Masaryk University
Faculty of Informatics

Beacon Network: A System for Global Genomic Data Sharing

Diploma Thesis

Miroslav Cupak

Brno, 2016

Declaration

Hereby I declare, that this paper is my original authorial work, which I have worked out by my own. All sources, references and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Miroslav Cupak

Advisor: Ing. Matej Lexa, Ph.D.

Acknowledgement

I would like to thank my advisor Matej Lexa for his supervision. My thanks go to the Global Alliance for Genomics and Health for supporting the project in various ways, and all the teams involved in the Beacon Project, without whom this work would not have been possible. Special thanks go to DNAstack for developing the client for the Beacon Network, and supporting the development as well as hosting of the system. I owe my sincere gratitude especially to Marc Fiume, who brought up the original idea for this work, lead the Beacon Project, and guided the development of the Beacon Network.

Abstract

Beacon is a genetic mutation sharing platform developed by the Global Alliance for Genomics and Health. It defines a web service that answers questions of the form "Do you have information about the following mutation?" and responds with one of "yes" or "no", and potentially more information. This work presents the Beacon Network, a search engine across the world's public beacons, and the driving force behind the specification of the Beacon API. The system enables global discovery of genetic mutations, federated across a large and growing network of shared genetic datasets.

Keywords

Beacon, Beacon Network, Global Alliance for Genomics and Health, data sharing, genomics.

Contents

1 Introduction

"Until the fountain of youth is discovered, all of us will have some conditions in our old age, only we don't know what they will be. I have a better guess than almost anyone else for what ills may be mine – and I have decades to prepare for it."

– Sergey Brin, President, Alphabet. Source: [28].

The cost of genome sequencing has fallen one-million fold in the past few years, leading to an explosion of information about the genetic basis of human health and disease, with the total amount of sequence data produced doubling approximately every seven months. If the growth continues at the current rate, it is predicted that we reach more than one exabase of sequence per year in the next five years and approach one zettabase of sequence per year by 2025. The promise of genomic medicine to revolutionize the diagnosis and treatment of disease suggests that 100 million to 2 billion human genomes could be sequenced in the next ten years, representing four to five orders of magnitude growth [103].

All this information can accelerate progress in biomedicine, but only if we have the technology, standards, and policies to analyze individual genome sequences alongside very large amounts of aggregated sequence and clinical data. Global Alliance for Genomics and Health (GA4GH), a coalition of over 375 institutions, was created to address this issue.

A key obstacle to solving many rare diseases is the difficulty of discovering additional affected families efficiently – there may only be a handful of similar patients worldwide, and their information may be stored in diverse clinical and research databases. With the amount of genomic data growing so rapidly, the traditional approach of data aggregation in a single centralized site does not help enough. A federated system capable of executing cross-dataset and cross-institution queries is needed.

To test the willingness of international sites to share genetic data in the simplest of all technical contexts, GA4GH started the Beacon Project. Beacon is a web service that answers questions of the form "Do you have information about the following mutation?" and responds with "yes" or "no". Several beacons became available quickly.

This work introduces the Beacon Network, a modern search engine across the world's public beacons. The system enables global discovery of genetic mutations, federated across a large and growing network of shared genetic datasets. It standardizes the way beacons are accessed and provides an interface allowing its users to query the services in parallel and from a common place on the web.

Although this work is focused on the Beacon Network as a technical system, certain expertise in genetics and computational biology is needed to understand its application. We provide the necessary background information in Chapter 2, together with the introduction of the Global Alliance and the Beacon Project. Other systems facilitating sharing of genomic data are briefly described in this chapter as well. Chapter 3 describes the design and implementation of the Beacon Network and its supporting tools. The system drove the development of new beacon specifications, which are highlighted in this chapter as well. The Beacon Network has been in production for over a year and collected some interesting data. This data is analyzed in Chapter 4 to evaluate how the system grows and what kind of information it presents. Limitations of the project in its current state as well as suggestions for future improvements are documented in Chapter 5, while the work is briefly concluded in Chapter 6.

2 Beacon

"The Beacon project is really exciting; it is a project to test the willingness of international sites to share genetic data in the simplest of all technical contexts."

– Julia Wilson, Associate Director External Relations, Wellcome Trust Sanger Institute. Source: [65].

2.1 Background in genomics

Even though this work is primarily written from software engineering perspective, a certain level of expertise in genetics is necessary to understand the space and functionality provided by the project. For this reason, we provide definitions of necessary terms in this section. Terminology not explicitly provided here is used according to the definitions in [66].

2.2 Variant structure

Since the traditional manipulation of gene sequence variations involved storing them in the form of text files, their structure is probably best explained using the standard formats they are typically stored in.

There are a number of file formats used to store genetic data. The traditional format for describing genes is General Feature Format (GFF). Its current version (GFF3) defines nine-column, tab-delimited, plain text files, whose columns represent ID of the landmark, source, type, start, end, score, strand, phase and a list of feature attributes, respectively [101]. This format is widely used and became a basis for many other more specific formats addressing its drawbacks, such as Genome Variation Format (GVF).

A major disadvantage of GFF is the fact that it stores all the genetic data, thus providing a lot of redundancy (data shared across the genomes). This lead to development of formats storing only differences with respect to a reference genome, such as Variant Call Format (VCF), which became a de-facto standard.

VCF is a specification of a generic format for storing DNA polymorphism data such as SNVs, insertions, deletions and structural variants, together with rich annotations. VCF is usually stored in a compressed manner and can be indexed for fast retrieval of variants from a range of positions in the reference genome. It consists of a header section and a data section. The header contains an arbitrary number of metainformation lines, each starting with characters ##, and a tab-delimited field definition line, starting with a single # character. The metainformation header lines provide a standardized description of tags and annotations used in the data section. They allow the information stored within a VCF file to be tailored to the data set in question. Metainformation can also be used to provide details about the means of file creation, date of creation, version of the reference sequence, software used and any other data relevant to the history of the file. The field definition line names eight mandatory columns, which correspond to data columns representing the chromosome (CHROM), a 1-based position of the

Headword	**Definition**
allele	An alternative form of a **gene** (one member of a pair) that is located at a specific position on a specific chromosome.
amino acids	A set of 20 different molecules used to build **proteins**.
chromosome	A structure composed of a very long molecule of DNA and associated **proteins** that carries hereditary information.
chromosome mutation	A type of **mutation** involving alterations in chromosome structure.
exome	The part of the **genome** formed by **exons**.
exon	The **protein**-coding DNA sequence of a **gene**.
gene	A unit of heredity associated with a location on a **chromosome**, which determines a particular trait of an organism.
genetic variation	A change of **genes** of an organisms within a population.
genetics	The branch of biology concerned with heredity and variation in organisms. Also the genetic features and constitution of any single organism, species, or group of organisms.
genome	The whole of the genetic information of an organism.
genomic context	**Genes** located in the same region of DNA with respect to a particular **gene**, essentially a neighborhood of a **gene**.
genomics	The systematic sequencing and characterization of complete **genomes**.
genotype	The total genetic constitution of an organism.
intron	A portion of a **gene** that does not code for **amino acids**.
mutation	The process by which genetic material undergoes a detectable and heritable structural change, or the result of such a change.
nucleobase	Also **base**. A constitutional unit of a **nucleotide** and a part of nucleic acids involved in pairing. The primary **nucleobases** are cytosine, guanine, adenine (DNA and RNA), thymine (DNA) and uracil (RNA), abbreviated as C, G, A, T, U.
nucleotide	A constitutional unit from which nucleic acids are considered to be built up.
phenotype	The totality of the observable characteristics of an organism as determined by interaction of the **genotype** of the organism with the environment in which it exists.
polymorphism	Natural variations in a **gene**.
protein	Any of a large group of organic compounds found as major macromolecular constituents of living organisms. Proteins are assembled from **amino acids** using information from **genes** through processes of **transcription** and **translation**.
sequence	The ordinal arrangement of constituent parts of a **protein** or nucleic acid.
sequencing	The act or process of determining the **sequence** of proteins or nucleic acids.

SNP	Single-nucleotide polymorphism. A type of **polymorphism** involving variation of a single base pair.
SNV	Single-nucleotide variant. The occurrence of single-base variations in the genetic code that occur about once every 1000 bases along the human **genome**.
variant	Something that deviates from the norm. Also **genomic variant**, a manifestation of **genetic variation**.

Table 2.1: Basic terminology. Definitions are based on [23], extended with data from [98] and modified according to [66] to fit the usage in this document (general meanings of the terms are narrowed down to more specific and suitable notions in some cases).

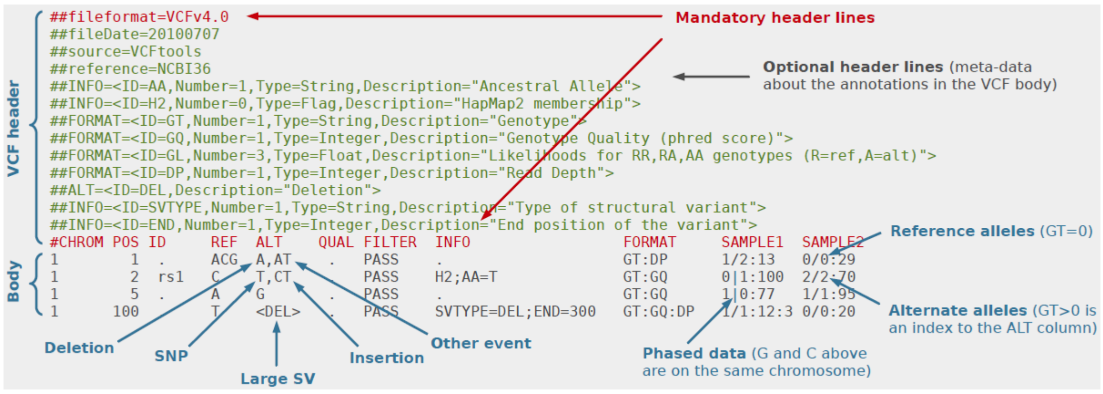

```
##fileformat=VCFv4.0
##fileDate=20100707
##source=VCFtools
##reference=NCBI36
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality (phred score)">
##FORMAT=<ID=GL,Number=3,Type=Float,Description="Likelihoods for RR,RA,AA genotypes (R=ref,A=alt)">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##ALT=<ID=DEL,Description="Deletion">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant">
#CHROM POS ID     REF  ALT    QUAL FILTER  INFO                 FORMAT      SAMPLE1    SAMPLE2
1        1  .     ACG  A,AT    .   PASS    .                    GT:DP       1/2:13     0/0:29
1        2  rs1   C    T,CT    .   PASS    H2;AA=T              GT:GQ       0|1:100    2/2:70
1        5  .     A    G       .   PASS    .                    GT:GQ       1|0:77     1/1:95
1      100        T    <DEL>   .   PASS    SVTYPE=DEL;END=300   GT:GQ:DP    1/1:12:3   0/0:20
```
VCF header
Body
Mandatory header lines
Optional header lines (meta-data about the annotations in the VCF body)
Reference alleles (GT=0)
Alternate alleles (GT>0 is an index to the ALT column)
Deletion
SNP
Large SV
Insertion
Other event
Phased data (G and C above are on the same chromosome)

Figure 2.1: Explanation of VCF file structure. Source: [27].

start of the variant (POS), a unique identifier of the variant (ID), a reference allele (REF), a comma separated list of alternate non-reference alleles (ALT), a phred-scaled quality score (QUAL), site filtering information (FILTER) and a semicolon-separated list of additional extensible annotations (INFO). In addition, if samples are present in the file, the mandatory header columns are followed by a FORMAT column and an arbitrary number of sample IDs defining the samples included in the VCF file. The FORMAT column is used to define the information contained in each subsequent genotype column, which consists of a colon-separated list of fields [26]. This structure is shown in Figure 2.1 [23].

From now on, when referring to the notion of variants, these are the properties we expect them to have.

2.3 Global Alliance for Genomics and Health

GA4GH is a coalition of over 375 leading institutions working in health care, research, disease advocacy, life science, and information technology. Its goal is to accelerate progress in human health by helping to establish a common framework of harmonized approaches to enable effective and responsible sharing of genomic and clinical data, and by catalyzing data sharing projects that drive and demonstrate the value of data sharing [43].

The conception of GA4GH traces to a meeting of 50 colleagues in January 2013. The cost of genome sequencing had already fallen one-million fold in the few previous years, which created opportunities to gain insight into diseases, improve prevention and early detection, define diagnostic categories, streamline clinical trials, and match patients to therapy. The industry, however, was not utilizing the wealth of genome sequence data to its full potential. Data was collected and studied in isolation, split by diseases, institutions or countries. Regulatory procedures did not anticipate developments in technology and the need for data sharing and aggregation. Tools were not standardized or compatible. The group concluded that in order to meet the need of patient, research and clinical communities, it would be necessary to create a global alliance of international partners, who would advance the idea that patients have a right to share genomic and clinical information, collaborate with governments and funders to create policies and regulations enabling this process, support the development of open technology standards and their reference implementations, provide a forum to evaluate and share best practices and create collaborative projects to accelerate the positive impact of genome sequence information on medicine, support a level playing field that enables technology and business innovation, and enable social networks to bring together patients and families with the genomic and medical communities [8].

From technical perspective, the Global Alliance aims to build foundation for scalable upload and storage of data from sequencing platforms together with clinical data, rapid processing with state-of-the-art generic and custom tools, management of security, privacy and user access, as well as downloading and controlled sharing of data and results. The platforms have standards and application programming interfaces (APIs) to securely interact with the data and results, enabling a wide variety of operating entities to serve users, and for developers to write applications customized to specific uses [8].

2.3.1 Current initiatives

As of December 2015, GA4GH consists of four main working groups, whose responsibilities are broken down into task and project teams. Task teams are small groups with clear, time-bound, and actionable deliverables, while project teams initiate and lead multi-stage, cross-cutting projects, or liaise with external projects in a supporting or a co-ownership role.

Clinical working group is focused on sharing clinical data and linking genomic information. Its work is distributed across four main teams. Catalog of Activities team provides a catalog of current activities related to data sharing to be used as a resource for researchers and clinicians. Phenotype Ontologies team brings together existing international efforts to develop and promote standard language and tools for recording patient clinical phenotypes and exploiting phenotype data for diagnostics and translational research. Clinical Cancer Genome team tries to harmonize the clinical sequencing efforts in the global cancer community. eHealth[1] team summarizes efforts in eHealth that attempt to collect and link genomic and phenotypic data [45].

Security working group works on developing and adopting standards for data se-

1. Della Mea, Vincenzo (2001). "What is e-Health (2): The death of telemedicine?". Journal of Medical Internet Research 3 (2): e22.

curity, privacy protection, and access control. Its current initiatives include Software Security, a team making recommendations for implementing the various elements of the security infrastructure, and Cloud Security, a team providing guidance around cloud security, including protections expected of cloud service providers [45].

Regulatory and Ethics working group concentrates on ethics, legal and social impact of the organization. Its work is split amongst 10 teams. Accountability Policy team works on defining governance and accountability of GA4GH members. Ageing and Dementia team works on tools and policies to facilitate international data sharing and promote health innovations related to dementia. Data Protection Regulation analyzes and responds to data protection regulation development around the world. Data Sharing Lexicon assembles a compilation of common or concordant terms related to data sharing. Ethics Review Equivalency develops models that allow for mutual recognition of ethics review. Individual Access creates an international survey to explore the attitudes of stakeholders relating to the communication of genomic data. Machine Readable Consent standardizes computer readable data use-types via consent forms. Pediatric team explores regulatory and ethical issues related to genomic newborn screening. Privacy Breach Notification team carries out a review of laws requiring researchers to notify authorities or participants of privacy breaches. Registered Access team defines a registered access level, how it is distinct from open access or controlled access, and when one can or should move from one access process to another [45].

Data working group is in charge of data representation, storage and analysis. It is probably the most developer-oriented of the working groups. Most of its work and communication is openly available on Git Hub[2], a popular version-control hosting service. The work of the group is broken up across 11 task teams. Reference Variation team works on describing how genomes differ from each other. Genotype To Phenotype team is focused on representation of genotype-phenotype associations. RNAseq and Gene Expression team provides APIs to process RNA sequence reads, computed transcript structures, and their expression levels. The mission of the Variant Annotation team is to develop standards for variant annotation, its formats, ontologies and vocabularies. Metadata team creates metadata schema shared across GA4GH working groups. File Formats team maintains popular genomic data formats – SAM/BAM, CRAM and VCF/BCF[3]. Containers and Workflows team works on defining portable data analysis workflows based on Common Workflow Language (CWL)[4]. Benchmarking team prepares benchmark toolkits for germline, cancer, and transcripts.

Aside from the products of individual groups and task teams, GA4GH develops three cross-cutting demonstration projects showcasing the value of data sharing, the development of which requires involvement of multiple working groups and task teams. This includes Matchmaker Exchange[5], a federated platform (exchange) to facilitate the identification of cases with similar phenotypic and genotypic profiles (matchmaking)

2. GitHub – Global Alliance for Genomics and Health. `https://github.com/ga4gh`. Accessed on December 22, 2015.
3. Specifications of SAM/BAM and related high-throughput sequencing file formats. `https://samtools.github.io/hts-specs`. Accessed on December 22, 2015.
4. CWL specification. `http://common-workflow-language.github.io`. Accessed on December 22, 2015.
5. Matchmaker Exchange. `http://www.matchmakerexchange.org`. Accessed on December 22, 2015.

through a standardized API and procedural conventions [83], BRCA Challenge[6], a project to advance understanding of the genetic basis of breast cancer and other cancers using data on BRCA[7] genetic variants, and the Beacon Project.

2.4 Definition

Beacon is defined as a simple web service allowing users to query institution's databases to determine whether they contain a genetic variant of interest. The service accepts questions of the form: *"Do you have any genomes with an 'A' at position 100,735 on chromosome 3?"*. It responds with either *yes* or *no* [44].

Since the beginning, there have been three main design principles behind beacons:

- A beacon has to be technically simple.

  It should not be a significant technical challenge or a big investment for an organization to implement a beacon, provided they have the data and it is organized in a reasonable way.

- A beacon has to minimize risks associated with genomic data sharing.

  The purpose of beacons is to be the first step in sharing of human genomic data. The service should return information so minimal that no reasonable human being would think that it violates anyone's privacy or exposes anyone to potential harms.

- It has to be possible to make a beacon publicly available.

  Beacons can be made public, and should initially be made public on purpose, so as not to trigger political and technical issues of requiring authentication and authorization.

The Beacon Project addresses the problem of general complexity of data sharing and is really a test of willingness of international sites to share genetic data in the simplest of all technical contexts. As one of the core demonstration projects in GA4GH, it is a collaborative effort of several working groups, primarily the Data, Regulatory and ethics, as well as Security working groups. The Beacon task team is responsible for leading the project and designing the API. The Software Security team is looking into different beacon access levels, authentication and authorization mechanism, as well as privacy protection and possible attacks. The Machine Readable Consent team develops standardized and computer-readable data use types in consent forms. The Beacon Project is one of the early adopters of consent support in GA4GH. The project is also an application of the research of the Registered Access team, whose responsibility is to explore the spectrum of data access (from open to registered to controlled), with the specific goal of defining a registered access level. The Metadata team with their metadata schema, the Genotype To Phenotype team with their language for genotype-phenotype associations, as well as the Variant Annotation team with their representations of information associated with particular regions of the genome, are potentially relevant for future extensions of the API.

