Infrastructure 3.3 Summary of Changes

Updated: May 22, 2019

This "change" document follows the Policy and Procedure processes for Specification Development within the A4L Community. This draft outlines the changes made to the Infrastructure v3.2.1 to deliver a v3.3 in conjunction with the "Unity" Specification release in North America.

V 3.3 Scope:

1. Privacy Integration
2. Version Indication/Negotiation
3. PESC JSON Adoption
4. Cleaner, More Consistent Schemas and Artifacts*
5. Clarifications*

*Not intended to change the meaning of the specification.

The released documents: https://www.a4l.org/general/custom.asp?page=Infrastructure_3-3

Privacy Integration:

The Privacy Obligations Document (POD) Utility Service stores meta-data about the various versions of these privacy controls and obligations and should not create any changes in interoperability in and of itself. The Privacy volume of the specification does lay out new services and data flows that can be used to provide greater control over interoperability. See Appendix A – Privacy Integration Requirements Summary for further details.

This impacts the existing specification in two important ways. First, these services did *not* fit well into any existing type(UTILITY, OBJECT, FUNCTIONAL, SERVICEPATH, XQUERYTEMPLATE), so another has been made valid (SERVICE) and may be conveyed on the wire when using Infrastructure 3.3 and above. Second, new headers (applicationKey, contractId, podId, podVersion and role) have been added in order to apply privacy rules correctly. If they are misused interoperability should be prevented. However, the use of these features is optional from a specification perspective.

Version Indication/Negotiation:

This addition adds needed functionality to indicate and/or negotiate not only the versions (numbers) of the specification in play for a payload but also the format that those versions take (XML/JSON). It has been labeled "experimental," because this is not just a new standard for us but for the web and is more likely to be changed by others as it is piloted around the globe than anything else we have ever adopted. This flexibility is needed to avoid partial interoperability amongst partners where one does not support a sufficient version of the standard and ensure that data will be conveyed in a format that is understood by the receiver(s). This should give

users a better understanding of interoperability issues and only prevent interoperability when attempting to do so would be counterproductive.

PESC JSON Adoption:

PESC JSON represents a clear path forward for us on the journey of treating JSON as a first-class citizen within the SIF ecosystem. As there are two JSON notations in the current specification (Goessner & PESC) the Version Indication/Negotiation feature above is used to clearly indicate which notation is used. This is done in order to keep the defaults the same and prevent problems with existing systems regardless of whether they use existing XML or JSON technologies. PESC JSON formally brings consistent structures and paths to our JSON, empowering much simpler and dependable development of consumers.

Cleaner, More Consistent Schemas and Artifacts:

When New Zealand chose to use the Specification Generation tool (SpecGen) rather than create schemas directly, it served to set a clear direction forward for the global community. Not only will resources be saved by focusing our global efforts around one tool for developing specifications, this move helps us to forge a path towards one signature process for creating and publishing our SIF Specifications.

Updates have been made to the XSD to help ensure consistency and improve tooling support across objects. While this move is *not* designed to change the meaning of the 3.3 Infrastructure Specification (how the data is expressed) there are a few small behind the scenes changes that may impact those generating code from our published schemas. For detailed information, please refer to SIF 3.3 XSD changes document.

Clarifications:

The documents supporting the infrastructure have been reviewed and 'minor' edits made to clarify meaning. These well tracked (on the Infrastructure community site) changes to the specification increases consistency and guidance for those who are writing software directly to the specification. All have been reviewed by the Infrastructure group and incorporated in the draft. The edits are not intended to alter any functionality.

Special Thanks need to go out to:
Nick Nicholas, NSIP: Readability and clarity of the Infrastructure Volumes.
Joerg Huber, Systemic: Technical consistency and clarity of these documents.
Linda Marshall, NSIP: Schema authoring with SpecGen (not a small task).
KamHay Fung, NSW Department of Education: Version Indication & Negotiation author.
Anthony Yaremenko, NSIP: Privacy co-lead and primary author.

Respectfully submitted,
John W. Lovell
Technology Director
A4L Community

Appendix A – Privacy Integration Requirements Summary:

The Consumer may be required to…

1. Retrieve its Data Privacy Marker (DPM) when it starts and whenever it receives a 403 error.
2. Retrieve the POD by including its DPM in the where clause of a query to this service.
3. Confirm it can function under the conditions of the POD.
4. Affirm it can either automate the requirements of the POD or notify the administrator of the tasks they must complete in order to conform to the POD.
5. Include the current dataPrivacyMarker header in all requests it makes.
6. Treat data retrieved or delivered through any mechanism (events, another API, human input, etc.) under the same Benchmarks.

The Provider and/or Broker may be required to…

1. Implement or relay request to the POD service.
2. Return the most specific POD based on the DPM.
3. Confirm the DPM is acceptable before responding to requests. Otherwise provide the Consumer with a 403/Forbidden HTTP error code.
4. Run every outgoing payload through the Privacy Enforcer Service (wherever it is implemented) *before* sending it on to the Consumer.
5. Include the dataPrivacyMarker header with the DPM used when the payload was processed.