Functional requirements for signature validator in EVSClient

General use case description

1. User chooses to validate an XML object in EVSClient
2. Schematron or model-based validation is performed to the object normally. Schematron rules may contain basic validation of the signature contents, such as checking whether correct digests have been used etc.
3. EVSClient detects whether the file contains an XML signature by checking if the file contains a <ds:Signature> element. There may be multiple signatures in the same file. If no signature is located, the use case stops here. Namespace prefix “ds” stands for "http://www.w3.org/2000/09/xmldsig#".
4. A link “Validate enveloped signature” is shown next to the object contents in the same way as a link to PDF validation is shown when the object contains an enveloped PDF.
5. User clicks the signature validation link.
6. Validation of signature(s) is performed and the result is shown to the user. Each signature is validated separately.
7. Alternatively (a better option) the validation may be performed immediately in parallel with other validation (XSD schema and schematron or model-based validation) and results are shown together with other results on the same page.

Validation rules

- Basic validation of signature structures is performed using a separate Schematron-based set of rules. There is no need for IHE-Europe to implement these rules.
- A validation engine for signature validation, to be called by EVSClient, needs to be implemented by IHE-Europe. Apache Santuario or another open source library should be used.
    - In a later project, additional validation may be implemented by building a client calling the Kanta signature validation service.
    - However, at this point of time this does not need to be implemented, as the Kanta signature validation service does not provide feedback on reference validation, and most of the experienced problems are connected with incorrect canonicalization of referenced content.
- Signature validation is performed in two steps: 1) validation of references, 2) validation of the signature value against the public key in the supplied certificate. The certificate is included as part of the signature.
- The minimum output as shown in EVSClient must contain the following data for each signature being validated:
    - For each reference, full referenced content, canonicalized and processed according to the algorithms declared in the ds:Reference element
    - For each reference, information about correctness of its declared digest (declared digest vs. actual calculated digest). The expected digest value is in the ds:DigestValue element, and the digest algorithm is declared in ds:DigestMethod.
    - Correctness of the signature value (correct/incorrect). The value is calculated over the SignedInfo element, canonicalized according to the

declared canonicalization algorithm (ds:CanonicalizationMethod) using the declared signature method (ds:SignatureMethod).

- Validity of the certificate is checked (validity dates only)
- There is no need for building certificate path validation.
  - This may be done in a later project.
- For references, either ID-based referencing or XPath Filter-2 referencing may be used. For enabling correct validation results with ID-based referencing the following ID attributes may need to be declared explicitly as being of type xsd:ID:
  - //hl7fi:signature/hl7fi:signatureTimestamp/@ID
  - //hl7fi:signature/hl7fi:multipleDocumentSignature/@ID
  - /hl7:ClinicalDocument/hl7:component/hl7:structuredBody/@ID
  - /hl7:ClinicalDocument/hl7:component/hl7:nonXMLBody/@ID
  - Note that all of these are local Finnish extensions of the original HL7 CDA schema. hl7fi namespace prefix stands for namespace "urn:hl7finland".
  - There should be a way of configuring the signature validation engine with XPaths of such ID attributes so that they can be taken into account for validation. For example, for assertion validation, saml:Assertion/@ID needs to be declared as a potential reference target.
  - Alternatively, signature validation may be schema-aware. In this case, the schema assigned to the object type should be passed to the validation engine.

Samples

- A set of example signatures is provided for testing reasons.
- A simple signature validator is provided, based on Apache Santuario. The implementation may be freely used and redistributed (ASLv2 license). Output needs to be improved for display in EVSClient.