6. BRCA Exchange. `http://brcaexchange.org`. Accessed on December 22, 2015.
7. BRCA1 and BRCA2: Cancer Risk and Genetic Testing. `http://www.cancer.gov/about-cancer/causes-prevention/genetics/brca-fact-sheet`. Accessed on December 22, 2015.

2.5 History

The origin of beacons traces back to a GA4GH meeting in March 2014. Dr. Jim Ostell (NCBI[8]) proposed a challenge application for institutions interested in engaging in sharing of human genomic data on an international scale. The challenge was to create a public web application aligned with the definition presented in Section 2.4, which would deliver information about real data.

The term beacon is a playful reference to the SETI[9] project, scientific effort primarily based on scanning the sky for beacons, sources of non-random patterns of electromagnetic emissions, such as radio or television waves, in order to detect another possible civilization somewhere else in the universe. Traditionally, extraterrestrial electromagnetic narrow-band (sine-wave) beacons were an excellent choice to get alien attention across interstellar distances. Continuous narrow-band beacons carry only one bit of information, the existence of an alien civilization [52]. Similarly, genomic beacons respond with a single bit of information about a human population (yes or no). Not unlike in the SETI project, many dedicated people have been scanning the universe of human research for signs of willing participants in far-reaching data sharing efforts, but despite many assurances of interest, it has remained a dark and quiet place.

The Beacon Project gained traction quickly. The first three institutions published their beacons no later than a month after the March 2014 meeting. Since then, 19 other organizations joined the project, exposing 64 public beacons in total.

The project clearly proved that people are willing to share data. The original concept has been extended to support other response metadata, such as the number of observed alleles or allele frequency, and the project is slowly evolving into an even more general query API.

2.6 Specification

When we introduced the Beacon Network, there was no formal specification of the Beacon API. In fact, the absence of such a specification was one of the main reasons the Beacon Network was needed in the first place. At the time, the API was defined merely by the description of the web service as provided in Section 2.4, and was later formalized as 0.1.

2.6.1 Beacon 0.1

Beacon 0.1 was an early attempt to standardize the Beacon API. It was published after the launch of the Beacon Network, in December 2014. At the time, it was a part of the GA4GH Core API[10], an attempt of the Data working group to standardize data models and APIs for genomic data.

8. National Center for Biotechnology Information. `http://www.ncbi.nlm.nih.gov`. Accessed on December 22, 2015.
9. Search for Extra-Terrestrial Intelligence. `http://history.nasa.gov/seti.html`. Accessed on December 22, 2015.
10. Schemas for the Data working group. `https://github.com/ga4gh/schemas`. Accessed on December 22, 2015.

GA4GH data models are specified using Apache Avro[11], an open-source schema-based data serialization system. Avro comes with a higher-level interface definition language, Avro IDL. The language is used to specify protocols and schemas, which can then be compiled to various programming languages. The schemas are thus focused on the internal model of the application rather than its API – records are compiled into classes, with certain API aspects (e.g. endpoint names) specified through documentation strings (comments).

Beacon 0.1 is defined as a simple protocol with 2 records, the specification of which is listed in Appendix A. *BEACONRequest* record specifies parameters of the beacon query – population/dataset identifier, reference genome, chromosome, position, and allele. *BEACONResponse* defines data provided in a reply to the beacon query – allele presence and frequency.

2.7 Implementations

When the Beacon Network was first introduced, there were four public beacons available, each providing their own, custom implementation of the API.

Algorithms, Machines and People Lab (AMPLab, UC Berkeley)[12] was the first organization exposing a beacon. The beacon queries public data from the 2011 release of the 1,000 Genomes Project[13], an effort to provide a deep characterization of human genome sequence variation as a foundation for investigating the relationship between genotype and phenotype [7]. The data is in the form of variants called by resource, stored in a VCF format, as described in Section 2.2. AMPLab chose to implement the query as an HTTP[14] POST request with five parameters – *population* (dataset identifier), *genome* (reference genome identifier, such as "hg19"), *chr* (chromosome identifier, autosome or X, prefixed with "chr"), *coord* (0-based position within a chromosome), and *allele* (allele identifier – one of A,C,G,T for SNVs, a string of the aforementioned letters to represent insertions, or DEL denoting an arbitrary deletion). A response is a fully formated HTML[15] document containing a sentence explaining whether a match was found. An example of such a query in cURL[16] format is listed in Listing 2.1.

NCBI set up a beacon serving data from NHLBI Exome Sequencing Project (GO-ESP)[17] and Phase 1 release of the 1,000 Genomes Project. This includes GO-ESP variants as well as raw sequence data from SRA[18], an international public archival resource for next-generation sequence data [74]. NCBI implements the API through an HTTP GET request with four parameters – *ref* (reference genome identifier, such as GRCh37), *chrom* (chromosome identifier without a prefix, autosome, X, Y, or MT), *pos* (1-based position within a chromosome), and *allele* (allele identifier – one of A,C,G,T

11. Apache Avro. `http://avro.apache.org`. Accessed on December 22, 2015.
12. AMPLab. `https://amplab.cs.berkeley.edu`. Accessed on December 22, 2015.
13. The 1,000 Genomes Project. `http://www.ncbi.nlm.nih.gov/bioproject/28889`. Accessed on December 22, 2015.
14. RFC 2616 – Hypertext Transfer Protocol – HTTP/1.1. `https://tools.ietf.org/html/rfc2616`. Accessed on December 22, 2015.
15. Hypertext Markup Language. `http://www.w3.org/TR/html/`. Accessed on December 22, 2015.
16. cURL. `http://curl.haxx.se`. Accessed on December 22, 2015.
17. NHLBI Grand Opportunity Exome Sequencing Project. `http://www.ncbi.nlm.nih.gov/bioproject/165957`. Accessed on December 22, 2015.
18. Sequence Read Archive. `http://www.ncbi.nlm.nih.gov/sra`. Accessed on December 22, 2015.

```
$ curl 'http://beacon.eecs.berkeley.edu/beacon.php' --data
'population=1000genomes&genome=hg19&chr=chr15&coord=41087869&allele=A'

<!DOCTYPE html>
<html>
<body>


Population: 1000genomes<br>
Genome: hg19<br>
Chromosome: chr15<br>
Coordinate: 41087869<br>
Allele: A<br>


<pre>Beacon found allele A at coordinate chr15:41087869 of hg19 with
    frequency 44
<pre>
</body>
</html>
```

Listing 2.1: Sample request and response of the AMPLab beacon in cURL format.

for SNVs, a string of the aforementioned letters to represent insertions, D and I denoting an arbitrary deletion and insertion, respectively). The response is a formatted JSON[19] document echoing the query, which also contains two boolean response fields, for sequence and variant data. A sample query is listed in Listing 2.2.

University of California, Santa Cruz (UCSC)[20] exposes a beacon serving data from LOVD (see Section 2.10.6), ClinVar (see Section 2.10.3), UniProt[21], an important collection of protein sequences and their annotations [105], and HGMD (see Section 2.10.3). UCSC beacon accepts an HTTP GET request with four parameters – *track* (dataset identifier), *chrom* (chromosome identifier, autosome or X/Y, prefixed with "chr" or INSDC[22] accession like CM000663.1), *pos* (0-based position within a chromosome), and *allele* (allele identifier – one of A,C,G,T). The reference genome is not specified in the request. A response is a simple "*Yes*" or "*No*" provided in plain text. For LOVD, an overlap with a deletion or duplication returns "*Maybe*". A sample query is listed in Listing 2.3.

EMBL-EBI[23] created a beacon as well. It initially exposed data from five datasets, such as UK10K[24], a project characterizing rare and low-frequency variation in the UK population, and studying its contribution to a broad spectrum of biomedically relevant

19. JavaScript Object Notation. http://www.json.org. Accessed on December 22, 2015.
20. University of California, Santa Cruz. http://www.ucsc.edu. Accessed on December 22, 2015.
21. UniProt. http://www.uniprot.org. Accessed on December 22, 2015.
22. International Nucleotide Sequence Database Collaboration. http://www.insdc.org. Accessed on December 22, 2015.
23. The European Bioinformatics Institute. http://www.ebi.ac.uk. Accessed on December 22, 2015.
24. UK10K – Rare Genetic Variants in Health and Disease. http://www.uk10k.org. Accessed on December 22, 2015.

```
$ curl 'http://www.ncbi.nlm.nih.gov/projects/genome/beacon/beacon.cgi?ref=
    GRCh37&chrom=9&pos=136132908&allele=T'

{
        "exist_gt": true,
        "exist_sra": true,
        "query": {
                "allele": "T",
                "chrom": "9",
                "pos": "136132908",
                "ref": "GRCh37"
        }
}
```

Listing 2.2: Sample request and response of the NCBI beacon in cURL format.

```
$ curl 'http://hgwdev-max.cse.ucsc.edu/cgi-bin/beacon/query?track=clinvar&
    chrom=chr1&pos=10042537&allele=T'

No
```

Listing 2.3: Sample request and response of the UCSC beacon in cURL format.

quantitative traits and diseases with different predicted genetic architectures [20], but the number quickly grew to 38 datasets of various species (as of December 2015). The data comes from the European Variation Archive (EVA)[25], a project providing a single access point for submissions, archiving, and access to high-resolution variation data of all types. EVA gathers, normalizes, and annotates the variants from externally hosted datasets as well as those submitted directly to EMBL-EBI [22]. The beacon accepts a query as an HTTP GET request with four parameters – *datasetIds* (dataset identifiers), *referenceName* (chromosome identifier without a prefix, 1-22 for autosomes, 23 and 24 for X and Y, respectively), *start* (1-based position within a chromosome) and *allele* (allele identifier – one of A,C,G,T for SNVs, a string of the aforementioned letters, INDEL or <DEL>). By default, the response is an unformatted JSON document echoing the request parameters together with a boolean response. A sample query is shown in Listing 2.4.

```
$ curl 'http://www.ebi.ac.uk/eva/webservices/rest/v1/ga4gh/beacon?
    referenceName=11&start=110993&allele=INDEL&datasetIds=PRJEB4019'

{"chromosome":"11","start":110993,"allele":"INDEL",
"datasetIds":"PRJEB4019","exists":true}
```

Listing 2.4: Sample request and response of the EBI beacon in cURL format.

25. European Variation Archive. http://www.ebi.ac.uk/eva. Accessed on December 22, 2015.

As the descriptions of the early beacons clearly indicate, the vague nature of the specification leads to significantly different implementations of the API. Beacons respond to various HTTP verbs, accept different sets of parameters, and name the parameters differently. The response is provided in several incompatible formats, many of which are focused on being human-readable as opposed to readable by machines. Beacons even expect different values for each of the parameters – chromosome names can be specified in several formats and certain values are not supported by certain beacons, positions can be 0-based or 1-based, there are multiple naming conventions for reference genomes and different notations for alleles, particularly indels.

2.8 Security

Although the initial idea was for beacons to be completely open, the need for a secured version quickly arose. Some datasets simply contain data too sensitive to be exposed through an open API, even if the API as as simple as beacon. This is especially useful as the query language is being extended and more sensitive information than a simple yes or no is getting exposed in the response.

Security in the context of genomic data is extremely important and complex beyond the scope of this work. The Security working group of GA4GH develops standards for secure data sharing applicable, but not limited to, all the Global Alliance projects, including the Beacon Project. Standards and implementation practices for protecting the privacy and security of shared genomic and clinical data are noted in the Global Alliance for Genomics and Health Security Infrastructure. The Security Infrastructure document contains sections on risk assessment, security technology infrastructure, information security responsibilities, privacy and security policy, identity management, authorization management, data security safeguards, cryptographic controls, physical and environmental security, operations security, communications security, service supplier assurances, information security aspects of business continuity management, and compliance [47]. It is based on ISO/IEC 27001:2013, a standard specifying the requirements for establishing, implementing, maintaining, and continually improving an information security management system within the context of the organization, which also includes requirements for the assessment and treatment of information security risks tailored to the needs of the organization [2]. Further guidance for the responsible sharing of clinical and genomic data throughout the world is provided in the Framework for Responsible Sharing of Genomic and Health Related Data, which provides the basis for the privacy and security policy that the security infrastructure seeks to uphold [64].

Beacons conform to the technology standards and operational guidelines defined in the Security Infrastructure and adhere to the Framework for Responsible Sharing of Genomic and Health Related Data [46].

2.8.1 Access levels

Typically, beacons operate with the notion of access levels closely tied to authentication and authorization. Three standard access levels have been defined.

Open access level is completely public and intended for anonymous Internet users, i.e. no authentication is required. This level is suitable for looking for any record of a variant anywhere in the database – typically, a beacon would only expose a simple yes/no answer here. This is the default access level for beacons, and almost all the beacons implement it.

Controlled access level is on the opposite side of the spectrum. It is the most restrictive of the access levels, requires thorough verification of the user's identity typically involving manual, human effort, as well as signing a legal contract. This kind of access is usually obtained through a data access committee (DAC). At this level, a beacon exposes identifiable, private information, such as a full VCF record, possibly even with patients' data. Controlled access level beacons were implemented by e.g. European Genome-Phenome Archive (EGA)[26], and provide access to individual-level data. In EGA, access to the data requires authentication and a separate data access authorization from the appropriate DAC governing the dataset.

Registered access level stands between the open and the controlled levels. It is designed for those who want to dig deeper than allowed by open access, but do not necessarily need access to raw patient's data. At this level, beacons typically expose information like allele frequency or non-stigmatizing health-related data about non-vulnerable individuals. Compared to controlled access, the approval process for registered access is much simpler in that some of the steps of the standard review procedure are not be necessary. These include, for example, undergoing scientific review and additional ethics review. Registered access does not involve the execution of a Data Access Agreement (DAA) between data providers and data users either. The precise specification of this level is an ongoing work of the Registered Access task team in the Global Alliance. The first implementations typically clear everyone who can show their identity, i.e. either has an account in a given system, or has an account with an approved identity provider. These accounts can often be created automatically, without any manual verification performed by a DAC. This definition of registered access has been adopted by e.g. DNAstack[27], whose beacons require a user to have a DNAstack account, or an account with one of the common identity providers, such as Google[28] or Facebook[29]. Alternatively, institutions only clear bone fide researchers for registered access, which typically requires the users to supply their name, title, position, affiliation, email address, institutional website, mailing address, and possibly sign certain agreements. EGA, for example, clears users with access to at least one controlled-access dataset, which involved communication with a DAC.

2.8.2 Authentication and authorization

Standardization of authentication and authorization mechanisms has been a hard problem for beacons, since various institutions support custom providers and protocols. The current rule-of-thumb recommendation for new beacons is to use OAuth 2.0 with OpenID Connect 1.0. OAuth 2.0 is an authorization framework enabling a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its own behalf [24]. OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 proto-

26. European Genome-Phenome Archive. `https://ega.crg.eu`. Accessed on December 22, 2015.
27. DNAstack. `http://dnastack.com`. Accessed on December 22, 2015.
28. Google. `https://www.google.com/about`. Accessed on December 22, 2015.
29. Facebook. `https://www.facebook.com/`. Accessed on December 22, 2015.

col. It enables clients to verify the identity of the end user based on the authentication performed by an authorization server, as well as to obtain basic profile information about the end user in an interoperable manner [91].

2.9 Privacy

As privacy is an important issue when it comes to sensitive data people are typically dealing with in genomics, the Global Alliance has been actively involved in creating policies that provide specific guidance for responsible sharing of genomic and clinical data. All its projects, including the Beacon Project, can follow GA4GH's Privacy and Security Policy, a guide for sharing of genomic and health-related data in a way that protects and promotes confidentiality, integrity, and availability of data and services, and the privacy of individuals, families, and communities whose data is shared. The policy explains and specifies several privacy-related tasks and terms, including consent, ensuring proportionate safeguards, re-identification, data quality, data disclosure and publication, access, data breach, accountability, transparency, complaints or inquiries, and vulnerable populations [72].

Best practices for consent management are further expanded upon in the Consent Policy. The policy focuses both on consent aspects that enable or encourage the use of data and provides guidance for data sharing in future protocols. It can help determine if data sharing is indeed covered in the existing consent or if re-consent or other forms of approvals or social engagement are necessary to enable data sharing [108].

2.9.1 Attribute disclosure

Although the beacon protocol was originally designed to expose so little information that it would not cause any privacy violations, recent research indicates the API does not necessarily guarantee this property, and adoption of privacy and consent policies is needed.

As shown in [93], even the basic API exposing only information about the presence of an allele in a beacon is susceptible to an attribute disclosure attack using DNA (ADAD), a form of re-identification, which is demonstrated by a likelihood ratio test of whether a given individual is present in a given genetic beacon. The test is not dependent on allele frequencies and is the most powerful test for a specified false positive rate. In a beacon with 1,000 individuals, re-identification is possible with just 5,000 queries. Re-identification is possible even in the presence of sequencing errors and variant calling differences. In a beacon constructed using 65 European individuals from the 1,000 Genomes Project, re-identification is possible with just 250 SNVs. With just 1,000 SNV queries, the presence of an individual from the Personal Genome Project in an existing beacon can be detected.

These findings present a non-trivial risk for a data sharing initiative like the Beacon Project. Even though current beacons host mostly public data, adoption of restricted data is crucial for the success of the project, and privacy leaks like this one can be dangerous. Similar situations were observed in the past – National Institutes of Health (NIH)[30] removed aggregate statistics from open-access databases [116] after it was

30. National Institutes of Health. `http://www.nih.gov`. Accessed on December 22, 2015.

shown that statistics across cohorts, such as allele frequency or genotype counts, do not mask identity within genome-wide association studies [54].

The problem with ADAD is that it is possible to execute it not only for individuals in the beacon, but also their relatives. Furthermore, beacons are often associated with phenotypes, e.g. serve data of individuals with a specific disease. While detecting membership does not breach privacy, the membership could allow the attacker to associate a person with the beacon's phenotype, which could constitute a serious breach. Hypothetically speaking, an insurance company could, for example, refuse to provide life insurance to a person that they identified as a patient with a terminal disease through a study exposed via a beacon.

Fortunately, this instance of ADAD has little probability of being employed as a privacy attack in the real world. The attacker is required to have all or most of the information about the genome of the person whose privacy they are attacking, or of a close relative. If someone has obtained most of an individual's entire genome sequence, they have already obtained more information than is made available through beacons. The only new information is whether or not the individual's genome is included in a beacon.

Moreover, several approaches can be used to mitigate the risk of ADAD. These include aggregating data from multiple beacons to increase the database size and obscure the database of origin, creating an information budgeting system to track the rate at which information is revealed and to restrict access when the information disclosed exceeds a certain threshold, introducing multiple tiers of secured access and requiring users to be authorized for data access as well as to agree not to attempt specific risky scenarios. To further decrease the risk of ADAD, beacons can choose to publish dataset metadata through restricted access levels, expose a single population as opposed to multiple populations in a single beacon, and share only small genomic regions (genes or exomes) instead of whole genomes. These strategies adhere to the best practices outlined in Privacy and Security Policy [46].

2.10 Other systems

Although beacons provide a unique approach to federated genomic data sharing, they exist in a large ecosystem, where many systems allow some sharing of genomic data. The number of tools is too high for the scope of this work – in fact, various research groups have spent significant resources cataloging system like these. As such, we divide the tools into basic categories and look at the most popular systems in each one of them.

