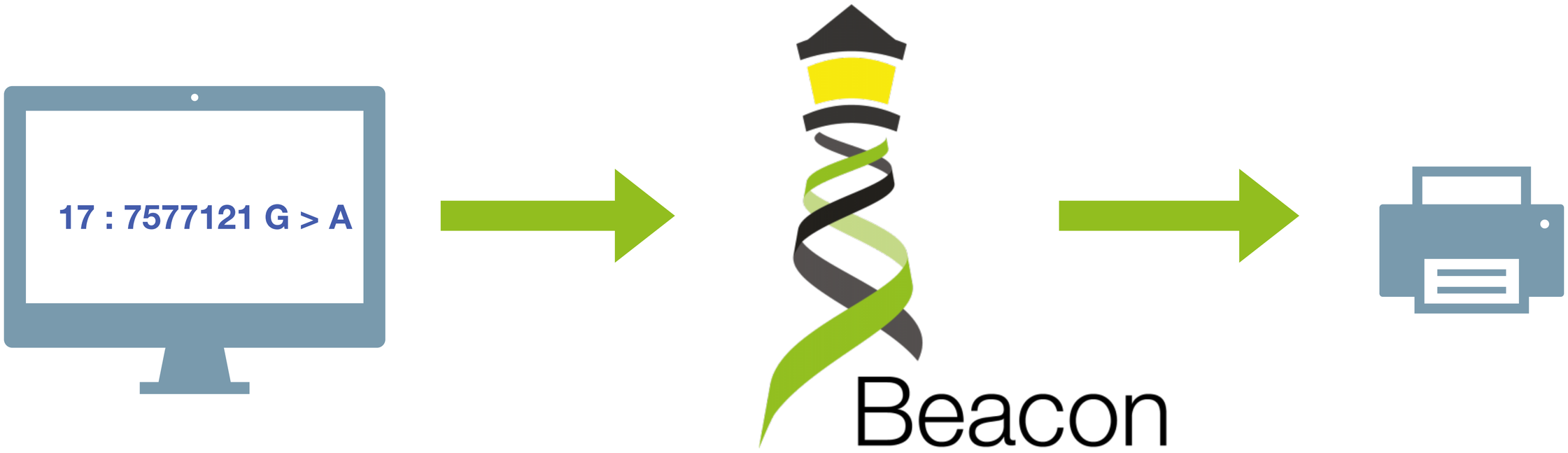
Beacon v2 && Phenopackets

Current relations & Future Alignment

Michael Baudis | 2022-11-22

Global Alliance
for Genomics & Health



A ***Beacon*** answers a query for a specific genome variant against individual or aggregate genome collections

YES | **NO** | **\0**

Beacon v1 Development

Beacon v2 Development

Related ...

2014 — **GA4GH founding event; Jim Ostell proposes Beacon concept including "more features ... version 2"**

2015

- beacon-network.org aggregator created by DNAstack

- ELIXIR starts Beacon project support

2016

- Beacon v0.3 release
- work on queries for structural variants (brackets for fuzzy start and end parameters...)

- Beacon+ concept implemented on progenetix.org
- concepts from GA4GH Metadata (ontologies…)
- entity-scoped query parameters ("individual.age")

2017

- OpenAPI implementation
- integrating CNV parameters (e.g. "startMin, statMax")

- Beacon+ demos "handover" concept

- GA4GH re-structuring (workstreams...)
- Beacon part of Discovery WS

2018

- Beacon v0.4 release in January; feature release for GA4GH approval process
- GA4GH Beacon v1 approved at Oct plenary

- new Beacon website (March)

2019

- ELIXIR Beacon Network

- Beacon hackathon Stockholm; settling on "filters"
- Barcelona goes Zurich developers meeting
- Beacon API v2 Kick off
- adopting "handover" concept

- Beacon publication at Nature Biotechnology

2020

- "Scouts" teams working on different aspects - filters, genomic variants, compliance ...
- discussions w/ clinical stakeholders

2021

- framework + models concept implemented
- range and bracket queries, variant length parameters
- starting of GA4GH review process

- Phenopackets v2 approved

2022

- further changes esp. in default model, aligning with Phenopackets and VRS
- unified beacon-v2 code & docs repository
- Beacon v2 approved at Apr GA4GH Connect

- *docs.genomebeacons.org*

Have you seen deletions in this region on chromosome 9 in Glioblastomas from a juvenile patient, in a dataset with unrestricted access?

