INTERNATIONAL STANDARD

ISO/IEC 30108-2

First edition
2023-10

Biometrics — Identity attributes verification services —

Part 2: RESTful specification

Biométrique — Services de vérification des attributs d'identité —

Partie 2: Spécification RESTful

Reference number
ISO/IEC 30108-2:2023(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 37, *Biometrics*.

A list of all parts in the ISO/IEC 30108 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

This document defines the architecture, operations, data elements and basic requirements for Identity Attributes Verification Services (IAVSs), thereby providing a framework for the implementation of generic identity services within a service-oriented environment. An identity in the context of IAVS comprises a subject, biographic data and biometric data. Other parts of the ISO/IEC 30108 series are intended to define specific IAVS implementations (or bindings) within specific environments, for example, Simple Object Access Protocol (SOAP) web services.

IAVS services are generic in nature, being modality-neutral and not targeted at any particular business application. These services include those related to the management, transformation and biometric comparison identity data. Services are invoked by an IAVS requester and implemented by an IAVS service provider (responder). IAVS does not prescribe the architecture or business logic of either the requester or service provider.

In IAVS two categories of identity services are defined: primitive and aggregate. Primitive services are more atomic and well-defined, whereas the aggregate services tend to be higher level and enable more flexibility on the part of the IAVS service provider.

In IAVS two identity models are also defined: person-centric and encounter-based. Person-centric systems maintain a single up-to-date record (set of data) for a given person, whereas an encounter-based system retains data related to each interaction the person has with the system.

This document represents a version of IAVS defined in ISO/IEC 30108-1, but using a representational state transfer (RESTful) approach.

Biometrics — Identity attributes verification services —

Part 2: RESTful specification

1 Scope

The ISO/IEC 30108 series defines biometric services used for identity assurance that are invoked over a services-based framework. It provides a generic set of biometric and identity-related functions and associated data definitions to allow remote access to biometric services.

Although focused on biometrics, the ISO/IEC 30108 series includes support for other related identity assurance mechanisms such as biographic and document capabilities. Identity attributes verification services (IAVSs) are intended to be compatible with and used in conjunction with other biometric standards as described in ISO/IEC 30108-1.

This document implements the specification provided in ISO/IEC 30108-1 using representational state transfer (REST).

Specification of biometric functionality is limited to remote (backend) services. Services between a client-side application and biometric capture devices are not within the scope of this document.

Integration of biometric services as part of an authentication service or protocol is not within the scope of this document.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 2382-37, *Information technology — Vocabulary — Part 37: Biometrics*

ISO/IEC 19785-1, *Information technology — Common Biometric Exchange Formats Framework — Part 1: Data element specification*

ISO/IEC 30108-1:2015[1)], *Information technology — Biometric Identity Assurance Services — Part 1: BIAS services*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 30108-1 and ISO/IEC 2382-37 apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

1) Under revision.

4 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviated terms given in ISO/IEC 30108-1 and the following apply.

BIR	biometric information record
BRA	ISO Biometric Registration Authority
CBEFF	Common Biometric Exchange Formats Framework (defined in ISO/IEC 19785 series)
EFTS	electronic fingerprint transmission specification
ID	identity / identification / identifier
IAVS	identity attributes verification services
JSON	JavaScript Object Notation
REST	representational state transfer
SOAP	Simple Object Access Protocol
UUID	universally unique identifier
YAML	Yet Another Markup Language[3]

5 Conformance

ISO/IEC 30108-1:2015, Annex A specifies the conformance requirements for systems/components claiming conformance to this document.

6 System context

This clause provides an overview of representational state transfer (REST), in the scope of IAVS.

The specification included in this document has been written in OpenAPI™,[2)] using YAML™ (Yet Another Markup Language)[3)] as the specification language. OpenAPI is an extended way to specify RESTful services, allowing a variety of tools to develop client and server applications, as well as using data in any of the data formats typically used in RESTful services, e.g. JSON (JavaScript Object Notation) or XML (eXtensible Markup Language).

A goal for this document is to be as open as possible. To that end, this specification is also available in an electronic OpenAPI file, available at https://standards.iso.org/iso-iec/30108/-2/ed-1.

The specification is written according to the following references:

— ECMA Standard ECMA-404 (2017 2nd Edition) The JSON Data Interchange Format ISO\IEC 21778

— Hyper Text Transfer Protocol (HTTP/1.1) RFC 7230 [https://httpwg.org/specs/rfc7230.html]

— Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content RFC 7231

— JSON Schema: A Media Type for Describing JSON Documents [http://json-schema.org/draft/2019-09/json-schema-core.html]

2) This trademark is provided for reasons of public interest or public safety. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO or IEC.

3) This trademark is provided for reasons of public interest or public safety. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO or IEC.

— YAML Ain't Markup Language (YAML™) Version 1.2: 3rd Edition, Patched at 2009-10-01 [https://yaml.org/spec/1.2/spec.html]

All subclauses within Clause 7 entitled "REST method" (numbered 7.x.x.2) indicate the HTTP method used, among the various possibilities: GET, POST, DELETE, PUT and PATCH.

Within this document, when a YAML specification is given (e.g. all subclauses numbered 7.x.x.5), it shall be placed within an OpenAPI file, in the section indicated, taking into account the template provided in Annex A.

7 Identity assurance services

7.1 General

This clause defines two categories of IAVS services: primitive and aggregate. Primitive services are lower-level operations that are used to request a specific capability. Aggregate services operate at a higher-level, performing a sequence of primitive operations in a single request. (An example of such a sequence would be a negative search where a 1:N identification which results in no matches being found is immediately followed by the addition of the biometric sample into that search population.) Implementers are not restricted to one or the other but they may provide (and requesters may use) a mixture of both primitive and aggregate types.

Aggregate services do not have to utilize primitive services.

Each service is identified by an *<interface>* tag and must include a *name* attribute. Service parameters are identified by a *<parameter>* tag and must include a *name*, *type* and *direction* attribute. The *direction* attribute specifies whether the parameter is an input parameter (*in*), an output parameter (*out*) or an input/output parameter (*inout*). Parameters may also include a *use* attribute to indicate if the parameter is required, optional or conditional. If the parameter is conditional, the service description must identify the conditions.

7.2 Primitive services

7.2.1 Introduction

IAVS specifies the following set of primitive services.

7.2.2 Add Subject To Gallery

7.2.2.1 Description

The *Add Subject To Gallery* service shall register a subject to a given gallery or population group. As an optional parameter, the value of the claim to identity by which the subject is known to the gallery may be specified. This claim to identity shall be unique across the gallery. If no claim to identity is specified, the subject ID (assigned with the *Create Subject* service) shall be used as the claim to identity. Additionally, in the encounter-centric model, the encounter ID associated with the subject's biometric sample that will be added to the gallery shall be specified.

NOTE In the IAVS model, the creation and management of galleries are the responsibility of the service provider implementation. Services are not exposed to the requester for this purpose.

7.2.2.2 REST method

POST

7.2.2.3 Parameters

— *Gallery ID* — the identifier of the gallery or population group to which the subject will be added.

— *Subject ID* — the identifier of the subject.

— *Identity Claim (optional)* — the identifier by which the subject is known to the gallery.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

7.2.2.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 1 (see 9.3 for the definitions of each error condition code).

Table 1 — Particular responses for *Add Subject To Gallery* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_IDENTITY_CLAIM	
		INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_GALLERY	
		UNKNOWN_SUBJECT	
		UNKNOWN_IDENTITY_CLAIM	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_STORE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.2.5 YAML specification

To be placed in section #/paths:

```
/addSubjectToGallery:
    post:
      summary: Add Subject To Gallery
      parameters:
        - name: galleryID
          in: query
          required: true
          schema:
            type: string
        - name: subjectID
          in: query
          required: true
          schema:
            type: string
        - name: identityClaim
          in: query
          required: false
          schema:
            type: string
        - name: encounterID
          in: query
          required: false #conditional
          schema:
            type: string
      responses:
        200:
          $ref: '#/components/responses/success'
```

```
400:
  description: Bad Request
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/invalidIdentityClaim'
          - $ref: '#/components/schemas/invalidSubjectId'
          - $ref: '#/components/schemas/invalidEncounterId'
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/unknownGallery'
          - $ref: '#/components/schemas/unknownSubject'
          - $ref: '#/components/schemas/unknownIdentityClaim'
          - $ref: '#/components/schemas/unknownEncounter'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotStoreData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.2.3 Check Quality

7.2.3.1 Description

The *Check Quality* service shall return a quality score for a given biometric sample or a specified subject. Either a biometric sample or a subject identifier (ID) shall be provided. The biometric input is provided in a Common Biometric Exchange Formats Framework (CBEFF, as defined in ISO/IEC 19785 series) basic structure or CBEFF record, which in this document is called a CBEFF-BIR (Biometric Information Record). The algorithm vendor and algorithm vendor product ID may be optionally provided in order to request a particular algorithm's use in calculating the biometric quality. If an algorithm vendor is provided, then the algorithm vendor product ID is required. If no algorithm vendor is provided, the implementing system shall provide the algorithm vendor and algorithm vendor product ID that were used to calculate the biometric quality as output parameters.

NOTE Algorithm vendors are registered with the ISO Biometric Registration Authority (BRA) and assigned unique identifiers as defined in ISO/IEC 19785-2. Algorithm Product IDs are assigned by the registered algorithm vendor.

7.2.3.2 REST method

GET

7.2.3.3 Parameters

— *BIR (conditional)* — data structure containing a single biometric sample for which a quality score is to be determined; required if no Subject ID is provided.

— *Subject ID (conditional)* — the identifier of the subject; required if no BIR is provided.

— *Algorithm Vendor (optional)* — the identifier of the vendor of the quality algorithm used to determine the quality.

— *Algorithm Vendor Product ID (conditional)* — the vendor-assigned ID for the algorithm used to determine the quality; required as input if algorithm vendor is provided.

7.2.3.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 2 (see 9.3 for the definitions of each error condition code).

Table 2 — Particular responses for *Check Quality* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Quality Score* — *Algorithm Version* as seen below table.
400	Bad Request	INVALID_BIR	
		INVALID_SUBJECT_ID	
		INVALID_INPUT	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
		UNKNOWN_FORMAT	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	UNKNOWN_SUBJECT	
500	Internal Server Error	CANNOT_CHECK_QUALITY	

A successful execution of this service will provide:

— *Quality Score* — the quality of the biometric, as defined by the Quality type in one of the JSON-coded patron formats ISO/IEC 19785-3.

— *Algorithm Version* — the version of the algorithm used to determine the quality

7.2.3.5 YAML specification

To be placed in section #/paths:

```
/checkQuality:
  get:
    summary: Check Quality
    parameters:
      - name: bir
        in: query
        required: false #conditional
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
      - name: subjectID
        in: query
        required: false #conditional
        schema:
          type: string
      - name: algorithmVendor
        in: query #inout
        required: false
        schema:
          type: string
```

```
    - name: algorithmVendorProductID
      in: query #inout
      required: false #conditional
      schema:
        type: string
  responses:
    200:
      description: Success. Returns quality score and algorithm version
      content:
        application/json:
          schema:
            type: object
            properties:
              qualityScore:
                type: string #RAUL: CBEFF-3 JSON
                description: >
                  The quality of the biometric, as defined by the Quality
                  type in one of the JSON-coded patron formats ISO/IEC
                  19785-3
              algorithmVersion:
                type: string
                description: >
                  The version of the algorithm used to determine the
                  quality
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidBir'
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/biometricTypeNotSupported'
              - $ref: '#/components/schemas/unknownFormat'
    403:
      description: Forbidden
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/birDecryptionFailure'
              - $ref: '#/components/schemas/birQualityError'
              - $ref: '#/components/schemas/birSignatureFailure'
    404:
      description: Unknown Subject
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/unknownSubject'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotCheckQuality'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```


7.2.4 Classify Biometric Data

7.2.4.1 Description

The *Classify Biometric Data* service shall attempt to classify a biometric sample. For example, a fingerprint biometric sample may be classified as a whorl, loop or arch (or other classification classes and sub-classes). The types of classification algorithms and classes are not specified here. Instead, they

are left for the implementing system to define. If no classification algorithm is input, then the IAVS service provider will make the selection.

7.2.4.2 REST method

GET

7.2.4.3 Parameters

— *BIR* — data structure containing a single biometric sample for which the classification is to be determined.

— *Classification Algorithm Type (optional/output)* — identifies the type of classification algorithm to be used (input) and that was used (output) to perform the classification (e.g. for fingerprints, Henry classification).

7.2.4.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 3 (see 9.3 for the definitions of each error condition code).

Table 3 — Particular responses for *Classify Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Classification*, as seen below table.
400	Bad Request	INVALID_BIR	
		INVALID_INPUT	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
		UNKNOWN_FORMAT	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
500	Internal Server Error	CANNOT_PROCESS_DATA	

A successful execution of this service will provide:

— *Classification* — the result of the classification.

7.2.4.5 YAML specification

To be placed in section #/paths:

```
/classifyBiometricData:
  get:
    summary: Classify Biometric Data
    parameters:
      - name: bir
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
      - name: classificationAlgorithmType
        in: query
        required: false
        schema:
```

```
        type: string
responses:
  200:
    description: Success. Returns classification
    content:
      application/json:
        schema:
          type: object
          properties:
            classification:
              type: string
              description: The result of the classification
  400:
    description: Bad Request
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/invalidBir'
            - $ref: '#/components/schemas/invalidInput'
            - $ref: '#/components/schemas/biometricTypeNotSupported'
            - $ref: '#/components/schemas/unknownFormat'
  403:
    description: Forbidden
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/birDecryptionFailure'
            - $ref: '#/components/schemas/birQualityError'
            - $ref: '#/components/schemas/birSignatureFailure'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotProcessData'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.2.5 Create Encounter

7.2.5.1 Description

The *Create Encounter* service shall, for the specified subject, create a new encounter record and associate an encounter ID to that record. If not provided by the requester, the ***Create Encounter*** service shall generate an encounter ID that uniquely identifies the encounter within the subject record in the system.

Typically, the IAVS service provider will assign the encounter ID. In the event that the requester assigns the encounter ID, it shall be used unless it duplicates an existing encounter ID in which case an error shall be returned.

The *Create Encounter* service is performed prior to a *Set Biographic Data*, *Set Biometric Data*, or *Set Document Data* operation.

NOTE When in encounter mode, for comparison operations it is not necessary to explicitly create an encounter. The IAVS service provider will create the encounter and will set the encounter type to "recognition".

7.2.5.2 REST method

POST

7.2.5.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter Type* — the category of encounter.

— *Encounter ID (optional)* — the identifier of the encounter.

7.2.5.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 4 (see 9.3 for the meaning of each error condition codes):

Table 4 — Particular responses for *Create Encounter* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Encounter ID*, as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_TYPE	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	DUPLICATE_ENCOUNTER_ID	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Encounter ID* — the identifier of the encounter.

7.2.5.5 YAML specification

To be placed in section #/paths:

```
/createEncounter:
  post:
    summary: Create Encounter
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterType
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/EncounterCategoryType'
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
    responses:
      201:
        description: Success. Returns encounter ID created
        content:
          application/json:
            schema:
              type: object
```

```
        properties:
          encounterID:
            type: string
            description: The identifier of the encounter
400:
  description: Bad Request
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/invalidSubjectId'
          - $ref: '#/components/schemas/invalidEncounterType'
          - $ref: '#/components/schemas/invalidEncounterId'
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/unknownSubject'
          - $ref: '#/components/schemas/unknownEncounter'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/duplicateEncounterId'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.2.6 Create Subject

7.2.6.1 Description

The *Create Subject* service shall create a new subject record and associate a subject ID to that record. The *Create Subject* service shall generate a subject ID that uniquely identifies the subject in the system.

