"attRef": {

"$ref": "#/definitions/dsb-AttachmentReferenceType"

},

"mimeType": {

"type": "string"

},

"IDREF": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of Any component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |

[component Any JSON schema details]

### Component SignatureProductionPlace

#### Semantics

[component SignatureProductionPlace normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional City element MUST contain a string. [sub component City details]
* The optional StateOrProvince element MUST contain a string. [sub component StateOrProvince details]
* The optional PostalCode element MUST contain a string. [sub component PostalCode details]
* The optional CountryName element MUST contain a string. [sub component CountryName details]

Non-normative Comment:

[component SignatureProductionPlace non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type SignatureProductionPlaceType SHALL implement the requirements defined in the SignatureProductionPlace component.

The SignatureProductionPlaceType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="SignatureProductionPlaceType">

<xsd:sequence>

<xsd:element minOccurs="0" name="City" type="xsd:string"/>

<xsd:element minOccurs="0" name="StateOrProvince" type="xsd:string"/>

<xsd:element minOccurs="0" name="PostalCode" type="xsd:string"/>

<xsd:element minOccurs="0" name="CountryName" type="xsd:string"/>

</xsd:sequence>

</xsd:complexType>

Each child element of SignatureProductionPlaceType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component SignatureProductionPlace XML schema details]

#### JSON Syntax

The SignatureProductionPlaceType JSON object SHALL implement in JSON syntax the requirements defined in the SignatureProductionPlace component.

The SignatureProductionPlaceType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-SignatureProductionPlaceType": {

"$xsd-full-type": "xades-rw:SignatureProductionPlaceType",

"type": "object",

"properties": {

"City": {

"type": "string"

},

"StateOrProvince": {

"type": "string"

},

"PostalCode": {

"type": "string"

},

"CountryName": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of SignatureProductionPlace component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| City | city | [] |
| StateOrProvince | stateOrProv | [] |
| PostalCode | postalCode | [] |
| CountryName | country | [] |

[component SignatureProductionPlace JSON schema details]

### Component ClaimedRolesList

#### Semantics

[component ClaimedRolesList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ClaimedRole element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section Any. [sub component ClaimedRole details]

Non-normative Comment:

[component ClaimedRolesList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type ClaimedRolesListType SHALL implement the requirements defined in the ClaimedRolesList component.

The ClaimedRolesListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="ClaimedRolesListType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="ClaimedRole" type="xades-rw:AnyType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of ClaimedRolesListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ClaimedRolesList XML schema details]

#### JSON Syntax

The ClaimedRolesListType JSON object SHALL implement in JSON syntax the requirements defined in the ClaimedRolesList component.

The ClaimedRolesListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-ClaimedRolesListType": {

"$xsd-full-type": "xades-rw:ClaimedRolesListType",

"type": "object",

"properties": {

"ClaimedRole": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-AnyType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of ClaimedRolesList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ClaimedRole | claimedRole | [] |

[component ClaimedRolesList JSON schema details]

### Component CRLIdentifier

#### Semantics

[component CRLIdentifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The Issuer element MUST contain one instance of a string. [sub component Issuer details]
* The IssueTime element MUST contain one instance of a date/time value. [sub component IssueTime details]
* The optional Number element MUST contain an integer. [sub component Number details]
* The optional URI element MUST contain one instance of a URI. [sub component URI details]

Non-normative Comment:

[component CRLIdentifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CRLIdentifierType SHALL implement the requirements defined in the CRLIdentifier component.

The CRLIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CRLIdentifierType">

<xsd:sequence>

<xsd:element name="Issuer" type="xsd:string"/>

<xsd:element name="IssueTime" type="xsd:dateTime"/>

<xsd:element minOccurs="0" name="Number" type="xsd:integer"/>

</xsd:sequence>

<xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>

</xsd:complexType>

Each child element of CRLIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CRLIdentifier XML schema details]

#### JSON Syntax

The CRLIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the CRLIdentifier component.

The CRLIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CRLIdentifierType": {

"$xsd-full-type": "xades-rw:CRLIdentifierType",

"type": "object",

"properties": {

"Issuer": {

"type": "string"

},

"IssueTime": {

"type": "integer",

"format": "utc-millisec"

},

"Number": {

"type": "integer"

},

"URI": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CRLIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Issuer | iss | [] |
| IssueTime | issTime | [] |
| Number | num | [] |
| URI | uri | [] |

[component CRLIdentifier JSON schema details]

### Component OCSPIdentifier

#### Semantics

[component OCSPIdentifier normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ResponderID element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ResponderID. [sub component ResponderID details]
* The ProducedAt element MUST contain one instance of a date/time value. [sub component ProducedAt details]
* The optional URI element MUST contain one instance of a URI. [sub component URI details]

Non-normative Comment:

[component OCSPIdentifier non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type OCSPIdentifierType SHALL implement the requirements defined in the OCSPIdentifier component.

The OCSPIdentifierType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="OCSPIdentifierType">

<xsd:sequence>

<xsd:element name="ResponderID" type="xades-rw:ResponderIDType"/>

<xsd:element name="ProducedAt" type="xsd:dateTime"/>

</xsd:sequence>

<xsd:attribute name="URI" type="xsd:anyURI" use="optional"/>

</xsd:complexType>

Each child element of OCSPIdentifierType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OCSPIdentifier XML schema details]

#### JSON Syntax

The OCSPIdentifierType JSON object SHALL implement in JSON syntax the requirements defined in the OCSPIdentifier component.

The OCSPIdentifierType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-OCSPIdentifierType": {

"$xsd-full-type": "xades-rw:OCSPIdentifierType",

"type": "object",

"properties": {

"ResponderID": {

"$ref": "#/definitions/xadesrw-ResponderIDType"

},

"ProducedAt": {

"type": "integer",

"format": "utc-millisec"

},

"URI": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of OCSPIdentifier component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ResponderID | respId | [] |
| ProducedAt | prodAt | [] |
| URI | uri | [] |

[component OCSPIdentifier JSON schema details]

### Component ResponderID

#### Semantics

[component ResponderID normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The ByName element MUST contain one instance of a string. [sub component ByName details]
* The ByKey element MUST contain one instance of base64 encoded binary data. [sub component ByKey details]

Non-normative Comment:

[component ResponderID non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type ResponderIDType SHALL implement the requirements defined in the ResponderID component.

The ResponderIDType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="ResponderIDType">

<xsd:choice>

<xsd:element name="ByName" type="xsd:string"/>

<xsd:element name="ByKey" type="xsd:base64Binary"/>

</xsd:choice>

</xsd:complexType>

Each child element of ResponderIDType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component ResponderID XML schema details]

#### JSON Syntax

The ResponderIDType JSON object SHALL implement in JSON syntax the requirements defined in the ResponderID component.

The ResponderIDType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-ResponderIDType": {

"$xsd-full-type": "xades-rw:ResponderIDType",

"type": "object",

"properties": {

"ByName": {

"type": "string"

},

"ByKey": {

"type": "string"

}

},

"minProperties": 1,

"maxProperties": 1

}

Properties in the JSON schema above SHALL implement sub-component of ResponderID component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| ByName | byName | [] |
| ByKey | byKey | [] |

[component ResponderID JSON schema details]

### Component DataObjectFormat

#### Semantics

[component DataObjectFormat normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional Description element MUST contain a string. [sub component Description details]
* The optional ObjectIdentifier element MUST contain sub-component. A given element MUST satisfy the requirements specified in section ObjectIdentifier. [sub component ObjectIdentifier details]
* The optional MimeType element MUST contain a string. [sub component MimeType details]
* The optional Encoding element MUST contain a URI. [sub component Encoding details]
* The ObjectReference element MUST contain one instance of a URI. [sub component ObjectReference details]

Non-normative Comment:

[component DataObjectFormat non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type DataObjectFormatType SHALL implement the requirements defined in the DataObjectFormat component.