Beacon *v2* API

The Beacon API v2 proposal opens the way for the design of a simple but powerful "**genomics API**".

Global Alliance
for Genomics & Health

Future?

Some proposals for a stepwise Beacon protocol extension

- Query language expansion, e.g. Boolean options for chaining filters

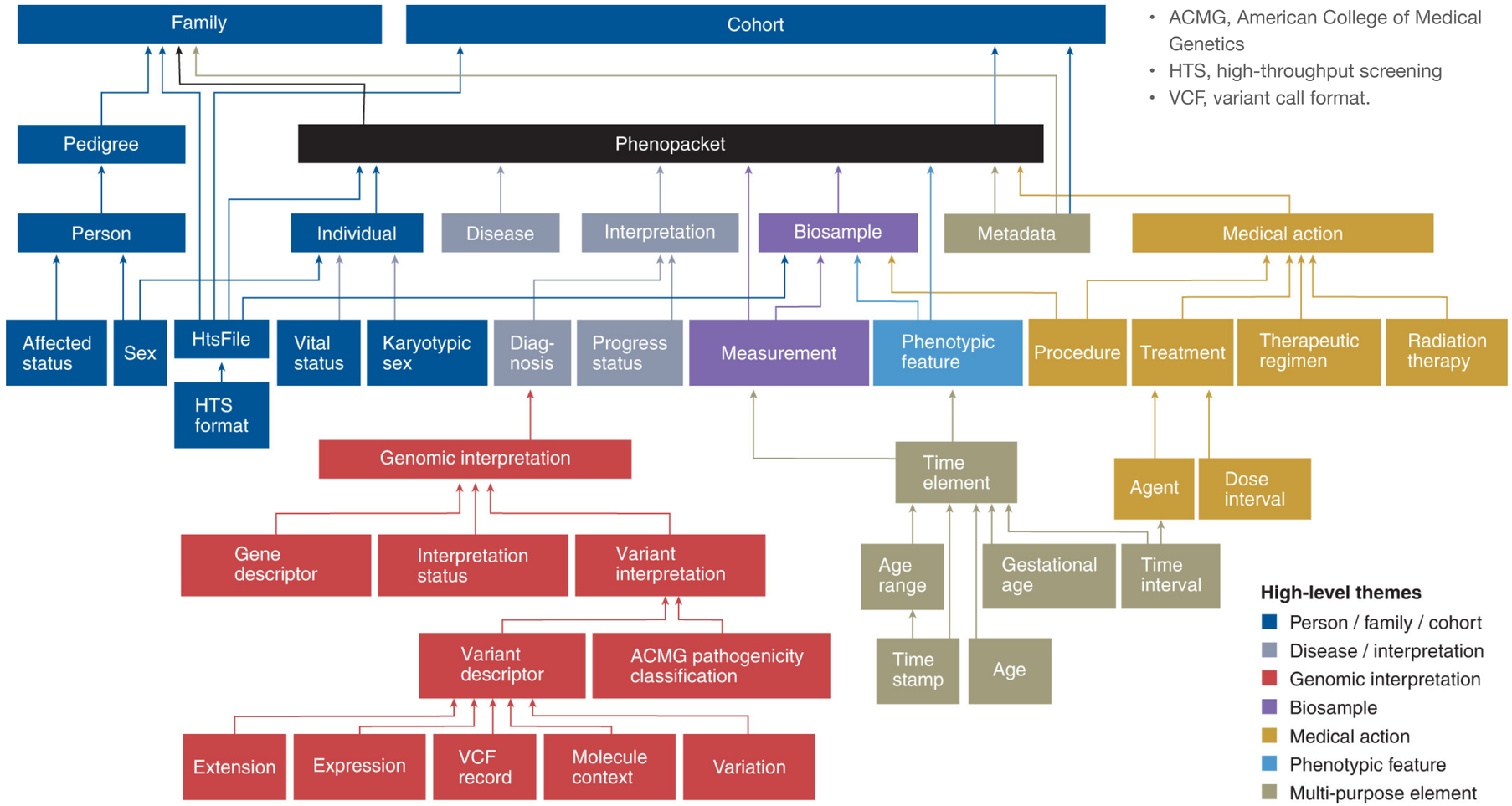
  ➡use of heterogeneous/alternative annotations within and across resources