When cross-system uniqueness is required, universally unique identifiers (UUIDs) should be used for Subject IDs.

7.2.6.2 REST method

POST

7.2.6.3 Parameters

None

7.2.6.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 5 (see 9.3 for the meaning of each error condition code):

Table 5 — Particular responses for *Create Subject* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Subject ID*, as seen below table.
500	Internal Server Error	INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Subject ID* — the identifier of the subject.

7.2.6.5 YAML specification

To be placed in section #/paths:

```
/createSubject:
  post:
    summary: Create Subject
    responses:
      201:
        description: Success. Returns the subject ID created
        content:
          application/json:
            schema:
              type: object
              properties:
                subjectID:
                  type: string
                  description: The identifier of the subject
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.7 Delete Biographic Data

7.2.7.1 Description

The *Delete Biographic Data* service shall erase all of the biographic data associated with a given subject record. In the encounter-centric model, the service shall erase all of the biographic data associated with a given encounter, and therefore the encounter ID shall be specified. If no encounter ID is specified, or it is null, biographic data will be removed from all encounters. If a gallery is specified, biographic data will be deleted from that gallery only. When deleting data, IAVS implementations may completely erase the information in order to prevent the ability to reconstruct a record in whole or in part, or they may track and record the deleted information for auditing and/or quality control purposes.

7.2.7.2 REST method

DELETE

7.2.7.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group from which the biographic information will be deleted.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

7.2.7.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 6 (see 9.3 for the meaning of each error condition code):

Table 6 — Particular responses for *Delete Biographic Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_INPUT	
		INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.7.5 YAML specification

To be placed in section #/paths:

```
/deleteBiographicData:
  post:
    summary: Delete Biographic Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidEncounterId'
      404:
        description: Not Found
```

```
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/unknownSubject'
            - $ref: '#/components/schemas/unknownGallery'
            - $ref: '#/components/schemas/unknownEncounter'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotDeleteData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.2.8 Delete Biometric Data

7.2.8.1 Description

The *Delete Biometric Data* service shall remove biometric data from a given subject record. In the encounter-centric model, the encounter ID shall be specified. If no encounter ID is specified, or it is null, biometric data will be removed from all encounters. If a gallery is specified, biometric data will be deleted from that gallery only. If a biometric type(s) is specified, then only biometric data of that type shall be deleted. When deleting data, IAVS implementations may completely erase the information in order to prevent the ability to reconstruct a record in whole or in part, or they may track and record the deleted information for auditing and/or quality control purposes.

7.2.8.2 REST method

DELETE

7.2.8.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group from which the biometric information will be deleted.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Biometric Type (optional)* — the type of biological or behavioural data to delete, as defined by the BiometricType schema.

7.2.8.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 7 (see 9.3 for the meaning of each error condition code):

Table 7 — Particular responses for *Delete Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.8.5 YAML specification

To be placed in section #/paths:

```
/deleteBiometricData:
  post:
    summary: Delete Biometric Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: biometricType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/BiometricType'
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidEncounterId'
                - $ref: '#/components/schemas/biometricTypeNotSupported'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownSubject'
```

```
            - $ref: '#/components/schemas/unknownGallery'
            - $ref: '#/components/schemas/unknownEncounter'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotDeleteData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.9 Delete Document Data

7.2.9.1 Description

The *Delete Document Data* service shall erase all of the document data of the specified category or categories associated with a given subject record. In the encounter-centric model the service shall erase all of the document data associated with a given encounter, and therefore the encounter ID shall be specified. If no encounter ID is specified, or it is null, document data will be removed from all encounters. If no categories are specified, then all categories (for the specified encounters) shall be deleted. When deleting data, IAVS implementations may completely erase the information in order to prevent the ability to reconstruct a record in whole or in part, or they may track and record the deleted information for auditing and/or quality control purposes.

7.2.9.2 REST method

DELETE

7.2.9.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Document Category (optional)* — the category(ies) of the identity documents to be deleted.

7.2.9.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 8 (see 9.3 for meaning of each error condition code):

Table 8 — Particular responses for *Delete Document Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_INPUT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
		UNKNOWN_DOCUMENT_CATEGORY	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.9.5 YAML specification

To be placed in section #/paths:

```
/deleteDocumentData:
  post:
    summary: Delete Document Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: documentCategory
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidEncounterId'
                - $ref: '#/components/schemas/invalidInput'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownSubject'
                - $ref: '#/components/schemas/unknownEncounter'
                - $ref: '#/components/schemas/unknownDocumentCategory'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotDeleteData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```


7.2.10 Delete Encounter

7.2.10.1 Description

The *Delete Encounter* service shall delete an existing encounter record from the system. When deleting an encounter, IAVS implementations may completely erase the encounter information in order to prevent the ability to reconstruct a record or records in whole or in part, or they may track and record the deleted information for auditing and/or quality control purposes.

7.2.10.2 REST method

DELETE

7.2.10.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID* — the identifier of the encounter.

7.2.10.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 9 (see 9.3 for the meaning of each error condition code):

Table 9 — Particular responses for *Delete Encounter* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.10.5 YAML specification

To be placed in section #/paths:

```
/deleteEncounter:
  post:
    summary: Delete Encounter
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidEncounterId'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
```

```
            - $ref: '#/components/schemas/unknownSubject'
            - $ref: '#/components/schemas/unknownEncounter'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotDeleteData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.11 Delete Subject

7.2.11.1 Description

The *Delete Subject* service shall delete an existing subject record and, in an encounter-centric model, any associated encounter information from the system. This service shall also remove the subject from any registered galleries. When deleting a subject, IAVS implementations may completely erase the subject information in order to prevent the ability to reconstruct a record or records in whole or in part, or they may track and record the deleted information for auditing and/or quality control purposes.

7.2.11.2 REST method

DELETE

7.2.11.3 Parameters

— *Subject ID* — the identifier of the subject.

7.2.11.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 10 (see 9.3 for the meaning of each error condition code):

Table 10 — Particular responses for *Delete Subject* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
404	Not Found	UNKNOWN_SUBJECT	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.11.5 YAML specification

To be placed in section #/paths:

```
  /deleteSubject:
    post:
      summary: Delete Subject
      parameters:
        - name: subjectID
          in: query
          required: true
          schema:
            type: string
```

```
responses:
  200:
    $ref: '#/components/responses/success'
  400:
    description: Invalid Subject ID
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/invalidSubjectId'
  404:
    description: Unknown Subject
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/unknownSubject'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotDeleteData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.2.12 Delete Subject From Gallery

7.2.12.1 Description

The *Delete Subject From Gallery* service shall remove the registration of a subject from a gallery or population group. The subject shall be identified by either the subject ID or the claim to identity that was specified in the *Add Subject To Gallery* service.

7.2.12.2 REST method

DELETE

7.2.12.3 Parameters

— *Gallery ID* — the identifier of the gallery or population group from which the subject will be deleted.

— *Subject ID (conditional)* — the identifier of the subject; required if an identity claim is not provided.

— *Identity Claim (conditional)* — the identifier by which the subject is known to the gallery; required if a subject ID is not provided.

7.2.12.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 11 (see 9.3 for the meaning of each error condition code):

Table 11 — Particular responses for *Delete Subject From Gallery* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_IDENTITY_CLAIM	
404	Not Found	UNKNOWN_GALLERY	
		UNKNOWN_SUBJECT	
		UNKNOWN_IDENTITY_CLAIM	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.12.5 YAML specification

To be placed in section #/paths:

```
/deleteSubjectFromGallery:
  post:
    summary: Delete Subject From Gallery
    parameters:
      - name: galleryID
        in: query
        required: true
        schema:
          type: string
      - name: subjectID
        in: query
        required: false
        schema:
          type: string
      - name: identityClaim
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidIdentityClaim'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownGallery'
                - $ref: '#/components/schemas/unknownSubject'
                - $ref: '#/components/schemas/unknownIdentityClaim'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
```

```
            - $ref: '#/components/schemas/cannotDeleteData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.13 Get Identify Subject Results

7.2.13.1 Description

The *Get Identify Subject Results* service shall retrieve the identification results for the specified token. This service is used in conjunction with the *Identify Subject* service. If the *Identify Subject* service is implemented as an asynchronous service, the implementing system returns a token, and the *Get Identify Subject Results* service is used to poll for the results of the original *Identify Subject* request.

7.2.13.2 REST method

GET

7.2.13.3 Parameters

— *Token* — a value used to retrieve the results of the *Identify Subject* request.

7.2.13.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses list in Table 12 (see 9.3 for the meaning of each error condition code):

Table 12 — Particular responses for *Get Identify Subject Results* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Candidate List*, as seen below table.
400	Bad Request	INVALID_TOKEN	
404	Not Found	TOKEN_EXPIRED	
		IDENTIFICATION_RESULT_NOT_YET_AVAILABLE	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Candidate List* — a rank-ordered list of candidates that have a likelihood of matching the input biometric sample.

7.2.13.5 YAML specification

To be placed in section #/paths:

```
/getIdentifySubjectResults:
  get:
    summary: Get Identify Subject Results
    parameters:
      - name: token
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/TokenType'
    responses:
```

```
200:
  description: Success. Returns the ranked Candidate List
  content:
    application/json:
      # a rank-ordered list of candidates that have a
      # likelihood of matching the input biometric sample
      schema:
        $ref: '#/components/schemas/CandidateListType'
400:
  description: Invalid Token
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/invalidToken'
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/tokenExpired'
          - $ref: '#/components/schemas/identificationResultNotYetAvailable'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotRetrieveData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.2.14 Identify Subject

7.2.14.1 Description

The *Identify Subject* service shall perform an identification search against a given gallery for a given biometric type, returning a rank-ordered candidate list of a given maximum size.

If the *Identify Subject* service is implemented as a synchronous service, the implementing system shall immediately process the request and return the results in the candidate list. If the *Identify Subject* service is implemented as an asynchronous service, the implementing system shall return a token, which is an indication that the request is being handled asynchronously. In this case, the *Get Identify Subject Results* service shall be used to poll for the results of the *Identify Subject* request.

The candidate list shall include only those candidates whose comparison score exceeds a system-defined threshold, thus indicating a likely match with the subject. In the event that there are more candidates than the requested Max List Size, the system shall determine which candidate is included in the last position in the candidate list.

NOTE When in encounter mode, for comparison operations, it is not necessary to explicitly create an encounter. The IAVS service provider will create the encounter and will set the encounter type to "recognition".

7.2.14.2 REST method

GET

7.2.14.3 Parameters

— *BIR* — data structure containing the biometric sample for the search.

NOTE When multiple samples are included as input (e.g. in a multimodal operation), a complex BIR is used.

— *Max List Size* — the maximum size of the candidate list that should be returned.

— *Gallery ID (optional)* — the identifier of the gallery or population group which will be searched; this parameter may also be used to identify an external system where the identification request should be forwarded, if this capability is supported by the implementing system. This parameter shall not be used in conjunction with *Gallery.*

— *Gallery (optional)* — a list of BIRs that shall be used instead of a stored gallery. This parameter shall not be used in conjunction with *Gallery ID.*

7.2.14.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 13 (see 9.3 for the meaning of each error condition code):

Table 13 — Particular responses for *Identify Subject* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Candidate List* — *Token*, as seen below table.
400	Bad Request	INVALID_BIR	
		INVALID_INPUT	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
		UNKNOWN_FORMAT	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	UNKNOWN_GALLERY	
		CANNOT_IDENTIFY_DATA	
		IDENTIFICATION_RESULT_NOT_YET_AVAILABLE	
500	Internal Server Error	CANNOT_PROCESS_DATA	
		CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Candidate List (conditional)* — a rank-ordered list of candidates that have a likelihood of matching the input biometric sample; returned with successful, synchronous request processing.

— *Token* — a token used to retrieve the results of the *Identify Subject* request, returned with asynchronous request processing. If set to zero, operation is processed synchronously and candidate list is returned. If set to a non-zero value, operation is processed asynchronously and *Get Identify Results* needs to be used to retrieve the results.

NOTE Multiple scores/candidates is already built in as a score comes with a CandidateType which is a member of Candidate List.

7.2.14.5 YAML specification

To be placed in section #/paths:

```
/identifySubject:
  get:
    summary: Identify Subject
    parameters:
      - name: bir
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
      - name: maxListSize
        in: query
        required: true
        schema:
          type: integer
          format: int32
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: gallery
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_ListType'
    responses:
      200:
        description: Success. Returns quality score and algorithm version
        content:
          application/json:
            schema:
              type: object
              properties:
                candidateList:
                  # A rank-ordered list of candidates that have a likelihood
                  # of matching the input biometric sample; returned with
                  # successful, synchronous request processing
                  $ref: '#/components/schemas/CBEFF_BIR_ListType'
                token:
                  # A token used to retrieve the results of the Identify
                  # Subject request; returned with asynchronous request
                  # processing. If set to zero, operation is processed
                  # synchronously and candidate list is returned. If set to a
                  # non-zero value, operation is processed asynchronously and
                  # Get Identify Results must be used to retrieve the results
                  $ref: '#/components/schemas/TokenType'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidBir'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/biometricTypeNotSupported'
                - $ref: '#/components/schemas/unknownFormat'
      403:
        description: Forbidden
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/birDecryptionFailure'
                - $ref: '#/components/schemas/birQualityError'
                - $ref: '#/components/schemas/birSignatureFailure'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
```

```
            - $ref: '#/components/schemas/unknownGallery'
            - $ref: '#/components/schemas/cannotIdentifyData'
            - $ref: '#/components/schemas/identificationResultNotYetAvailable'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotProcessData'
                - $ref: '#/components/schemas/cannotRetrieveData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.15 List Biographic Data

7.2.15.1 Description

The *List Biographic Data* service shall list the biographic data elements stored for a subject using the Biographic Data Elements output parameter. Note that no actual biographic data is returned by this service (see the *Retrieve Biographic Data* service to obtain the biographic data). In the encounter-centric model, an encounter ID may be specified to indicate that only the biographic data elements stored for that encounter should be returned. If an encounter ID is not specified and encounter data exists for the subject, the service shall return the list of encounter IDs which contain biographic data using the Encounter List output parameter, and the Biographic Data Elements output parameter shall be empty.

7.2.15.2 REST method

GET

7.2.15.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of encounter.

7.2.15.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 14 (see 9.3 for the meaning of each error condition code):

Table 14 — Particular responses for *List Biographic Data* service

HTTP status code	HTTP ctatus code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Biographic Data Elements* — *Encounter List* as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	

Table 14 *(continued)*

HTTP status code	HTTP ctatus code meaning	Error condition code	Description
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_PROCESS_DATA	
		CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Biographic Data Elements (conditional)* — a list of biographic data elements associated with a subject or encounter; non-empty if the service was successful, biographic data exists, and either:

a) the person-centric model is being used, or

b) the encounter-centric model is being used and an encounter identifier was specified.

— *Encounter List (conditional)* — a list of encounter IDs associated with a subject and which contain biographic data; non-empty if the service was successful, biographic data exists, the encounter-centric model is being used, and an encounter identifier was not specified

7.2.15.5 YAML specification

To be placed in section #/paths:

