|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Title\*:** | Comments from TC ESI to OASIS DSS-X TC on DSS-X V2 | | |
|  |  | | |
| from **Source**\*: | ETSI | | |
| Contact: | Sonia Compans | | |
|  |  | | |
| input for **Committee**\***:** | ESI | | |
|  |  | | |
| Contribution **For\*:** | Decision | **X** |  |
|  | Discussion |  |  |
|  | Information |  |  |
|  |  | | |
| Submission date**\***: | 2018-09-24 | | |
|  |  | | |
| Meeting & Allocation: | **ESI-Review comments on draft OASIS DSS V2 spec** - | | |
| Relevant WI(s), or deliverable(s): |  | | |
|  | | | |

**Decision/action requested:** Please approve

**ABSTRACT:***following steering committee call on 17/09 and addition of 2 other editorial comments, the resulting pre-agreed comments are now for ESI approval for submission to OASIS DSS-X TC by 28 September*

|  |  |
| --- | --- |
| **Comments from ETSI TC ESI for OASIS DSS-X TC on core v2.0 Committee Specification**  **v01 Public Review Draft v 01 28th September 2018** | |
| Comment #1 (T) | |
| Clause 4.1.5 Component AttachmentReference. Second paragraph is misleading:  COMMENT:  Original text: “Below follows a list of the sub-components that MAY be present within this component”, and then follows a list of two subcomponents, one of which is AttRefURI  Then the XML schema piece in clause 4.1.5.2 shows that the AttRefURI attribute is MANDATORY.  In summary the introductory text uses “the sub-components that MAY be present” while one of them is actually mandatory.  In addition to that the first sentence after the former paragraph is: “The optional DigestInfo element MAY occur zero or more times containing a sub-component.” So the leading sentence says “sub-components that MAY be present:”, and the sentences start speaking of elements that contain sub-components…..this seems an unadequate wording that does not clarify what is the relationship between element and component, and clearly indicates that are not the same.  REQUEST: modify the introductory text so that the MAY disappears. Below follows a proposal:  “This component:  MAY contain zero or more DigestInfo sub-components……  …  MUST contain one AttRefURI sub-component. The value of this sub-component MUST be an URI…  “  By doing so, there is no possible ambiguity in the reading. The text says whether the sub-component is mandatory (MUST contain) or optional (MAY contain).  In the case that a component has a choice and also mandatory and optional components outside the choice, the text should read something like:  “This component:  MUST contain EITHER:  One XXX sub-compoonent (….)OR  One YYYY sub-component (….) OR  One (or one or more) ZZZZ sub-component  MAY contain one (or the suitable cardinality) of AAA sub-component (….)  MUST contain one (or the suitable cardinality) of BBB sub-component (….)  “ | |
|  | |
| This type of misleading constructs appear in a number of other clauses. Below follows a list of the ones detected. | |
| 1. Clause 4.1.11 Component ResponseBase. Result is mandatory. | |
| 2. Clause 4.2.3 Document. Base64Data is mandatory. | |
| 3. Clause 4.2.4 TransformedData. Base64Data is mandatory | |
| 4. Clause 4.2.5 DocumentHash. Base64Data is mandatory | |
| 5. Clause 4.2.8 SignatureObject. In this case the content is a choice between two sub-components and additionally there may be another sub-component. Try to specify exactly this semantics. Otherwise the syntax goes beyond the semantics (the semantics does not say anything about Base64Signature and SignaturePtr being a choice). | |
| 6. Clause 4.3.9, ClaimedIdentity. Name is mandatory | |
| 7. | |
| 8. | |
| 9. | |
| CONCLUSION: Accepted to pass this comment to DSS-X TC | |
| Comment #2 (T) | |
| Clause 4.2.1. “An input document can also be a <ds:Manifest>, allowing the client to handle manifest creation while using the server to create the rest of the signature.”  An initial search of the keywords ds:Manifest and Manifest does not hit any sentence explaining or developing how this situation can be managed and what does “allowing the client to handle manifest creation” actually means.  REQUEST: CLARIFY THE TEXT, MAYBE ADDING SOME ADDITIONAL TEXT IN THE PROCESSING CLAUSES FOR THE CASE THE INPUTDOCUMENT CONTAINS A DS:MANIFEST, IF NOT, DELETE IT. | |
| AK: Just took text as it was in DSS v1.0: it was as it is. There is no problem in adding more text. Most of the text was taken from old v1.0 to prevent additional problems to compliant implementations. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #3 (T) | |
