Guide to implementing Criterion OpenAPI definition

Version: **1.4**
Date: **04 June 2025**
Distribution: **Industry**
Document Name: **Guide_to_implementing_Criterion_OpenAPI_definition.pdf**

DISCLAIMER

Criterion believes it has employed personnel using reasonable skill and care in the creation of this document. However, this document is provided to the reader 'as is' without any warranty (express or implied) as to accuracy or completeness. Criterion cannot be held liable for any errors or omissions in this document or any losses, damages or expenses arising consequent to the use of this document by the reader.

CHANGE HISTORY

DATE	VERSION	DESCRIPTION
29 November 2023	1.0	Initial guide.
01 May 2024	1.1	Following consultation with STAG, update approaches to end point versioning, security and extensibility.
11 December 2024	1.2	Change references from OpenAPI *Specification* to Open API *Definition*.
30 April 2025	1.3	Add sections for *Business rules* and *Data field security*.
04 June 2025	1.4	New introductory text and diagram in first 'How to use...' section.

CONTENTS

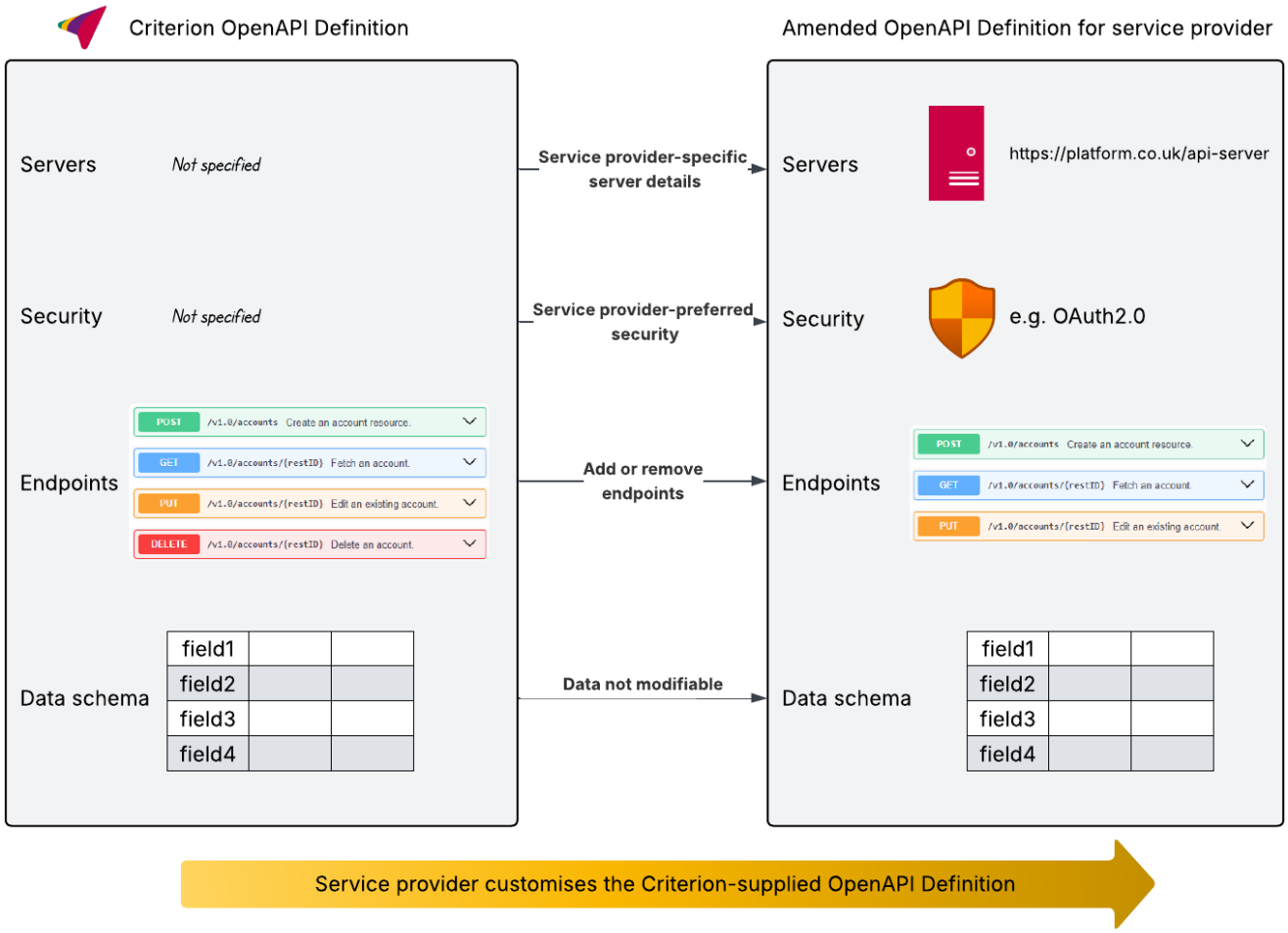
1 HOW TO USE CRITERION OPENAPI DEFINITIONS

Criterion does not host services but supplies API definitions written using the industry-standard *OpenAPI Specification*. A service provider wishing to host an API will take the Criterion-supplied API definition and modify it to suit their needs as set out below. An online service will then be built from the modified API definition, and the definition will be shared with partners wishing to use the API.