2.10.1 Matchmakers

Matchmakers represent the category of tools probably closest to beacons. The Matchmaker Exchange is a protocol and platform for exchange of phenotypic and genotypic information. Like the Beacon Network, the Matchmaker Exchange uses a federated approach to data sharing. However, the system is very patient-focused and requires the users to contribute a case to one of its services before they can execute a search. Once a matchmaker service obtains a case, it queries other services on behalf of the user. These services use the structured patient data in the query to identify and return descriptions of similar cases within their respective databases. They are not permitted to store request data for uses other than analytics and diagnostics, i.e. the data exchanged over the API does not become a part of the data stored by the receiving services. Similar cases found through the API are then reported to the users for evaluation. The users can then follow up with each other on any promising matches using contact information provided with the query and response [15]. The Beacon Network does not require such symmetry and joining it is much less constrained.

Several matchmaking implementations are available. One of the most prominent ones is PhenomeCentral, a platform which identifies similar patients in the database based on semantic similarity between clinical features, automatically prioritized genes from whole-exome data, and candidate genes entered by the users, enabling both hypothesis-free and hypothesis-driven matchmaking [16].

DECIPHER (DatabasE of genomiC variation and Phenotype in Humans using Ensembl Resources) is a web-based platform for secure deposition, analysis, and sharing of plausibly pathogenic genomic variants from well-phenotyped patients suffering from genetic disorders [37]. It aids clinical interpretation of these rare sequence and copy-number variants by providing tools for variant analysis and identification of other patients exhibiting similar genotype-phenotype characteristics [17].

GenomeConnect is a patient portal supported by the Clinical Genome Resource (ClinGen), which provides an opportunity for patients to add to the knowledge base by securely sharing their health history and genetic test results. Data can be matched with queries from clinicians, laboratory personnel, and researchers to better interpret the results of genetic testing and build a foundation to support genomic medicine [62].

2.10.2 Genome browsers

Genome browsers are tools for visualizing annotations within the context of a linearized genome. They are focused on spatial relationships of data, which often indicate functional relationships in the genome. Historically, they have been successfully used to share annotation data.

Dozens of genomes browser implementations are available today [39]. One of the first, and a de-facto standard for researchers today, is the UCSC Genome Browser. The browser is a mature web tool for rapid and reliable display of any requested portion of the genome at any scale, together with many aligned annotation tracks, assembly contigs and gaps, mRNA and expressed sequence tag alignments, multiple gene predictions, cross-species homologies, single nucleotide polymorphisms, sequence-tagged sites, radiation hybrid data, transposon repeats etc. Text and sequence-based searches provide quick and precise access to any region of specific interest. Secondary links from individual features lead to sequence details and supplementary off-site databases [61].

Over time, the UCSC Genome Browser database has become one of the most important resources for bioinformaticians. It hosts a large repository of genomes with 166 assemblies that represent over 93 different organisms, and aggregates hundreds of annotation tracks – in 2015 alone, UCSC added over 200 new and updated tracks [100].

The UCSC Genome Browser laid foundations for other web-based genome browsers, most of which provide a very similar user interface. Traditionally, browsers render images on the server side, which are subsequently displayed in the client. A different approach is used e.g. by JBrowse, a JavaScript-based genome browser, which can be used to navigate genome annotations over the web. JBrowse helps preserve the user's sense of location by avoiding discontinuous transitions, instead offering smoothly animated panning, zooming, navigation, and track selection [94].

Several desktop browsers exist as well, and often deliver better performance than web-based solutions, which is important particularly for very large datasets. The most popular are probably Integrative Genomics Viewer, a cross-platform genome browser with support for concurrent visualization of diverse datasets [88], and Savant, which provides some innovative visualization modes and navigation interfaces for several genomic data types, and synergizes visual and automated analyses in a way that is powerful yet easy even for non-expert users [38].

In many ways, genome browsers are similar to the Beacon Network. They are a popular tool for sharing annotation data. Their queries are specified in a similar way (chromosome and interval of positions), since the whole genome is too large to show without zooming to a specific interval, and they aggregate data from various reference databases. Obviously, they show much more data than beacons through various visualizations. They do not, however, support federation of queries to the extent of the Beacon Network.

2.10.3 Platforms for reference data sharing

Several public projects have been created to aggregate reference data. These include both genotypic and phenotypic databases and portals, which are often used by researchers.

One of the most widely used databases is ClinVar, an archive of reports of relationships among medically important variants and phenotypes. It is highly curated and contains rich, structured details of phenotype, interpretation of functional and clinical significance, methodology used to capture variant calls and supporting evidence. Submissions are categorized by type of data capture (e.g. clinical testing, literature evaluation, and research) and level of review status (single submission, expert panel, and practice guideline). ClinVar thus gives users access to a broader set of clinical interpretations than they may have collected on their own and the promise of a comprehensive site for obtaining current and historical data. The system is available to individual users and organizations that want to incorporate the data into local applications and workflows. The database can be downloaded for free via FTP[31] or accessed programmatically [71].

ClinVar is tightly coupled with dbSNP and dbVar, and uses phenotypic descriptions maintained by MedGen. dbSNP is a public catalog of genomic variation addressing the large-scale sampling designs required by association studies, gene mapping and evolutionary biology. Once variations are identified and cataloged in the database, additional laboratories can use the sequence information around the polymorphism and the specific experimental conditions for further research applications. The scope of dbSNP includes disease-causing clinical mutations as well as neutral polymorphisms. The database can be downloaded via FTP and accessed programmatically [97]. While dbSNP contains

31. RFC 959 – File Transfer Protocol. `https://www.ietf.org/rfc/rfc959.txt`. Accessed on December 22, 2015.

short nucleotide variations, dbVar is focused on large-scale genomic variants (generally 50 bp or more), such as insertions, deletions, translocations, and inversions [3]. MedGen is a portal providing information about human disorders and other phenotypes having a genetic component. It is structured to serve health care professionals, the medical genetics community, and other interested parties by providing centralized access to diverse types of content [85].

NCBI offers several other specialized databases for reference data, such as Reference Sequence Database (RefSeq), a non-redundant collection of sequences representing genomic data, transcripts and proteins. Although the goal is to provide a comprehensive dataset representing the complete sequence information for any given species, the database pragmatically includes sequence data that are currently publicly available in the archival databases. Sequences are annotated to include coding regions, conserved domains, variation, references, names, database cross-references, and other features. The RefSeq collection is unique in providing a curated, non-redundant, explicitly linked nucleotide and protein database representing significant taxonomic diversity. The collection is derived from the primary submissions available in GenBank, a comprehensive public database of nucleotide sequences and supporting bibliographic and biological annotation [13]. GenBank is a redundant archival database that represents sequence information generated at different times, and may represent several alternate views of the protein, names or other information. In contrast, RefSeq represents a nearly non-redundant collection that is a synthesis and summary of available information, and represents the current view of the sequence information, names and other annotations [86].

NCBI's Assembly database provides stable accessioning and data tracking for genome assembly data. Multiple sequencing groups may produce different genome assemblies for the same organism and any one group may release different versions of an assembly as they generate more sequence data, close gaps, correct misassemblies or make other improvements to the assembly. The multiple assembly versions for an organism need to be clearly identified, differentiated and tracked in a way that allows researchers to unambiguously refer to the set of sequences that comprise a particular version of a genome assembly. The database stores the names and identifiers for the sequences in each genome assembly and records the organization of the component sequences into scaffolds and chromosomes. This enables the database to report the assembly structure and to provide mappings between names, synonyms and identifiers for assemblies, chromosomes or scaffolds. In addition, the database calculates numerous statistics from the sequences in each assembly so that users can evaluate different assemblies and tracks updates so that users can see the history of previous versions of an assembly. The database also records a variety of metadata about genome assemblies such as names, dates, the degree of assembly, the group that generated the assembly and details about the sequenced organism and sample [63].

Another interesting database is HapMap, a catalog of common genetic variants that occur in humans. It describes what these variants are, where they occur in our DNA, and how they are distributed among people within populations and among populations in different parts of the world. The goal of HapMap is not to establish connections between particular genetic variants and diseases – rather, it tries to provide information that other researchers can use to link genetic variants to the risk for specific illnesses, which can lead to new methods of preventing, diagnosing, and treating disease [57].

Ensembl is a bioinformatics project to organize biological information around the sequences of large genomes. It is a comprehensive source of stable automatic annotation of individual genomes, and of the synteny and orthology relationships between them. It is also a framework for integration of any biological data that can be mapped onto features derived from the genomic sequence [14]. In many ways, Ensembl is similar to the UCSC Genome Browser, in that it provides a similar graphical view and acts as a hub for reference data. However, we decided to put in the category of reference data providers, as we view it primarily as a system for genome annotation, analysis, and storage [113].

Interesting resources also include Online Mendelian Inheritance in Man (OMIM), a catalog of associations between human phenotypes and their causative genes (the Morbid Map of the Genome), which tries to classify and name newly recognized disorders [10]. It has cataloged human Mendelian diseases for over 40 years [79], and contains a variety of textual information including patient symptoms and signs [53].

Clinical Genomic Database (CGD) is a manually curated database of conditions with known genetic causes, focusing on medically significant genetic data with available interventions. It contains genes organized clinically by affected organ systems and interventions (including preventive measures, disease surveillance, and medical or surgical interventions) that could be reasonably warranted by the identification of pathogenic mutations. To aid independent analysis and optimize new data incorporation, the CGD also includes all genetic conditions for which genetic knowledge may affect the selection of supportive care, informed medical decision-making, prognostic considerations, reproductive decisions, and allow avoidance of unnecessary testing, but for which specific interventions are not otherwise currently available [99].

Additional gene-related information is provided via GenAtlas and Human Gene Mutation Database (HGMD). GenAtlas is a database containing information on human genes, markers and phenotypes [41]. HGMD is a comprehensive collection of germline mutations in nuclear genes that underlie, or are associated with, human inherited diseases [102].

The catalog of published Genome-Wide Association Studies (GWAS) is a resource describing SNP-trait associations and is manually curated from literature by the National Human Genome Research Institute (NHGRI) [112].

Database for non-synonymous SNVs' functional predictions (dbNSFP) and Functional Single Nucleotide Polymorphism (F-SNP) database are great resources for functional effects of variants. dbNSFP is a compilation of prediction scores from multiple popular algorithms [75]. F-SNP database integrates information obtained from databases about the functional effects of SNPs. These effects are predicted and indicated at the splicing, transcriptional, translational and post-translational level. In particular, users can retrieve SNPs that disrupt genomic regions known to be functional, including splice sites and transcriptional regulatory regions. Users can also identify non-synonymous SNPs that may have deleterious effects on protein structure or function, interfere with protein translation or impede post-translational modification [73].

SolveBio[32] is a platform for delivery of critical reference data used by hospitals and

32. SolveBio. `https://www.solvebio.com`. Accessed on December 22, 2015.

companies to run genomic applications. It distributes data from many sources listed in this section, such as CGD.

When describing reference databases, molecular databases should not be completely omitted. They are, however, a bit far from the concept of genomic beacons, so we only mention them briefly. A popular database in this category is Molecular modeling database (MMDB) and Protein Data Bank (PDB). PDB is a world-wide repository of experimentally determined structures of proteins, nucleic acids, and complex biomolecular assemblies, which is curated and annotated [90]. MMDB mirrors the contents of PDB and links protein 3D structure data with sequence data and sequence classification resources [76]. It also integrates PubChem, an open repository for chemical structures and their biological test results [110]. Another interesting database in this category is ChemBank, a small-molecule database dedicated to storage of raw screening data, rigorous definition of screening experiments in terms of statistical hypothesis testing, and metadata-based organization of screening experiments into projects involving collections of related assays. ChemBank stores an increasingly varied set of measurements derived from cells and other biological assay systems treated with small molecules [92].

Overall, reference databases are not an alternative to beacons or the Beacon Network, even though many of them aggregate data from various sources and provide programmatic APIs for accessing this data. In fact, some of them, such as ClinVar and CGD, are exposed as beacons, and plug into the federation enabled by the Beacon Network nicely.

2.10.4 Platforms for case-level data sharing

Various institutions provide case-level, sample-based data. This includes both public data, such as the 1,000 Genomes Project, and controlled data, such as UK10K.

One of the major providers is the Exome Aggregation Consortium (ExAC), a coalition of investigators seeking to aggregate and harmonize exome sequencing data from a wide variety of large-scale sequencing projects, and to make summary data available for the wider scientific community. ExAC provides a browser for high-quality exome sequence data for 60,706 individuals of diverse ethnicities and can be used to calculate objective metrics of pathogenicity for sequence variants, and to identify genes subject to strong selection against various classes of mutation [34].

The Database of Genotypes and Phenotypes (dbGaP) archives, distributes and supports submission of data that correlate genomic characteristics with observable traits. It is a public repository for individual-level phenotype, exposure, genotype, and sequence data, and the associations between them. To protect the confidentiality of study subjects, dbGaP accepts only de-identified data and requires investigators to go through an authorization process in order to access individual-level data. Study documents, protocols, and subject questionnaires are available without restriction [77].

Kaviar is a database of genomic variation from personal genomes, family genomes, transcriptomes, SNV databases and population surveys, which was created to facilitate testing for the novelty and frequency of observed variants. It contains 169 million SNV sites (including 37M not in dbSNP), and incorporates data from 34 projects encompassing 77,238 individuals [49].

The Cancer Genome Atlas (TCGA) is a comprehensive and coordinated effort to accelerate our understanding of the molecular basis of cancer through the application of genome analysis technologies, including large-scale genome sequencing. TCGA Data Portal provides a platform for researchers to search, download, and analyze data sets related to various tumor types [111].

The International Cancer Genome Consortium (ICGC) is a collaborative effort to characterize genomic abnormalities in 50 different cancer types. To make this data available, ICGC has created the ICGC Data Portal. The Data Portal allows each member institution to manage and maintain its own databases locally, while presenting all the data in a single access point for users. Available open access data types include simple somatic mutations, copy number alterations, structural rearrangements, gene expression, microRNAs, DNA methylation, and exon junctions. Additionally, simple germline variations are available as controlled access data [117].

RD-Connect is an infrastructure project connecting databases, registries, biobanks, and clinical bioinformatics data used in rare disease research into a central resource for researchers. It provides an integrated platform to host and analyze genomic and clinical data from research projects, clinical bioinformatics tools for analysis and integration of molecular and clinical data to discover new disease genes and pathways, common infrastructures and data elements for rare disease patient registries, common standards and catalog for rare disease biobanks, and best ethical practices with a regulatory framework for linking medical and personal data related to rare diseases [106].

Google Genomics is a cloud-based platform for processing human genomic data, such as reads, variants, references, and annotations. The platform does not have a user-friendly interface, but provides a powerful developer-friendly API. It hosts a handful of datasets including 1,000 Genomes data, TCGA data, and MSSNG Database for Autism Researchers, whole-genome sequence data of families with autism spectrum disorder [115].

Many more initiatives are active. Similarly to the reference data providers, these platforms for genomic data sharing are compatible with the beacon protocol. In fact, data from all the systems listed in this section is available through beacons (see Section 4.3).

2.10.5 Platforms for consumer data sharing

While most of the tools in the space of genomic data sharing are designed for experts, such as researchers and clinicians, systems for making data available to non-expert consumers exist as well.

The most prominent system in this category is 23andMe[33]. 23andMe sequences a part of user's genome and provides a detailed report on around 100 conditions and traits as well as ancestry information. The portal provides a web-based interface with user-friendly visualizations around all the data, which is also available for download in text form. The company regularly publishes research findings.

Similarly to other data-sharing platforms, a part of 23andMe data has been exposed through a beacon.

33. 23andMe. `https://www.23andme.com`. Accessed on December 22, 2015.

2.10.6 Data-sharing software

Unlike the systems described Sections 2.10.3 to 2.10.5, which are primarily run as hosted services, certain tools are designed to be downloaded, installed, and run with custom data.

One of these tools is Leiden Open Variation Database (LOVD), a flexible, web-based, open-source tool for gene-centered collection and display of DNA variations. The latest release extends this idea to also provides patient-centered data storage and storage of next-generation sequencing data, even of variants outside of genes [40]. In a way, LOVD is similar to a database like MySQL[34] – it can be set up as a shared instance with custom users, and filled with custom data.

Similarly to LOVD, Cafe Variome is run as a service, but also provides a web-based data-hosting system which can be set up locally. Cafe Variome is a general-purpose data discovery tool that can be installed by a genotype-phenotype data owner, or network of data owners, to make safe or sensitive content appropriately discoverable [70].

The setup of these systems is similar to the Beacon Network – while they are run as services, people can host their own local instances as well.

2.10.7 Distributed search and retrieval systems

Although almost all the platforms listed in Sections 2.10.3 to 2.10.5 support data search and retrieval, we decided to create a special category for systems taking this functionality further, and integrating with other systems in the aforementioned sections.

BioMart enables scientists to perform advanced querying of biological data sources through a single web interface. It is an open-source data-management system that comes with a range of query interfaces allowing users to group and refine data based upon many different criteria. In addition, the software features a built-in query optimizer for fast data retrieval. A BioMart installation can provide domain-specific querying of a single data source or function as a web portal to a wide range of data sources. This is where the power of the system comes from – if any datasets share common identifiers, such as Ensembl gene IDs, or even mappings to a common genome assembly, these can be used to link the sources together in queries. Additionally, these datasets do not have to be located on the same server, or even in the same geographical location [96]. BioMart is a very generic and scalable system and has become an integral part of large data resources including Ensembl, HapMap and ICGC. With the federation of queries and aggregation of results, it is similar to the Beacon Network.

NCBI databases are accessible through Entrez, an integrated database retrieval system. Entrez supports text searching, downloading of data in various formats and linking of records between databases based on biological relationships. In their simplest form, these links may be cross-references between a sequence and the abstract of the paper in which it is reported, or between a protein sequence and its coding DNA sequence or its 3D structure [3]. Entrez is a cross-database federated search engine, not unlike the Beacon Network. Its search capabilities are much more extensive, but limited to NCBI resources.

DBGET is an integrated database retrieval system for major biological databases. It

34. MySQL. `https://www.mysql.com`. Accessed on December 22, 2015.

provides the basis of GenomeNet[35]. It is distributed, open, and suitable for incorporating local databases as well as establishing a specialized server. While its primary focus is on molecular databases, it does integrate with several sources of human genomic data, such as RefSeq [42].

Distributed Annotation System (DAS) is a specification of a protocol for requesting and returning annotation data for genomic regions. It addresses the problem of scaling genome annotation curation, which is typically performed by centralized groups with limited resources. DAS allows sequence annotation to be stored in a decentralized manner, by multiple third-party annotators, and integrated on an as-needed basis by client-side software. The basic system is composed of a genome server, one or more annotation servers, and an annotation viewer. The genome server is responsible for serving genome maps, sequences, and information related to the sequencing process. Annotation servers are responsible for responding to requests on a region and delivering annotations. The client, an annotation viewer, is a lightweight application whose behavior is analogous to a web browser. The viewer communicates with the genome and annotation servers using a well defined language specification [30]. Among other systems, DAS powers Ensembl, where it is used in integrate external data, as well as export data to other applications. Even though DAS is not a traditional search engine, it is designed to look up and retrieve annotations in a decentralized way, and thus we are listing it in this category.

35. GenomeNet. `http://www.genome.jp`. Accessed on December 22, 2015.

3 Design and implementation

"The Beacon Network allows someone to identify where in the world this mutation has been observed before. It's kind of a data discovery engine, which one can follow up."

– Marc Fiume, Chief Executive Officer, DNAstack. Source: [68].