- **Phenopackets** support as a (the?) default format for biodata export
- **Phenopackets** as **request** documents
- Focus on service & **resource discovery**
- **ELIXIR Beacon Network**, including translations for federated queries to Beacon and Beacon-like resources

The GA4GH Phenopackets v2 Standard
A Computable Representation of Clinical Data

The GA4GH Phenopacket schema consists of several optional elements, each containing information about a certain topic, such as phenotype, variant or pedigree. An element can contain other elements, which allows a hierarchical representation of data.

For instance, Phenopacket contains elements of type *Individual*, *PhenotypicFeature*, *Biosample* and so on. Individual elements can therefore be regarded as **building blocks** of larger structures.

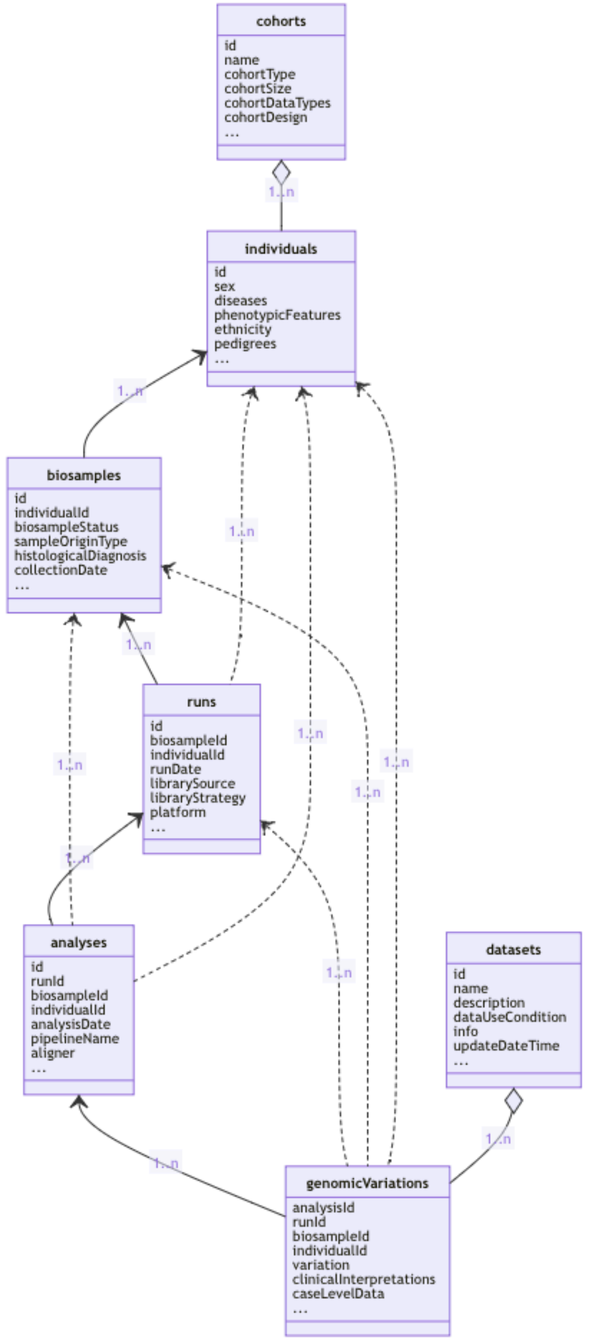


Jacobsen JOB, Baudis M, Baynam GS, Beckmann JS, Beltran S, Buske OJ, Callahan TJ, *et al.* 2022.
The GA4GH Phenopacket Schema Defines a Computable Representation of Clinical Data.
Nature Biotechnology 40 (6): 817–20.

Beacon Default v2 Model

Similarities & Differences …

- The Beacon framework describes the overall structure of the API requests, responses, parameters, the common components, etc.
- Beacon Models describe the set of concepts included in a Beacon, like individual or biosample, and also the relationships between them.
- Besides the overall logical concepts, the Beacon models represent the schemas for data delivery in Beacons supporting “record” granularity
- Beacon explicitly allows the use of other models besides its version specific default.
- The domains defined in the standard model overall correspond well to phenopackets “themes” (since both have been influenced by GA4GH task teams etc.)



Beacon Default v2 Model

Similarities & Differences …

- In the Beacon v2 default data model, many schemas are either directly compatible to Phenopackets v2 building blocks or at least reflect them but with some adjustments.
- While the Beacon v2 default model's schemas do not per se have to reflect PXF schemas, we target an as-close-as-possible alignment to promote/leverage GA4GH-wide standardization.
- The Phenopackets model is centered around the Phenopacket, which is the collector and integrator of all sub-schemas (with the addition of the external Family and Cohort schemas). While Phenopacket usually describes information related to a subject - which is defined in an Individual - and the top level elements in Phenopacket relate to a specific proband (measurements as "Measurements performed in the proband"), the phenopacket itself does not explicitely represent an individual.