Each OpenAPI-based Criterion Standard is comprised of one or more OpenAPI Definitions (OAD), or 'resources', conforming to the *OpenAPI Specification*. Each OAD contains a standardised data schema which must not be modified by implementors. However, it is possible to add data not included in the data schema – see Extensibility for more information. This must be done with caution as it reduces the value of standardisation.

In addition to the data schema, a set of API endpoints are included. These represent endpoints which Criterion expect would be used, but implementors are free to add or remove endpoints as required. Implementors are requested to inform Criterion in such cases, so the Criterion-supplied OAD could potentially be amended and thus enhance standardisation across the industry.

The Criterion-supplied OADs do not contain any server or security details. Server details will be inserted by the API service provider. Likewise, every API service provider will have an in-house security preference, and thus it is not beneficial for Criterion to be prescriptive in this regard.



1.1 SERVICE PROVIDER

The OAD is designed to be downloaded, edited and then hosted by the service provider.

The service provider must:

- Keep reference to Criterion, including links and copyright information;
- The Criterion version of the resources must be included alongside each endpoint.

The service provider should:

- Use the version number in the server section to reflect the service provider's internal versioning;
- Keep the Criterion tag names;
- Not change any of the data structures or their meaning (see section Trading partner specific data below);
- Send a copy of the adapted OAD to Criterion to help aid future development (under NDA if required);
- Inform Criterion of any problems found;
- Make their versions of adapted standards available to their clients (under limitations set out in the licensing contract);
- Use the hypermedia links where provided as a primary way to extend and layout the offering;
- Where REST IDs are defined, these should be used as surrogates to human identifiable identifiers.

The service provider could:

- Add in digital signatures;
- Add additional commentary to descriptions;
- Set up an OAD SwaggerUI (or other) test interface for use by their clients (under limitations set out in the licensing contract);
- Provide examples of usage (under limitations set out in the licensing contract);
- Integrate the Criterion OAD with other OAD in the same domain when appropriate.

1.2 SERVICE CLIENT

The client should:

- Dominantly use the OAD of the service providers not Criterion's OAD directly in implementation;
- Use Criterion's OAD as a Rosetta stone when dealing with multiple service providers;
- Use Criterion's OAD when initially evaluating the need for the Standard;
- Use Criterion's OAD prior to service provider engagement;
- Use Criterion's OAD prior to service provider implementation;
- Request changes to be made by Criterion where appropriate, rather than directly with multiple service providers;
- Use the hypermedia links where provided, rather than statically storing URLs.

2 OPENAPI DEFINITION BREAKDOWN

OpenAPI definitions within Criterion Standards (as specified in accordance with the OpenAPI Specification https://swagger.io/docs/specification/about/) provide a machine-readable API definition, allowing quicker service implementations. OpenAPI definitions consist of the components detailed in the table below. The "prescriptive" column indicates if the content supplied by Criterion for this Standard is prescriptive or not.

OPENAPI COMPONENT	DETAILS	PRESCRIPTIVE
openapi	The OpenAPI specification version used. See https://swagger.io/specification/#appendix-a-revision-history	Yes
info	High-level description of the Criterion Standard: • title; • description; • version (specified in Criterion Standards Versioning Policy terms, e.g. v1.0.); • contact; • licence.	Yes
servers	The service provider must specify the API server and base URL. More than one server can be defined, for example one for production and one for a sandbox server. All paths are relative to server URLs. This cannot be prescriptive for obvious reasons. e.g. https://api.serviceprovider.com/	No
security	The security scheme for securing the service must be agreed between the trading partners and specified accordingly here.	No
paths	Contains the paths/operations available in the API: • defining individual endpoints in the API, and the HTTP methods supported by these endpoints; • includes all the details of the message exchange patterns and refers to the input/output data structures for each message; • defines the handled return code values (via RFC 7807); • can support semantic versioning (https://semver.org/). The service provider should remove any endpoints they do not wish to implement. The service provider should produce specific error messages including when functionality that is not supported is used. The service provider should consult with Criterion before extending into new end points, to aid in a universal approach.	No
components/ schemas	Defines common data structures (schemas) referred to elsewhere in the API definition e.g., the paths/operations can refer to schema components for definitions of the message structures.	Yes
components/ examples	Example data for each endpoint, including HATEOAS links.	No
components/ securitySchemes	Defines common security schemes that can be used by the API's operations.	No

OpenAPI components that are NOT prescriptive can be customised for the specific requirements of an implementation. The API definition supplied as part of the Criterion Standard can be used as a basis for implementing a service conforming to the Criterion Standard. There will be no specific entries relating to securing the service or semantic versioning of the service operations/paths.