```
/listBiographicData:
  get:
    summary: List Biographic Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/EncounterCategoryType'
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
              type: object
              properties:
                biographicDataElements:
                  # A list of biographic data elements associated with a
                  # subject or encounter; non-empty if the service was
                  # successful, biographic data exists, and either (a) the
                  # person-centric model is being used or (b) the
                  # encounter-centric model is being used and an encounter
                  # identifier was specified
                  $ref: '#/components/schemas/BiographicDataType'
                encounterList:
```

```
            # a list of encounter ID's associated with a subject and
            # which contain biographic data; non-empty if the service
            # was successful, biographic data exists, the
            # encounter-centric model is being used, and an encounter
            # identifier was not specified
            $ref: '#/components/schemas/EncounterListType'
400:
  description: Bad Request
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/invalidSubjectId'
          - $ref: '#/components/schemas/invalidEncounterId'
          - $ref: '#/components/schemas/invalidEncounterType'
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/unknownSubject'
          - $ref: '#/components/schemas/unknownEncounter'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotProcessData'
          - $ref: '#/components/schemas/cannotRetrieveData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.2.16 List Biometric Data

7.2.16.1 Description

The *List Biometric Data* service shall list the biometric data elements stored for a subject using the Biometric Data Elements output parameter. Note that no actual biometric data is returned by this service (see the *Retrieve Biometric Data* service to obtain the biometric data). In the encounter-centric model, an encounter ID may be specified to indicate that only the biometric data elements stored for that encounter should be returned. If an encounter ID is not specified and encounter data exists for the subject, the service shall return the list of encounter IDs which contain biometric data using the Encounter List output parameter, and the Biometric Data Elements output parameter shall be empty.

An optional parameter may be used to indicate a filter on the list of returned data. Such a filter may indicate that only biometric types should be listed (e.g. face, finger, iris, etc.) or that only biometric subtypes for a particular biometric type should be listed (e.g. all fingerprints: left slap, right index, etc.). If a filter is not specified, all biometric type and biometric subtype information shall be listed (e.g. left index finger, right iris, face frontal, etc.).

7.2.16.2 REST method

GET

7.2.16.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of encounter.

— *List Filter (optional)* — indicates what biometric information should be returned.

7.2.16.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 15 (see 9.3 for the meaning of each error condition code).

Table 15 — Particular responses for *List Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Biometric Data Elements* — *Encounter List* as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	
		INVALID_INPUT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_PROCESS_DATA	
		CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Biometric Data Elements (conditional)* — a list of biometric data elements associated with a subject or encounter; non-empty if the service was successful, biometric data exists, and either

  a) the person-centric model is being used, or

  b) the encounter-centric model is being used and an encounter identifier was specified.

— *Encounter List (conditional)* — a list of encounter IDs associated with a subject and which contain biometric data; non-empty if the service was successful, biometric data exists, the encounter-centric model is being used, and an encounter identifier was not specified.

7.2.16.5 YAML specification

To be placed in section #/paths:

```
/listBiometricData:
  get:
    summary: List Biometric Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
```

```
      required: false
      schema:
        $ref: '#/components/schemas/EncounterCategoryType'
    - name: listFilter
      in: query
      required: false
      schema:
        $ref: '#/components/schemas/ListFilterType'
  responses:
    200:
      description: >
        Success. Returns Biographic Data Elements and/or Encounter List
      content:
        application/json:
          schema:
            type: object
            properties:
              biometricDataElements:
                # A list of biometric data elements associated with a
                # subject or encounter; non-empty if the service was
                # successful, biometric data exists, and either (a) the
                # person-centric model is being used or (b) the
                # encounter-centric model is being used and an encounter
                # identifier was specified
                $ref: '#/components/schemas/BiometricDataListType'
              encounterList:
                # a list of encounter ID's associated with a subject and
                # which contain biographic data; non-empty if the service
                # was successful, biographic data exists, the
                # encounter-centric model is being used, and an encounter
                # identifier was not specified
                $ref: '#/components/schemas/EncounterListType'
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidEncounterId'
              - $ref: '#/components/schemas/invalidEncounterType'
              - $ref: '#/components/schemas/invalidInput'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownEncounter'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotProcessData'
              - $ref: '#/components/schemas/cannotRetrieveData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```


7.2.17 List Document Data

7.2.17.1 Description

The *List Document Data* service shall list the document categories stored for a subject using the Document Data Elements output parameter. Note that no other document data is returned by this

service (see the *Retrieve Document Data* service to obtain document data by category). In the encounter-centric model, an encounter ID may be specified to indicate that only the document data elements stored for that encounter should be returned. If an encounter ID is not specified and encounter data exists for the subject, the service shall return the list of encounter IDs which contain document data using the Encounter List output parameter, and the Document Data Elements output parameter shall be empty.

7.2.17.2 REST method

GET

7.2.17.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of the encounter.

7.2.17.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 16 (see 9.3 for the meaning of each error condition code):

Table 16 — Particular responses for *List Document Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Document Data Categories* — *Encounter List* as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
		NON_EXISTENT_DATA	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Document Data Categories (conditional)* — a list of document categories associated with a subject or encounter; non-empty if the service was successful, document data exists, and either

  a) the person-centric model is being used, or

  b) the encounter-centric model is being used and an encounter identifier was specified.

— *Encounter List (conditional)* — a list of encounter IDs associated with a subject and which contain document data; non-empty if the service was successful, document data exists, the encounter-centric model is being used, and an encounter identifier was not specified.

7.2.17.5 YAML specification

To be placed in section #/paths:

```
/listDocumentData:
  get:
    summary: List Document Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/EncounterCategoryType'
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
              type: object
              properties:
                documentDataCategories:
                  description: >
                    a list of document categories associated with a subject
                    or encounter; non-empty if the service was successful,
                    document data exists, and either (a) the person-centric
                    model is being used or (b) the encounter-centric model
                    is being used and an encounter identifier was specified
                  type: string
                encounterList:
                  # a list of encounter ID's associated with a subject and
                  # which contain document data; non-empty if the service
                  # was successful, biographic data exists, the
                  # encounter-centric model is being used, and an encounter
                  # identifier was not specified
                  $ref: '#/components/schemas/EncounterListType'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidEncounterId'
                - $ref: '#/components/schemas/invalidEncounterType'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownSubject'
                - $ref: '#/components/schemas/unknownEncounter'
                - $ref: '#/components/schemas/nonExistentData'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
```

```
            - $ref: '#/components/schemas/cannotRetrieveData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.18 Perform Fusion

7.2.18.1 Description

The *Perform Fusion* service shall accept either comparison score or comparison decision information and create a fused comparison result. The FusionInformationListType, through the FusionInformationType, provides specific elements for comparison score input and comparison decision input for a single identity, while the FusionIdentityListType provides the ability to submit multiple identities to the Perform Fusion service (see 8.6). The fusion method and processes are left to the implementing system.

7.2.18.2 REST method

GET

7.2.18.3 Parameters

— *Fusion Input* — score or decision input information to the fusion method for each identity.

7.2.18.4 Responses

The *Perform Fusion* service returns one of the return codes defined in Clause 9.

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 17 (see 9.3 for the meaning of each error condition code):

Table 17 — Particular responses for *Perform Fusion* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Match*, as seen below table.
400	Bad Request	INVALID_INPUT	
500	Internal Server Error	CANNOT_PROCESS_DATA	

A successful execution of this service will provide:

— *Match* — indicates the result of the fusion method.

7.2.18.5 YAML specification

To be placed in section #/paths:

```
/performFusion:
  get:
    summary: Perform Fusion
    parameters:
      - name: fusionInput
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/FusionIdentityListType'
    responses:
      200:
        description: >
```

```
      Success. Returns Biographic Data Elements and/or Encounter List
    content:
      application/json:
        schema:
          type: object
          properties:
            match:
              description: >
                indicates the result of the fusion methodd
              type: boolean
  400:
    description: Invalid Input
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/invalidInput'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotProcessData'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.2.19 Query Capabilities

7.2.19.1 Description

The *Query Capabilities* service shall return a list of the capabilities, options, galleries, etc. that are supported by the IAVS implementation. Subclause 8.6.2 provides a list of capabilities. Refer to ISO/IEC 30108-1:2015, Annex A, for conformance requirements regarding which capability names an implementation is required to use in the *Query Capabilities* service. If the implementing system does not support a capability item, the Capability Value can be set to null in the response.

Proprietary and additional information can be returned by returning capabilities that are not part of the listed capabilities in Table 1. When returning capabilities that are not listed in Table 1, the Capability Description shall describe the capability.

7.2.19.2 REST method

GET

7.2.19.3 Parameters

None.

7.2.19.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 18 (see 9.3 for the meaning of each error condition code).

Table 18 — Particular responses for *Query Capabilities* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Capability List*, as seen below table.

A successful execution of this service will provide:

— *Capability List* – a list of capabilities supported by the IAVS implementation.

7.2.19.5 YAML specification

To be placed in section #/paths:

```
/queryCapabilities:
  get:
    summary: Query Capabilities
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
              type: object
              properties:
                capabilityList:
                  # a list of capabilities supported by the BIAS
                  # implementation
                  $ref: '#/components/schemas/CapabilityListType'
      500:
        description: Unknown Error
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.20 Retrieve Biographic Data

7.2.20.1 Description

The *Retrieve Biographic Data* service shall retrieve the list of biographic data associated with a subject ID. In the encounter-centric model, the encounter ID may be specified and the service shall return the set of biographic data associated with that encounter (list contains a single set). If the encounter ID is not specified in the encounter-centric model, the service shall return the list of biographic information associated with the most recent encounter. If no gallery ID is specified, a list of biographic data from all galleries shall be returned.

7.2.20.2 REST method

GET

7.2.20.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group from which the biographic information will be retrieved.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of encounter.

7.2.20.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 19 (see 9.3 for the meaning of each error condition code).

Table 19 — Particular responses for *Retrieve Biographic Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Biographic Data List*, as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
		NON_EXISTENT_DATA	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Biographic Data List* — a list of biographic data associated with the subject or encounter.

7.2.20.5 YAML specification

To be placed in section #/paths:

```
/retrieveBiographicData:
  get:
    summary: Retrieve Biographic Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/EncounterCategoryType'
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
              type: object
              properties:
                biographicDataList:
                  # A list of biographic data elements associated with a
                  # subject or encounter
                  $ref: '#/components/schemas/BiographicDataListType'
```

```
400:
  description: Bad Request
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/invalidSubjectId'
          - $ref: '#/components/schemas/invalidInput'
          - $ref: '#/components/schemas/invalidEncounterId'
          - $ref: '#/components/schemas/invalidEncounterType'
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/unknownSubject'
          - $ref: '#/components/schemas/unknownGallery'
          - $ref: '#/components/schemas/unknownEncounter'
          - $ref: '#/components/schemas/nonExistentData'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotRetrieveData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.2.21 Retrieve Biometric Data

7.2.21.1 Description

The *Retrieve Biometric Data* service shall retrieve the biometric data associated with a subject ID. In the encounter-centric model, the encounter ID may be specified and the service shall return the biometric data associated with that encounter. If the encounter ID is not specified in the encounter-centric model, the service shall return the biometric information associated with the most recent encounter. If no gallery ID is specified, biometric data from all galleries shall be returned. The service provides an optional input parameter to specify that only biometric data of a certain type should be retrieved.

7.2.21.2 REST method

GET

7.2.21.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group from which the biometric information will be retrieved.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of encounter.

— *Biometric Type (optional)* — the type of biological or behavioural data to retrieve, as defined by the Multiple-types type in one of the JSON-coded patron formats ISO/IEC 19785-3.

7.2.21.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 20 (see 9.3 for the meaning of each error condition code).

Table 20 — Particular responses for *Retrieve Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *BIR List*, as seen below
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
		NON_EXISTENT_DATA	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *BIR List* — data structure containing the retrieved biometric samples.

7.2.21.5 YAML specification

To be placed in section #/paths:

```
/retrieveBiometricData:
  get:
    summary: Retrieve Biometric Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/EncounterCategoryType'
      - name: biometricType
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/BiometricType'
    responses:
      200:
```

```
    description: >
      Success. Returns Biographic Data Elements and/or Encounter List
    content:
      application/json:
        schema:
          type: object
          properties:
            birList:
              # data structure containing the retrieved biometric
              # samples
              $ref: '#/components/schemas/CBEFF_BIR_ListType'
  400:
    description: Bad Request
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/invalidSubjectId'
            - $ref: '#/components/schemas/invalidInput'
            - $ref: '#/components/schemas/invalidEncounterId'
            - $ref: '#/components/schemas/invalidEncounterType'
            - $ref: '#/components/schemas/biometricTypeNotSupported'
  404:
    description: Not Found
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/unknownSubject'
            - $ref: '#/components/schemas/unknownGallery'
            - $ref: '#/components/schemas/unknownEncounter'
            - $ref: '#/components/schemas/nonExistentData'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotRetrieveData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```


7.2.22 Retrieve Document Data

7.2.22.1 Description

The *Retrieve Document Data* service shall retrieve the list of document data associated with a subject ID for the category or categories specified. In the encounter-centric model, the encounter ID may be specified and the service shall return the list of document data associated with that encounter. If the encounter ID is not specified in the encounter-centric model, the service shall return the list of document information associated with the most recent encounter for which document data exists. If no gallery ID is specified, document data from all galleries shall be returned. If no document category is specified, all documents associated with the subject (and encounter ID, if present) shall be returned.

7.2.22.2 REST method

GET

7.2.22.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID* — the identifier of the gallery or population group from which the biographic information will be retrieved.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Encounter Type (optional)* — the category of encounter.

— *Document Category (optional)* — the category(ies) of the identity documents to be retrieved.

7.2.22.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 21 (see 9.3 for the meaning of each error condition code).

Table 21 — Particular responses for *Retrieve Document Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Document Data List*, as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		INVALID_ENCOUNTER_TYPE	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
		UNKNOWN_DOCUMENT_CATEGORY	
		NON_EXISTENT_DATA	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Document Data List* — a list of document data associated with the subject or encounter.