The DataObjectFormatType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="DataObjectFormatType">

<xsd:sequence>

<xsd:element minOccurs="0" name="Description" type="xsd:string"/>

<xsd:element minOccurs="0" name="ObjectIdentifier" type="xades-rw:ObjectIdentifierType"/>

<xsd:element minOccurs="0" name="MimeType" type="xsd:string"/>

<xsd:element minOccurs="0" name="Encoding" type="xsd:anyURI"/>

</xsd:sequence>

<xsd:attribute name="ObjectReference" type="xsd:anyURI" use="required"/>

</xsd:complexType>

Each child element of DataObjectFormatType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component DataObjectFormat XML schema details]

#### JSON Syntax

The DataObjectFormatType JSON object SHALL implement in JSON syntax the requirements defined in the DataObjectFormat component.

The DataObjectFormatType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-DataObjectFormatType": {

"$xsd-full-type": "xades-rw:DataObjectFormatType",

"type": "object",

"properties": {

"Description": {

"type": "string"

},

"ObjectIdentifier": {

"$ref": "#/definitions/xadesrw-ObjectIdentifierType"

},

"mimeType": {

"type": "string"

},

"Encoding": {

"type": "string"

},

"ObjectReference": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of DataObjectFormat component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| Description | desc | [] |
| ObjectIdentifier | objId | [] |
| MimeType | mimeType | [] |
| Encoding | enc | [] |
| ObjectReference | objRef | [] |

[component DataObjectFormat JSON schema details]

### Component CommitmentTypeIndication

#### Semantics

[component CommitmentTypeIndication normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CommitmentTypeId element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section ObjectIdentifier. [sub component CommitmentTypeId details]
* The ObjectReference element MUST occur 1 or more times containing a URI. [sub component ObjectReference details]
* The AllSignedDataObjects element MUST contain one instance of a boolean. [sub component AllSignedDataObjects details]
* The optional CommitmentTypeQualifiers element MUST contain sub-component. A given element MUST satisfy the requirements specified in section CommitmentTypeQualifiersList. [sub component CommitmentTypeQualifiers details]

Non-normative Comment:

[component CommitmentTypeIndication non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CommitmentTypeIndicationType SHALL implement the requirements defined in the CommitmentTypeIndication component.

The CommitmentTypeIndicationType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CommitmentTypeIndicationType">

<xsd:sequence>

<xsd:element name="CommitmentTypeId" type="xades-rw:ObjectIdentifierType"/>

<xsd:choice>

<xsd:element maxOccurs="unbounded" name="ObjectReference" type="xsd:anyURI"/>

<xsd:element name="AllSignedDataObjects" type="xs:boolean"/>

</xsd:choice>

<xsd:element minOccurs="0" name="CommitmentTypeQualifiers" type="xades-rw:CommitmentTypeQualifiersListType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of CommitmentTypeIndicationType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CommitmentTypeIndication XML schema details]

#### JSON Syntax

The CommitmentTypeIndicationType JSON object SHALL implement in JSON syntax the requirements defined in the CommitmentTypeIndication component.

The CommitmentTypeIndicationType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CommitmentTypeIndicationType": {

"$xsd-full-type": "xades-rw:CommitmentTypeIndicationType",

"type": "object",

"properties": {

"CommitmentTypeId": {

"$ref": "#/definitions/xadesrw-ObjectIdentifierType"

},

"ObjectReference": {

"type": "array",

"items": {

"type": "string"

}

},

"AllSignedDataObjects": {

"type": "boolean"

},

"CommitmentTypeQualifiers": {

"$ref": "#/definitions/xadesrw-CommitmentTypeQualifiersListType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CommitmentTypeIndication component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CommitmentTypeId | commTypeId | [] |
| ObjectReference | objRef | [] |
| AllSignedDataObjects | allSigDataObj | [] |
| CommitmentTypeQualifiers | commTypeQuals | [] |

[component CommitmentTypeIndication JSON schema details]

### Component CommitmentTypeQualifiersList

#### Semantics

[component CommitmentTypeQualifiersList normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional CommitmentTypeQualifier element MAY occur zero or more times containing sub-component. If present each instance MUST satisfy the requirements specified in section Any. [sub component CommitmentTypeQualifier details]

Non-normative Comment:

[component CommitmentTypeQualifiersList non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CommitmentTypeQualifiersListType SHALL implement the requirements defined in the CommitmentTypeQualifiersList component.