| Clause 4.2.1.2. The new XML Schema does not allow that in a certain request appear sub-components of different type. With the current XML Schema one instance of InputDocumentsType can only have EITHER one or more Document, OR one or more TransformedData, OR one or more DocumentHash. An InputDocumentsType instance MAY NOT have, with the current definition one Document, one TransformedData and one DocumentHash, for instance. Core v1.0 actually ALLOWED THE PRESENCE OF INPUTS OF DIFFERENT NATURE.  REQUEST: MODIFY THE XML SCHEMA FOR ALLOWING PRESENCE OF INPUTS OF DIFFERENT NATURE (A CHOICE WITH MAXOCCURS=2UNBOUNDED” WOULD MAKE IT. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #4. (E) | |
| Clause 4.2.6. SignRequest. When one reads the text “Below follows a list of the sub-components that MAY be present within this component:” one thinks that it is misleading, as this component inherits from RequestBaseType and other sub-components may appear. At the end of all the requirements for the new subcomponents, the text fixes the situation “A set of sub-components is inherited from component 5.1.10 and is not repeated here”  REQUEST: makes it clear since the very beginning that the sub-components of RequestBaseType and the new ones listed below may appear. Also include the name of the base type RequestBaseType and not only the clause number for facilitating reading. | |
| This situation appears in a good number (if not in all) those clauses where a type is derived from a base type.  REQUEST: make changes in all these clauses as indicated above. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #5 (E) | |
| Clause 4.2.7 SignResponse. As previous comment. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #6 (T) | |
| Clause 4.3.5.  COMMENT: The definition of the component ReturnAugmentedSignature allows for multiple occurrences of this component. One could think that this essentially means that in theory one request could order to the server to augment one signature to one level, another signature to another level, and so on. However, the contents of the type AugmentSignatureInstructionType DOES NOT allow to refer to any signature in the request, so in the end it seems as if one would like to request to the server to augment all the signatures in the request to different levels and retrieve them (i.e., would be like requesting to augment one signature to XAdES-B-T level, then to XAdES-B-LT, and then to XAdES-B-LTA, and return the three augmented signatures). In principle, this does not seem to make lot of sense.  ETSI ESI has decided that in one request the ReturnAugmentedSignature will appear ONLY ONE TIME, with ONE URI value and this will mean that the server is requested to augment all the signatures to the identified level, if possible. The rationale behind is to make things easier for the server.  REQUEST: Modify the specification of the ReturnAugmentedSignature and its type to make it compatible with ETSI ESI, i.e., make maxOccurs of ReturnAugmentedSignature =”1”, and specify that the server shall try to augment all the signatures to the level identified in that component. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #6. (T) | |
| Clause 4.3.11. AugmentSignatureInstruction.  COMMENT:   1. There is a lack of coherence in the names of certain clauses: for instance clause 4.3.5.2 specify an element called ReturnAugmentedSignature, of type AugmentSignatureInstructionType. There is NO component AugmentSignatureInstruction. There are instances of AugmentSignatureInstructionType. Avoid missunderstandings changing the titles of the clauses. 2. The name ReturnAugmentedSignatureType, which is the name selected by ESI for the type in TS 119 442, illustrates with much more precission what the type is about than AugmentSignatureInstructionType does. 3. The attribute “Type” is misleading and does not match the terminology used in the ENs that define the AdES signatures: the different combinations of properties/attributes receive the name of “level”, not “type”. So, they speak of XAdES level B-B, XAdES level E-BES, etc. 4. The attribute “Type” is optional. This essentially means that this attribute could not be present. The document does not provide any specification of to what level the server has to augment the signatures.   REQUEST :   1. Change the name of the type AugmentSignatureInstructionType to ReturnAugmentedSignatureType. 2. Change the name clause 4.3.11 to “Instances of ReturnAugmentedSignatureType”, the name of clause 4.3.11.1 to Instances of ReturnAugmentedSignatureType – JSON syntax, and the name of clause 4.3.11.2 to Instances of ReturnAugmentedSignatureType – XML syntax 3. Change the name of the attribute from “Type” to “Level”. 