7.2.22.5 YAML specification

To be placed in section #/paths:

```
/retrieveDocumentData:
  get:
    summary: Retrieve Document Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: encounterType
        in: query
        required: false
```

```
      schema:
        $ref: '#/components/schemas/EncounterCategoryType'
    - name: documentCategory
      in: query
      required: false
      schema:
        type: string
  responses:
    200:
      description: >
        Success. Returns Biographic Data Elements and/or Encounter List
      content:
        application/json:
          schema:
            type: object
            properties:
              documentDataList:
                # a list of document data associated with the subject
                # or encounter
                $ref: '#/components/schemas/DocumentDataListType'
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/invalidEncounterId'
              - $ref: '#/components/schemas/invalidEncounterType'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownGallery'
              - $ref: '#/components/schemas/unknownEncounter'
              - $ref: '#/components/schemas/unknownDocumentCategory'
              - $ref: '#/components/schemas/nonExistentData'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotRetrieveData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```


7.2.23 Set Biographic Data

7.2.23.1 Description

The *Set Biographic Data* service shall associate biographic data to a given subject record. Depending on the identity model in use, the biographic information should replace any existing biographic information (person-centric model) or it shall be added within a new encounter (encounter-centric model). If biometric or document data was also collected during the same encounter, the same encounter ID shall be specified by the caller in order to link the biographic data with the associated biometric and/or document information (using the *Set Biometric Data* and *Set Document Data* services). If the encounter ID is omitted for the encounter-centric model, an error shall be returned.

NOTE Biographic data is typically set within the master database. However, this is implementation-specific.

For encounter-based systems, the *Create Encounter* service shall be called prior to *Set Biographic Data*. The Encounter ID assigned as a result (and returned by the *Create Encounter* service) shall be used as input to this service.

7.2.23.2 REST method

PATCH

7.2.23.3 Parameters

— *Subject ID (input)* — the identifier of the subject.

— *Gallery ID (input, optional)* — the identifier of the gallery or population group to which the biographic will be added.

— *Encounter ID (input, conditional)* — the identifier of the encounter.

— *Biographic Data (input)* — a list of biographic data to associate with the subject or encounter.

7.2.23.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 22 (see 9.3 for the meaning of each error condition code).

Table 22 — Particular responses for *Set Biographic Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		UNKNOWN_BIOGRAPHIC_FORMAT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_STORE_DATA	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.23.5 YAML specification

To be placed in section #/paths:

```
/setBiographicData:
  post:
    summary: Set Biographic Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
```

```
    - name: encounterID
      in: query
      required: false
      schema:
        type: string
    - name: biographicData
      in: query
      required: true
      schema:
        $ref: '#/components/schemas/BiographicDataType'
  responses:
    200:
      $ref: '#/components/responses/success'
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/invalidEncounterId'
              - $ref: '#/components/schemas/unknownBiographicFormat'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownGallery'
              - $ref: '#/components/schemas/unknownEncounter'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotStoreData'
              - $ref: '#/components/schemas/cannotUpdateData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.24 Set Biometric Data

7.2.24.1 Description

The *Set Biometric Data* service shall associate biometric data to a given subject record. Depending on the identity model in use, the biometric information should replace any existing biometric information (person-centric model) or it shall be added within a new encounter (encounter-centric model). If biographic or document data was also collected during the same encounter, the same encounter ID shall be specified by the caller in order to link the biometric data with the associated biographic and/or document information (using the *Set Biographic Data* and *Set Document Data* services). If the encounter ID is omitted for the encounter-centric model, an error shall be returned.

NOTE Biometric data is typically set within the master database. The updating of the comparison engine database (if separate) is governed by service provider policy.

For encounter-based systems, the *Create Encounter* service shall be called prior to *Set Biometric Data*. The Encounter ID assigned as a result (and returned by the *Create Encounter* service) shall be used as input to this service.

7.2.24.2 REST method

PATCH

7.2.24.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group to which the biometric will be added.

— *Encounter ID (optional)* — the identifier of the encounter.

— *BIR List* — data structure containing the new biometric sample(s).

7.2.24.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 23 (see 9.3 for the meaning of each error condition code).

Table 23 — Particular responses for *Set Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Encounter ID*, as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		INVALID_BIR	
		UNKNOWN_FORMAT	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_STORE_DATA	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Encounter ID* — the identifier of the encounter.

7.2.24.5 YAML specification

To be placed in section #/paths:

```
/setBiometricData:
  post:
    summary: Set Biometric Data
    parameters:
```

```
    - name: subjectID
      in: query
      required: true
      schema:
        type: string
    - name: galleryID
      in: query
      required: false
      schema:
        type: string
    - name: encounterID
      in: query
      required: false
      schema:
        type: string
    - name: birList
      in: query
      required: true
      schema:
        $ref: '#/components/schemas/CBEFF_BIR_ListType'
  responses:
    200:
      description: Success. Returns encounter ID created
      content:
        application/json:
          schema:
            type: object
            properties:
              encounterID:
                type: string
                description: The identifier of the encounter
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/invalidEncounterId'
              - $ref: '#/components/schemas/invalidBir'
              - $ref: '#/components/schemas/unknownFormat'
              - $ref: '#/components/schemas/biometricTypeNotSupported'
    403:
      description: Forbidden
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/birDecryptionFailure'
              - $ref: '#/components/schemas/birQualityError'
              - $ref: '#/components/schemas/birSignatureFailure'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownGallery'
              - $ref: '#/components/schemas/unknownEncounter'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotCheckQuality'
              - $ref: '#/components/schemas/cannotStoreData'
              - $ref: '#/components/schemas/cannotUpdateData'
              - $ref: '#/components/schemas/internalDatabaseError'
```

```
            - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.25 Set Document Data

7.2.25.1 Description

The *Set Document Data* service shall associate identity document data to a given subject record. Depending on the identity model in use, the document information should replace any existing document information for the same document category (person-centric model) or it shall be added within a new encounter (encounter-centric model). If biographic or biometric data was also collected during the same encounter, the same encounter ID shall be specified by the caller in order to link the document data with the associated biographic and/or biometric information (using the *Set Biographic Data* and *Set Biometric Data* services). If the encounter ID is omitted for the encounter-centric model, an error shall be returned.

NOTE Document data is typically set within the master database only. However, this is implementation-specific.

For encounter-based systems, the *Create Encounter* service shall be called prior to *Set Document Data*. The Encounter ID assigned as a result (and returned by the *Create Encounter* service) shall be used as input to this service.

7.2.25.2 REST method

PATCH

7.2.25.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Gallery ID (optional)* — the identifier of the gallery or population group to which the biographic will be added.

— *Encounter ID (optional)* — the identifier of the encounter.

— *Document Data* — a list of document data to associate with the subject or encounter.

7.2.25.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 24 (see 9.3 for the meaning of each error condition code).

Table 24 — Particular responses for *Set Document Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Encounter ID*, as seen below table.
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	

Table 24 *(continued)*

HTTP status code	HTTP status code meaning	Error condition code	Description
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_GALLERY	
		UNKNOWN_ENCOUNTER	
		UNKNOWN_DOCUMENT_CATEGORY	
500	Internal Server Error	CANNOT_STORE_DATA	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Encounter ID* — the identifier of the encounter.

7.2.25.5 YAML specification

To be placed in section #/paths:

```
/setDocumentData:
  post:
    summary: Set Document Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: documentData
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/DocumentDataListType'
    responses:
      200:
        description: Success. Returns encounter ID created
        content:
          application/json:
            schema:
              type: object
              properties:
                encounterID:
                  type: string
                  description: The identifier of the encounter
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidEncounterId'
      404:
```

```
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownGallery'
              - $ref: '#/components/schemas/unknownEncounter'
              - $ref: '#/components/schemas/unknownDocumentCategory'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotStoreData'
              - $ref: '#/components/schemas/cannotUpdateData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.26 Transform Biometric Data

7.2.26.1 Description

The *Transform Biometric Data* service shall transform or process a given biometric in one format into a new target format. Examples of transformations include:

— feature extraction;

— centring or cropping biometric images;

— standard biometric data format conversion.

The service includes an optional parameter to allow the requestor to specify certain controls for a particular transform operation (e.g. compression ratio, target record size, or target resolution). See the *Query Capabilities* service for a description of the use of the Transform Operation and Transform Control parameters.

7.2.26.2 REST method

GET

7.2.26.3 Parameters

— *Input BIR* — data structure containing the biometric information to be transformed.

— *Transform Operation* — value indicating the type of transformation to perform.

— *Transform Control (optional)* — specifies controls for the requested transform operation.

7.2.26.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 25 (see 9.3 for the meaning of each error condition code).

Table 25 — Particular responses for *Transform Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Output BIR*, as seen below table.
400	Bad Request	INVALID_BIR	
		INVALID_INPUT	
		INVALID_PROCESSING_OPTION	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
500	Internal Server Error	CANNOT_PROCESS_DATA	

A successful execution of this service will provide:

— *Output BIR* — data structure containing the new, transformed biometric information.

7.2.26.5 YAML specification

To be placed in section #/paths:

```
/transformBiometricData:
  get:
    summary: Classify Biometric Data
    parameters:
      - name: bir
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
      - name: transformOperation
        in: query
        required: true
        schema:
          type: integer
          format: int64
      - name: transformControl
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        description: Success. Returns classification
        content:
          application/json:
            schema:
              type: object
              properties:
                outputBIR:
                  # data structure containing the new, transformed
                  # biometric information
                  $ref: '#/components/schemas/CBEFF_BIR_Type'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidBir'
```

```
            - $ref: '#/components/schemas/invalidInput'
            - $ref: '#/components/schemas/invalidProcessingOptions'
            - $ref: '#/components/schemas/biometricTypeNotSupported'
    403:
      description: Forbidden
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/birDecryptionFailure'
              - $ref: '#/components/schemas/birQualityError'
              - $ref: '#/components/schemas/birSignatureFailure'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotProcessData'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.27 Update Biographic Data

7.2.27.1 Description

The *Update Biographic Data* service shall update the biographic data for an existing subject record. The service shall replace any existing biographic data with the new biographic data. In the encounter-centric model, the encounter ID shall be specified.

7.2.27.2 REST method

PUT

7.2.27.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Biographic Data* — list of updated biographic data elements.

7.2.27.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 26 (see 9.3 for the meaning of each error condition code).

Table 26 — Particular responses for *Update Biographic Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		UNKNOWN_BIOGRAPHIC_FORMAT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	

Table 26 *(continued)*

HTTP status code	HTTP status code meaning	Error condition code	Description
500	Internal Server Error	CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.27.5 YAML specification

To be placed in section #/paths:

```
/updateBiographicData:
  post:
    summary: Update Biographic Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: biographicData
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/BiographicDataListType'
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidEncounterId'
                - $ref: '#/components/schemas/unknownBiographicFormat'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownSubject'
                - $ref: '#/components/schemas/unknownEncounter'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotUpdateData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.2.28 Update Biometric Data

7.2.28.1 Description

The *Update Biometric Data* service shall update a single biometric sample for an existing subject record. The service includes an optional parameter indicating if the new biometric sample should be merged with the existing biometric sample. If this parameter is set to "False" or is not used in the service request, the service shall replace the existing biometric sample with the new biometric sample. In the encounter-centric model, the encounter ID shall be specified.

NOTE The term "merge" is implementation-specific. However, it may include either adding the sample to a multi-sample record or performing some level of biometric fusion (e.g. sample or feature level fusion).

7.2.28.2 REST method

PUT

7.2.28.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Merge (optional)* — value indicating if the input biometric sample should be merged with any existing biometric information.

— *BIR* — data structure containing the new biometric sample.

7.2.28.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 27 (see 9.3 for the meaning of each error condition code).

Table 27 — Particular responses for *Update Biometric Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
		INVALID_BIR	
		UNKNOWN_FORMAT	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.28.5 YAML specification

To be placed in section #/paths:

```
/updateBiometricData:
  post:
    summary: Update Biometric Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: merge
        in: query
        required: false
        schema:
          type: boolean
      - name: bir
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
    responses:
      200:
        $ref: '#/components/responses/success'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidEncounterId'
                - $ref: '#/components/schemas/invalidBir'
                - $ref: '#/components/schemas/unknownFormat'
                - $ref: '#/components/schemas/biometricTypeNotSupported'
      403:
        description: Forbidden
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/birDecryptionFailure'
                - $ref: '#/components/schemas/birQualityError'
                - $ref: '#/components/schemas/birSignatureFailure'
      404:
        description: Not Found
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/unknownSubject'
                - $ref: '#/components/schemas/unknownEncounter'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotCheckQuality'
                - $ref: '#/components/schemas/cannotUpdateData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
```

```
        $ref: '#/components/responses/x503'
```

7.2.29 Update Document Data

7.2.29.1 Description

The *Update Document Data* service shall update the document data for an existing subject record. The service shall replace any existing document data of the same category with the new document data. In the encounter-centric model, the encounter ID shall be specified.

7.2.29.2 REST method

PUT

7.2.29.3 Parameters

— *Subject ID* — the identifier of the subject.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Document Data* — list of updated document data.

7.2.29.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 28 (see 9.3 for the meaning of each error condition code).