The CommitmentTypeQualifiersListType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CommitmentTypeQualifiersListType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" minOccurs="0" name="CommitmentTypeQualifier" type="xades-rw:AnyType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of CommitmentTypeQualifiersListType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CommitmentTypeQualifiersList XML schema details]

#### JSON Syntax

The CommitmentTypeQualifiersListType JSON object SHALL implement in JSON syntax the requirements defined in the CommitmentTypeQualifiersList component.

The CommitmentTypeQualifiersListType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CommitmentTypeQualifiersListType": {

"$xsd-full-type": "xades-rw:CommitmentTypeQualifiersListType",

"type": "object",

"properties": {

"CommitmentTypeQualifier": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-AnyType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CommitmentTypeQualifiersList component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CommitmentTypeQualifier | commTypeQual | [] |

[component CommitmentTypeQualifiersList JSON schema details]

### Component CompleteCertificateRefs

#### Semantics

[component CompleteCertificateRefs normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CertRefs element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section CertIDList. [sub component CertRefs details]
* The optional Id element MUST contain one instance of a unique identifier. [sub component Id details]

Non-normative Comment:

[component CompleteCertificateRefs non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CompleteCertificateRefsType SHALL implement the requirements defined in the CompleteCertificateRefs component.

The CompleteCertificateRefsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CompleteCertificateRefsType">

<xsd:sequence>

<xsd:element name="CertRefs" type="xades-rw:CertIDListType"/>

</xsd:sequence>

<xsd:attribute name="Id" type="xsd:ID" use="optional"/>

</xsd:complexType>

Each child element of CompleteCertificateRefsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CompleteCertificateRefs XML schema details]

#### JSON Syntax

The CompleteCertificateRefsType JSON object SHALL implement in JSON syntax the requirements defined in the CompleteCertificateRefs component.

The CompleteCertificateRefsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CompleteCertificateRefsType": {

"$xsd-full-type": "xades-rw:CompleteCertificateRefsType",

"type": "object",

"properties": {

"CertRefs": {

"$ref": "#/definitions/xadesrw-CertIDListType"

},

"Id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CompleteCertificateRefs component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CertRefs | certRefs | [] |
| Id | id | [] |

[component CompleteCertificateRefs JSON schema details]

### Component CompleteRevocationRefs

#### Semantics

[component CompleteRevocationRefs normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The optional CRLRefs element MUST contain sub-component. A given element MUST satisfy the requirements specified in section CRLRefs. [sub component CRLRefs details]
* The optional OCSPRefs element MUST contain sub-component. A given element MUST satisfy the requirements specified in section OCSPRefs. [sub component OCSPRefs details]
* The optional OtherRefs element MUST contain sub-component. A given element MUST satisfy the requirements specified in section OtherCertStatusRefs. [sub component OtherRefs details]
* The optional Id element MUST contain one instance of a unique identifier. [sub component Id details]

Non-normative Comment:

[component CompleteRevocationRefs non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CompleteRevocationRefsType SHALL implement the requirements defined in the CompleteRevocationRefs component.

The CompleteRevocationRefsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CompleteRevocationRefsType">

<xsd:sequence>

<xsd:element minOccurs="0" name="CRLRefs" type="xades-rw:CRLRefsType"/>

<xsd:element minOccurs="0" name="OCSPRefs" type="xades-rw:OCSPRefsType"/>

<xsd:element minOccurs="0" name="OtherRefs" type="xades-rw:OtherCertStatusRefsType"/>

</xsd:sequence>

<xsd:attribute name="Id" type="xsd:ID" use="optional"/>

</xsd:complexType>

Each child element of CompleteRevocationRefsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CompleteRevocationRefs XML schema details]

#### JSON Syntax

The CompleteRevocationRefsType JSON object SHALL implement in JSON syntax the requirements defined in the CompleteRevocationRefs component.

The CompleteRevocationRefsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CompleteRevocationRefsType": {

"$xsd-full-type": "xades-rw:CompleteRevocationRefsType",

"type": "object",

"properties": {

"crlrefs": {

"$ref": "#/definitions/xadesrw-CRLRefsType"

},

"ocsprefs": {

"$ref": "#/definitions/xadesrw-OCSPRefsType"

},

"CRLRefs": {

"$ref": "#/definitions/xadesrw-CRLRefsType"

},

"OCSPRefs": {

"$ref": "#/definitions/xadesrw-OCSPRefsType"

},

"otherRefs": {

"$ref": "#/definitions/xadesrw-OtherCertStatusRefsType"

},

"Id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CompleteRevocationRefs component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CRLRefs | crlRefs | [] |
| OCSPRefs | ocspRefs | [] |
| OtherRefs | otherRefs | [] |
| Id | id | [] |

[component CompleteRevocationRefs JSON schema details]

### Component CRLRefs

#### Semantics

[component CRLRefs normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The CRLRef element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section CRLRef. [sub component CRLRef details]

Non-normative Comment:

[component CRLRefs non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CRLRefsType SHALL implement the requirements defined in the CRLRefs component.

The CRLRefsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CRLRefsType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="CRLRef" type="xades-rw:CRLRefType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of CRLRefsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CRLRefs XML schema details]

#### JSON Syntax

The CRLRefsType JSON object SHALL implement in JSON syntax the requirements defined in the CRLRefs component.

The CRLRefsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CRLRefsType": {

"$xsd-full-type": "xades-rw:CRLRefsType",

"type": "object",

"properties": {

"crlref": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CRLRefType"

}

},

"CRLRef": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-CRLRefType"

}

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CRLRefs component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| CRLRef | crlRef | [] |

[component CRLRefs JSON schema details]

### Component CRLRef

#### Semantics

[component CRLRef normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The DigestAlgAndValue element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section DigestAlgAndValue. [sub component DigestAlgAndValue details]
* The optional CRLIdentifier element MUST contain sub-component. A given element MUST satisfy the requirements specified in section CRLIdentifier. [sub component CRLIdentifier details]

Non-normative Comment:

[component CRLRef non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type CRLRefType SHALL implement the requirements defined in the CRLRef component.

The CRLRefType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="CRLRefType">

<xsd:sequence>

<xsd:element name="DigestAlgAndValue" type="xades-rw:DigestAlgAndValueType"/>

<xsd:element minOccurs="0" name="CRLIdentifier" type="xades-rw:CRLIdentifierType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of CRLRefType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component CRLRef XML schema details]

#### JSON Syntax

The CRLRefType JSON object SHALL implement in JSON syntax the requirements defined in the CRLRef component.

The CRLRefType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-CRLRefType": {

"$xsd-full-type": "xades-rw:CRLRefType",

"type": "object",

"properties": {

"crlidentifier": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

},

"DigestAlgAndValue": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

},

"CRLIdentifier": {

"$ref": "#/definitions/xadesrw-CRLIdentifierType"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of CRLRef component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| DigestAlgAndValue | digAlgVal | [] |
| CRLIdentifier | crlId | [] |

[component CRLRef JSON schema details]

### Component OCSPRefs

#### Semantics

[component OCSPRefs normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The OCSPRef element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section OCSPRef. [sub component OCSPRef details]

Non-normative Comment:

[component OCSPRefs non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type OCSPRefsType SHALL implement the requirements defined in the OCSPRefs component.

The OCSPRefsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="OCSPRefsType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="OCSPRef" type="xades-rw:OCSPRefType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of OCSPRefsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OCSPRefs XML schema details]

#### JSON Syntax

The OCSPRefsType JSON object SHALL implement in JSON syntax the requirements defined in the OCSPRefs component.

The OCSPRefsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-OCSPRefsType": {

"$xsd-full-type": "xades-rw:OCSPRefsType",

"type": "object",

"properties": {

"ocspref": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-OCSPRefType"

}

},

"ocspRef": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-OCSPRefType"

}

}

},

"required": ["ocspRef"]

}

Properties in the JSON schema above SHALL implement sub-component of OCSPRefs component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OCSPRef | ocspRef | [] |

[component OCSPRefs JSON schema details]

### Component OCSPRef

#### Semantics

[component OCSPRef normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The OCSPIdentifier element MUST contain one instance of sub-component. This element MUST satisfy the requirements specified in section OCSPIdentifier. [sub component OCSPIdentifier details]
* The optional DigestAlgAndValue element MUST contain sub-component. A given element MUST satisfy the requirements specified in section DigestAlgAndValue. [sub component DigestAlgAndValue details]

Non-normative Comment:

[component OCSPRef non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type OCSPRefType SHALL implement the requirements defined in the OCSPRef component.

The OCSPRefType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="OCSPRefType">

<xsd:sequence>

<xsd:element name="OCSPIdentifier" type="xades-rw:OCSPIdentifierType"/>

<xsd:element minOccurs="0" name="DigestAlgAndValue" type="xades-rw:DigestAlgAndValueType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of OCSPRefType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OCSPRef XML schema details]

#### JSON Syntax

The OCSPRefType JSON object SHALL implement in JSON syntax the requirements defined in the OCSPRef component.

The OCSPRefType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-OCSPRefType": {

"$xsd-full-type": "xades-rw:OCSPRefType",

"type": "object",

"properties": {

"ocspidentifier": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"ocspId": {

"$ref": "#/definitions/xadesrw-OCSPIdentifierType"

},

"digAlgVal": {

"$ref": "#/definitions/xadesrw-DigestAlgAndValueType"

}

},

"required": ["ocspId"]

}

Properties in the JSON schema above SHALL implement sub-component of OCSPRef component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OCSPIdentifier | ocspId | [] |
| DigestAlgAndValue | digAlgVal | [] |

[component OCSPRef JSON schema details]

### Component OtherCertStatusRefs

#### Semantics

[component OtherCertStatusRefs normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The OtherRef element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section Any. [sub component OtherRef details]

Non-normative Comment:

[component OtherCertStatusRefs non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type OtherCertStatusRefsType SHALL implement the requirements defined in the OtherCertStatusRefs component.

The OtherCertStatusRefsType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="OtherCertStatusRefsType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="OtherRef" type="xades-rw:AnyType"/>

</xsd:sequence>

</xsd:complexType>

Each child element of OtherCertStatusRefsType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component OtherCertStatusRefs XML schema details]

#### JSON Syntax

The OtherCertStatusRefsType JSON object SHALL implement in JSON syntax the requirements defined in the OtherCertStatusRefs component.

The OtherCertStatusRefsType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-OtherCertStatusRefsType": {

"$xsd-full-type": "xades-rw:OtherCertStatusRefsType",

"type": "object",

"properties": {

"otherRef": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-AnyType"

}

}

},

"required": ["otherRef"]

}

Properties in the JSON schema above SHALL implement sub-component of OtherCertStatusRefs component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| OtherRef | otherRef | [] |

[component OtherCertStatusRefs JSON schema details]

### Component UnsignedDataObjectProperties

#### Semantics

[component UnsignedDataObjectProperties normative details]

Below follows a list of the sub-components that MAY be present within this component:

* The UnsignedDataObjectProperty element MUST occur 1 or more times containing sub-component. Each instance MUST satisfy the requirements specified in section Any. [sub component UnsignedDataObjectProperty details]
* The optional Id element MUST contain one instance of a unique identifier. [sub component Id details]

Non-normative Comment:

[component UnsignedDataObjectProperties non normative details]

#### XML Syntax

The XML element is defined in the XML namespace 'urn:oasis:names:tc:dss-x:2.0:xades:rewritten' .The XML type UnsignedDataObjectPropertiesType SHALL implement the requirements defined in the UnsignedDataObjectProperties component.

The UnsignedDataObjectPropertiesType XML element SHALL be defined as in XML Schema file [FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE XSD], and is copied below for information.