Trading partner specific HTTP headers can be supplied as part of the customisation of the OpenAPI definition.

3 BUSINESS RULES

Criterion OADs do not enforce any validation or business rules. All data fields are optional, meaning that there is no minimum set of data in any OAD which must be sent or returned.

Similarly - with the exception of country and currency - there are no enumerations of any data items, for example the OAD does not stipulate valid values for a person's title such as Mr, Ms etc.
For country and currency, OADs use a three-character code for the ISO standard representation of the country or currency, and therefore inherently have validation built-in. Country data items use an ISO 3166-1 alpha-3 code, and currency data items use an ISO 4217 code.

It is up to implementors to agree any required business rules and validation as part of setting up their API.

4 END POINT VERSIONING

There are two distinct versions that need to be captured in an API call, the Criterion Standard version and the implementer's service version. The implementor version should be specified in the URL and the Criterion Standard version in the header.

The implementor version in the endpoint URL is free for implementors to use however they want;
e.g. `GET /v1.0/persons`
The version contained in the header should specify the version of the Criterion Standard being used;
e.g. Platform Account Opening **v1.0**.

Criterion will version both the OAD and each contained end point.

5 SECURITY

Criterion is not prescriptive about security, as companies usually want to handle security in their own way. As the security landscape is constantly evolving, it is better not to have security details baked into the Standard. Therefore, trading partners should ensure they have implemented adequate security, and include the security details in their OAD copy.
REST ids should be used to ensure no identifiable data is included in the URL. For example, instead of directly using a policy number as a URL parameter, use a GUID which would have no external meaning. Note that these IDs could be formed using a lookup table, or an encryption of a recognisable identifier.

6 DATA FIELD SECURITY

All *properties* within the schemas of Criterion OADs have been restricted as much as possible to minimise the risk of security exploits. For example, all string fields have a maximum length defined to guard against denial of service attacks via unbounded lengths. Additionally, the allowed character set for all string fields is restricted via regular expressions which again minimises the possibility of attack from cross-site scripting or SQL injection attacks. Anyone implementing the resources must still treat all incoming data as untrusted, and perform suitable escaping etc.

7 EXTENSIBILITY

Criterion recognise the need for trading partners to exchange data which is not included in the OAD. To facilitate this, two options are provided as described below.

7.1 LINKS

For RESTful APIs where significant chunks of data need to be sent, a HATEOAS (Hypermedia As The Engine Of Application State) approach can be used whereby one or more links are provided to handle extra data.
For more details on this, please see the REST guide which is available to licenced users of Criterion REST Standards on the Criterion website.

7.2 TRADING PARTNER SPECIFIC DATA

Where HATEOAS is not suitable, for example where small bits of data are required throughout the message, or the data is contained inside an array, then trading partner specific data containers (called tpsData) can be used.

By using a named tpsData container, it is obvious where trading partner specific data is being deployed. Trading partners can use this extensibility feature in order to exchange information which is sensitive in competitive terms, or which has not been supported in the definition of this version of the Standard yet.

Criterion OADs contain a skeleton tpsData schema object which is not referenced by the main data schema.
Users of Standards are welcome to insert tpsData objects as required. These tpsData containers should appear as the last property in a particular object definition.

This should be used with caution as it could reduce the benefit of a standardised approach to data.

7.2.1 HOW TO USE TPSDATA

Within tpsData there should be a hierarchy, with a root object identifying the sender domain of the tpsData, and the payload contained under that. This facilitates handling similar tpsData from multiple trading partners, as they can be separated out in the OAD copy under different hierarchies.

This allows receivers to combine multiple tpsData into a single OAD so that tpsData processing can be branched off anywhere in the process flow rather than up front at ingestion, relieving the need for multiple OADs.

7.2.2 EXAMPLE INSTANCE

This example instance shows tpsData included in two places. This represents an example of how an adviser request message could look. In this case the request comes from *best-advisers.co.uk*.