Table 28 — Particular responses for *Update Document Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	
400	Bad Request	INVALID_SUBJECT_ID	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
		UNKNOWN_FORMAT	
		UNKNOWN_DOCUMENT_CATEGORY	
500	Internal Server Error	CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

7.2.29.5 YAML specification

To be placed in section #/paths:

```
/updateDocumentData:
  post:
    summary: Update Document Data
    parameters:
      - name: subjectID
        in: query
        required: true
        schema:
          type: string
      - name: encounterID
        in: query
```

```
      required: false
      schema:
        type: string
    - name: documentData
      in: query
      required: true
      schema:
        $ref: '#/components/schemas/DocumentDataListType'
  responses:
    200:
      $ref: '#/components/responses/success'
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidSubjectId'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/invalidEncounterId'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownSubject'
              - $ref: '#/components/schemas/unknownEncounter'
              - $ref: '#/components/schemas/unknownFormat'
              - $ref: '#/components/schemas/unknownDocumentCategory'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotUpdateData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.2.30 Verify Subject

7.2.30.1 Description

The *Verify Subject* service shall perform a 1:1 verification comparison between a given biometric and either a claim to identity in a given gallery or another provided biometric. As such, either the Identity Claim or Reference BIR input parameters are required.

NOTE When in encounter mode, for comparison operations, it is not necessary to explicitly create an encounter. The IAVS service provider will create the encounter and will set the encounter type to "recognition".

7.2.30.2 REST method

GET

7.2.30.3 Parameters

— *Input BIR* — data structure containing the biometric sample for the search.

NOTE 1 When multiple samples are included as input (e.g. in a multimodal operation), a complex BIR is used.

— *Reference BIR (conditional)* — data structure containing the biometric sample that will be compared to the Input BIR, required if no Identity Claim is provided.

— *Identity Claim (conditional)* — the identifier by which the subject is known to the gallery, required if no Reference BIR is provided.

NOTE 2 An identity claim can be the Subject ID or a different unique piece of biographic data used by the gallery as a key (e.g. account number or username).

— *Gallery ID (optional)* — the identifier of the gallery or population group of which the subject needs to be a member.

7.2.30.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 29 (see 9.3 for the meaning of each error condition code).

Table 29 — Particular responses for *Verify Subject* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Match* — *Score* as seen below table.
400	Bad Request	INVALID_BIR	
		INVALID_INPUT	
		INVALID_IDENTITY_CLAIM	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
		UNKNOWN_FORMAT	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	UNKNOWN_GALLERY	
		UNKNOWN_IDENTITY_CLAIM	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_VERIFY_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Match* — indicates if the Input BIR matched either the biometric information associated with the Identity Claim or the Reference BIR.

— *Score (optional)* — the comparison score, if the biometric information matched.

7.2.30.5 YAML specification

To be placed in section #/paths:

```
/verifySubject:
  get:
    summary: Verify Subject
    parameters:
      - name: inputBIR
        in: query
        required: true
```

```
      schema:
        $ref: '#/components/schemas/CBEFF_BIR_Type'
    - name: referenceBIR
      in: query
      required: true
      schema:
        $ref: '#/components/schemas/CBEFF_BIR_Type'
    - name: identityClaim
      in: query
      required: false
      schema:
        type: string
    - name: galleryID
      in: query
      required: false
      schema:
        type: string
  responses:
    200:
      description: >
        Success. Returns Biographic Data Elements and/or Encounter List
      content:
        application/json:
          schema:
            type: object
            properties:
              match:
                description: >
                  indicates if the Input BIR matched either the biometric
                  information associated with the Identity Claim or the
                  Reference BIR
                type: boolean
              score:
                description: >
                  the comparison score, if the biometric information
                  matched
                type: number
                format: float
    400:
      description: Bad Request
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/invalidBir'
              - $ref: '#/components/schemas/invalidInput'
              - $ref: '#/components/schemas/invalidIdentityClaim'
              - $ref: '#/components/schemas/biometricTypeNotSupported'
              - $ref: '#/components/schemas/unknownFormat'
    403:
      description: Forbidden
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/birDecryptionFailure'
              - $ref: '#/components/schemas/birQualityError'
              - $ref: '#/components/schemas/birSignatureFailure'
    404:
      description: Not Found
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/unknownGallery'
              - $ref: '#/components/schemas/unknownIdentityClaim'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
```

```
            oneOf:
              - $ref: '#/components/schemas/cannotCheckQuality'
              - $ref: '#/components/schemas/cannotProcessData'
              - $ref: '#/components/schemas/cannotVerifyData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.3 Aggregated Services

IAVS offers the following set of aggregate services. The intent of IAVS is to standardize the service request. System requirements and organizational business rules will determine how the service is implemented. While the description for an aggregate service may provide examples of how each one may be implemented using the primitive services defined in 7.2, service providers are not required to utilize any of the primitive services when implementing the aggregate services.

7.3.1 Delete

7.3.1.1 Description

The *Delete* aggregate service shall delete an existing subject or, in an encounter-centric model, an existing encounter from the system. This may be accomplished in a number of different ways according to system requirements and/or resources. For example, this aggregate service may initiate one or more *Delete Subject from Gallery, Delete Biographic Data, Delete Biometric Data* and *Delete Subject* primitive services to delete subject information from the system.

If the *Delete* aggregate service is implemented as a synchronous service, the implementing system shall immediately process the request and return the results in the Return Data parameter. If the *Delete* aggregate service is implemented as an asynchronous service, the implementing system shall return a token in the Return Data parameter, which is an indication that the request is being handled asynchronously. In this case, the *Get Deletion Results* service shall be used to poll for the results of the *Delete* request.

7.3.1.2 REST method

DELETE

7.3.1.3 Parameters

— *Processing Options* — options that guide how the service request is processed.

— *Subject ID (conditional)* — the identifier assigned to the subject.

— *Encounter ID (conditional)* — the identifier of the encounter, required for encounter-centric models.

— *Input Data (optional)* — contains an input data record, which may include biometric data.

7.3.1.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 30 (see 9.3 for the meaning of each error condition code).

Table 30 — Particular responses for *Delete* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Token* — *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_INPUT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Token (conditional)* — a token used to retrieve the results of the *Delete* request; returned with asynchronous request processing. If set to zero, operation is processed synchronously. If set to a non-zero value, operation is processed asynchronously and *Get Deletion Results* are used to retrieve the results.

— *Return Data (optional)* — contains a return data record.

7.3.1.5 YAML specification

To be placed in section #/paths:

```
/delete:
  get:
    summary: Retrieve Biometric Data
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: subjectID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: inputData
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/InformationType'
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
```

```
          type: object
          properties:
            token:
              # a token used to retrieve the results of the Delete
              # request; returned with asynchronous request processing.
              # If set to zero, operation is processed synchronously.
              # If set to a non-zero value, operation is processed
              # asynchronously and Get Deletion Results must be used
              # to retrieve the results.
              $ref: '#/components/schemas/TokenType'
            returnData:
              # contains a return data record
              $ref: '#/components/schemas/InformationType'
  400:
    description: Bad Request
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/invalidProcessingOptions'
            - $ref: '#/components/schemas/invalidSubjectId'
            - $ref: '#/components/schemas/invalidEncounterId'
            - $ref: '#/components/schemas/invalidInput'
  404:
    description: Not Found
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/unknownSubject'
            - $ref: '#/components/schemas/unknownEncounter'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotDeleteData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.3.2 Enrol

7.3.2.1 Description

The *Enrol* aggregate service shall add a new subject or, in an encounter-centric model, a new encounter to the system. This may be accomplished in a number of different ways according to system requirements and/or resources. For example, this aggregate service may initiate one or more *Identify Subject* primitive service requests to determine if the given subject is already known to the system. If the subject is not previously known to the system, any or all of the *Create Subject*, *Set Biographic Data*, *Set Biometric Data*, and *Add Subject to Gallery* primitive services may be utilized to add subject information to the system. If the subject is previously known to the system, the service may:

a) do nothing;

b) initiate an *Update Biographic Data* and/or *Update Biometric Data* primitive service request in a person-centric model; or

c) initiate a *Set Biographic Data* and/or *Set Biometric Data* primitive service request in an encounter-centric model.

For encounter-centric models, the encounter ID may optionally be specified by the caller. If the encounter ID is omitted for the encounter-centric model, the service shall return a system-assigned encounter ID.

If the *Enrol* aggregate service is implemented as a synchronous service, the implementing system shall immediately process the request and return the results in the Return Data parameter. If the *Enrol* aggregate service is implemented as an asynchronous service, the implementing system shall return a non-zero token in the Token parameter, which is an indication that the request is being handled asynchronously. In this case, the *Get Enrol Results* service shall be used to poll for the results of the *Enrol* request.

7.3.2.2 REST method

POST

7.3.2.3 Parameters

— *Processing Options* — options that guide how the service request is processed.

— *Input Data* — contains a subject enrolment record.

— *Input Encounter ID (optional)* — the identifier of the encounter; required for encounter-centric models.

7.3.2.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 31 (see 9.3 for the meaning of each error condition code).

Table 31 — Particular responses for *Enrol* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Output Encounter ID* — *Subject ID* — *Token* — *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_INPUT	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_CREATE_TEMPLATE	
		CANNOT_PROCESS_DATA	
		CANNOT_STORE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Output Encounter ID (conditional)* — the identifier of the encounter; required for encounter-centric models.

— *Subject ID (conditional)* — the identifier assigned to the subject.

— *Token (conditional)* — a token used to retrieve the results of the *Enrol* request; returned with asynchronous request processing. If set to zero, operation is processed synchronously. If set to a

non-zero value, operation is processed asynchronously and *Get Enrol Results* is used to retrieve the results.

— *Return Data (optional)* — contains a return data record.

7.3.2.5 YAML specification

To be placed in section #/paths:

```
/enrol:
  get:
    summary: Enrol
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: inputData
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/InformationType'
      - name: inputEncounterID
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        description: >
          Success. Returns Biographic Data Elements and/or Encounter List
        content:
          application/json:
            schema:
              type: object
              properties:
                outputEncounterID:
                  description: >
                    The identifier of the encounter, required for
                    enounter-centric models
                  type: string
                subjectID:
                  description: >
                    The identifier of the subject
                  type: string
                token:
                  # a token used to retrieve the results of the Enrol
                  # request; returned with asynchronous request
                  # processing. If set to zero, operation is processed
                  # synchronously. If set to a non-zero value, operation
                  # is processed asynchronously and Get Enrol Results
                  # must be used to retrieve the results.
                  $ref: '#/components/schemas/TokenType'
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidProcessingOptions'
                - $ref: '#/components/schemas/invalidInput'
                - $ref: '#/components/schemas/invalidEncounterId'
      404:
        description: Unknown Encounter
        content:
          application/json:
```

```
        schema:
          $ref: '#/components/schemas/unknownEncounter'
    500:
      description: Internal Server Error
      content:
        application/json:
          schema:
            oneOf:
              - $ref: '#/components/schemas/cannotCreateTemplate'
              - $ref: '#/components/schemas/cannotProcessData'
              - $ref: '#/components/schemas/cannotStoreData'
              - $ref: '#/components/schemas/internalDatabaseError'
              - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

7.3.3 Get Deletion Results

7.3.3.1 Descriptions

The *Get Deletion Results* aggregate service shall retrieve the deletion results for the specified token. This service is used in conjunction with the *Delete* aggregate service. If the *Delete* aggregate service is implemented as an asynchronous service, the implementing system returns a token, and the *Get Deletion Results* service is used to poll for the results of the original *Delete* request.

7.3.3.2 REST method

GET

7.3.3.3 Parameters

— *Token* — a value used to retrieve the results of the *Delete* request.

7.3.3.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 32 (see 9.3 for the meaning of each error condition code).

Table 32 — Particular responses for *Get Deletion Results* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Return Data* as seen below table.
400	Bad Request	INVALID_TOKEN	
404	Not Found	TOKEN_EXPIRED	
500	Internal Server Error	CANNOT_DELETE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Return Data (optional)* — contains a return data record.

7.3.3.5 YAML specification

To be placed in section #/paths:

```
  /getDeletionResults:
    get:
```

```
summary: Get Deletion Results
parameters:
  - name: token
    in: query
    required: true
    schema:
      $ref: '#/components/schemas/TokenType'
responses:
  200:
    description: >
      Success. Returns Deletion Results
    content:
      application/json:
        schema:
          type: object
          properties:
            returnData:
              # contains a return data record
              $ref: '#/components/schemas/InformationType'
  400:
    description: Invalid Token
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/invalidToken'
  404:
    description: Token Expired
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/tokenExpired'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotDeleteData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.3.4 Get Enrol Results

7.3.4.1 Descriptions

The *Get Enrol Results* aggregate service shall retrieve the enrolment results for the specified token. This service is used in conjunction with the *Enrol* aggregate service. If the *Enrol* aggregate service is implemented as an asynchronous service, the implementing system returns a token, and the *Get Enrol Results* service is used to poll for the results of the original *Enrol* request.

7.3.4.2 REST method

GET

7.3.4.3 Parameters

— *Token* — a value used to retrieve the results of the *Enrol* request.

7.3.4.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 33 (see 9.3 for the meaning of each error condition code).

Table 33 — Particular responses for *Enrol* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Encounter ID* — *Subject ID* — *Return Data* as seen below table.
400	Bad Request	INVALID_TOKEN	
404	Not Found	TOKEN_EXPIRED	
500	Internal Server Error	CANNOT_CREATE_TEMPLATE	
		CANNOT_PROCESS_DATA	
		CANNOT_STORE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Encounter ID (conditional)* — the identifier of the encounter, if assigned.

— *Subject ID (conditional)* — the identifier assigned to the subject.

— *Return Data (optional)* — contains a return data record.

7.3.4.5 YAML specification

To be placed in section #/paths:

```
/getEnrolResults:
  get:
    summary: Get Enrol Results
    parameters:
      - name: token
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/TokenType'
    responses:
      200:
        description: >
          Success. Returns Enrolment Results
        content:
          application/json:
            schema:
              type: object
              properties:
                encounterID:
                  description: >
                    The identifier of the encounter, if assigned
                  type: string
                subjectID:
                  description: >
                    The identifier assigned to the subject
                  type: string
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Invalid Token
        content:
          application/json:
```

```
      schema:
        $ref: '#/components/schemas/invalidToken'
404:
  description: Token Expired
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/tokenExpired'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotCreateTemplate'
          - $ref: '#/components/schemas/cannotProcessData'
          - $ref: '#/components/schemas/cannotStoreData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.3.5 Get Identify Results

7.3.5.1 Description

The *Get Identify Results* aggregate service shall retrieve the identification results for the specified token. This service is used in conjunction with the *Identify* aggregate service. If the *Identify* aggregate service is implemented as an asynchronous service, the implementing system returns a non-zero token, and the *Get Identify Results* service is used to poll for the results of the original *Identify* request.

7.3.5.2 REST method

GET

7.3.5.3 Parameters

— *Token* — a value used to retrieve the results of the *Identify* request

7.3.5.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 34 (see 9.3 for the meaning of each of the error condition code).

Table 34 — Particular responses for *Get Identify Results* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Candidate List* — *Encounter ID* — *Return Data* as seen below table.
400	Bad Request	INVALID_TOKEN	
404	Not Found	TOKEN_EXPIRED	
		IDENTIFICATION_RESULT_NOT_YET_AVAILABLE	

Table 34 *(continued)*

HTTP status code	HTTP status code meaning	Error condition code	Description
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_IDENTIFY_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Candidate List (conditional)* — a rank-ordered list of candidates that have a likelihood of matching the input biometric sample.

— *Encounter ID (conditional)* — the identifier of the encounter, if assigned.

— *Return Data (optional)* — contains a return data record.