<xsd:complexType name="UnsignedDataObjectPropertiesType">

<xsd:sequence>

<xsd:element maxOccurs="unbounded" name="UnsignedDataObjectProperty" type="xades-rw:AnyType"/>

</xsd:sequence>

<xsd:attribute name="Id" type="xsd:ID" use="optional"/>

</xsd:complexType>

Each child element of UnsignedDataObjectPropertiesType XML element SHALL implement in XML syntax the sub-component that has a name equal to its local name. [component UnsignedDataObjectProperties XML schema details]

#### JSON Syntax

The UnsignedDataObjectPropertiesType JSON object SHALL implement in JSON syntax the requirements defined in the UnsignedDataObjectProperties component.

The UnsignedDataObjectPropertiesType JSON object SHALL be defined as in JSON Schema file [JSON SCHEMA FILE NAME] whose location is detailed in clause [CLAUSE FOR LINK TO THE JSON SCHEMA FILE], and is copied below for information.

"xadesrw-UnsignedDataObjectPropertiesType": {

"$xsd-full-type": "xades-rw:UnsignedDataObjectPropertiesType",

"type": "object",

"properties": {

"UnsignedDataObjectProperty": {

"type": "array",

"items": {

"$ref": "#/definitions/xadesrw-AnyType"

}

},

"Id": {

"type": "string"

}

}

}

Properties in the JSON schema above SHALL implement sub-component of UnsignedDataObjectProperties component mapped by names as shown in the table below.

|  |  |  |
| --- | --- | --- |
| Element | Implementing JSON member name | Comments |
| UnsignedDataObjectProperty | unsignedDataObjProps | [] |
| Id | id | [] |

[component UnsignedDataObjectProperties JSON schema details]

# Conformance

This profile defines two conformance levels:

* Level 1 ‑ “Basic”,
* Level 2 ‑ “Comprehensive”

## Level 1 – “Basic”

The conformance level “Basic” allows to return individual verification results for each signature contained in a VerifyRequest component. For this purpose the VerifyResponse component MUST contain in OptionalOutputs a VerificationReport element, as specified in Section 3.2. The VerificationReport element MUST contain an IndividualSignatureReport element (see Section 3.3) for each signature or time stamp (i.e. SignatureObject component) contained in the VerifyRequest component.

The Details component within IndividualSignatureReport MAY contain other elements, such as the Optional Outputs defined in Section 4.5 of **[DSSCore]**.

## Level 2 – “Comprehensive”

The conformance level “Comprehensive” comprises all requirements of conformance Level 1 (“Basic”), as explained in Section 4.1. Furthermore, the Details component within each IndividualReport MUST contain exactly one object-specific element, which documents the detailed verification results for the signatures or validation data under consideration. It is REQUIRED in this conformance level that certificate values and revocation values are included into the verification report if requested by the IncludeCertificateValues- and IncludeRevocationValues-element within the ReturnVerifcationReport component (cf. Section 3.1).

The object-specific detail elements defined in this specification are given as follows:

* DetailedSignatureReport (cf. Section 3.5) ‑ is used for the verification of (advanced) electronic signatures.
* IndividualTimeStampReport (cf. Section 3.5.5) – is used for the verification of individual time stamps according to **[RFC3161]**, which are not included in a signature.
* IndividualCertificateReport (cf. Section 3.5.6) – is used for the verification of individual certificates according to **[RFC5280]**, which are not included in a signature.
* IndividualAttributeCertificateReport (cf. Section 3.5.7) ‑ is used for the verification of individual attribute certificates according to **[RFC3281]**, which are not included in a signature.
* IndividualCRLReport (cf. Section 3.5.8) ‑ is used for the verification of individual CRLs according to **[RFC5280]**, which are not included in a signature.
* IndividualOCSPReport (cf. Section 3.5.9) ‑ is used for the verification of individual OCSP-responses according to **[RFC2560]**, which are not included in a signature.
* EvidenceRecordReport (cf. Section 3.5.10) – is used for the verification of evidence records according to **[RFC4998]**.

Other object-specific detail elements MAY be defined in other profiles.

1. Index

DateTime, 12

1. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| [Rev number] | [Rev Date] | Andreas Kuehne and Stefan Hagen | Initial Draft version with feedback from the TC |