- In contrast, the Beacon v2 default model uses a hierarchy in which biosamples reference individuals directly (if existing). For most purposes one can equate Beacon's Individual with a merge of Phenopacket's core Phenopacket and Individual parameters.

http://docs.genomebeacons.org/formats-standards/#phenopackets

Documentation Home | Search | progenetix-web ☆3 ⑂1

Progenetix Documentation

- Documentation Home
  - Progenetix Source Code
    - bycon
    - progenetix-web
    - PGX
    - Additional Projects
- News & Changes
- Pages & Forms
- Services & API
- Use Case Examples
- Classifications, Ontologies & Standards
- Publication Collection
- Data Review
- Beacon+ & bycon
- Technical Notes
- Progenetix Data
- Baudisgroup @ UZH

Progenetix Source Code ¶

With exception of some utility scripts and external dependencies (e.g. MongoDB
the software (from database interaction to website) behind Progenetix and Beac

`bycon`

- Python based service based on the GA4GH Beacon protocol
- software powering the Progenetix resource
- Beacon+ implementation(s) use the same code base

`progenetix-web`

- website for Progenetix and its Beacon+ implementations
- provides Beacon interfaces for the `bycon` server, as well as other Progenetix sevices (e.g. the publ
- implemented as React / Next.js project
- contains this documentation tree here as `mkdocs` project, with files in the `docs` directory

Org.progenetix | Search | beacon ☆2

Base `/biosamples`

`/BIOSAMPLES/` + QUERY

- /biosamples?filters=cellosaurus:CVCL_0004
- this example retrieves all biosamples having an annotation for the Cellosaurus *CVCL_0004* identifier (K562)

es/pgxbs-kftva5c9

a single biosample

TMODE=TRUE

es?testMode=true

some random samples

- for testing API responses

`/BIOSAMPLES/{ID}/G_VARIANTS`

- /biosamples/pgxbs-kftva5c9/g_variants/
- retrieval of all variants from a single biosample

Base `/individuals`

`/INDIVIDUALS` + QUERY ¶

/individuals?filters=NCIT:C7541

Rapidly evolving documentation of both the Beacon API itself and its use and technical implementation on
docs.genomebeacons.org
docs.progenetix.org

Shoutout to Laure(e)n Fromont & Manuel Rueda for being instrumental in the Beacon v2 documentation!