3.1 Requirements

Beacon Network was developed to a address a set of very specific issues in the genomic data-sharing ecosystem. As such, certain key requirements were identified before the design started:

- Federation of queries across beacons.

  The system has to support queries for multiple beacons. As such, the Beacon Network has to be able to receive a query and instantiate it across a network of beacons.

- Integration of publicly available beacons.

  The network needs to support all the public beacons available at the start of its development. This includes accounting for differences in their APIs, i.e. converting the queries to a beacon-specific form a particular participant can understands.

- Aggregation of data from multiple sources.

  The application needs to be able to collect data coming from beacons participating in a query. It needs to account for differences in the beacons' responses, i.e. it needs to be able to support variable names of fields and a range of formats.

- Online distribution of queries without the need to store genomic data.

  Storage of genomic data introduces a range of regulatory and technical issues. To avoid these, the system needs to distribute the queries across beacons in an online manner, without hosting any genomic data itself (with the exception of intermediary caching, logging, audit trail).

- Registry of public beacons.

  The Beacon Network should become a de-facto global registry of public beacons. It should integrate new beacons as they pop up and serve as the go-to application for multi-beacon queries.

- Programmatically accessible.

  The system has to provide a machine-readable API in a standard format, such as a JSON-based REST API.

- Easily accessible.

  The network should be accessible globally with minimal restrictions. It should be hosted on the Internet, accessible through standard browsers and allow public, anonymous access without the need to create an account.

- Unified beacon API.

  The network should hide the API nuances of individual beacons and support single-beacon queries via a standard, unified API.

- Push for standardization of the Beacon API.

  As a prominent application built on top of the Beacon API, the Beacon Network should be a driving force in standardizing APIs of individual beacons, even the ones not participating in the network.

- Performance.

  One of the crucial ideas behind the concept of beacons was their ability to respond within a short amount of time (near real time), as opposed to a callback after a longer processing task, both due to simplicity and scalability in a distributed environment. The system needs to extend this property to multi-beacon queries and offer relatively short response times (in the order of seconds). To achieve this over a large number of beacons, extensive parallelization is necessary.

- Scalability.

  Being the de-facto registry of beacons, the network needs to be able to scale both in terms of the number of beacons in the network and the number of users asking the queries. The system should be heavily parallelized and allow for an easy distributed setup.

- Modern software engineering practices.

  When it comes to bioinformatics, many legacy tools are still in use. As a core application on top of the Beacon API, the Beacon Network needs to demonstrate modern enterprise engineering practices. This includes extensive automated testing as well as the use of mature standard technologies.

- Modularity and extensibility.

  Institutions may choose to break compliance of their beacon with the specification for various reasons, or only support older versions of the specification. For cases like these, the Beacon Network needs to be easily extensible to support non-standard APIs as well as small differences between various beacon implementations. To make this easier, the code base should be organized into independently updatable and easily replaceable modules.

- Logging and audit trail.

  The system should provide query logging for auditing purposes and analysis of potential attacks.

- Beacon monitoring.

  An organization might decide to change the API of its beacons without notifying an upstream application, such as the Beacon Network. For this reason, the system should have an automatic way of checking the availability and compatibility of beacons' APIs and periodically run these checks to mitigate the risk of giving faulty responses.

- Lower barrier of entry for beacon developers.

  Institutions are often willing to share their data, but do not have the technical expertise to do so. The Beacon Network should make it easier for developers to implement beacons by providing a set of templates and APIs.

- Development under the umbrella of GA4GH.

  Like the Beacon Project, the Beacon Network needs to be developed under the umbrella of the Global Alliance. The feedback from its members, the traction of its APIs, the expertise of the Beacon task team and the alignment of the system with the development of the Beacon Project itself are crucial for the success of the network.

3.2 Architecture

With modularity and extensibility in mind, we designed the Beacon Network system as a collection of 9 logical subsystems comprising 18 modules in total. Each subsystem consists of 2 modules – an API module and a module implementing the API. This hierarchy is shown in Figure 3.1.

The system is organized using Apache Maven[1], a popular project management tool which encompasses a project object model, a set of standards, a project lifecycle, a dependency management system, and logic for executing plugin goals at defined phases in a lifecycle [19]. Modules in our system are not only logically independent units, but also actual Maven modules, independent compilation units as defined by a multi-module project in [19].

Subsystems are split into API and implementation modules primarily for flexibility. API modules are thin layers with no non-API dependencies. Implementation modules are decoupled from their respective API modules and can be compiled and packaged separately. This is an advantage for modularity, since implementation modules are likely to change more often than the relatively stable API. Implementation modules have no compile-time dependencies on other implementation modules, they depend only on API modules. The actual implementations are supplied automatically by the runtime. This creates a very clean, loosely-coupled and extensible structure, where each non-API change affects a single module, and that is the only unit that needs to be recompiled. This dependency hierarchy is shown in Figure 3.2.

As for the runtime, we are referring to a Java EE 7[2] runtime, in our case provided

1. Apache Maven Project. `https://maven.apache.org`. Accessed on December 22, 2015.
2. Java Platform, Enterprise Edition (Java EE) 7. `http://docs.oracle.com/javaee/7`. Accessed on December 22, 2015.

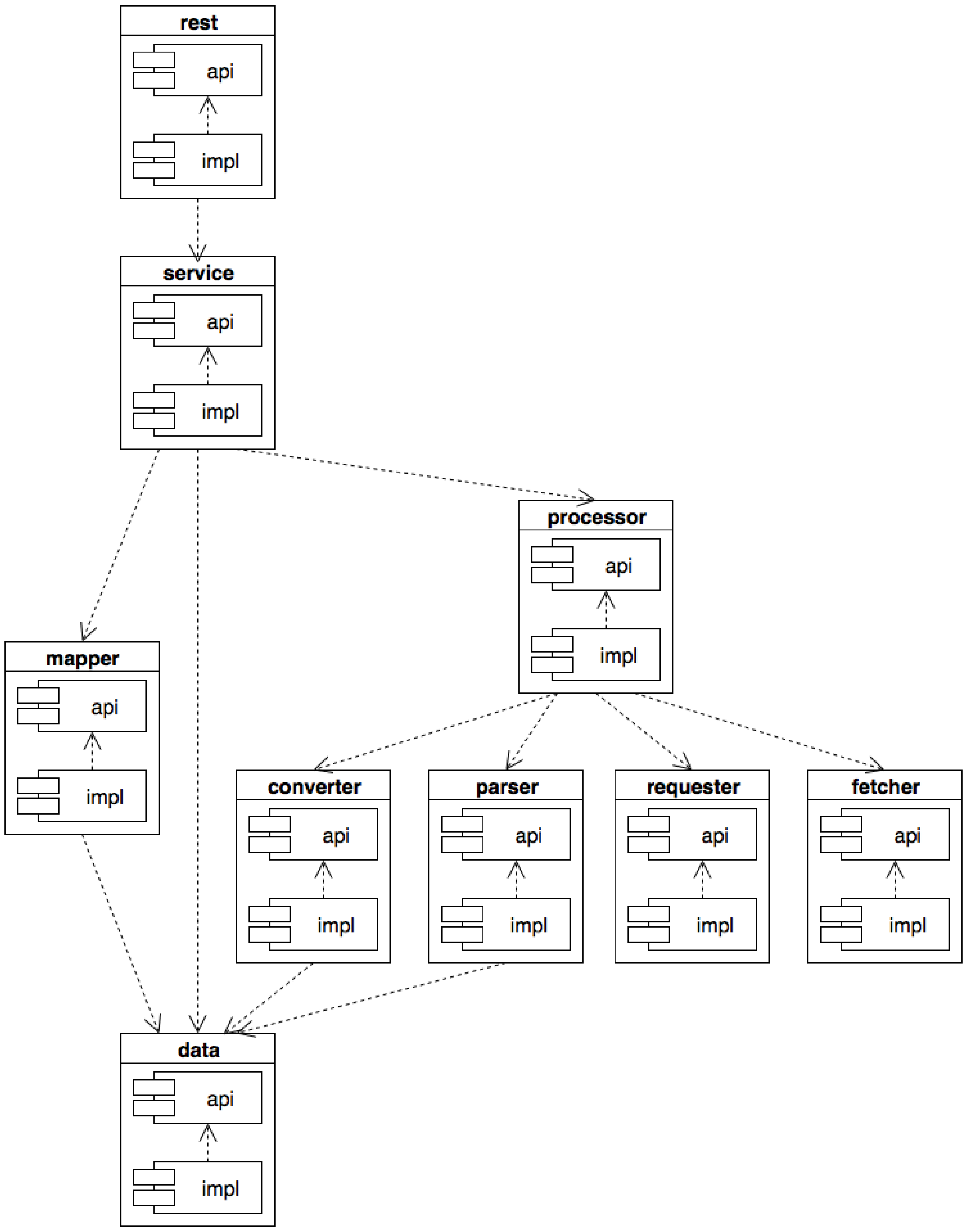


Figure 3.1: Architecture of the Beacon Network showing the organization of 9 subsystems with their API (*api*) and implementation (*impl*) modules.

by the WildFly[3] application server. Our system is written in Java 8[4] and utilizes many

3. WildFly. `http://wildfly.org`. Accessed on December 22, 2015.
4. The Java Language Specification – Java SE 8 Edition. `https://docs.oracle.com/javase/specs/jls/se8/html`. Accessed on December 22, 2015.

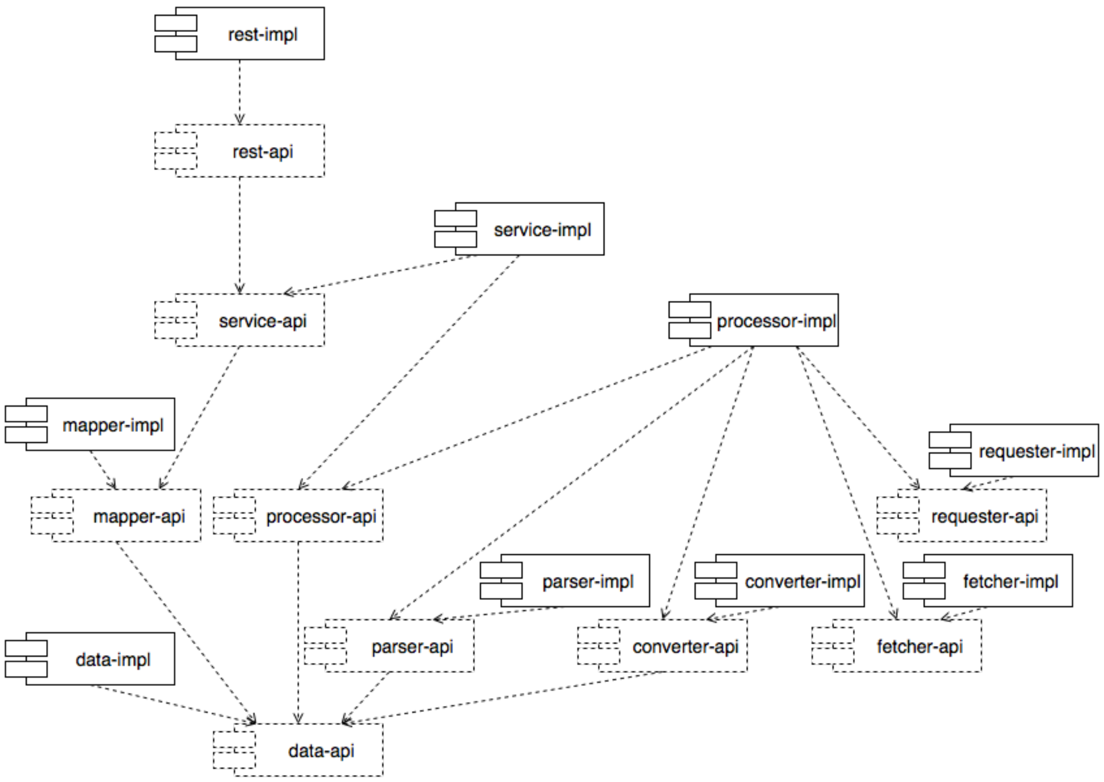


Figure 3.2: Compile-time module dependencies of the Beacon Network. For clarity, the API modules are displayed in dashed, while the implementation modules are displayed in solid containers.

of the cutting-edge features of the release, such as lambdas, method references, stream API and optionals. It uses several specifications from the Java EE 7 stack and heavily relies on a compatible runtime.

3.3 Data

The Data subsystem represents the persistence layer of the system – it provides access to the data stored in a relational database.

From the implementation perspective, the Data subsystem uses Java Persistence API (JPA) 2, a specification from the Java EE stack. JPA defines management of persistence and object-relational mapping. It is based on the notion of entities as lightweight persistent domain objects. As for the basic terminology, an entity is defined by its entity class (and possibly other helper classes), which is a non-final class with a public or protected no-argument constructor. The persistent state of an entity is determined by its persistent fields and properties (attributes of the class) usually accessed via respective getter and setter methods. Associations between entities are modeled via polymorphic entity relationships (one-to-one, one-to-many, many-to-one, or many-to-many). A set of entity instances with a unique instance for any persistent entity is known as the persistence context. Within a persistence context, entity instances and their lifecycle are managed. Interaction with the persistence context is performed through an entity manager. A set of all the classes grouped by the application and collocated in their mapping to a single database is defined by a persistence unit. The specification also describes operations on entities, container and provider contracts for deployment and metadata. It introduces a language to define queries over entities and their persistent state – Java Persistence Query Language (JPQL) [59]. Our system uses Hibernate, a popular implementation of JPA provided by the WildFly runtime.

The API module in the Data subsystem provides isolation of the upper application layers from the persistence implementation details. It contains entities forming the data model of the application as well as interfaces of the data access objects, the implementations of which based on the entity manager are provided via the implementation module. The module executes all the database queries defined in entities using JPQL, and takes care of data validation before writes. The validation uses Bean Validation 1.1 specification from the Java EE stack, which defines an annotation-based object-level constraint declaration and validation facility, as well as a constraint metadata repository and query API [12]. The data model of the system is explained in Figure 3.3.

To keep the code base readable and compact, the persistence layer extensively relies on Lombok[5] and uses its code-generation capabilities to provide constructors, builders, getter/setter and standard *equals* and *hashCode* entity methods.

Portability is important. Our system uses a MySQL database, but thanks to the isolation provided by JPA and our API modules, the actual database can be easily replaced by an alternative relational storage. The same is true for Hibernate and other implementations of the JPA standard.

3.4 Service

The Service subsystem is the core of the business logic in the Beacon Network. Its primary responsibilities include:

- Communication with other subsystems.

  The Service subsystem is the only unit the REST subsystem talks to. All the operations pass through this layer before they are directed to other subsystems for further processing.

- Query normalization.

  The Beacon Network supports several notations for the query parameters to account for various conventions users of other systems are used to. Chromosome values can be specified with an arbitrary prefix, such as *chr* or *chrom*, where the case does not matter. Indels can be specified as *INS*/*DEL*, or *I*/*D*. Reference genome identifiers, such as *GRCh37* and *HG19*, are supported as well. Before the query is processed, the Service subsystem normalizes these parameters into an unambiguous representation and validates the query.

  Our system uses genomic coordinates internally, but it also supports Locus Reference Genomic (LRG) sequence format in the queries. The Service subsystem takes care of this translation. LRG is a manually curated record that contains

5. Project Lombok. `https://projectlombok.org`. Accessed on December 22, 2015.

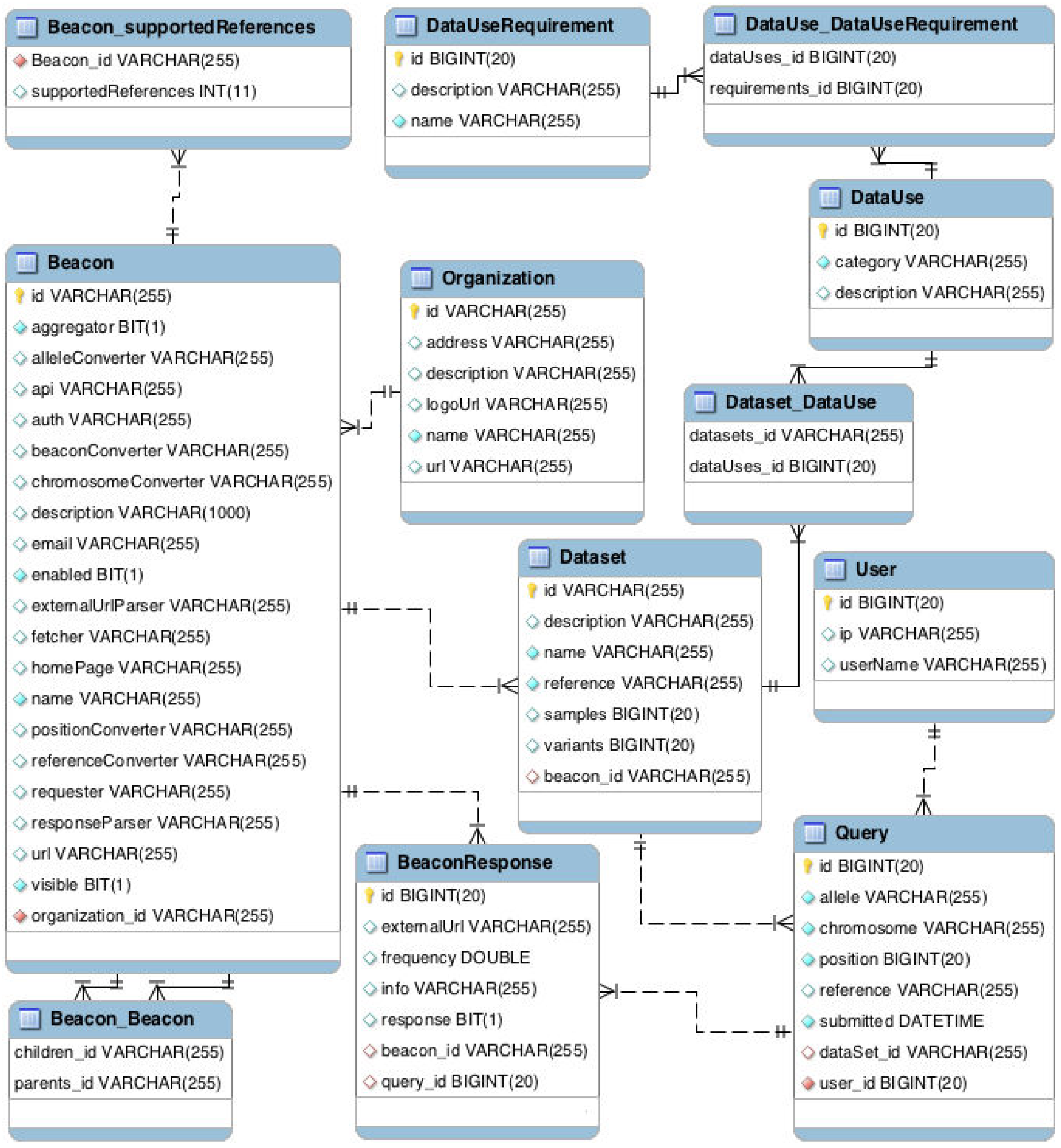


Figure 3.3: Entity-relationship diagram showing the data model of the Beacon Network.

stable, unversioned reference sequences designed specifically for reporting sequence variants with clinical implications. The LRG system provides a genomic DNA sequence representation of a single gene that is idealized, has a permanent ID (with no versioning), and core content that never changes (nucleotide sequence, transcripts, exons, start and stop codon positions). This core annotation is known as the fixed-annotation layer. Additional annotations, known as the updatable-annotation layer, that may change with time, provide ancillary information about a gene [25].