7.3.5.5 YAML specification

To be placed in section #/paths:

```
/getIdentifyResults:
  get:
    summary: Get Identify Results
    parameters:
      - name: token
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/TokenType'
    responses:
      200:
        description: >
          Success. Returns Identify Results
        content:
          application/json:
            schema:
              type: object
              properties:
                candidateList:
                  # a rank-ordered list of candidates that have
                  # a likelihood of matching the input biometric
                  # sample
                  $ref: '#/components/schemas/CandidateListType'
                encounterID:
                  description: >
                    The identifier of the encounter, if assigned
                  type: string
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Invalid Token
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/invalidToken'
      404:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/tokenExpired'
                - $ref: '#/components/schemas/identificationResultNotYetAvailable'
```

```
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotCheckQuality'
          - $ref: '#/components/schemas/cannotProcessData'
          - $ref: '#/components/schemas/cannotIdentifyData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.3.6 Get Update Results

7.3.6.1 Descriptions

The *Get Update Results* aggregate service shall retrieve the update results for the specified token. This service is used in conjunction with the *Update* aggregate service. If the *Update* aggregate service is implemented as an asynchronous service, the implementing system returns a token, and the *Get Update Results* service is used to poll for the results of the original *Update* request.

7.3.6.2 REST method

GET

7.3.6.3 Parameters

— *Token* — a value used to retrieve the results of the *Update* request.

7.3.6.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 35 (see 9.2 for the meaning of each error condition code).

Table 35 — Particular responses for *Get Update Results* service

HTTP status code	**HTTP status code meaning**	**Error condition code**	**Description**
200	OK	SUCCESS	Adding also *Return Data* as seen below.
400	Bad Request	INVALID_TOKEN	
404	Not Found	TOKEN_EXPIRED	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Return Data (optional)* — contains a return data record.

7.3.6.5 YAML specification

To be placed in section #/paths:

```
/getUpdateResults:
  get:
    summary: Get Update Results
    parameters:
      - name: token
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/TokenType'
    responses:
      200:
        description: >
          Success. Returns Update Results
        content:
          application/json:
            schema:
              type: object
              properties:
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Invalid Token
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/invalidToken'
      404:
        description: Token Expired
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/tokenExpired'
      500:
        description: Internal Server Error
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/cannotCheckQuality'
                - $ref: '#/components/schemas/cannotProcessData'
                - $ref: '#/components/schemas/cannotUpdateData'
                - $ref: '#/components/schemas/internalDatabaseError'
                - $ref: '#/components/schemas/unknownError'
      503:
        $ref: '#/components/responses/x503'
```

7.3.7 Get Verify Results

7.3.7.1 Description

The *Get Verify Results* aggregate service shall retrieve the verification results for the specified token. This service is used in conjunction with the *Verify* aggregate service. If the *Verify* aggregate service is implemented as an asynchronous service, the implementing system returns a non-zero token, and the *Get Verify Results* service is used to poll for the results of the original *Verify* request.

7.3.7.2 REST method

GET

7.3.7.3 Parameters

— *Token* — a value used to retrieve the results of the *Verify* request.

7.3.7.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 36 (see 9.3 for the meaning of each error condition code).

Table 36 — Particular responses for *Get Verify Results* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Match* — *Score* as seen below table.
400	Bad Request	INVALID_TOKEN	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	
404	Not Found	TOKEN_EXPIRED	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_VERIFY_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Match* — indicates if the Input BIR matched either the biometric information associated with the Identity Claim or the Reference BIR.

— *Score (optional)* — the comparison score, if the biometric information matched.

— *Encounter ID (conditional)* — the identifier of the encounter, if assigned.

— *Return Data (optional)* — contains a return data record.

7.3.7.5 YAML specification

To be placed in section #/paths:

```
/getVerifyResults:
  get:
    summary: Get Verify Results
    parameters:
      - name: token
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/TokenType'
    responses:
      200:
        description: >
          Success. Returns Verify Results
        content:
          application/json:
            schema:
```

```
        type: object
        properties:
          match:
            description: >
              indicates if the Input BIR matched either the
              biometric information associated with the Identity
              Claim or the Reference BIR
            type: boolean
          score:
            description: >
              the comparison score, if the biometric information
              matched
            type: number
            format: float
          encounterID:
            description: >
              The identifier of the encounter, if assigned
            type: string
          returnData:
            # contains a return data record
            $ref: '#/components/schemas/InformationType'
400:
  description: Invalid Token
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/invalidToken'
403:
  description: Forbidden
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/birDecryptionFailure'
          - $ref: '#/components/schemas/birQualityError'
          - $ref: '#/components/schemas/birSignatureFailure'
404:
  description: Token Expired
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/tokenExpired'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotCheckQuality'
          - $ref: '#/components/schemas/cannotProcessData'
          - $ref: '#/components/schemas/cannotVerifyData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```


7.3.8 Identify

7.3.8.1 Description

The *Identify* aggregate service shall perform an identification function according to system requirements and/or resources. For example, a system may have multiple galleries of subjects, and may utilize any or all of these galleries, via calls to the *Identify Subject* primitive service, to perform a system-level identification function. The system may perform additional actions based on input data and/or results of the *Identify Subject* primitive service requests. For example, in an encounter-centric model, this aggregate service may search three separate galleries of subjects, and if a match is found it may then utilize the *Set Biographic Data* and/or *Set Biometric Data* primitive services to create a new encounter for the subject. If this occurs, the service shall return a system-assigned encounter ID.

If the *Identify* aggregate service is implemented as a synchronous service, the implementing system shall immediately process the request and return the results in the Return Data parameter. If the *Identify* aggregate service is implemented as an asynchronous service, the implementing system shall return a non-zero token in the Token parameter, which is an indication that the request is being handled asynchronously. In this case, the *Get Identify Results* service shall be used to poll for the results of the *Identify* request.

NOTE When in encounter mode, for comparison operations, it is not necessary to explicitly create an encounter. The IAVS service provider will create the encounter and will set the encounter type to "recognition".

7.3.8.2 REST method

GET

7.3.8.3 Parameters

— *Processing Options* — options that guide how the service request is processed.

— *Input Data* — contains an input data record, which at a minimum must include biometric data.

— *Gallery ID (optional)* — the identifier of the gallery or population group which will be searched. This parameter may also be used to identify an external system where the identification request should be forwarded, if this capability is supported by the implementing system.

— *Max List Size* — the maximum size of the candidate list that should be returned.

7.3.8.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 37 (see 9.3 for the meaning of each error condition code).

Table 37 — Particular responses for *Identify* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Candidate List* — *Encounter ID* — *Token* — *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_INPUT	
404	Not Found	UNKNOWN_GALLERY	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_IDENTIFY_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Candidate List (conditional)* — a rank-ordered list of candidates that have a likelihood of matching the input biometric sample; returned with successful, synchronous request processing.

— *Encounter ID (conditional)* — the identifier of the encounter, if assigned.

— *Token (conditional)* — a token used to retrieve the results of the *Identify* request; returned with asynchronous request processing. If set to zero, operation is processed synchronously and candidate list is returned. If set to a non-zero value, operation is processed asynchronously and *Get Identify Results* is used to retrieve the results.

— *Return Data (optional)* — contains a return data record.

7.3.8.5 YAML specification

To be placed in section #/paths:

```
/identify:
  get:
    summary: Identify
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: inputData
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/InformationType'
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
      - name: maxListSize
        in: query
        required: true
        schema:
          type: integer
          format: int32
    responses:
      200:
        description: >
          Success. Returns Identify-related results
        content:
          application/json:
            schema:
              type: object
              properties:
                candidateList:
                  # A rank-ordered list of candidates that have a likelihood
                  # of matching the input biometric sample; returned with
                  # successful, synchronous request processing
                  $ref: '#/components/schemas/CBEFF_BIR_ListType'
                encounterID:
                  description: >
                    The identifier of the encounter, if assigned
                  type: string
                token:
                  # a token used to retrieve the results of the Identify
                  # request; returned with asynchronous request processing.
                  # If set to zero, operation is processed synchronously
                  # and candidate list is returned. If set to a non-zero
                  # value, operation is processed asynchronously and Get
                  # Identify Results must be used to retrieve the results
                  $ref: '#/components/schemas/TokenType'
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Bad Request
        content:
          application/json:
```

```
      schema:
        oneOf:
          - $ref: '#/components/schemas/invalidProcessingOptions'
          - $ref: '#/components/schemas/invalidInput'
404:
  description: Unknown GAllery
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/unknownGallery'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotCheckQuality'
          - $ref: '#/components/schemas/cannotProcessData'
          - $ref: '#/components/schemas/cannotIdentifyData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.3.9 Retrieve Data

7.3.9.1 Description

The *Retrieve Data* aggregate service shall retrieve requested information about a subject, or in an encounter-centric model about an encounter. In a person-centric model, this aggregate service may be used to retrieve both biographic and biometric information for a subject record. In an encounter-centric model, this aggregate service may be used to retrieve biographic and/or biometric information for either a single encounter or all encounters. Either a subject ID or encounter ID is specified.

7.3.9.2 REST method

GET

7.3.9.3 Parameters

— *Processing Options* — options that guide how the service request is processed, and may identify what type(s) of information should be returned.

— *Subject ID (conditional)* — the identifier of the subject; required if no encounter ID is provided.

— *Encounter ID (conditional)* — the identifier of the encounter; required if no subject ID is provided.

7.3.9.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 38 (see 9.3 for the meaning of each error condition code).

Table 38 — Particular responses for *Retrieve Data* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
		NON_EXISTENT_DATA	
500	Internal Server Error	CANNOT_RETRIEVE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Return Data* — contains a return data record.

7.3.9.5 YAML specification

To be placed in section #/paths:

```
/retrieveData:
  get:
    summary: Retrieve Data
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: subjectID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        description: >
          Success. Returns Data retrieved
        content:
          application/json:
            schema:
              type: object
              properties:
                returnData:
                  # contains a return data record
                  $ref: '#/components/schemas/InformationType'
      400:
        description: Bad Request
        content:
          application/json:
            schema:
              oneOf:
                - $ref: '#/components/schemas/invalidProcessingOptions'
                - $ref: '#/components/schemas/invalidSubjectId'
                - $ref: '#/components/schemas/invalidEncounterId'
```

```
404:
  description: Not Found
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/unknownSubject'
          - $ref: '#/components/schemas/unknownEncounter'
          - $ref: '#/components/schemas/nonExistentData'
500:
  description: Internal Server Error
  content:
    application/json:
      schema:
        oneOf:
          - $ref: '#/components/schemas/cannotRetrieveData'
          - $ref: '#/components/schemas/internalDatabaseError'
          - $ref: '#/components/schemas/unknownError'
503:
  $ref: '#/components/responses/x503'
```

7.3.10 Update

7.3.10.1 Description

The *Update* aggregate service shall update specified information about a subject, or in an encounter-centric model about an encounter. In a person-centric model, this aggregate service may be used to update both biographic and biometric information for a subject record. In an encounter-centric model, this aggregate service may be used to update biographic and/or biometric information for either a single encounter or all encounters. Either a subject ID or encounter ID is specified.

7.3.10.2 REST method

PATCH

7.3.10.3 Parameters

— *Processing Options* — options that guide how the service request is processed, and may identify what type(s) of information should be returned.

— *Subject ID (conditional)* — the identifier of the subject; required if no encounter ID is provided.

— *Encounter ID (conditional)* — the identifier of the encounter; required if no subject ID is provided.

— *Input Data* — contains subject information to update.

7.3.10.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 39 (see 9.4 for the meaning of each error condition code).

Table 39 — Particular responses for *Update* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Token* — *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_SUBJECT_ID	
		INVALID_ENCOUNTER_ID	
		INVALID_INPUT	
404	Not Found	UNKNOWN_SUBJECT	
		UNKNOWN_ENCOUNTER	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_UPDATE_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Token (conditional)* — a token used to retrieve the results of the Update request; returned with asynchronous request processing. If set to zero, operation is processed synchronously. If set to a non-zero value, operation is processed asynchronously and Get Update Results is used to retrieve the results.

— *Return Data (optional)* — contains a return data record.

7.3.10.5 YAML specification

To be placed in section #/paths:

```
/update:
  get:
    summary: Update
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: subjectID
        in: query
        required: false
        schema:
          type: string
      - name: encounterID
        in: query
        required: false
        schema:
          type: string
      - name: inputData
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/InformationType'
    responses:
      200:
        description: >
```

```
      Success. Returns Update-related data
    content:
      application/json:
        schema:
          type: object
          properties:
            token:
              # a token used to retrieve the results of the Update
              # request; returned with asynchronous request
              # processing. If set to zero, operation is processed
              # synchronously. If set to a non-zero value, operation
              # is processed asynchronously and Get Update Results
              # must be used to retrieve the results.
              $ref: '#/components/schemas/TokenType'
            returnData:
              # contains a return data record
              $ref: '#/components/schemas/InformationType'
  400:
    description: Bad Request
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/invalidProcessingOptions'
            - $ref: '#/components/schemas/invalidSubjectId'
            - $ref: '#/components/schemas/invalidEncounterId'
            - $ref: '#/components/schemas/invalidInput'
  404:
    description: Not Found
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/unknownSubject'
            - $ref: '#/components/schemas/unknownEncounter'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotCheckQuality'
            - $ref: '#/components/schemas/cannotProcessData'
            - $ref: '#/components/schemas/cannotUpdateData'
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
  503:
    $ref: '#/components/responses/x503'
```

7.3.11 Verify

7.3.11.1 Description

The *Verify* aggregate service shall perform a 1:1 verification function between a given biometric and either a claim of identity in a given gallery or another provided biometric, and according to system requirements and/or resources. Either the Identity Claim or Reference BIR input parameters are required. The system may perform additional actions based on input data and/or results of verification. For example, in an encounter-centric model, this aggregate service may initiate a request to the *Verify Subject* primitive service, and if a match is found it may then utilize the *Set Biographic Data* and/or *Set Biometric Data* primitive services to create a new encounter for the subject. If this occurs, the service shall return a system-assigned encounter ID.

For encounter-centric models, the encounter ID may optionally be specified by the caller. If the encounter ID is omitted for the encounter-centric model, the service shall return a system-assigned encounter ID.

NOTE When in encounter mode, for comparison operations, it is not necessary to explicitly create an encounter. The IAVS service provider will create the encounter and will set the encounter type to "recognition".

If the *Verify* aggregate service is implemented as a synchronous service, the implementing system shall immediately process the request and return the results in the Return Data parameter. If the *Verify* aggregate service is implemented as an asynchronous service, the implementing system shall return a non-zero token in the Token parameter, which is an indication that the request is being handled asynchronously. In this case, the *Get Verify Results* service shall be used to poll for the results of the *Verify* request.

7.3.11.2 REST method

GET

7.3.11.3 Parameters

— *Processing Options* — options that guide how the service request is processed and may identify what type(s) of information should be returned.

— *Input Data* — contains an input data record, which at a minimum is required to include biometric data.

— *Reference BIR (conditional)* — data structure containing the biometric sample that will be compared to the Input BIR, required if no Identity Claim is provided.

— *Identity Claim (conditional)* — the identifier by which the subject is known to the gallery, required if no Reference BIR is provided.

— *Gallery ID (optional)* — the identifier of the gallery or population group of which the subject is required to be a member.

7.3.11.4 Responses

In addition to the generic HTTP responses defined in 9.2, the execution of this service shall provide one of the responses listed in Table 40 (see 9.3 for the meaning of each error condition code).

Table 40 — Particular responses for *Verify* service

HTTP status code	HTTP status code meaning	Error condition code	Description
200	OK	SUCCESS	Adding also — *Match* — *Score* — *Encounter ID* — *Token* — *Return Data* as seen below table.
400	Bad Request	INVALID_PROCESSING_OPTIONS	
		INVALID_INPUT	
		INVALID_BIR	
		INVALID_IDENTITY_CLAIM	
		BIOMETRIC_TYPE_NOT_SUPPORTED	
		UNKNOWN_FORMAT	
403	Forbidden	BIR_DECRYPTION_FAILURE	
		BIR_QUALITY_ERROR	
		BIR_SIGNATURE_FAILURE	

Table 40 *(continued)*

HTTP status code	HTTP status code meaning	Error condition code	Description
404	Not Found	UNKNOWN_GALLERY	
		UNKNOWN_IDENTITY_CLAIM	
500	Internal Server Error	CANNOT_CHECK_QUALITY	
		CANNOT_PROCESS_DATA	
		CANNOT_VERIFY_DATA	
		INTERNAL_DATABASE_ERROR	

A successful execution of this service will provide:

— *Match* — indicates if the Input BIR matched either the biometric information associated with the Identity Claim or the Reference BIR.

— *Score (optional)* — the comparison score, if the biometric information matched.

— *Encounter ID (conditional)* — the identifier of the encounter, if assigned.

— *Token (conditional)* — a token used to retrieve the results of the *Verify* request; returned with asynchronous request processing. If set to zero, operation is processed synchronously. If set to a non-zero value, operation is processed asynchronously and *Get Verify Results* is used to retrieve the results.

— *Return Data (optional)* — contains a return data record.

7.3.11.5 YAML specification

To be placed in section #/paths:

