Data Standards Body
Technical Working Group
Decision 004 – Versioning Strategy

Contact: James Bligh

Publish Date: 30th August 2018

Decision Approved By Chairman: 6th September 2018

Context

To support the evolution of the APIs a versioning strategy is required so that changes to the API end points, or the overall standards, can be made with a minimisation of impact to the clients that have been built and deployed.

The UK standards have a model for versioning that can be described as "Block Versioning" (more detail on this below). This aspect of the UK standards was critiqued in some of the feedback to the Farrell report. As such there are multiple options for how versioning can be handled.

There are two conceptual levels of versioning in the API standards. One level is the high level versioning of the overall standards themselves. The second level is the more granular versioning of individual end points. These two levels can be handled together or separately depending on the approach taken.

Decision To Be Made

Determine the versioning and backwards compatibility approach to be employed for the API standards.

This decision does not include mandatory requirements for provider compliance with new versions and maintenance of support for older versions.

Feedback Provided

The original proposal and the associated feedback can be found at:
https://github.com/ConsumerDataStandardsAustralia/open-banking/issues/4

A small number of feedback submissions were provided for this decision but they were well considered and detailed in nature. The decision proposal articulated a number of alternatives so the feedback gave very useful insight into opinions on the various alternatives.

In general a clear path emerged but there was one deficiency identified - accommodation for versioning of provider extensions - in the decision proposal. This deficiency has been addressed in the final decision below.

Decision For Approval

The versioning strategy to be adopted by the API standards is as follows:

Standard Versioning

As articulated in *Decision 002 – URI Structure* the version of the overall standard would be embedded in the URI structure.

For example:

http://www.bank.com.au/api/cds-au/**v1**/banking/accounts
http://www.bank.com.au/api/cds-au/**v1**/banking/products
http://www.energyretailer.com.au/api/cds-au/**v1**/energy/usage

End Point Versioning

A specific end point version will be requested by a client using a HTTP header. This header will be supported by all end points under the API standards.

For example:

GET http://www.bank.com.au/api/cds-au/v1/banking/accounts **(HEADER x-v = 1)**
GET http://www.bank.com.au/api/cds-au/v1/banking/products **(HEADER x-v = 3)**
GET http://www.energyretailer.com.au/api/cds-au/v1/energy/usage **(HEADER x-v = 4)**

The header for end point version would be "**x-v**" and would have a value of a single integer representing the version. No major or minor versions would be used. Each new version would increment the value of the version. Each HTTP method for an end point would be versioned separately.

Minimum Version

An optional header "**x-min-v**" will be supported for all end points that can be used by the client to indicate that older versions are also acceptable for the response if the requested version is not supported.

By providing this header the client is indicating that any version between the requested version and the minimum version is acceptable. For example, if **x-v = 4** and **x-min-v = 2** then the client will accept versions 2, 3 or 4 but will not accept version 1 or versions above 4.

Response Header

A supplied version will be specified in the response header using the header field "**x-v**".

Extension Versioning

An optional header "**x-<PID>-v**" will be supported for all end points that can be used by the client to indicate a specific version of extension fields to include in the response. Note that <PID> is the specific provider string used by the provider to label extensions as articulated in *Decision 003 – Extensibility Model*.

This header applies only to the specific version requested and is mutually exclusive with the minimum version header field. A client should not supply "**x-<PID>-v**" and "**x-min-v**".