- Aggregators.

  The Beacon Network introduces the concept of aggregators, virtual beacons representing a group of physical or other virtual beacons. The relationship between an aggregator and a beacon it covers resembles the one between a parent and a child in a class hierarchy of object-oriented programming languages supporting multi-inheritance. Each beacon can be a child of an aggregating beacon, and each aggregating beacon can be a parent of other beacons, thus creating a tree-like structure of beacons (a directed acyclic graph in a general case). A child can have multiple parents, and a parent can have multiple children.

A beacon can be exposed on its own, or hidden behind a visible aggregator. This allows us to anonymize certain beacons in the network, while still keeping their data available.

An aggregator does not host any genomic data on its own, it merely acts as a proxy – whenever it receives a query, it sends the query to all its children, collects their responses and provides a single global result, i.e. answers *yes* when at least one its children answers *yes*, all transparently to the user.

The Beacon Network contains several aggregators. One of them is the Beacon of Beacons, an aggregator covering all the beacons in the network, and the root of the beacon hierarchy. Similarly to the definition of beacons in Section 2.4, the Beacon of Beacons answers the question *"Does anybody have any genomes with an allele X at position Y on chromosome Z?"*. The network also exposes an aggregator for each organization exposing multiple beacons, which answers the question *"Does this organization have any genomes with an allele X at position Y on chromosome Z?"*.

- Participant resolution.

  Our system supports multi-beacon queries. Whenever a query is received, the Service subsystem computes a list of beacons participating in the query. While this is easy for regular non-aggregate beacons, aggregators are analyzed by a breadth-first search on the beacon graph or its subgraph. The subsystem makes sure a list of all the necessary beacons is computed and their duplicates are removed, so that each of them is contacted at most once, and only the beacons needed to answer the query are actually talked to. This optimization provides a significant improvement in response times, particularly for queries involving large aggregators.

- Query distribution.

  The Service subsystem distributes the queries across beacons and summarizes the results as they come in. This task is parallelized – each beacon is queried asynchronously in a separate thread (or a set of threads) from a container-managed thread pool.

- Audit trail.

  The Service subsystem logs important actions affecting the state of the system (e.g. modification of beacons). The information is logged to a database and a file, the latter being achieved using Apache Log4j[6], a popular logging framework.

6. Apache Log4j 2. `http://logging.apache.org/log4j`. Accessed on December 22, 2015.

The API module of the Service subsystem consists of interfaces describing the services, which are implemented in the implementation module using Enterprise JavaBeans (EJB) 3.2 specification from the Java EE stack. EJB is an architecture for component-based transaction-oriented enterprise applications [33]. Our services are stateless session beans, business objects that do not have state associated with them, whose lifecycle is managed by the container provided by the runtime. Using container-managed concurrency and properties of stateless beans, we guarantee the required level of isolation and safety – the services are thread-safe, and concurrent users access different instances.

While most of the services are synchronous, beacon queries are implemented using asynchronous EJBs. All the operations are transactional, with transactions being controlled declaratively using Java Transaction API (JTA) 1.2, a specification in the Java EE stack for local interfaces between a transaction manager and the parties involved in a distributed transaction system: the application, the resource manager, and the application server [81].

3.5 Processor

The Processor subsystem is responsible for executing a query against a beacon and processing its response. A processor instance receives a query from the service subsystem and controls its execution against a single beacon.

To account for the specifics of individual beacons and differences in their APIs, the query execution is, in fact, a processing pipeline consisting of four stages – conversion performed by the Converter subsystem (see Section 3.6), request construction addressed by the Requester subsystem (see Section 3.7), data retrieval done by the Fetcher subsystem (see Section 3.8), and response parsing executed by the Parser subsystem (see Section 3.9). A processor is the computational unit executing this pipeline. This data flow is shown in Figure 3.4.

Due to the differences between implementations, each beacon needs an individually crafted pipeline with a converter, a requester, a fetcher, and a parser behaving in a specific way. The processor needs to be able to know how to construct this pipeline for its beacon. As there are many beacons in the network and even more combinations of behaviors of pipeline stages, we needed to come up with a more flexible solution than an assignment of statically constructed pipelines to beacons. Moreover, the behavior of the pipelines might even depend on the query parameters themselves, which gets too unwieldy for common mechanisms of static configuration. Essentially, we were looking for a way to store business logic in the database.

Instead of a naive approach involving implementation of a large set of switches carefully crafting the behavior of a few general methods, we addressed this issue by implementing converters, requesters, fetchers, and parsers as separate classes. Each class can contain arbitrary Java code executing a particular pipeline stage, thus allowing for maximum flexibility. All the classes in the same category, such as converters, implement the same interface, which typically requires the use of Java's generic interfaces and type parameters, thus allowing for strict type checking.

When a beacon is registered, its converter, requester, fetcher, and parser are specified, and the references to their classes become a part of the beacon's database record (see the *Beacon* table in Figure 3.3). Before the processor triggers the query execution,

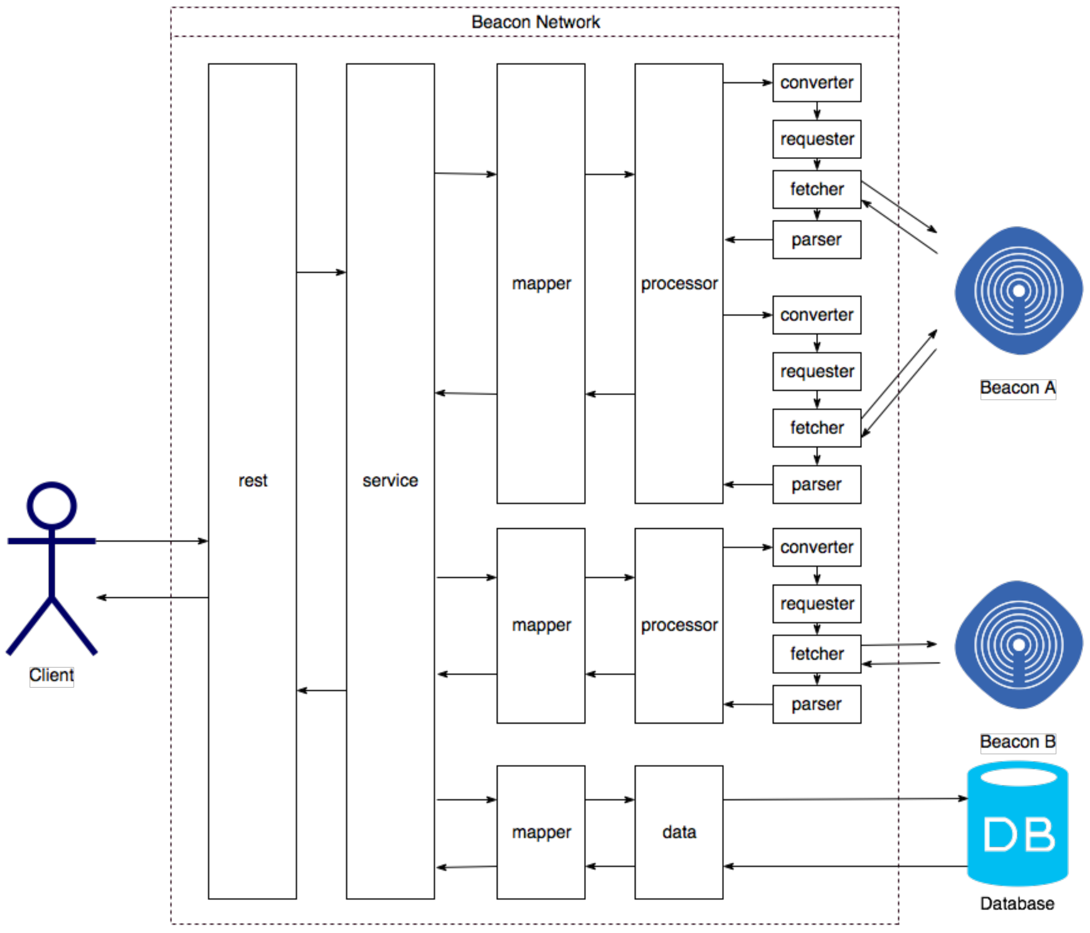


Figure 3.4: Data flow during a query execution in the Beacon Network. The diagram shows a query involving 2 beacons – *Beacon A* hosting 2 datasets/reference genomes, and a single-dataset *Beacon B*.

it fetches the beacon record, resolves the types of the implementations of the pipeline stages, loads them, and assigns them to the pipeline.

The Processor subsystem supports two mechanisms for resolving and loading pipeline stages. The choice is made based on whether the class that needs to be loaded is an EJB or a CDI bean. CDI, or Contexts and Dependency Injection, is a Java EE specification defining a set of complementary services that help improve the structure of application code, such as a lifecycle for stateful objects bound to lifecycle contexts, a type-safe dependency injection mechanism, an event notification model, and a service provider interface allowing portable extensions to integrate with the container. Various kinds of objects are injectable, also known as beans, such as EJB 3 session beans, managed beans and Java EE resources [21]. The Processor subsystem, as well as all the other subsystems, uses CDI 1.1 and its reference implementation, Weld[7], provided by the runtime.

EJBs, such as fetchers, are looked up via Java Naming and Directory Interface

7. Weld. `http://weld.cdi-spec.org`. Accessed on December 22, 2015.

(JNDI)[8] in the application's namespace. Plain CDI beans, such as converters, are looked up by the canonical name of their class, and instantiated using CDI's bean manager.

This dynamic model of pipeline stages can be easily extended. As the Beacon Network bundles a large number of converters, requesters, fetchers,and parsers, new beacons will likely be able to compose their pipeline from the existing elements. If that were not the case and a beacon required e.g. a new converter, we could easily write a new implementation of the converter interface. The implementation module of the Converter subsystem would be the only module requiring recompilation, no other units of code would be affected.

The Processor subsystem adds another level of parallelization on top of the parallelization implemented in the Service subsystem. The Beacon Network is capable of executing cross-assembly beacon queries to provide truly global, all-inclusive results. However, this functionality is not supported by the beacon specification. Many beacons host several datasets, sometimes with different reference genomes, and in practice, we have seen beacons hosting composite datasets with data stored with respect to multiple assemblies. In cases like these, the Beacon Network needs to submit several requests against a beacon to be able to answer a query. This is done asynchronously and in parallel by the processor, which itself is implemented as an asynchronous EJB.

3.6 Converter

The Converter subsystem is in charge of translating query parameters from the system's canonical internal representation to a form a particular beacon understands. It is the first stage in the query execution pipeline run by the Processor subsystem.

There are currently five types of converters – chromosome converters processing chromosomes, position converters changing coordinate bases, reference converters translating assembly names, allele converters taking care of nucleotide strings, and beacon converters computing a beacon identifier from a beacon representation.

Looking at the Beacon API in Sections 2.7 and 3.17, one might wonder why there are no dataset converters. The reason is that the Beacon Network treats beacons and datasets the same way, and thus this functionality is handled by the beacon converters. The standardization of datasets, or rather the lack of it, is actually a major issue, which caused differences in early beacon implementations. Firstly, there is no requirement on the atomicity of the datasets – while some beacons have a different dataset for each patient or sample, many beacons provide large composite datasets consisting of data from various sources. Secondly, a dataset is simply a collection of data, and so is a beacon, in a way. In fact, many currently available beacons have no notion of datasets and simply function as standalone units. An example of this phenomenon is SolveBio, an organization exposing tens of single-dataset beacons. As a result, we decided to merge the two concepts – beacons with several independent datasets are exposed as a set of beacons with an aggregator in the Beacon Network.

The API module of the Converter subsystem contains interfaces for all five types of converters and one parent interface necessary for reliable type checking during lookup. The interfaces define a single conversion method, which is all that needs to be provided

8. Java Naming and Directory Interface. `http://www.oracle.com/technetwork/java/jndi`. Accessed on December 22, 2015.

by the implementation. The implementation module currently provides 14 converters capturing nuances of parameters of current beacons. The converters are implemented as CDI beans.

3.7 Requester

The Requester subsystem is responsible for constructing beacon requests based on their URIs[9] and parameters produced by the converters. It is the second stage in the query execution pipeline run by the Processor subsystem.

The API module of the Requester subsystem provides an interface with two methods for constructing the query URL with parameters and the payload that needs to be sent. The implementation module bundles seven requesters based on various sets of parameters beacons require. The requesters are implemented as CDI beans.

3.8 Fetcher

The Fetcher subsystem is the unit actually talking to the API of beacons. It is responsible for submitting requests over the network and obtaining the raw response. It is the third stage in the query execution pipeline run by the Processor subsystem.

The Fetcher subsystem is likely the most stable of all the subsystems involved in the query execution pipeline and beacons are very unlikely to require a new fetcher. The API module defines a simple single-method interface while the implementation module provides two basic fetchers – a fetcher submitting an HTTP GET request with the necessary headers and the parameters from the requester, and a fetcher submitting an HTTP POST query request with the payload from the requester. Both implementations take care of HTTP and network errors, such as timeouts.

Fetchers are implemented as asynchronous stateless EJB session beans. The actual network calls are performed using Apache HttpComponents[10], an open-source set of low-level tools focused on HTTP and associated protocols, and Apache HttpClient in particular, the component providing basic client-side functionality for accessing resources via HTTP.

3.9 Parser

The Parser subsystem takes care of extracting the information of interest from the raw response obtained by a fetcher. It is the last stage in the query execution pipeline run by the Processor subsystem.

There are currently two types of parsers – parsers of beacons' yes/no response, and parsers of links to external systems, where users can see more information about a queried variant.

The API module of the Parser subsystem contains interfaces for all the parser types. In order to create a parser, a single method needs to be implemented. The implementation module bundles 14 parsers, which deal with various formats, such as

9. RFC 3986 – Uniform Resource Identifier (URI): Generic Syntax. `https://www.ietf.org/rfc/rfc3986.txt`. Accessed on December 22, 2015.
10. Apache HttpComponents. `http://hc.apache.org`. Accessed on December 22, 2015.

JSON, HTML, or arbitrary plain text. These parsers are capable of extracting values from various fields in the response, dealing with multiple dataset responses in the same document, as well normalizing non-standard response values. For example, a beacon provided by the Wellcome Trust Sanger Institute[11] returns five different plain-text responses – *yes* (allele observed in datasets), *no* (allele not observed in datasets), *ref* (this is the reference allele at the given location), *error* (error retrieving data from a data source), and *invalid* (one of the parameters passed is invalid, the location supplied is invalid, and/or one of the required parameters is missing), which are ultimately mapped to a Boolean field.

Parsers are parallelized and implemented as asynchronous stateless EJB session beans.

3.10 Mapper

The Mapper subsystem sits between the Service subsystem on one side and Data and Processor subsystems on the other side. It handles translation between different representations of objects.

The system uses the Data Transfer Object (DTO) design pattern. DTO is a serializable plain old Java object (POJO) encapsulating business data, the purpose of which is to transport data to the client. DTOs are objects directly serialized and exposed through the REST API. In general, everything above the service layer API uses DTOs, while the lower layers use JPA-aware entities. This helps us design our application as a set of decoupled components, where the higher layers of the system and the client are not aware of how the data is stored, and are not affected by any changes in the lower application layers. This provides a lot of flexibility – the API can expose objects different from our data model to minimize the number of network calls the client needs to make, without putting any constraints on how the data is actually stored.

The API module of the Mapper subsystem contains DTOs representing all the basic domain objects as well as interfaces for translation of DTOs to entities and back. The implementation module provides mappers for all the domain objects as CDI beans.

3.11 REST

The REST subsystem provides the top, presentation layer of the application exposing the API of the Beacon Network consumable by the client. This is the subsystem a user of the system interacts with.

REST in the name of the subsystem refers to Representational State Transfer, an architectural style for distributed hypermedia systems. REST emphasizes a few basic mandatory constraints – client-server model (separation of concerns), stateless communication (self-contained requests not requiring storage of any client context on the server side), caching (marking responses as cacheable or not cacheable), layered system structure (hierarchy of components, which cannot see beyond the immediate layer with which they are interacting), and uniform interface. Uniform interface between components is a central feature of REST guided by four interface constraints – identification of resources (individual resources are identified in requests and are separate from their

11. Sanger Institute. `http://www.sanger.ac.uk`. Accessed on December 22, 2015.

representations), manipulation of resources through representations (resource representations contain enough information to modify a resource), self-descriptive messages (messages define how they should be processed), hypermedia as the engine of application state (state transitions for clients are dynamically identified as part of hypermedia by the server) [36].

All these constraints make REST particularly suitable for machine-readable APIs, and the stateless nature of the protocol allows such systems to scale nicely. In practice, resources are typically identified using URIs, accessed over HTTP, with operations triggered using various HTTP verbs. Responses are typically represented in JSON or XML[12] formats (or other valid internet media types) and include metadata expressed through HTTP headers.

The Beacon Network is REST-compatible, or RESTful, and exposes JSON and XML APIs, depending on the request headers. To serialize objects into these formats, the subsystem uses Java Architecture for XML Binding (JAXB) 2 specification from the Java EE stack. JAXB specifies an XML-based data-binding facility consisting of a schema compiler binding a source schema to a set of schema-derived program elements, a schema generator mapping a set of existing program elements to a derived schema, and a binding runtime framework responsible for unmarshalling (reading)/marshalling (writing) of content objects from/to XML documents [58]. To be able to provide both XML and JSON APIs, the REST subsystem uses EclipseLink MOXy[13] implementation of JAXB and its XML and JSON bindings.

The implementation of the REST subsystem is based on JAX-RS 2 specification from the Java EE stack. JAX-RS defines a set of Java APIs for the development of web services built according to REST. It is based on the notion of resource classes, POJOs annotated to implement a web resource, which use resource methods to handle requests [60]. The Beacon Network uses RESTEasy[14] JAX-RS implementation, which is provided by the WildFly runtime.

The API module of the REST subsystem defines methods for manipulation of various domain objects and maps them and their arguments to particular HTTP methods and request parameters. The overview of the API is provided in Table 3.1. The implementation module supplies the actual resources for their respective endpoints, and takes care of object serialization into the requested format, setting up the response headers (such as to enable cross-origin resource sharing), extracting information from the request headers for the purpose of logging etc. The module also provides a set of comparators for ordering elements in responses and a universal exception handler translating uncaught exceptions to the JSON or XML representation of errors.

3.12 Testing and monitoring

Automated tests are a necessary part of modern software development. Beacon Network provides an extensive test suite with unit, integration, and functional tests.

As the Beacon Network is an enterprise application, we built our test suite on top

12. Extensible Markup Language. `http://www.w3.org/XML`. Accessed on December 22, 2015.
13. EclipseLink. `https://eclipse.org/eclipselink/#moxy`. Accessed on December 22, 2015.
14. RESTEasy. `http://resteasy.jboss.org`. Accessed on December 22, 2015.

Method	**URI**	**Description**
GET	/	Lists endpoints.
GET	/beacons	Lists beacons.
POST	/beacons	Creates a beacon.
GET	/beacons/{beacon}	Shows a beacon.
PUT	/beacons/{beacon}	Updates a beacon.
DELETE	/beacons/{beacon}	Removes a beacon.
GET	/organizations	Lists organizations.
POST	/organizations	Creates an organization.
GET	/organizations/{organization}	Shows an organization.
PUT	/organizations/{organization}	Updates an organization.
DELETE	/organizations/{organization}	Removes an organization.
GET	/responses	Queries beacons.
GET	/responses/{beacon}	Queries a beacon.
GET	/chromosomes	Lists supported chromosome.
GET	/alleles	Lists supported alleles.
GET	/references	Lists supported reference genomes.