```
/deleteBiographicData:
/verify:
  get:
    summary: Verify
    parameters:
      - name: processingOptions
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/ProcessingOptionsType'
      - name: inputData
        in: query
        required: true
        schema:
          $ref: '#/components/schemas/InformationType'
      - name: referenceBIR
        in: query
        required: false
        schema:
          $ref: '#/components/schemas/CBEFF_BIR_Type'
      - name: identityClaim
        in: query
        required: false
        schema:
          type: string
      - name: galleryID
        in: query
        required: false
        schema:
          type: string
    responses:
      200:
        description: >
```

```
      Success. Returns Verify Results
    content:
      application/json:
        schema:
          type: object
          properties:
            match:
              description: >
                indicates if the Input BIR matched either the
                biometric information associated with the Identity
                Claim or the Reference BIR
              type: boolean
            score:
              description: >
                the comparison score, if the biometric information
                matched
              type: number
              format: float
            encounterID:
              description: >
                The identifier of the encounter, if assigned
              type: string
            token:
              # a token used to retrieve the results of the Verify
              # request; returned with asynchronous request
              # processing. If set to zero, operation is processed
              # synchronously. If set to a non-zero value, operation
              # is processed asynchronously and Get Verify Results
              # must be used to retrieve the results
              $ref: '#/components/schemas/TokenType'
            returnData:
              # contains a return data record
              $ref: '#/components/schemas/InformationType'
  400:
    description: Bad Request
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/invalidProcessingOptions'
            - $ref: '#/components/schemas/invalidInput'
            - $ref: '#/components/schemas/invalidBir'
            - $ref: '#/components/schemas/invalidIdentityClaim'
            - $ref: '#/components/schemas/biometricTypeNotSupported'
            - $ref: '#/components/schemas/unknownFormat'
  403:
    description: Forbidden
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/birDecryptionFailure'
            - $ref: '#/components/schemas/birQualityError'
            - $ref: '#/components/schemas/birSignatureFailure'
  404:
    description: Not Found
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/unknownGallery'
            - $ref: '#/components/schemas/unknownIdentityClaim'
  500:
    description: Internal Server Error
    content:
      application/json:
        schema:
          oneOf:
            - $ref: '#/components/schemas/cannotCheckQuality'
            - $ref: '#/components/schemas/cannotProcessData'
            - $ref: '#/components/schemas/cannotVerifyData'
```

```
            - $ref: '#/components/schemas/internalDatabaseError'
            - $ref: '#/components/schemas/unknownError'
    503:
      $ref: '#/components/responses/x503'
```

8 Data elements and data types

8.1 Introduction

One goal of IAVS is to be flexible to the amount and types of biographic and biometric information available to and used by a system. The parameters “Biographic Data” and “Biometric Data” are intended to be general in this sense in order to allow this flexibility. This clause includes information on how this flexibility can be specified and supported by implementing systems.

8.2 Biographic data

8.2.1 General

IAVS defines three data types to provide flexibility for the amount and types of biographic data supported by implementing systems. The Biographic Data Item Type shall represent a single biographic data item, and the Biographic Data Set Type shall represent a set of biographic information in a specified format. The Biographic Data Type is a common type that shall represent either a set or list of biographic data.

8.2.2 Biographic Data Item Type

8.2.2.1 Description

The Biographic Data Item Type defines a single biographic data element. The biographic data item *name* and *type* are required elements, while the *value* is optional.

8.2.2.2 Definitions

— *Name* — the name of the biographic data item (e.g. "PersonName").

— *Type* — the data type for the biographic data item (e.g. "string").

— *Value (optional)* — the value assigned to the biographic data item (i.e. "John Doe").

8.2.2.3 YAML specification

To be placed in section #/components/schemas:

```
BiographicDataItemType:
  type: object
  required:
    - name
    - type
  properties:
    name:
      type: string
    type:
      type: string
    value:
      type: string
```

8.2.3 Biographic Data List Type

8.2.3.1 Description

The Biographic Data List Type shall provide a list of biographic data.

8.2.3.2 Definitions

— *Biographic Data* — data structure containing information about a biographic record.

8.2.3.3 YAML specification

To be placed in section #/components/schemas:

```
BiographicDataListType:
  type: array
  items:
    oneOf:
      - $ref: '#/components/schemas/BiographicDataType'
      - $ref: '#/components/schemas/BiographicDataItemType'
      - $ref: '#/components/schemas/BiographicDataSetType'
```

8.2.4 Biographic Data Set Type

8.2.4.1 Description

The Biographic Data Set Type defines a set of biographic data that is formatted according to the specified format. This data type allows biographic information in an electronic fingerprint transmission specification (EFTS) or pre-defined JSON format, for example, to be used with IAVS messages.

8.2.4.2 Definitions

This data type is an array of objects with the following elements:

— *name* — the name of the biographic data format (e.g. "EFTS").

— *version (optional)* — the version of the biographic data format (e.g. "7.1" or "1.0").

— *source* — reference to a URI describing the biographic data format.

— *type* — the biographic data format type (e.g. "JSON").

8.2.4.3 Common biographic data format references

In order to provide a consistent representation of some common biographic data formats, this specification will establish common values for the attributes used in the Biographic Data Set Type. The common biographic data format shall have the *name, source, and type* attributes in common. The *version* attribute, if used, shall reflect the source organization's assigned version. These shall be registered with the BRA. Samples of common biographic data formats are shown in ISO/IEC 30108-1:2015, Table B.1.

8.2.4.4 YAML specification

To be placed in section #/components/schemas:

```
BiographicDataSetType:
  type: array
  items:
    type: object
    required:
      - name
      - source
```

```
    - type
  properties:
    name:
      type: string
    version:
      type: string
    source:
      type: string
    type:
      type: string
```

8.2.5 Biographic Data Type

8.2.5.1 Description

The Biographic Data Type defines a set of biographic data elements, utilizing either the Biographic Data Item Type to represent a list of elements or the Biographic Data Set Type to represent a complete, formatted set of biographic information. This type also includes optional fixed fields for representing first and last names, two common biographic data elements.

8.2.5.2 Definitions

— *Last Name (optional)* — the last name of a subject.

— *First Name (optional)* — the first name of a subject.

— *Biographic Data Item* — a single biographic data element.

— *Biographic Data Set* — a set of biographic data information.

8.2.5.3 YAML specification

To be placed in section #/components/schemas:

```
BiographicDataType:
  type: object
  properties:
    lastName:
      type: string
    firstName:
      type: string
    biographicDataItem:
      $ref: '#/components/schemas/BiographicDataItemType'
    biographicDataSet:
      $ref: '#/components/schemas/BiographicDataSetType'
```

8.3 Biometric Data

8.3.1 General

This subclause defines how to represent biometric data in the IAVS services.

8.3.2 Biometric Data Element Type

8.3.2.1 Description

The Biometric Data Element Type shall provide descriptive information about biometric data, such as the biometric type, subtype, and format, contained in the BDB of the CBEFF-BIR.

8.3.2.2 Definitions

— *Biometric Type* — the type of biological or behavioural data stored in the biometric record, as defined by CBEFF.

— *Biometric Type Count (optional)* — the number of biometric records having the biometric type recorded in the biometric type field.

— *BDB Format Owner* — identifies the standards body, working group, industry consortium, or other CBEFF biometric organization that has defined the format for the biometric data.

— *BDB Format Type* — identifies the specific biometric data format specified by the CBEFF biometric organization recorded in the BDB Format Owner field.

8.3.2.3 YAML specification

To be placed in section #/components/schemas:

```
BiometricDataElementType:
  type: object
  required:
    - biometricType
    - bdbFormat
  properties:
    biometricType:
      $ref: '#/components/schemas/BiometricType'
    biometricTypeCount:
      type: integer
      format: int32
    bdbFormat:
      type: object
      properties:
        bdbOwner:
          type: string
        bdbType:
          type: string
```

8.3.3 Biometric Data List Type

8.3.3.1 Description

The Biometric Data List Type shall provide a list of biometric data elements.

8.3.3.2 Definitions

— *Biometric Data Element* — data structure containing information about a biometric record.

8.3.3.3 YAML specification

To be placed in section #/components/schemas:

```
BiometricDataListType:
  type: array
  items:
    $ref: '#/components/schemas/BiometricDataElementType'
```

8.3.4 Biometric Type

8.3.4.1 Description

The Biometric Type defines the type and subtype of biometric modality that is being referred to. It is used in several services. Coding is based on ISO/IEC 19785-3. As the resulting value for type comes from

a combination of types, a simple integer is used rather than enumeration. The values can be obtained in ISO/IEC 19785-3.

8.3.4.2 Definitions

— *Type* — type of biometric data, as defined in ISO/IEC 19785-3.

— *Subtype* — subtype of biometric data, as defined in ISO/IEC 19785-3.

8.3.4.3 YAML specification

To be placed in section #/components/schemas:

```
BiometricType:
  type: object
  properties:
    biotype:
      type: integer
      format: int32
    subtype:
      type: integer
      format: int32
```

8.3.5 CBEFF BIR Type

8.3.5.1 Description

Biometric information shall be packaged as a biometric information record (BIR) in a CBEFF structure, called a CBEFF-BIR in this document, with the biometric data embedded in the biometric data block, as defined by ISO/IEC 19785-1. This document requires the use of either the "Self-identifying, Tag-oriented Simple BIR" or the "Self-identifying, Tag-oriented Complex BIR" patron formats, defined in ISO/IEC 19785-3.

Annex B represents these two patron formats in YAML for the direct integration into this OpenAPI specification.

The CBEFF BIR Type schema shall be used to represent biometric information. This schema provides a reference to CBEFF-BIR representation in YAML.

8.3.5.2 Definitions

bir-info — contains information about the CBEFF-BIR.

bdb-info — contains information about the BDB in a simple CBEFF-BIR.

sb-info — contains information about the security block, if used, in a simple CBEFF-BIR.

format-owner — identifies the Patron format owner.

format-type — identifies the Patron format type.

8.3.5.3 YAML specification

Refer to Annex B.

8.3.6 CBEFF BIR List Type

8.3.6.1 Description

The CBEFF BIR List Type shall provide a list of CBEFF-BIR elements.

NOTE As an alternative to a BIR List, a complex BIR can be used which contains multiple BIRs/modalities.

8.3.6.2 Definitions

— *BIR* — CBEFF structure containing biometric data and associated metadata.

8.3.6.3 YAML specification

To be placed in section #/components/schemas:

```
CBEFF_BIR_ListType:
  type: array
  items:
      $ref: '#/components/schemas/CBEFF_BIR_Type'
```

8.4 Candidate Lists

8.4.1 General

Candidate lists are returned in the response to a biometric identification request. IAVS defines two data types to represent candidate lists. The Candidate Type shall represent a single candidate, and the Candidate List Type shall represent a set or list of candidates.

8.4.2 Candidate List Type

8.4.2.1 Description

The Candidate List Type defines a set of candidates, utilizing the Candidate Type to represent each element in the set.

8.4.2.2 Definitions

— *Candidate* — a single candidate.

8.4.2.3 YAML specification

To be placed in section #/components/schemas:

```
CandidateListType:
  type: array
  items:
    $ref: '#/components/schemas/CandidateType'
```

8.4.3 Candidate Type

8.4.3.1 Description

The Candidate Type defines a single candidate as a possible match in response to a biometric identification request. The *Subject ID* is a required element, while the *score* and *biographic data* are optional.

8.4.3.2 Definitions

— *Rank* — the rank of the candidate in relation to other candidates for the same biometric identification operation.

— *Subject ID* — the identifier of the subject.

— *ScoreList (optional)* — a list of comparison score(s) and optionally the type and subtype of the relating biometric.

— *Biographic Data (optional)* — biographic data associated with the matching candidate.

8.4.3.3 YAML specification

To be placed in section #/components/schemas:

```
CandidateType:
  type: object
  required:
    - rank
    - subjectID
  properties:
    rank:
      type: integer
    subjectID:
      type: string
    scoreList:
      type: array
      items:
        type: object
        properties:
          scores:
            type: array
            items:
              type: number
              format: float
          biometricType:
            $ref: '#/components/schemas/BiometricType'
        required:
          - scores
    biographicData:
      $ref: '#/components/schemas/BiographicDataType'
```

8.5 Document Data

8.5.1 General

IAVS includes metadata about one or more identity documents as well as (optionally) images of scanned identity documents.

8.5.2 Document Data List Type

8.5.2.1 Description

The Document Data List Type shall provide a list of documents.

8.5.2.2 Definitions

— *Document* — data structure containing information about a document and optionally an image of that document.

8.5.2.3 YAML specification

To be placed in section #/components/schemas:

```
DocumentDataListType:
  type: array
  items:
      $ref: '#/components/schemas/DocumentDataType'
```

8.5.3 Document Data Type

8.5.3.1 Description

The Document Data Type defines a set of document data elements providing information about the presented identity document. This type also includes an optional field for an image of the document.

8.5.3.2 Definitions

— *Document Category* — the type of identity document presented (e.g. passport).

— *Document ID Number (optional)* — the number associated with the identity document (e.g. passport number).

— *Document Issuance Country Code (optional)* — the ISO 3166-1 alpha-2 code for the country which issued the document or from within which it was issued.

— *Document Issuing Organization (optional)* — the entity which issued the identity document.

— *Document Issuance Date (optional)* — the date upon which the identity document was issued.

— *Document Expiration Date (optional)* — the date upon which the identity document is no longer valid (expires).

— *Document Last Name (optional)* — the family name of the person to whom the identity document was issued, as contained within the document itself.

— *Document First Name (optional)* — the first given name of the person to whom the identity document was issued, as contained within the document itself.

— *Document Middle Name (optional)* — the second given name of the person to whom the identity document was issued, as contained within the document itself.

— *Document Validity (optional)* — the assessed validity of the identity document (e.g. as the result of local or online validity checks).

— *Document Validity Text (optional)* — details or remarks associated with the assessed validity (e.g. description of validity issue).

— *Document Image (optional)* — a scanned image of the subject document (e.g. passport picture page).

8.5.3.3 YAML specification

To be placed in section #/components/schemas:

```
DocumentDataType:
  type: object
  properties:
    category:
      type: string
    idNumber:
      type: string
    issuanceCountryCode:
      type: string
```