Beacon API

Beacon-style JSON responses

The Progenetix resource's API utilizes the `bycon` framework for data query and delivery and represents a custom implementation of the Beacon *v2* API.

The standard format for JSON responses corresponds to a generic Beacon v2 response, with the `meta` and `response` root elements. Depending on the endpoint, the main data will be a list of objects either inside `response.results` or (mostly) in `response.resultSets.results`. Additionally, most API responses (e.g. for biosamples or variants) provide access to data using *handover* objects.

Beacon v2 Documentation | Search | beacon-v2 ☆2 ⑂8

Org.progenetix

Progenetix & Beacon+

The Beacon+ implementation - developed in the Python & MongoDB based `bycon` project - implements an expanding set of Beacon v2 paths for the Progenetix resource.

Scoped responses from query object

In queries with a complete `beaconRequestBody` the type of the delivered data is independent of the path and determined in the `requestedSchemas`. So far, Beacon+ will compare the first of those to its supported responses and provide the results accordingly; it doesn't matter if the endpoint was `/beacon/biosamples/` or `/beacon/variants/` etc.

Below is an example for the standard test "small deletion CNVs in the CDKN2A locus, in gliomas" Progenetix test query, here responding with the matched variants. Exchanging the `entityType` entry to

- `{ "entityType": "biosample", "schema:": "https://progenetix.org/services/schemas/Biosample/"}`

would change this to a biosample response. The example ccan be tested by POSTing this as `application/json` to `http://progenetix.org/beacon/variants/` or `http://progenetix.org/beacon/biosamples/`.

```
{
    "$schema":"beaconRequestBody.json",
    "meta": {
        "apiVersion": "2.0",
        "requestedSchemas": [
            {
                "entityType": "genomicVariant",
                "schema:": "https://progenetix.org/services/schemas/genomicVariant"
            }
        ]
    },
    "query": {
        "requestParameters": {
```

Beacon+: Phenopackets

Testing alternative response schemas...