Table 3.1: REST API of the Beacon Network. All URIs are relative to the root of the API (/api). Parameters are omitted for simplicity.

of Arquillian[15], the go-to framework for testing Java EE applications. The framework supports multiple test runners, such as JUnit[16], the de-facto standard for developing unit tests in Java [104]. Arquillian addresses the challenge of testing business components, where a vanilla unit test is often not sufficient. Components in an enterprise application rarely perform strictly self-contained operations – instead, they interact with a greater system, often through declarative functionality applied at runtime, such as dependency injection and transaction control [6]. Arquillian allows us to utilize and test this functionality in a container-managed environment with CDI support. In a way, the framework brings unit and integration testing together.

Our system uses Arquillian's JUnit runner with Weld CDI implementation provided by WildFly. This environment is very similar to our production setup, which allows us to test more reliably and discover relevant bugs early.

Arquillian does not only take care of the runtime management, but also allows us to create minimal archives containing only the classes related to our tests, which can then be deployed automatically before the actual test execution. This is accomplished through ShrinkWrap[17], Java API for archive manipulation.

Our tests not only cover the integration with the container, they also test communication with external services, such as the database. For testing the persistence layer, we take advantage of Arquillian's extensibility and utilize its Transaction[18] and

15. Arquillian. `http://arquillian.org`. Accessed on December 22, 2015.
16. JUnit. `http://junit.org`. Accessed on December 22, 2015.
17. ShrinkWrap. `http://arquillian.org/modules/shrinkwrap-shrinkwrap`. Accessed on December 22, 2015.
18. Arquillian Transaction Extension. `http://arquillian.org/modules/transaction-extension`.

Persistence[19] extensions. The Transaction extension allows for declarative transaction control over database operations otherwise performed at the service layer of the application. The Persistence extension integrates DbUnit[20], a JUnit extension taking care of putting the database into a known state between test runs, which helps decouple the tests and avoid situations where a test corrupts the database for other tests. The extension also takes care of seeding the database from JSON files, which allows us to decouple the data initialization from the test logic.

All our tests use AssertJ[21], a fluent alternative to standard JUnit assertions, which improves code readability and maintainability.

Our unit/integration tests are tied to a specific component of the system in order to test its behavior. They use Arquillian in in-container mode, which gives us the ability to communicate with the test, enrich it, and execute it in the remote container. On the other hand, our functional test suite covers end-to-end functionality of the application. It uses Arquillian in client mode – the tests run in their Java virtual machine (JVM) and test the container from the outside, as the clients see it. In other words, the tests essentially simulate a client executing a request against each REST endpoint and reading the response in order to check the request was processed properly as it was propagated through the application stack. Our functional tests also use ShrinkWrap, Apache HttpComponents, Apache Log4j, and JAXB.

As the number of beacons in the network grew and new versions of the API were proposed, a need for monitoring of participating beacons arose. Beacons started updating their APIs without any obligation to notify the Beacon Network of the changes. Small beacons were typically experimental projects of organizations running on relatively unreliable servers, and we have seen many of them go down fairly frequently. We decided to periodically check for the API changes of beacons in order to keep delivering correct results, as well as monitor their availability as an additional service for beacon developers.

We implemented this functionality by parameterizing our functional test suite and extending it to support an external database of queries. Typically, parameterized tests are implemented using a parameterized JUnit runner. This however, was not an option for us, due to our dependency on Arquillian and its JUnit runner. We simulated this effect by providing our own implementation of a method rule capable of evaluating parameters. We filled in the database with 15-18 queries per beacon, each testing beacon's handling of a specific API detail, such as a positive/negative response, missing or invalid parameters, or indels. The test suite also chooses parameters to test multi-beacon queries. Monitoring tasks are run on a nightly basis using Jenkins, currently the most popular modern continuous integration tool [95].

3.13 Beacon development kits

Institutions often have the data and are willing to share it, but do not have the technical expertise to do so, or simply resources to invest into a demonstration project like

Accessed on December 22, 2015.

19. Arquillian Persistence Extension. `http://arquillian.org/modules/persistence-extension`. Accessed on December 22, 2015.

20. DbUnit. `http://dbunit.sourceforge.net`. Accessed on December 22, 2015.

21. AssertJ. `http://joel-costigliola.github.io/assertj`. Accessed on December 22, 2015.

beacons. To make it easier for developers to light beacons on top of their data, the Beacon Network provides a set of quick-start applications. We call these applications beacon development kits (BDKs).

BDKs are distributed with the Beacon Network and provide a template with a compliant beacon service the Beacon Network can talk to. The code provides the API and clearly marks the places where beacon-specific functionality should be plugged in. Essentially, BDKs can be viewed as sample, dummy beacons.

The code base of our system includes three BDKs written in Java, Python[22], and JavaScript[23]. The Java BDK is the most advanced of our BDKs and provides a REST API, a sample beacon implementation, conversion of parameters to a normalized form, a sample navigation web page and a test suite. The implementation requires a Java EE runtime and uses CDI, JAX-RS, JAXB, and Arquillian with ShrinkWrap. The Python BDK is built using Flask[24] web framework. The sample beacon bundles an embedded web server for easy setup. The JavaScript BDK is built using Node.js[25] JavaScript runtime and Restify[26] module for building RESTful web services. An embedded web server is provided as well.

Several beacons in the network are currently based on our BDKs, particularly the Java BDK.

3.14 Client

The Beacon Network comes with a thin JavaScript client providing a user-friendly interface to access beacons. The client in its current form was contributed by DNAstack, and thus is not a part of the system as presented in this work. However, the client is a primary driver of traffic to our system, and as such, it deserves to be mentioned.

The application provides a nice query interface allowing the users to send requests to beacons and list their responses asynchronously as they come in. A user can target a specific set of beacons and filter their responses on the client side. A list of participating beacons together with the map of their organizations and a list of recent news and publications related to the Beacon Network is provided as well.

The client is implemented using AngularJS[27] framework. Its user interface is shown in Appendix D.

The production instance is available at `https://beacon-network.org/`.

3.15 Deployment

Stability, scalability, and maintainability of a system is ensured not only through development, but also operations. Our system is deployed in cloud environment based

22. Python. `https://www.python.org`. Accessed on December 22, 2015.
23. JavaScript. `https://www.javascript.com/`. Accessed on December 22, 2015.
24. Flask. `http://flask.pocoo.org`. Accessed on December 22, 2015.
25. Node.js. `https://nodejs.org`. Accessed on December 22, 2015.
26. Restify – API guide. `http://restify.com`. Accessed on December 22, 2015.
27. AngularJS – Superheroic JavaScript MVW Framework. `https://angularjs.org`. Accessed on December 22, 2015.

on Amazon Web Services (AWS)[28], specifically Elastic Compute Cloud (EC2)[29]. This allows us to quickly scale capacity and balance the load in an elastic way as more users and beacons join our system. The service is also highly reliable – Amazon EC2 Service Level Agreement commitment is 99.95% availability for each region.

To simplify maintenance and increase portability of the system, the application runs in Docker, a platform for developers and sysadmins to develop, ship, and run applications. Its engine is lightweight and powerful open-source container virtualization technology combined with a work flow for building and containerizing applications [29]. The Beacon Network is distributed as a Docker image and runs in an isolated container on an EC2 host.

In order to operate, the Beacon Network actually needs a set of containers – a database container, and one or more application containers with a Java EE runtime. To manage and orchestrate the containers, we use Docker Compose[30], a tool for defining and running multi-container Docker applications. This allows us to configure the services in a single file and start them with a single command, thus reducing the deployment time of the system.

The production instance of the Beacon Network is available at `https://beacon-network.org/api/`.

3.16 Security and privacy

The Beacon Network is a service and as such, security is an important part of its operation. However, its API is public, and allowing access to anonymous users is an important system's feature. Since no authentication and authorization is in place, the security model of the application is relatively simple compared to other systems.

As explained in Section 3.15, the system is running in AWS, which is responsible for its physical security. AWS offers highly secure data centers utilizing state-of-the art electronic surveillance and multi-factor access control systems. Data centers are staffed 24x7 by trained security guards, and access is authorized strictly on a least privileged basis. Environmental systems are designed to minimize the impact of disruptions to operations [78]. AWS also takes on additional responsibilities in infrastructure management and is very focused on managing business continuity, storage and network security [9]. In fact, the platform can provide environment compliant with guidelines and best practices set by government and grant funding agencies such as the NIH [84].

While AWS takes care of the security of its hosts, the guest operating system is under our control. The operating system is Red Hat Enterprise Linux[31] with the SELinux[32] security enhancement configured to block actions that were not explicitly authorized. The operating system was set up according to [107] with standard security practices in mind. Access to the guest system as well as AWS management console is secured and restricted to a small group of people. The system is protected by a firewall allow-

28. Amazon Web Services. `http://aws.amazon.com`. Accessed on December 22, 2015.
29. Elastic Compute Cloud – virtual server hosting. `http://aws.amazon.com/ec2`. Accessed on December 22, 2015.
30. Docker Compose. `https://www.docker.com/docker-compose`. Accessed on December 22, 2015.
31. Red Hat Enterprise Linux. `https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux`. Accessed on December 22, 2015.
32. SELinux. `http://selinuxproject.org`. Accessed on December 22, 2015.

ing inbound traffic only through explicitly selected and necessary ports and protocols. The rules of the firewall on the guest systems are automatically updated to block IP addresses showing malicious signs, such as multiple failed log-in attempts.

AWS provides instance isolation. We provide another level of isolation through Docker containers. The database and the application run in distinct secured containers. The database itself is secured and only accessible from the host machine. Traffic is only allowed on the necessary ports and the database port is bound to the container with the application through Docker Compose. The application runs inside a Java EE runtime in a JVM and benefits from the security and maturity of both. Memory limitations and thread pool sizes are carefully configured so that the system does not crash under heavy load. When the limits of the system are reached, the nature of the problem is logged for further investigation. The communication with the outside world is encrypted, thus effectively preventing man-in-the-middle type of attacks. The Beacon Network provides an SSL certificate and its API is only accessible through the HTTPS[33] protocol. The container with the application is running behind an instance of Apache HTTP Server[34], which is further configured to limit potential attacks through careful setup of permissions, timeouts, request limits, and simultaneous connections. In combination with the firewall rules, JVM, and runtime restrictions, this provides us with basic protection against common attacks, such as denial of service.

The security of our system is also supported via thorough testing, as shown in Section 3.12. The software is built from signed Maven artifacts downloaded from trusted repositories using encrypted connection to lower the risk of incorporating malicious external libraries. For transparency, a public mailing list is in place to inform users and beacon operators of known vulnerabilities and other problems.

The availability of our system is checked periodically. In case of corruption, Docker and AWS allow us to replace a container or an instance quickly from backup.

Beacons adhere to the technology standards and operational guidelines described in the Security Infrastructure [47] and Framework for Responsible Sharing of Genomic and Health Related Data [64]. As the Beacon Network wraps the API of existing beacons without storing any extra genomic data, it easily conforms to the same principles.

The Beacon Network features extensive logging set up not only for problem diagnostics, but also for auditing purposes. This allows us to retroactively analyze an action that took place in the system. Proactively, the system does not impose any limits on the number of queries beyond the tools described in the previous paragraphs. Instead, this responsibility is shifted to individual beacons. Some beacons, such as Mike Lin's personal beacon, chose to impose a limit on the query frequency. In cases like these, our system does not have any special privileges and has to respect the same limits as any other clients. This ensures the security risk is not greater when the data is accessed through our system, and a beacon's compliance with security standards is not broken.

As the Beacon Network does not host any genomic data on its own, privacy concerns are primarily applicable to individual beacons. Our system would never give a potential attacker access to more data than they could get by querying the beacon directly. It does, however, provide unified API on top of beacons, and thus can be used as a proxy

33. RFC 2818 – HTTP Over TLS. `https://tools.ietf.org/html/rfc2818`. Accessed on December 22, 2015.
34. Apache HTTP Server Project. `https://httpd.apache.org`. Accessed on December 22, 2015.

in a potential attack. In fact, the ADAD attack described in Section 2.9.1 used the Beacon Network for this purpose.

To lower the risk of privacy breaches through ADAD, participants in the network can be aggregated into virtual beacons which represent a group of beacons. This allows us to hide the individual services and expose them collectively as a bigger dataset, which makes re-identification of an individual harder. In fact, the system provides a virtual beacon for this purpose (Conglomerate), and some beacons chose to join it. This also disassociates the data from the beacon's phenotype, thus reducing the value of information that can be obtained through ADAD to a minimum.

In order to prevent ADAD, the system's terms of use explicitly prohibit re-identifying of individuals whose data is possibly served through the service [48].

3.17 Beacon API

As shown in Section 2.7, early beacons did not provide mutually compatible APIs despite the simplicity of the beacon definition. When the Beacon Network was first put online, the existing beacons already exposed different interfaces, and divergence increased as new beacons showed up. The incompatibility was typically caused by differences falling in the following categories:

- Request method.

  As web services, beacons typically accept queries over the HTTP protocol. However, they tend to support different HTTP methods, usually GET, POST, or both.

- Supported parameters.

  Aside from the core set of parameters, i.e. chromosome, position, allele, and assembly, beacons often support other mandatory or optional parameters, e.g. reference bases (see Section 2.2), or output format.

- Parameter names.

  Names for core parameters are not standardized, e.g. *chromosome/reference*, *position/coord*, or *assembly/genome*.

- Chromosome identifiers.

  Beacons use various identifiers for chromosomes. Prefixes, such as *chrom* or *chr* are often attached to the numeric value of the autosomes. X/Y chromosomes can be specified lowercase or uppercase, or are denoted as 23/24, respectively.

- Positional base.

  Some beacons support 0-based positions, while other beacons support 1-based coordinates.

- Assembly notation.

  Some beacons support HG-based identifiers, such as HG19, while others prefer GRCh-based nomenclature, such as GRCh37. Rarely, beacons do not support parameters denoting the reference genome, and automatically use the assembly the data is associated with.

- Supported alleles.

  Some beacons support queries with the atomic nucleotides, while others support an arbitrary string of nucleotides. In general, the query format for indels is not standardized.

- Dataset support.

  While some services support multiple datasets and allow the user to query a specific one, others have no concept of datasets and support a single data source.

- Response format.

  Responses are provided in various formats – plaintext, HTML, XML, or JSON.

- Data included in the response.

  Aside from the basic yes/no responses required by the beacon specification, beacons optionally disclose other information about the data, such as allele frequencies.

The situation did not get much better with the introduction of the 0.1 API (see Section 2.6). The core API was not mature enough to get significant traction at the time, and many beacons never adopted 0.1 specification. Moreover, 0.1 API suffered from many of the problems of the early, informal beacon definition – it did not specify the request method, the format of the chromosome and reference genome identifiers, supported alleles (particularly the format of indels), and the output format. Moreover, the specification did not clarify how records map the actual API endpoints and noted each parameter and response value as optional, thus making it unclear how to implement queries.

The Beacon Network addressed most of the flaws of the 0.1 specification by providing a unified query interface (see Section 3.11) on top of the APIs of individual beacons. As one of the most prominent applications built on top of the Beacon API, its interface became the basis of the 0.2 specification, the purpose of which was to fix and extend the 0.1 API at the beacon level.

3.17.1 Beacon 0.2

Using the Beacon Network API as a template, we drafted Beacon 0.2 specification and opened it for review and comments. After incorporating feedback from the members of the Global Alliance and addressing major issues of the 0.1 API, the specification was noted in Avro IDL, and became a part of the core API.

Beacon 0.2 specification is listed in Appendix B and defines a protocol with 9 records. *QueryResource* specifies the main beacon query. Aside from changing the names of parameters and making the core parameters mandatory in comparison to the 0.1 API, the record also changes the notation for alleles. Like in the VCF format (see

Section 2.2), a beacon query was updated to contain both reference and alternate bases, which clarifies how to express indels.

A query response is specified via *ResponseResource*. Serialization into JSON is implied. The record echoes the beacon identifier and the query. The query is listed as it is interpreted by the beacon, thus allowing it to point out potential errors made by the user. Both these fields are included for practical reasons – when building an application on top of the Beacon API, an asynchronous library can be used to make requests, and responses can be reconstructed trivially without having to maintain a lot of state.

The main fields in the response are wrapped in a *ResponseResource* record. The original yes/no indication of allele presence in the *exists* field has been extended with an *overlap* value to describe a situation when there is a sequence match on a queried position, but the match is not equal to the queried allele. This is especially useful for indels. Aside from the basic response, optional fields for allele frequency and the number of observed alleles are included. Similarly to the Beacon Network API, an optional *ErrorResource* is included to describe problems, such as an invalid query.

Another idea we took from the Beacon Network API is an endpoint describing a beacon. The metadata is captured by the *BeaconInformationResource*. The record notes a variety of fields, many of which were specified to make it easier for us to add beacons to the Beacon Network. Most importantly, this is where the ID of a beacon and its datasets can be found.

Dataset information is captured by *DataSetResource*. Aside from the basic metadata, a dataset can also describe its size in number of variants and samples via *DataSizeResource* and its data use conditions via *DataUseResource* and *DataUseRequirementResource*. Without prescribing a set of standard categories, this allows a beacon to specify restrictions on how its data can be used. An example could include a research-only data use category with a time limit on data use specified via a data use requirement.

3.17.2 Beacon 0.3

Even though the number of public beacons rose to over 60 during the time of the 0.2 API, we did not see great adoption rate of the specification. Most beacons were very close to being compliant, but chose to implement a certain variation of the specification for pragmatic reasons, or simply never upgraded from 0.1. We also saw several new, previously unanticipated data and fields. Strictly speaking, only one of the currently available beacons would be compliant. In other words, the 0.2 specification did not seem to reflect the needs of the beacon operators.

Learning from our experience with the Beacon Network and its members, we proposed a refined Beacon 0.3 specification. Beacon 0.3 fixes bugs in 0.2, and introduces new fields and features. It serves as a draft to create a basis for the first stable release, Beacon 1.0. With Beacon 1.0, we want to be reasonably confident we can guarantee backward compatibility. With the 0.x release line, breaking changes are allowed, provided they lead to a nice, clean and adoptable version of the API. With this in mind and in order to accommodate naming and structural conventions, we renamed and refactored most of the 0.2 API for the 0.3 release.

For compatibility with the rest of the GA4GH APIs, Beacon 0.3 is specified using Avro IDL. Modularity was an important goal for this release, which lead us to splitting the specification into three protocols – beacons, data use conditions and beacon methods. The draft of the specification is listed in Appendix C and is currently under review.

Beacon protocol introduces seven records. The main query record was renamed to a more accurate *BeaconAlleleRequest* in order to allow for easy naming of possible future endpoints. Its attributes were renamed to match the standard VCF notation (see Section 2.2). The record includes all the fields from its counterpart in the 0.2 API, but it specifies the notations and value domains unambiguously. All the core parameters from the 0.2 API are present and mandatory. Furthermore, beacons now support multi-dataset queries.

Similarly to the 0.2, the main response in the form of *BeaconAlleleResponse* echoes the beacon ID and the query. Due to the support for cross-dataset queries, the response was extended to include a list of responses from the individual datasets specified in the query. Optionally, a beacon can provide a global yes/no answer, similarly to the 0.2 API.