Trading partners should update their copy of the Standard OAD to define the expected payload within tpsData. Note the use of an object with the name 'best-advisers.co.uk' in the example below. It is proposed that all tpsData payload is provided within an object which identifies the sender. As shown in the subsequent example, the OAD copy would be updated with multiple such hierarchies in order to define expected data from multiple different trading partners.

```
{
  "requestor_reference": "abc123",
  "current_product_provider": "Acme Provider",
  "adviser": {
    "full_name": "John Doe",
    "adviser_reference_number": "1a2b3c",
    "adviser_issuing_authority_name": "FCA"
  },
  "policyholder": {
    "person": {
      "given_names": ["Jane"],
      "family_name": "Smith"
    },
    "tpsData": {
      "best-advisers.co.uk": {
        "policyholderProfile": "Novice",
        "commsPreference": "email",
        "previousAuthExists": true
      }
    }
  },
  "include_specified_contracts_only_ind": true,
  "authorisations": [
    {
      "authorisation_type": "HandWrittenScanned",
      "image_file_format": "PNG",
      "base64_binary": "asdjfay74385uk4tkjklsdjg"
    }
  ],
    "tpsData": {
      "best-advisers.co.uk": {
        "urgencyLevel": "high",
        "riskRating": "3",
        "validTo": "2025-05-01",
        "authorisedProcesses": [
          "PropertySell",
          "PropertyValuation",
          "Admin"
        ]
      }
    }
}
```

7.2.3 EXAMPLE OAD

This example OAD excerpt shows how the tpsData identifier can be used to distinguish similar data from multiple trading partners. This represents an example of how a provider OAD copy could look. The original Criterion OAD would not have tpsData defined.

In this case a provider defines data structures for tpsData received from three different advisers. Notice in the first tpsData below that *urgencyLevel* is defined as a *string* for best-advisers.co.uk, whereas it is defined as an *integer* for another-adviser.org.uk. This illustrates how adding an identifying object adds flexibility, as the provider can accept different tpsData payloads from different trading partners and validate them all against their copy of the OAD.

```
RequestBody:
  type: object
  additionalProperties: false
  properties:
    requestor_reference:
      type: string
    current_product_provider:
      type: string
    adviser:
      $ref: "#/components/schemas/Adviser"
    policyholder:
      $ref: "#/components/schemas/Policyholder"
    include_specified_contracts_only_ind:
      type: boolean
    authorisations:
      type: array
      minItems: 1
      maxItems: 100
      items:
        $ref: '#/components/schemas/Authorisation'
    tpsData:
      type: object
      additionalProperties: false
      properties:
        best-advisers.co.uk:
          type: object
          additionalProperties: false
          properties:
            urgencyLevel:
              type: string
            riskRating:
              type: integer
            validTo:
              type: number
              format: date
            authorisedProcesses:
              type: array
              minItems: 1
```

```
              maxItems: 100
              items:
                type: string
                enum:
                  - Admin
                  - PropertySell
                  - PropertyValuation
        another-adviser.org.uk:
          type: object
          additionalProperties: false
          properties:
            urgencyLevel:
              type: integer
            risky:
              type: boolean
            validTo:
              type: number
              format: date
            authorisedProcesses:
              type: array
              minItems: 1
              maxItems: 2
              items:
                type: string
                enum:
                  - Admin
                  - All
                  - Restricted
Adviser:
  type: object
  additionalProperties: false
  properties:
    full_name:
      type: string
    adviser_reference_number:
      type: string
    adviser_issuing_authority_name:
      type: string
Policyholder:
  type: object
  additionalProperties: false
  properties:
    person:
      $ref: '#/components/schemas/Person'
    tpsData:
      type: object
      additionalProperties: false
      properties:
```

```
        best-advisers.co.uk:
          type: object
          additionalProperties: false
          properties:
            policyholderProfile:
              type: string
            commsPreference:
              type: string
            previousAuthExists:
              type: boolean
        ace-advice.co.uk:
          type: object
          additionalProperties: false
          properties:
            disclosureComplete:
              type: boolean
Person:
  type: object
  additionalProperties: false
  properties:
    given_names:
      type: array
      minItems: 1
      maxItems: 10
      items:
        type: string
    family_name:
      type: string
Authorisation:
  - $ref: '#/components/schemas/HandWrittenScanned'
HandWrittenScanned:
  type: object
  additionalProperties: false
  properties:
    authorisation_type:
      type: string
      enum:
        - HandWrittenScanned
    image_file_format:
      type: string
      enum:
        - PNG
        - JPEG
    base64_binary:
      minLength: 1
      maxLength: 7000000
```

8 GENERAL NOTES

8.1 EMPTY DATA ITEMS

The specification validation rules ensure that no empty data items are sent.