4. Make the attribute (Level) mandatory (required).   ETSI ESI exchanged information with the editors of DSS core v2.0 about the suitability of not using the term upgrade, which the document has done, however the name does not seem the more appropriate. Instead ReturnAugmentedSignature clearly indicates the purpose of the input: it is an order to the server to augment a signature and to return it. AugmentSignatureInstruction is much more ambiguous as the word Instruction can actually be anything…and from the text that specifies this component one always concludes that the goal is that the server augments a signature and returns it to the client.  The request for changing the name of the attribute to Level is even stronger: “Level” is the formalized term used by all the ETSI ENs on signature formats (CAdES, PAdES, and XAdES) for identifying different combinations of incorporated attributes/properties to the signature. Use of “Type” in the DSS-X protocols will certainly generate at least doubts among users of ETSI ENs, which are used to speak of XAdES-BES level signatures, XAdES-B-B level signatures, etc. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #7 (T) | |
| Clause 4.3.4.2. XML Schema for OptionalInputsSignType.    The definition of SignRequest includes the following sub-component:  <xs:element minOccurs="0" name="OptionalInputs"  type="dss2:OptionalInputsSignType"/>  And then the definition of OptionalInputsSignType is as follows:  <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="SignatureQualityLevel"  type="xs:anyURI"/>  </xs:choice>  </xs:sequence>  </xs:extension>  </xs:complexContent> </xs:complexType>  With these definitions one SignRequest CAN HAVE ONLYONE OPTIONAL INPUT (PLEASE NOTE THAT IN THE DEFINITION OF OptionalInputsBaseType THE SEQUENCE ONLY HAS ONE ELEMENT: A CHOICE BETWEEN A NUMBER OF OPTIONS, BUT ONLY ONE. Note also the absence of maxOccurs in the first schema piece for the element OptionalInputs.  REQUEST: EITHER SUPPRESS THE CHOICE WITHIN THE SEQUENCE OR ADD A MAXOCCURS TO THE SEQUENCE. THE FIRST OPTION WOULD RESTRICT THE ORDER OF APPEARANCE OF THE OPTIONALINPUTS IN THE REQUEST. THE SECOND WOULD ALLOW ANY ORDER. | |
| AK: the problem is not to require strict order that the JSON syntax can not impose. | |
| JC: OK, but then if there are restrictions to the number of sub-components that are not captured by the Schemas, there must be text requirements imposing them. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #8 (T) | |
| Clause 4.3.5.2. XML schema for OptionalInputsVerifyType  The definition of VerifyRequest includes the following sub-component  <xs:element minOccurs="0" name="OptionalInputs"  type="dss2:OptionalInputsVerifyType"/>  And clause 4.3.5.2 defines:  <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="ReturnAugmentedSignature"  type="dss2:AugmentSignatureInstructionType"/>  <xs:element maxOccurs="unbounded" minOccurs="0"  name="ReturnTransformedDocument"  type="dss2:ReturnTransformedDocumentType"/>  <xs:element maxOccurs="1" minOccurs="0"  name="ReturnTimestampedSignature"  type="dss2:AugmentSignatureInstructionType"/>  <xs:element default="false" maxOccurs="1" minOccurs="0"  name="VerifyManifests" type="xs:boolean"/>  </xs:choice>  </xs:sequence>  </xs:extension>  </xs:complexContent> </xs:complexType>  Again, the sequence only has ONE child, which is a choice of the children of the <choice> component, but ONLY ONE, and consequently this schema does not allow to have more than one optional input.  REQUEST: EITHER SUPPRESS THE CHOICE WITHIN THE SEQUENCE OR ADD A MAXOCCURS TO THE SEQUENCE. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #9 (T) | |
| Clause 4.3.6.2 OptionalOutputsBaseType.  Same comment as above and same request.  REQUEST: EITHER SUPPRESS THE CHOICE WITHIN THE SEQUENCE OR ADD A MAXOCCURS TO THE SEQUENCE. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #10 (T) | |
| Clause 4.3.8.2 OptionalOutputsVerifyType.  Same comment as above and same request.  REQUEST: EITHER SUPPRESS THE CHOICE WITHIN THE SEQUENCE OR ADD A MAXOCCURS TO THE SEQUENCE. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #11 (T) | |
| Clause 6.3.1 Sub process ‘update Signature’.  COMMENT: in the title and within the text of the clause the term “update of signature” is used, which seems a problem inherited from version 1.0. Now the new term for this process is augment and augmented signature, according to AdES terminology in ETSI ENs.  REQUEST:   1. Replace the title by “augmentation of signature”. 2. Replace “update” by “augment” or “augmentation” depending on the context. 3. Replace ReturnUpdatedSignature by ReturnAugmentedSignature. | |