A dataset-level response includes a dataset identifier as well as a standard indicator of allele presence. The number of allelic observations was split into three fields denoting the number of variants, samples and calls. The response can also include allele frequency, a link to an external system providing more information about the given query (such as an alternative beacon behind an authentication server exposing more data to authorized users), a custom note and arbitrary metadata in the form of key-value pairs. In case of an error, a dataset can specify a *BeaconError* value. We chose this custom error record instead of Avro's standard error declaration similar to exceptions thrown in methods in order to be able to provide an error per dataset. For example, a beacon can serve multiple datasets, each with a different assembly. When querying several of these datasets for a particular reference genome, the beacon has to be able to provide valid information for one of the datasets, and specify errors for others.

The main *Beacon* record standardizes more metadata in the 0.3 API and allows for specifying an arbitrary set of non-standard data through key-value pairs. One of the more significant features is an extensible structured record for information about the organization hosting a beacon provided via *BeaconOrganization*, which can be nicely utilized for the implementation of programmatic beacon registration in the Beacon Network.

Extensibility has been a major requirement for the 0.3 API, and beacons can specify arbitrary metadata for their datasets via *BeaconDataset* as well. Among other fields, the dataset record now adds mandatory fields for creation and modification date. This can be utilized by periodic dataset updates and refreshes in the Beacon Network. The record describing the size of the dataset has been replaced by independent dataset-level fields specifying the number of variants, calls and samples.

One of the major issues of the 0.2 API is a rather vague specification of the data use restrictions. In fact, we are not aware of any 0.2 beacons using this part of the API. As a result, we proposed a set of records for consent-based data use conditions. These records are specified as part of a data use conditions protocol separate from the main beacon protocol, so that they can be easily used by the rest of the Core API.

The data use conditions protocol was designed to hold conditions based on consent permissions as found in the datasets of the main public genome archives. This set of consent codes was compiled based on a study conducted by the Global Alliance, which resulted in a system of 19 data use conditions divided into categories and requirements. Requirements are conditions that need some kind of an additional agreement on the part of the data provider or data user, or additional criteria in order for the data to be reused. Data use categories were further divided into primary and secondary categories. All datasets should fall into only one primary category, whereas a number of secondary categories and requirements may also be applied [32]. To provide an example, let's take no restrictions (NRES) as a primary category, research use only (RUO) as a secondary category, and publication required (PUB) as a data use requirement.

The protocol holds the data use conditions in 2 records – *DataUseCondition*, a record containing the abbreviated consent code with an optional description, and *DataUse*, a record wrapping a single primary category and lists of secondary categories as well as requirements. In order to encourage a conscious decision regarding the data use for each dataset, all these fields are mandatory, although the lists of secondary categories and requirements may be empty. Such a data use record is a mandatory field in a beacon dataset record.

A particular issue with the 0.2 specification was the complete absence of the request methods and endpoints. This is fixed in the 0.3 API through the beacon methods protocol, which names the endpoints, specifies HTTP methods, and ties the request records to the response records.

4 Evaluation

"The unfolding calamity in genomics is that a great deal of this life-saving information, though already collected, is inaccessible."

– David Shaywitz, Chief Medical Officer, DNAnexus. Source: [87].

The Beacon Network has been in production since August 2014. In January 2015, we added logging capabilities, which allowed us to track queries submitted to the system. In this chapter, we use the 12 months of data to analyze the usage and the traction of the network in 2015. With the Beacon Network being a global search engine, this provides us with unique information larger than the scope of any single dataset, beacon, or institution.

4.1 Size

As of December 2015, 23 organizations with 64 beacons are participating in the Beacon Network. The beacons are hosted in 7 countries on 4 continents, as shown in Figure 4.1, which makes the Beacon Network a global search engine of genomic data.

Together, the participants of the network expose 160 datasets. The word dataset refers to the notion of a beacon dataset, which can be different from the dataset stored underneath the beacon. In fact, many beacons are known to serve data from a large number of sources, but expose them as a single logical unit. Beacon datasets are not compared across institutions – if the same data is hosted by two beacons from two different institutions, the data is a part of two different datasets, since the metadata associated with it might differ.

In order to illustrate the reach of the Beacon Network, we estimated the total number of samples searched. Since most of the data hosted by the beacons is publicly available, we were able to look up this information for almost all the datasets. Extrapolating from this, we estimate the total number of samples searched by the network to be around 2 million. To our knowledge, this makes the Beacon Network the largest search engine of human genomic variations in the world.

To put the size of the network into perspective, we attempted to compute an estimate for the total number of genomic variants searched. Based on the information available for public datasets, we estimate the total size of the network to be around 2 billion variants. The notion of a genomic variant is similar to the notion of a beacon dataset in terms of uniqueness – a variant is typically unique within a dataset (beacon) and can be contained in multiple samples, but there is no way for us to compare variants across beacons or organizations. In other words, common variants are often exposed in the network several times. For example, several beacons serve data from the 1,000 Genomes Project.

Genomic variants are often covered by multiple samples or calls, an thus often provide several unique data points. We estimate the total number of such call-based data points covered by the network to be around 300 billion.

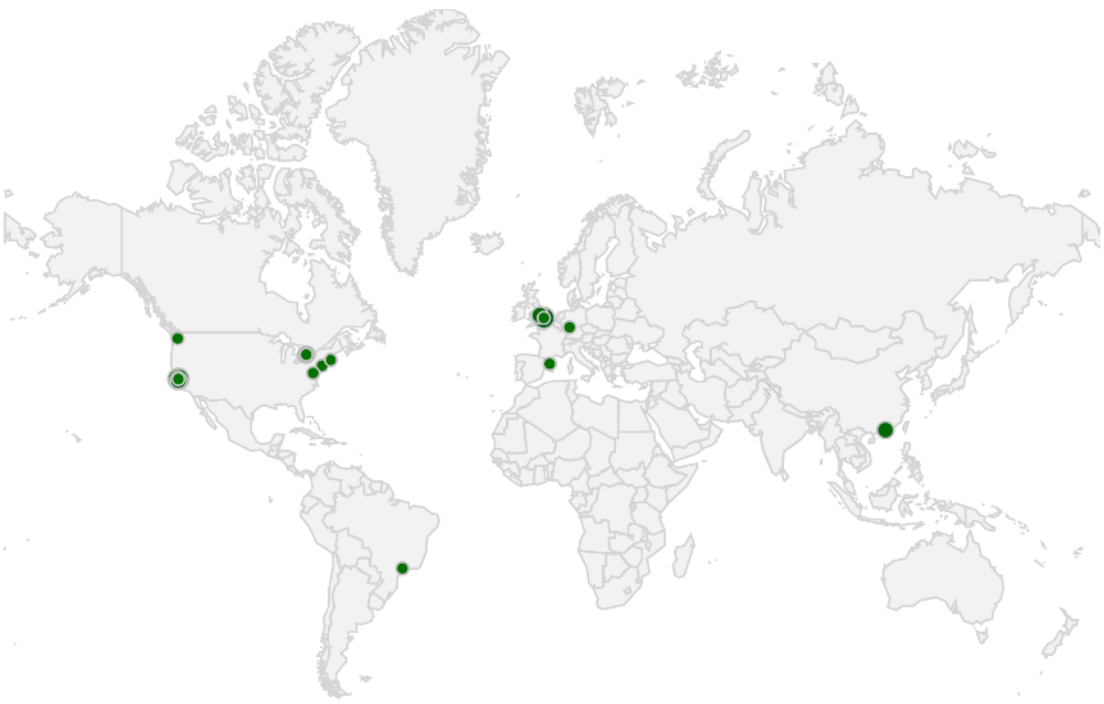


Figure 4.1: Geographic location of beacons in the Beacon Network around the world.

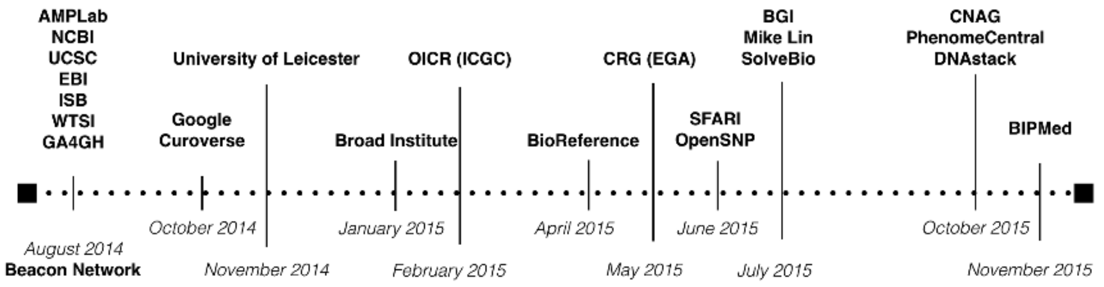


Figure 4.2: Timeline of organizations joining the Beacon Network.

4.2 Adoption

When we first started developing the Beacon Network, four organizations were exposing public beacons, as explained in Section 2.7. By the time the Beacon Network went into production and became publicly available in August 2014, another two organizations and their two beacons had joined the network. Since then, the number of participating organizations has grown to 23, which is captured by the timeline in Figure 4.2.

Over time, the number of participating beacons has risen to 64, which is shown in Figure 4.3.

4.3 Participants

After Algorithms, Machines and People Lab, National Center for Biotechnology Information, University of California, Santa Cruz, and European Bioinformatics Institute,

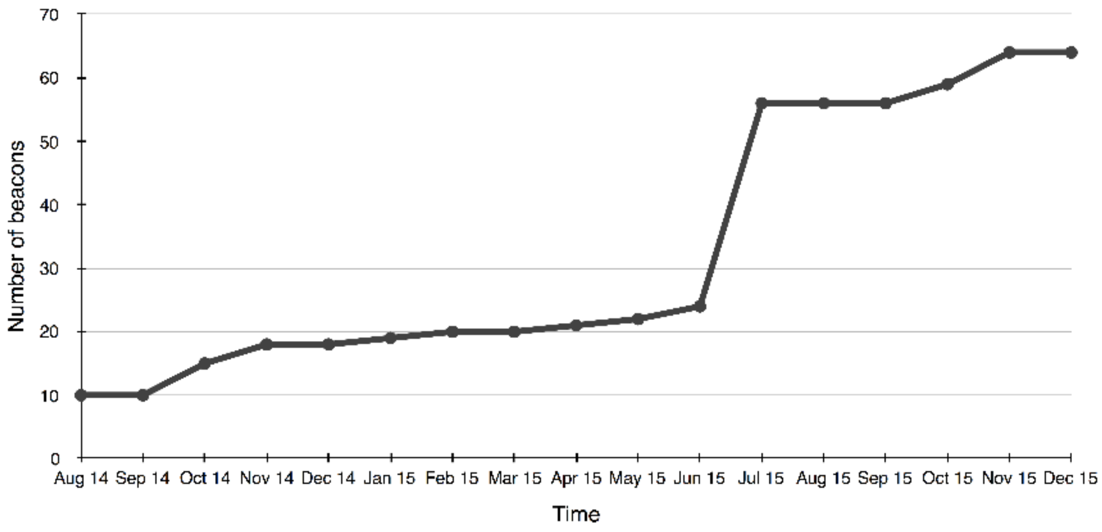


Figure 4.3: Number of beacons in the Beacon Network over time.

the list of beacon providers was extended by the Institute for Systems Biology (ISB)[1]. ISB exposed Kaviar as a beacon.

Wellcome Trust Sanger Institute (WTSI) provides a beacon serving data from whole genomes of Crohn's disease patients, samples from Native American and Egyptian individuals, and UK10K data. At the end of 2015, WTSI added a collection of beacons querying data from the Catalog of Somatic Mutations in Cancer (COSMIC), a database that holds somatic mutation data and associated information related to cancer [11].

When the Beacon Network was released, it imported all the public beacons known to date. In some cases, such as UCSC, it created a set of beacons and an aggregator for the organization. The system also introduced two aggregate beacons on behalf of GA4GH – Beacon of Beacons covering all the other beacons in the network, and Conglomerate, an aggregate of beacons not exposed independently, i.e. beacons which chose to remain anonymous.

Google introduced four beacons (three datasets and an aggregator) based on public data in Google Genomics – 1,000 Genomes data, 1,000 Genomes Phase 3 data, and Illumina Platinum Genomes[2].

Curoverse[3] exposed several individuals from the Personal Genome Project (PGP), an effort to create a standard open collection of genomic data of 100,000 volunteers, which can be used to standardize new tools and enable research into personalized medicine [18].

Bioinformatics Research Group at the University of Leicester[4] runs a beacon for resources in Cafe Variome.

Broad Institute[5] hosts data from ExAC, a collection of exome sequencing data from

1. Institute for Systems Biology. `http://www.systemsbiology.org`. Accessed on December 22, 2015.
2. Platinum Genomes. `http://www.illumina.com/platinumgenomes`. Accessed on December 22, 2015.
3. Curoverse. `https://curoverse.com`. Accessed on December 22, 2015.
4. University of Leicester. `https://le.ac.uk`. Accessed on December 22, 2015.
5. Broad Institute of MIT and Harvard. `https://www.broadinstitute.org`. Accessed on December

various large-scale sequencing projects.

Ontario Institute for Cancer Research (OICR)[6] exposes a beacon for the International Cancer Genome Consortium with the data from a range of cancer projects.

BioReference Laboratories[7] exposed high-confidence variants from samples submitted to a genetic testing company GeneDx[8].

Centre for Genomic Regulation (CRG)[9] hosts data from EGA. The beacon serves 22 datasets related to various conditions. The Beacon Network searches two UK10K datasets, which the beacon exposes publicly.

Simons Foundation Autism Research Initiative (SFARI)[10] exposes a collection of simplex families, each of which has only one individual with an autism spectrum disorder.

OpenSNP[11] hosts thousands of personal genomics datasets consisting of mostly 23andMe SNP genotyping data and a few complete exomes.

BGI Hong Kong[12] joined the network with three beacons set up on behalf of GigaScience[13].

Mike Lin joined the network with a beacon exposing his personal whole-genome sequencing data obtained via Illumina's Understand Your Genome[14] program.

SolveBio hosts several datasets with information related to various populations and diseases. Each genomic dataset is exposed as a beacon, and the company thus contributes 28 beacons to our system.

Centro Nacional de Análisis Genómico (CNAG)[15] hosts a beacon for RD-Connect (see Section 2.10.4). The beacon is based on one of our BDKs described in Section 3.13.

PhenomeCentral is a platform for data sharing within the rare disorder community. Its beacon exposes hundreds of exomes.

DNAstack is a platform for management, sharing, and analysis of genomic datasets. The number of DNAstack beacons varies – the platform allows its users to create beacons from the datasets they uploaded and processed in the system, and public beacons are automatically published to the Beacon Network through its API. The platform is still pending its final release, and we expect a rapid increase of beacons in the Beacon Network once it launches.

Brazilian Initiative on Precision Medicine (BIPMed)[16] joined the system with information on variants identified on Brazilian subjects who belong to the reference population. The beacon is hosted by Instituto de Química, Universidade Estadual de Campinas[17].

22, 2015.

6. Ontario Institute for Cancer Research. `http://oicr.on.ca`. Accessed on December 22, 2015.
7. BioReference Laboratories. `http://www.bioreference.com`. Accessed on December 22, 2015.
8. GeneDx. `http://www.genedx.com`. Accessed on December 22, 2015.
9. CRG. `http://www.crg.eu`. Accessed on December 22, 2015.
10. SFARI. `http://sfari.org`. Accessed on December 22, 2015.
11. OpenSNP. `https://opensnp.org`. Accessed on December 22, 2015.
12. BGI. `www.genomics.cn`. Accessed on December 22, 2015.
13. GigaScience. `http://www.gigasciencejournal.com`. Accessed on December 22, 2015.
14. Understand Your Genome Community. `https://www.understandyourgenome.com`. Accessed on December 22, 2015.
15. CNAG. `http://www.cnag.cat`. Accessed on December 22, 2015.
16. Brazilian Initiative on Precision Medicine. `http://bipmed.iqm.unicamp.br`. Accessed on December 22, 2015.
17. Instituto de Química. `http://iqm.unicamp.br`. Accessed on December 22, 2015.

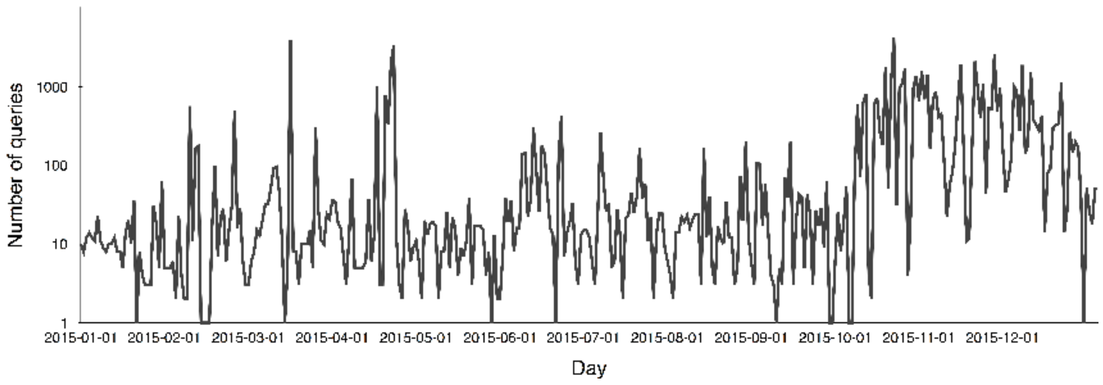


Figure 4.4: Daily number of queries submitted to the Beacon Network over time.

4.4 Queries

Throughout 2015, the Beacon Network's API received over 96,000 API requests, out of which 69.35% (over 67,000 requests) constituted valid beacon queries. The rest of the requests were primarily used to obtain the list of beacons in the network (25.53%).

On average, the system responded to over 183 requests a day. In peak times, however, the API often had to handle 50-90 concurrent requests, with an absolute maximum of 108 so far.

We analyzed the parameters of the beacon queries submitted to the network over time and cross-referenced it with the information about the status of the beacons at the time the query was submitted. Based on this data, we estimate the total number of queries facilitated by the network, i.e. dispatched to individual beacons, to be over 827,000. On average, 12.35 beacons were involved in a query.

Overall, the number of queries the system was receiving was increasing throughout the year, thus proving the increasing traction of the Beacon Network. This is obvious from the daily and monthly number of requests plotted in Figures 4.4 and 4.5, respectively. The data indicates a relatively significant increase in the average number of requests on March 17 and April 17-23, 2015. This traffic originated from Stanford's network and is most likely related to the analysis performed by the authors of [93]. The number of requests the system served also increased in October 2015, and has not dropped since. We believe this is related to the increased popularity of the Beacon Network and the Beacon Project overall, as well as the roll-out of the new version of the Beacon Network client. The new web application queries all the beacons in batches by default.

4.5 Users

Even though our system does not track individual user information for privacy reasons, it keeps some aggregate statistics related to where the queries are coming from.

In 2015, the system was accessed by 1073 unique users. On average, 6.67 different users access the network every day, with peak days reaching up to 80 unique users. In peak times, the network is being accessed by over 10 concurrent users.