```
    issuingOrganization:
      type: string
    issuanceDate:
      type: string
    expirationDate:
      type: string
    lastName:
      type: string
    firstName:
      type: string
    middleName:
      type: string
    validity:
      type: string
    validityText:
      type: boolean
    image:
      type: string
      format: byte
  required:
    - category
```

8.6 Capabilities

Implementing systems will have various capabilities to support IAVS services. These capabilities include information on supported biometric comparison capabilities, supported galleries, supported processing options for the aggregate services, supported biographic formats, etc. IAVS defines two data types to represent these capabilities. The Capability Type shall represent a single capability, and the Capability List Type shall represent a set of capabilities.

8.6.1 Capability List Type

8.6.1.1 Description

The Capability List Type defines a set of capabilities, utilizing the Capability Type to represent each element in the set.

8.6.1.2 Definitions

— *Capability* — a single capability.

8.6.1.3 YAML specification

To be placed in section #/components/schemas:

```
CapabilityListType:
  type: array
  items:
    $ref: '#/components/schemas/CapabilityType'
```

8.6.2 Capability Type

8.6.2.1 Description

The Capability Type defines a single capability supported by an implementing system. Each supported capability shall be identified by a *Capability Name*. Some supported capabilities will have an associated *Capability Value* (e.g. supported galleries will have a Gallery ID value), while others will not (e.g. biometric comparison support for a specific biometric type).

8.6.2.2 Definitions

— *Capability Name* — the name of the capability, as defined by the implementing system.

— *Capability ID (optional)* — an identifier assigned to the capability by the implementing system.

— *Capability Description (optional)* — a description of the capability.

— *Capability Value (optional)* — a value assigned to the capability.

— *Capability Supporting Value (optional)* — a secondary value supporting the capability.

— *Capability Additional Info (optional)* — contains additional information for the supported capability.

For each capability described in Table 41, the Capability Name shall be set to that shown in Column 1.

Table 41 — List of Capability Items

Capability Name	Capability Description
AggregateInputDataOptional	A capability information item shall be provided for each data element accepted as optional input by the implementing system for the aggregate services. The Capability Value shall be equal to the name of the data element accepted by the aggregate services. The Capability Supporting Value shall indicate which aggregate services support the data element, using one or more of the following values, each separated by a comma: — "Delete" — “Enrol” — “Identify” — “Verify” — “All”
AggregateInputDataRequired	A capability information item shall be provided for each data element required as input by the implementing system for the aggregate services. The Capability Value shall be equal to the name of the data element required by the aggregate services. The Capability Supporting Value shall indicate which aggregate services support the data element, using one or more of the following values, each separated by a comma: — "Delete" — “Enrol” — “Identify” — “Verify” — “All”
AggregateProcessingOption	A capability information item shall be provided for each processing option supported by the implementing system for the aggregate services. The Capability Value shall be equal to the option identifier, or "key" field, for the Processing Option parameter in the aggregate services. The Capability Supporting Value shall be equal to the option value, or "value" field, for the Processing Option parameter in the aggregate services, if applicable. The Capability Additional Info shall indicate which aggregate services support the processing option, using one or more of the following values, each separated by a comma: — "Delete" — “Enrol” — “Identify” — “Verify” — “Retrieve” — “All”

Table 41 *(continued)*

Capability Name	Capability Description
AggregateReturnData	A capability information item shall be provided for each data element returned by the implementing system for the aggregate services. The Capability Value shall be equal to the name of the data element returned by the aggregate services. The Capability Supporting Value shall indicate which aggregate services support the data element, using one or more of the following values, each separated by a comma: — "Delete" — "Enrol" — "Identify" — "Verify" — "Retrieve" — "All"
AggregateServiceDescription	A capability information item shall be provided for each aggregate service supported by the implementing system, describing the processing logic of the aggregate system. The Capability Value shall contain a description of the processing logic of the aggregate services. The Capability Supporting Value shall indicate which aggregate service is described by this capability information item, using one of the following values: — "Delete" — "Enrol" — "Identify" — "Verify" — "Retrieve"
BiographicDataSet	A capability information item shall be provided to identify the biographic data sets supported by the implementing system. The Capability Value shall contain the name of the supported biographic data format (e.g. "EFTS" or "NIEM"). The Capability Supporting Value shall contain the version of the supported biographic data format. The Capability Additional Info shall contain the supported biographic data format type (e.g. ASCII or JSON).
CBEFFPatronFormat	A capability information item shall be provided for each patron format supported by the implementing system. The Capability Value shall contain the format owner. The Capability Supporting Value shall contain the format type.
ClassificationAlgorithmType	A capability information item shall be provided for each classification algorithm type supported by the implementing system. The Capability Value shall be set to the name of the supported classification algorithm type.

Table 41 *(continued)*

Capability Name	Capability Description
ConformanceClass	A capability information item shall be provided to identify the conformance class of the IAVS implementation (see ISO/IEC 30108-1:2015, Annex A for more information). The Capability Value shall be set to one of the following: — "1" (for Class 1 conformance) — "2" (for Class 2 conformance) — "3" (for Class 3 conformance) — "4" (for Class 4 conformance) — "5" (for Class 5 conformance) — "6" (for Class 6 conformance) — "7" (for Class 7 conformance)
Gallery	A capability information item shall be provided for each gallery or population group supported by the implementing system. The Capability Value shall be equal to the value for the Gallery ID parameter in the *Add Subject to Gallery*, *Delete Biographic Data, Delete Biometric Data, Delete Subject From Gallery*, *Identify Subject*, *Retrieve Biographic Data, Retrieve Biometric Data, Retrieve Document Data*, *Set Biographic Data, Set Biometric Data, Set Document Data* and *Verify Subject* services.
IdentityModel	A capability information item shall be provided to identify whether the implementing system is person-centric or encounter-centric based. The Capability Value shall be set to one of the following: — "person" — "encounter"
ComparisonAlgorithm	A capability information item shall be provided for each comparison algorithm vendor and algorithm vendor product ID supported by the implementing system. The Capability Value shall contain the algorithm vendor. The Capability Supporting Value shall contain the algorithm vendor product ID. The Capability Additional Info shall be set to the biometric type, as defined in one of the JSON-coded patron formats ISO/IEC 19785-3, that corresponds to the comparison algorithm. The Capability Description shall contain the software version of the comparison algorithm.
ComparisonScore	A capability information item shall be provided to identify the use of comparison scores returned by the implementing system. The Capability Value shall be set to the end-of-score-range that signifies a match. The Capability Supporting Value shall be set to the end-of-score-range that signifies a no-match. The Capability Additional Info shall be set to the biometric type, as defined in one of the JSON-coded patron formats ISO/IEC 19785-3, that corresponds to the comparison score range.

Table 41 *(continued)*

Capability Name	Capability Description
QualityAlgorithm	A capability information item shall be provided for each quality algorithm vendor and algorithm vendor product ID supported by the implementing system. The Capability Value shall contain the algorithm vendor. The Capability Supporting Value shall contain the algorithm vendor product ID. The Capability Additional Info shall be set to the biometric type, as defined in one of the JSON-coded patron formats ISO/IEC 19785-3, that corresponds to the quality algorithm. The Capability Description shall contain the software version of the quality algorithm.
SupportedBiometric	A capability information item shall be provided for each biometric type supported by the implementing system. The Capability Value shall be set to the biometric type, as defined in one of the JSON-coded patron formats in ISO/IEC 19785-3 (for example, the biometric type for finger is represented as "finger"). The Capability Supporting Value shall indicate if the implementing system supports comparison for the biometric type, using one of the following values: — "1" (identification) — "2" (verification) — "3" (identification and verification) — "4" (no comparison supported)
TransformOperation	A capability information item shall be provided for each transform operation type and transform control combination supported by the implementing system. The Capability Value shall be equal to the value for the Transform Operation parameter in the *Transform Biometric Data* service. The Capability Supporting Value shall specify the value of the Transform Control parameter in the *Transform Biometric Data* service. The value returned may be either a single value or a range of values. If a range of values is returned, the Capability Description shall specify additional information for the value of the Transform Control parameter. If the Transform Operation does not support a Transform Control, the Capability Supporting value shall be set to "NotApplicable".

8.6.2.3 YAML specification

To be placed in section #/components/schemas:

```
CapabilityAggregateInputDataOptional:
  type: object
  properties:
    name:
      type: string
      default: AggregateInputDataOptional
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: array
      items:
        enum:
```

```
            - 'Delete'
            - 'Enrol'
            - 'Identify'
            - 'Verify'
            - 'All'
      additionalInfo:
        type: string
    required:
      - name

  CapabilityAggregateInputDataRequired:
    type: object
    properties:
      name:
        type: string
        default: AggregateInputDataRequired
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        type: array
        items:
          enum:
            - 'Delete'
            - 'Enrol'
            - 'Identify'
            - 'Verify'
            - 'All'
      additionalInfo:
        type: string
    required:
      - name

  CapabilityAggregateProcessingOption:
    type: object
    properties:
      name:
        type: string
        default: AggregateProcessingOption
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        type: string
      additionalInfo:
        type: array
        items:
          enum:
            - 'Delete'
            - 'Enrol'
            - 'Identify'
            - 'Verify'
            - 'Retrieve'
            - 'All'
    required:
      - name

  CapabilityAggregateReturnData:
    type: object
    properties:
      name:
        type: string
        default: AggregateReturnData
      id:
        type: string
```

```
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: array
      items:
        enum:
          - 'Delete'
          - 'Enrol'
          - 'Identify'
          - 'Verify'
          - 'Retrieve'
          - 'All'
    additionalInfo:
      type: string
  required:
    - name

CapabilityAggregateServiceDescription:
  type: object
  properties:
    name:
      type: string
      default: AggregateServiceDescription
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: array
      items:
        enum:
          - 'Delete'
          - 'Enrol'
          - 'Identify'
          - 'Verify'
          - 'Retrieve'
    additionalInfo:
      type: string
  required:
    - name

CapabilityBiographicDataSet:
  type: object
  properties:
    name:
      type: string
      default: BiographicDataSet
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: string
    additionalInfo:
      type: string
  required:
    - name

CapabilityCBEFFPatronFormat:
  type: object
  properties:
    name:
      type: string
      default: CBEFFPatronFormat
    id:
```

```
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: string
    additionalInfo:
      type: string
  required:
    - name

CapabilityClassificationAlgorithmType:
  type: object
  properties:
    name:
      type: string
      default: ClassificationAlgorithmType
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: string
    additionalInfo:
      type: string
  required:
    - name

CapabilityConformanceClass:
  type: object
  properties:
    name:
      type: string
      default: ConformanceClass
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      enum:
        - '1'
        - '2'
        - '3'
        - '4'
        - '5'
    supportingValue:
      type: string
    additionalInfo:
      type: string
  required:
    - name

CapabilityGallery:
  type: object
  properties:
    name:
      type: string
      default: Gallery
    id:
      type: string
    capabilityDesc:
      type: string
    value:
      type: string
    supportingValue:
      type: string
    additionalInfo:
      type: string
```

```
    required:
      - name

  CapabilityIdentityModel:
    type: object
    properties:
      name:
        type: string
        default: IdentityModel
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        enum:
          - 'person'
          - 'encounter'
      supportingValue:
        type: string
      additionalInfo:
        type: string
    required:
      - name

  CapabilityComparisonAlgorithm:
    type: object
    properties:
      name:
        type: string
        default: ComparisonAlgorithm
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        type: string
      additionalInfo:
        type: string
    required:
      - name

  CapabilityComparisonScore:
    type: object
    properties:
      name:
        type: string
        default: ComparisonScore
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        type: string
      additionalInfo:
        type: string
    required:
      - name

  CapabilityQualityAlgorithm:
    type: object
    properties:
      name:
        type: string
        default: QualityAlgorithm
      id:
        type: string
      capabilityDesc:
```

```
        type: string
      value:
        type: string
      supportingValue:
        type: string
      additionalInfo:
        type: string
    required:
      - name

  CapabilitySupportedBiometric:
    type: object
    properties:
      name:
        type: string
        default: SupportedBiometric
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        enum:
          - '1'
          - '2'
          - '3'
          - '4'
      additionalInfo:
        type: string
    required:
      - name

  CapabilityTransformOperation:
    type: object
    properties:
      name:
        type: string
        default: TransformOperation
      id:
        type: string
      capabilityDesc:
        type: string
      value:
        type: string
      supportingValue:
        type: string
      additionalInfo:
        type: string
    required:
      - name

  CapabilityType:
    oneOf:
      - $ref: '#/components/schemas/CapabilityAggregateInputDataOptional'
      - $ref: '#/components/schemas/CapabilityAggregateInputDataRequired'
      - $ref: '#/components/schemas/CapabilityAggregateProcessingOption'
      - $ref: '#/components/schemas/CapabilityAggregateReturnData'
      - $ref: '#/components/schemas/CapabilityAggregateServiceDescription'
      - $ref: '#/components/schemas/CapabilityBiographicDataSet'
      - $ref: '#/components/schemas/CapabilityCBEFFPatronFormat'
      - $ref: '#/components/schemas/CapabilityClassificationAlgorithmType'
      - $ref: '#/components/schemas/CapabilityConformanceClass'
      - $ref: '#/components/schemas/CapabilityGallery'
      - $ref: '#/components/schemas/CapabilityIdentityModel'
      - $ref: '#/components/schemas/CapabilityComparisonAlgorithm'
      - $ref: '#/components/schemas/CapabilityComparisonScore'
      - $ref: '#/components/schemas/CapabilityQualityAlgorithm'
      - $ref: '#/components/schemas/CapabilitySupportedBiometric'
      - $ref: '#/components/schemas/CapabilityTransformOperation'
```

8.7 Fusion Information

8.7.1 Introduction

Fusion information is sent as input to fusion services. IAVS defines two data types to represent fusion information. The Fusion Information Type shall represent a single set of fusion information, and the Fusion Information List Type shall represent a set or list of fusion information elements.

8.7.2 Fusion Identity List Type

8.7.2.1 Description

The Fusion Identity List Type shall contain fusion input elements for one or more identities, utilizing the Fusion Information List Type to represent a single set of fusion information for each identity.

8.7.2.2 Definitions

— *Fusion Identity* — a set of fusion information for a single identity.

8.7.2.3 YAML specification

To be placed in section #/components/schemas:

```
FusionIdentityListType:
  type: array
  items:
    $ref: '#/components/schemas/FusionInformationListType'
  minItems: 1
```

8.7.3 Fusion Information List Type

8.7.3.1 Description

The Fusion Information List Type shall contain at a minimum two sets of fusion input elements, utilizing the Fusion Information Type to represent a single set of fusion information.

8.7.3.2 Definitions

— *Fusion Element* — a set of fusion information.

8.7.3.3 YAML specification

To be placed in section #/components/schemas:

```
FusionInformationListType:
  type: array
  items:
    $ref: '#/components/schemas/FusionInformationType'
  minItems: 2
```

8.7.4 Fusion Information Type

8.7.4.1 Description

The Fusion Information Type represents the information necessary to perform a fusion operation. It shall include the biometric type and subtype, if applicable, for which the biometric comparison was performed, the comparison algorithm identifying information, and either a score (for score-level fusion) or a decision (for decision-level fusion).