| CONCLUSION: Pass the comment to DSS-X. | |
| Comment #13 (T) | |
| Clause 4.3.8. OptionalOutputsVerify and Clause 4.3.32 AugmentedSignature  COMMENT:  Clause 4.3.8 only allows one instance of AugmentedSignature, and AugmentedSignature may only contain one SignatureObject. It is not possible to return more than one AugmentedSignature if no changes are done to the XML schema. In addition to that if the augmented signature is within it is returned within the component DocumentWithSignature, which makes it difficult to notice that it is an augmented signature.  REQUESTS:   1. Make it possible that a response may contain more than one AugmentedSignature occurrences. 2. Make it possible that an item of AugmentedSignature may also contain a DocumentWithSignature sub-component…in other words, make its content a choice of either a SignatureObject OR a DocumentWithSignature, so that it is clear that regardless where the signature appears in the response it is clear that it is an augmented signature. | |
| AK: The requirement for returning detailed information of validation of several signatures is more an item for profiles than for the core. | |
| JC: very good point. I fully agree. TS 119 442 is a profile of the DSS-X core 2.0 that defines new types/elements and restricts the usage of certain elements in the core: expands it and restricts it. | |
| CONCLUSION: Pass the comment to DSS-X. | |
|  | |
| Comment #14 (T) | |
| Assigned time | [Conductor] |
|  |  |
| LR: I have not been able to read the specification, but I plan to do it. | |
| Frederick: We have some few comments. Maybe we could bring them later on or present them now | |
| Clause 4.3.4.  COMMENTS:  They are SignatureType refer to 7.1, which does not exist. It should be 9.1. When I look to the URIs we are missing URIs for PDF signatures | |
| AK: we took the URIs coming from DSS 1.0. | |
| JC: after DSS 1.0 became a OASIS standard, a number of profiles were created for managing PDF signatures (generation) as well. | |
| LR: there is a component drafted that you can pass to the server with a pdf document asking to the server to generate a PAdES signature. | |
| AK: the list is informal in the core, not limiting anything. I will take a look to the ETSI Profile for signing. | |
| CONCLUSION: raise the comment to OASIS DSS-X | |
| Comment #16 (T) | |
| Clause 7  QUESTION: in this model is the server the one that decides whether the management is done synchronous or asynchronous. Could it be possible to consider the possibility of allowing the client to have the possibility of deciding this? | |
| AK: are there use cases for this? | |
| FP, LR: there are few use cases for this…for instance when the client does not want to wait until the server finishes, but it prefers to leave this, do something else and come back later on. This is more for server to server communication. | |
| AK: this is a good point and we should think about it. | |
| JC: I guess that this would also impact the validation protocol, not only the sign protocol. | |
| AK: indeed, clause 7 affects both. | |
| CONCLUSION: Pass this request to DSS-X TC | |
| Comment #17 (E) | |
| COMMENT:  Some references in some places of the text are wrong. In clause 4.X.X there is the text saying: “requirements specified in this document in section 5:XX” when it should read “4.X.X”  REQUEST: Should be fixed. | |
| CONCLUSION: Pass this request to DSS-X TC | |
| Comment #18 (E) | |
| Clause 4.1.11 Component ResponseBase. Incomplete sentence at the end of the clause:  COMMENT:  At the end of the clause there is an incomplete sentence.  « The optional ResponseID element MUST contain one instance of a string. » [DSS-4.1.11-4].  The ResponseID element … | |
| CONCLUSION: Pass this request to DSS-X TC | |
| Comment #19 (E) | |
| Clause 4.3.6 Component OptionalOutputsBase. Statement non very clear.  COMMENT:  It is stated “« If a server does not recognize or can not handle any optional input, it MUST reject the request with a ResultMajor code of RequesterError and a ResultMinor code of NotSupported. »”. The statement refers to OptionalInputBase component, it is not clear if it is a repetition (in such case it doesn’t seem appropriate in the clause concerning OptionalOutputBase component) or it should be referred to the client instead of the server (in this case it seems strange having a statement imposing a condition to the client) or something else. | |
| CONCLUSION: Pass this request to DSS-X TC | |