http://progenetix.org/beacon/biosamples/pgxbs-kftvhyvb/phenopackets

- the v2 default schemas are mostly aligned w/ Phenopackets v2
- creating phenopackets can be done mostly by re-wrapping of Beacon entities (individual, biosample)
- variants can be included through file resource URLs; in Beacon+ this is done through *ad hoc* handover URIs


```
"id": "pgxpxf-kftx3tl5",
"metaData": {
  "phenopacketSchemaVersion": "v2",
  "resources": [
    {
      "id": "NCIT",
      "iriPrefix": "http://purl.obolibrary.org/obo/NCIT_",
      "name": "NCIt Plus Neoplasm Core",
      "namespacePrefix": "NCIT",
      "url": "http://purl.obolibrary.org/obo/ncit/neoplasm-core.c
      "version": "2022-04-01"
    },


"subject": {
  "dataUseConditions": {
    "id": "DUO:0000004",
    "label": "no restriction"
  },
  "diseases": [
    {
      "clinicalTnmFinding": [],
      "diseaseCode": {
        "id": "NCIT:C3099",
        "label": "Hepatocellular Carcinoma"
      },
      "onset": {
        "age": "P48Y9M26D"
      },
      "stage": {
        "id": "NCIT:C27966",
        "label": "Stage I"
      }
    }
  ],
  "id": "pgxind-kftx3tl5",
  "sex": {
    "id": "PATO:0020001",
    "label": "male genotypic sex"
  },
  "updated": "2018-12-04 14:53:11.674000",
  "vitalStatus": {
    "status": "UNKNOWN_STATUS"
  }
}
```

```
"biosamples": [
  {
    "biosampleStatus": {
      "id": "EFO:0009656",
      "label": "neoplastic sample"
    },
    "dataUseConditions": {
      "id": "DUO:0000004",
      "label": "no restriction"
    },
    "description": "Primary Tumor",
    "externalReferences": [
      {
        "id": "pgx:TCGA-0004d251-3f70-4395-b175-c94c2f5b1b81",
        "label": "TCGA case_id"
      },
      {
        "id": "pgx:TCGA-TCGA-DD-AAVP",
        "label": "TCGA submitter_id"
      },
      {
        "id": "pgx:TCGA-9259e9ee-7279-4b62-8512-509cb705029c",
        "label": "TCGA sample_id"
      },
      {
        "id": "pgx:TCGA-LIHC",
        "label": "TCGA LIHC project"
      }
    ],
    "files": [
      {
        "fileAttributes": {
          "fileFormat": "pgxseg",
          "genomeAssembly": "GRCh38"
        },
        "uri": "https://progenetix.org/beacon/biosamples/pgxbs-kftvhyvb/variants/?output=pgxseg"
      }
    ],
    "histologicalDiagnosis": {
      "id": "NCIT:C3099",
      "label": "Hepatocellular Carcinoma"
    },
    "id": "pgxbs-kftvhyvb",
    "individualId": "pgxind-kftx3tl5",
    "pathologicalStage": {
      "id": "NCIT:C27966",
      "label": "Stage I"
    },
    "sampledTissue": {
      "id": "UBERON:0002107",
      "label": "liver"
    },
    "timeOfCollection": {
      "age": "P48Y9M26D"
    },
```

Beacon+: Phenopackets

Testing alternative response schemas...

http://progenetix.org/beacon/biosamples/pgxbs-kftvhyvb/phenopackets

- the v2 default schemas are mostly aligned w/ Phenopackets v2
- creating phenopackets can be done mostly by re-wrapping of Beacon entities (individual, biosample)
- variants can be included through file resource URLs; in Beacon+ this is done through *ad hoc* handover URIs

Beacon+: Phenopackets

Testing alternative response schemas...

http://progenetix.org/beacon/biosamples/pgxbs-kftvhyvb/phenopackets

- the v2 default schemas are mostly aligned w/ Phenopackets v2
- creating phenopackets can be done mostly by re-wrapping of Beacon entities (individual, biosample)
- variants can be included through file resource URLs; in Beacon+ this is done through *ad hoc* handover URIs

```
bios_s = data_db["biosamples"].find({"individual_id":ind["id"]})

for bios in bios_s:

    bios.update({
        "files": [
            {
                "uri": "{}/beacon/biosamples/{}/variants/?output=pgxseg".format(server, bios["id"]),
                "file_attributes": {
                    "genomeAssembly": "GRCh38",
                    "fileFormat": "pgxseg"
                }
            }
        ]
    })
    for k in bios_pop_keys:
        bios.pop(k, None)

    clean_empty_fields(bios)

    pxf_bios.append(bios)
```

```
def remap_phenopackets(ds_id, r_s_res, byc):

    if not "phenopacket" in byc["response_entity_id"]:
        return r_s_res

    mongo_client = MongoClient()
    data_db = mongo_client[ds_id]
    pxf_s = []

    for ind_i, ind in enumerate(r_s_res):

        pxf = phenopack_individual(ind, data_db, byc)
        pxf_s.append(pxf)

    return pxf_s
```

Beacon & Phenopackets

Where to go?

- Adding a Phenopacket model to Beacon v2.n
  - ➡ going for fast, correct implementation over completeness
- Beacon will (?) include defined handover types; PXF could be an early one
  - ➡ does not require definition in the Beacon models itself
- Driver project using PXF over Beacon v2 for data delivery
  - ➡ existing draft proposal …
- PXF as the default schema for Beacon data delivery in v2.n…