EDSSI

IIA V6 Signing and Approving Behaviour

Document title	**European Digital Student Service Infrastructure** **IIA V6 Signing and Approving Behaviour**
Version	**Draft**
Status	**Draft**
Author(s)	**EDSSI**

IIA Partners Approval Behaviour

Assumptions

Partner A has all needed IIA APIs in the latest version

- IIA Get API
- IIA List API
- IIA CNR API
- IIA Approval CNR API
- IIA Approval API

Partner B has all needed IIA APIs in the latest version

- IIA Get API
- IIA List API
- IIA CNR API
- IIA Approval CNR API
- IIA Approval API

Behaviour

1. Partner A creates an IIA in their system
2. **Partner A** should contact the **IIA CNR** of **Partner B** and receive a successful answer, meaning that the CNR has been successfully received
   2.1. **Partner A** should be able to send new **IIA CNR** to **Partner B** and receive a successful answer, meaning that the CNR has been successfully received
3. **Partner B** does a request to **Partner A IIA Get API** , either after receiving the CNR or at any moment in time
   3.1.1. When **Partner A** is generating the IIA XML should **calculate the Hash of the Cooperation Conditions** and append it in the XML
   3.1.2. When **Partner B** retrieves the XML should **calculate the Hash of the Cooperation Conditions** received and **compare** to the Hash received (appended in 3.1.1)
   3.1.2.1. If they match it is ok. Perhaps here the partner can check is there were any changes from the previous get they have made
   3.1.2.2. If they do not match, should not proceed
4. **Partner B** edits the IIA in their
5. **Partner B** should contact the **IIA CNR** of **Partner A** and receive a successful answer, meaning that the CNR has been successfully received
   5.1. **Partner B** should be able to send new **IIA CNR** to **Partner A** and receives a successful answer, meaning that the CNR has been successfully received
6. **Partner A** does a request to **Partner B IIA Get API** , either after receiving the CNR or at any moment in time
   6.1.1. When **Partner B** is generating the IIA XML should **calculate the Hash of the Cooperation Conditions** and append it in the XML
   6.1.2. When **Partner A** retrieves the XML should **calculate the Hash of the Cooperation Conditions** received and **compare** to the Hash received (appended in 3.1.1)

6.1.2.1. If they match it is ok. Perhaps here the partner can check is there were any changes from the previous get they have made

6.1.2.2. If they do not match, should not proceed

7. **Partner A** is happy with the IIA received by **Partner B**
   7.1. **Partner A** puts the **Partner B Hash** listed in its **IIA Approval API**
   7.2. **Partner A** contacts the **Partner B IIA Approval CNR API** and receives a successful answer, meaning that the CNR has been successfully received
   7.3. Partner B contacts **Partner A IIA Approval API** checking if it's hash for an IIA ID is present there, meaning that **Partner A** has approved **Partner B** version of a specific IIA
8. **Partner B** is also happy with the IIA received by **Partner A**
   8.1. **Partner B** puts the **Partner A Hash** listed in its **IIA Approval API**
   8.2. **Partner B** contacts the **Partner A IIA Approval CNR API** and receives a successful answer, meaning that the CNR has been successfully received
   8.3. Partner A contacts **Partner B IIA Approval API** checking if it's hash for an IIA ID is present there, meaning that **Partner B** has approved **Partner A** version of a specific IIA
9. After the same IIA is approved by both partners they are confirmed and acknowledged as valid IIAs

Calculation of the Hash when generating the XML

When receiving a request, to it's **IIA GET API** from **Partner A**, **Partner B** has to generate an XML document with all the information, including the Cooperation Conditions of that specific agreement, and has to **calculate the Hash of those Cooperation Conditions**, appending it to the XML document, on its specific tag.
For this calculation, **Partner B** has to take into account the following points:

- The hash **should not** change if the cooperation conditions are not really changing.
- The `sending-contact` and `receiving-contact` subelements are **not** taken into account when calculating hash.
- Before calculating the hash, the cooperation-conditions element should be normalized using Exclusive XML Canonicalization.

Hash calculation pseudocode may be found in the "Hash calculation" section.

Calculation of the Hash when receiving an XML

Upon making a request to a Partner B IIA Get API, if successful, Partner A receives an XML document that, if valid (please validate in the Schema Validator[1]), will contain a Hash tag that was calculated by Partner B, focusing on the Cooperation Conditions, during the generation of the document.
For this calculation, **Partner B** has to take into account the following points:

- The `sending-contact` and `receiving-contact` subelements are **not** taken into account when calculating hash.
- Before calculating the hash, the cooperation-conditions element should be normalized using Exclusive XML Canonicalization.

[1] https://dev-registry.erasmuswithoutpaper.eu/schemaValidator

After the calculation, **Partner A** will have both its **calculated hash** and **Partner B hash**. At this point Parner A can check if the hash of Partner B was correctly calculated and that Cooperation Conditions haven't been altered since last request.

Hash calculation pseudocode may be found in the "Hash calculation" section.

Hash calculation

There is an important consideration to the Exclusive XML Canonicalization. The cooperation-conditions element has to contain the same namespace aliases as the XML response to the IIA GET method. If a namespace alias is autogenerated when marshalling to XML, then it might be a good idea to set the namespace alias to a predefined value.

An a example pseudocode for hash calculation:

- Extract cooperation conditions from the IIA GET response object.
- Remove sending and receiving contacts elements from the extracted object.
- Marshall the object to an XML by applying Exclusive XML Canonicalization (take into consideration namespace aliases as mentioned above!).
- Hash the calculated XML with SHA256.
- Add to the IIA GET response a conditions-hash element with the calculated hash.

Useful Link

https://en.wikipedia.org/wiki/Canonical_XML

https://www.w3.org/TR/xml-c14n/

www.edssi.eu