59.13% of the visits are generated by new users, whereas 40.87% of visits belong to

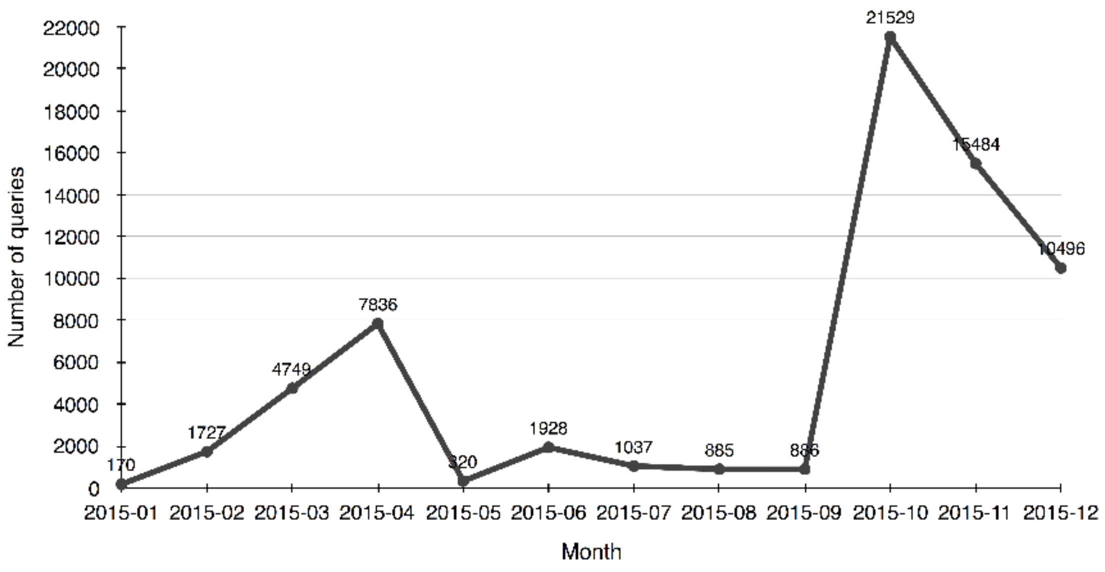


Figure 4.5: Monthly number of queries submitted to the Beacon Network over time.

returning users. This matches our expectations that many people are happy to try out the system, but do not necessarily use it regularly.

During 2015, the network has been accessed by people from 74 countries. The top 10 countries generating the most requests are shown in Figure 4.6, together with how much traffic they generated. Unsurprisingly, 5 of the 7 countries hosting beacons are also featured among the top traffic-generating sites. Interestingly enough, the top two countries accessing the system, USA and Canada, with their 46.23% share of the traffic, host 73.44% of the world's beacons.

4.6 Assemblies

Looking at the parameters of the queries, we can conclude that the vast majority of requests were looking for data with respect to GRCh37 (hg19) reference. This was the case of 74.13% of requests, followed by 9.63% of requests covering GRCh38 (hg20). This is not surprising, since the vast majority of beacons store data with respect to GRCh37. In fact, 92.72% of beacons support it, followed by 27.27% support of GRCh38. As is obvious from the percentage, several beacons support multiple assemblies.

4.7 Chromosomes

The most searched chromosome in the system is chromosome 1, which was covered by overwhelming 42.49% of requests. While this percentage might seem very high at first, it is not completely surprising. Chromosome 1 is the largest of the human chromosomes, containing approximately 8% of all human genetic information. Because of its size, we can expect it to be more representative of the human genome than some other chromosomes with respect to genomic landscape and genetic properties. Chromosome 1 is gene-dense, with 3,141 genes. It is medically important, as over 350 human diseases are associated with disruptions in the sequence of this chromosome – including cancers,

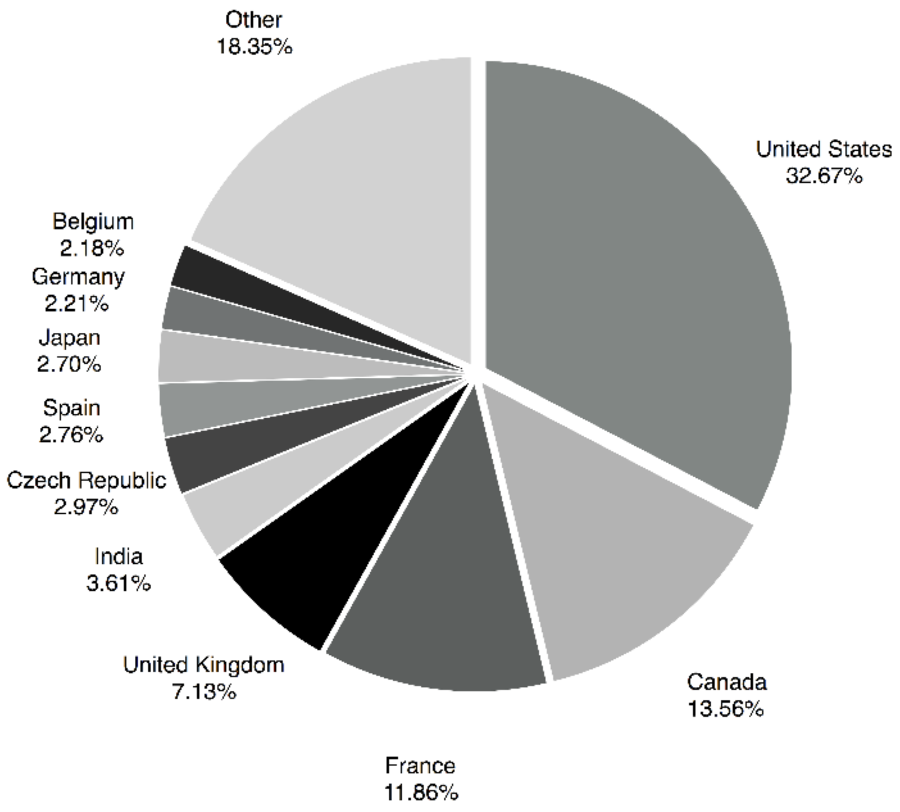


Figure 4.6: Countries that generated the most Beacon Network traffic in 2015.

neurological and developmental disorders, and Mendelian conditions – for which many of the corresponding genes are unknown. There are also important biological implications of the size of chromosome 1 – it is approximately 6 times longer than the smallest human chromosomes (21, 22 and Y), which raises the question of how all human genetic information is replicated in a coordinated manner before each cell division [51].

Being targeted by 21.75% of the queries, chromosome 13 is the second most searched chromosomes in the system. This might seem surprising, since chromosome 13 has relatively low and variable gene density compared to other autosomes. However, it is the largest acrocentric human chromosome. It carries genes involved in cancer including the breast cancer type 2 (BRCA2) and retinoblastoma (RB1) genes, is frequently rearranged in B-cell chronic lymphocytic leukaemia, and contains the DAOA locus associated with bipolar disorder and schizophrenia [31]. We believe there are several factors contributing to the popularity of this chromosome in our searches. Breast cancer is a very popular research direction in genomics, and BRCA2 gene in particular is of great interest to the community. The traction of BRCA Challenge (see Section 2.3.1) certainly helps – in fact, a separate version of the Beacon Network specifically focused on

No.	Chromosome	Position	Allele	Percentage
1	13	32,954,207	A	18.61
2	1	0	C	10.80
3	1	880,237	A	6.55
4	1	69,509	G	2.25
5	1	99,999	C	2.01
6	11	6,892,865	A	1.49
7	1	9,999	C	1.19

Table 4.1: Alleles covering over 1% of queries submitted to the Beacon Network in 2015 (GRCh37).

BRCA1 and BRCA2 was running in production for several months as a demonstration of the alignment of the Beacon Project with the BRCA Challenge. To this day, a query for BRCA2 is an example query in the client of the Beacon Network, and thus new users are likely to execute it. This is further supported by Sections 4.8 and 4.11.

Each of the other chromosomes have been a subject of a significantly lower number of requests. The third most searched chromosome is chromosome 3 covered by 3.86% of requests.

4.8 Alleles

In 2015, the system searched for 14,298 distinct alleles. Alleles constituting over 1% of the requests each are listed in Table 4.1.

The list of the most popular alleles reveals several interesting facts. Firstly, over 18% of the requests target a single variant. This variant is associated with the BRCA2 gene and is used as an example in the client for the Beacon Network. The query probably gets executed often by new users of the system, while they are trying the client out. Returning users might start their session with this query as well, just to fill in the inputs in the query form, which they later modify. The query was added as an example to our client when we rolled out its latest version, i.e. in October 2015. Analysis of the timestamps of the queries shows they were all submitted during or after October 2015, thus confirming our suspicion of the popularity of the query being linked to the fact that it is suggested to users by the client.

In fact, several queries indicate they were typed by users to try out the Beacon Network rather than look for an interesting allele. While the API accepts 0-based positions, the client lets the users type a 1-based position. This implies chromosome 1 and positions 1, 100,000, and 10,000 specified on the client side for queries for variants 2, 5, and 7, respectively. These numbers are likely the first user inputs used to see how the system works.

This leaves us with 3, 4, and 6 as the likely candidates for variants of scientific interest. Query 3 is actually a request for reference allele. The coordinates point to NOC2L gene, NOC2-like nucleolar associated transcriptional repressor representing a histone-deacetylases-independent inhibitor of histone acetyltransferase [55]. Alleles 4 and 6 are well known and found in the Beacon Network. Variant 4 falls into the OR4F5 gene, olfactory receptor family 4 subfamily F member 5. Olfactory receptor proteins interact with odorant molecules in the nose, initiating a neuronal response that triggers the perception of a smell [114]. Allele 6 is a popular mutation triggering a positive response in 17 network beacons. It belongs to the OR10A2 gene, olfactory receptor family 10 subfamily A member 2. Interestingly enough, this variant is associated with cilantro aversion. Although cilantro (also known as coriander or Chinese parsley) is widely used to flavor food around the world, many people are unable to appreciate this herb's fruity qualities and instead taste a mouthful of soap. This particular mutation indicates 1.1 times higher odds of the an individual disliking cilantro [35].

4.9 Deleteriousness

A non-synonymous SNV is a single nucleotide variant that causes an amino acid change of its corresponding protein. Because of their direct link to protein changes, non-synonymous SNVs are believed to be the major contributor to human diseases among all types of variants in the human genome [75]. In fact, non-synonymous variants constitute more than 50% of the mutations known to be involved in human hereditary diseases [67]. Since experimental investigation of the functional effect of variants is too costly and time-consuming, algorithms have been developed to help predict the potential of a variant being functional or deleterious. Different prediction algorithms use different information and each has its own strength and weakness. It has been suggested that investigators should use predictions from multiple algorithms instead of relying on a single one [75].

One of the classic algorithms for functional prediction is SIFT (Sorting Intolerant From Tolerant). SIFT assumes that important positions in a protein sequence have been conserved throughout evolution, and therefore substitutions at these positions may affect protein function. By using sequence homology, it predicts the effects of all possible substitutions at each position in the protein sequence [69].

Another standard prediction algorithm is PolyPhen-2 (Polymorphism Phenotyping version 2). PolyPhen-2 predicts possible impact of an amino acid substitution on the structure and function of a human protein using straightforward physical and comparative considerations. It uses eight sequence-based and three structure-based predictive features selected automatically by an iterative greedy algorithm. Majority of these features involve comparison of a property of the wild-type (ancestral, normal) allele and the corresponding property of the mutant (derived, disease-causing) allele, which together define an amino acid replacement. Most informative features characterize how well the two human alleles fit into the pattern of amino acid replacements within the multiple sequence alignment of homologous proteins, how distant the protein harboring the first deviation from the human wild-type allele is from the human protein, and whether the mutant allele originated at a hyper-mutable site. A set of homologous sequences is selected for the analysis using a clustering algorithm. The functional significance of an allele replacement is predicted from its individual features by Naive Bayes classifier. There are currently two versions of PolyPhen-2 differing in the dataset the algorithm was trained on – HDIV and HVAR. Diagnostics of Mendelian diseases requires distinguishing mutations with drastic effects from all the remaining human variation, including abundant mildly deleterious alleles. HVAR PolyPhen-2 should be

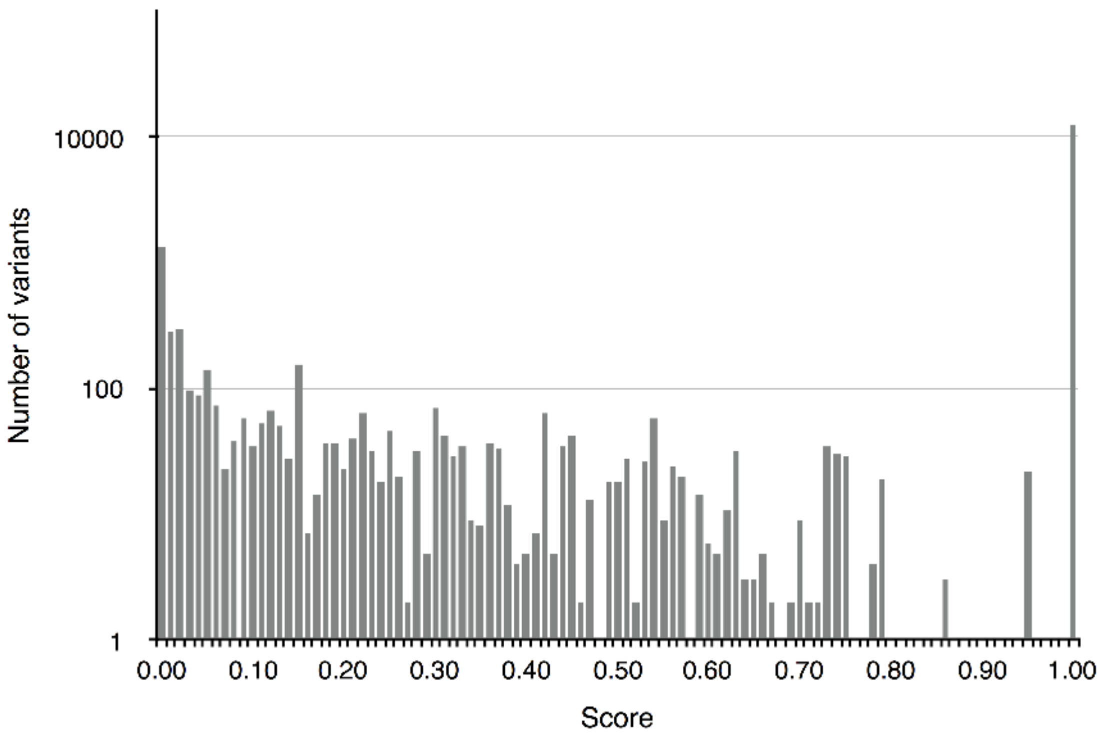


Figure 4.7: Distribution of SIFT scores for variants submitted to the Beacon Network in 2015.

used for this task. In contrast, HDIV PolyPhen-2 should be used for evaluating rare alleles at loci potentially involved in complex phenotypes, dense mapping of regions identified by genome-wide association studies, and analysis of natural selection from sequence data, where even mildly deleterious alleles must be treated as damaging [5].

We analyzed the variants using ANNOVAR, a tool for annotating variants with their functional consequence on genes, inferring cytogenetic bands, reporting functional importance scores, or finding variants in conserved regions [109]. Usage of multiple predictions from multiple algorithms is a good practice – using filter-based annotation, we annotated the variants with SIFT and PolyPhen-2 scores from dbNSFP. As our variants correspond to queries targeting many different datasets, we used both HDIV and HVAR versions of PolyPhen-2, for comparison.

The distribution of SIFT, PolyPhen-2 HDIV, and PolyPhen-2 HVAR scores is shown in Figures 4.7 to 4.9, respectively. The algorithms were not able to rank all our variants, for example due to some of them being intergenic. SIFT provided a score for 16,762 variants, out of which 13.49% were classified as damaging (score less than or equal to 0.05), and 86.51% as tolerated (score more than 0.05). PolyPhen-2 provided a score for 4,259 variants. The HDIV version classified 43.84% of them as probably damaging (score between 0.957 and 1), 23.03% as possibly damaging (score between 0.453 and 0.956), and 33.13% as benign (score between 0 and 0.452). The HVAR version classified 35.03% of variants as probably damaging (score between 0.909 and 1), 18.81% as possibly damaging (score between 0.447 and 0.908), and 46.16% as benign (score between 0 and 0.446).

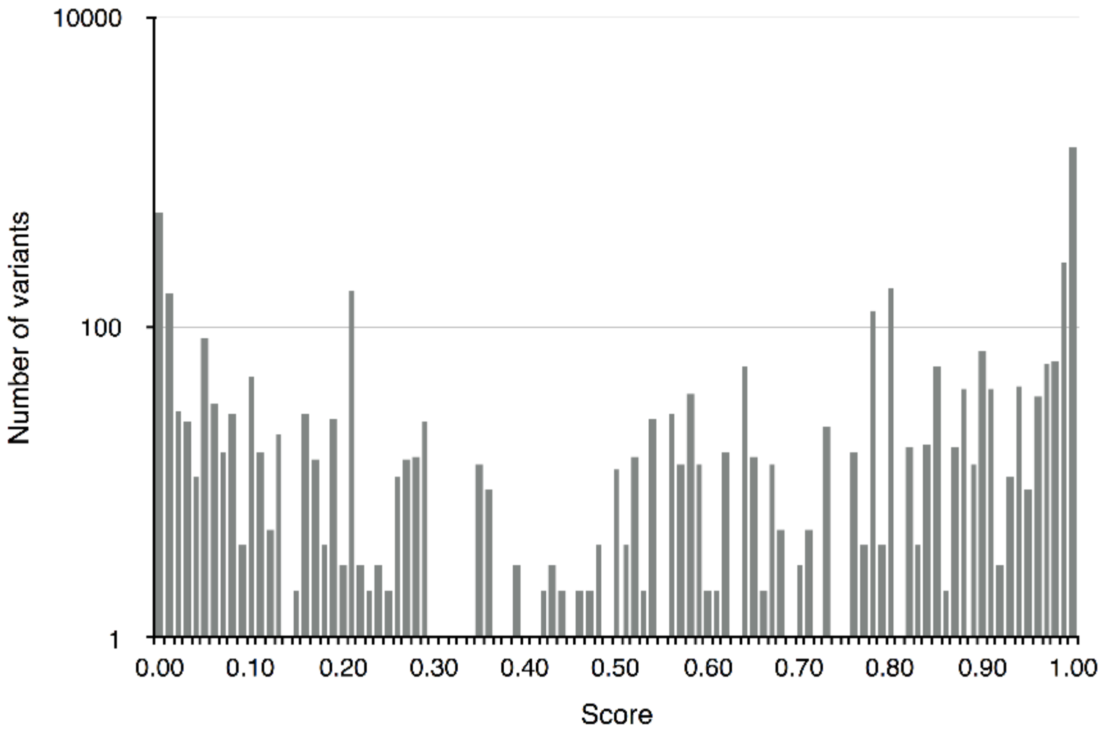


Figure 4.8: Distribution of PolyPhen-2 HDIV scores for variants submitted to the Beacon Network in 2015.

Overall, the relatively high number of damaging variants suggests an interesting use case for the Beacon Network – people use the system to see who else has a particular variant they possibly found in their own dataset and suspect to cause a problem. This validates the system's function as a discovery engine.

4.10 Rarity

The Beacon Network was created with rare diseases in mind. Common variants can usually be found in many public databases – for rare variants, it is important to search across databases and institutions in order to discover where they are and figure out who to contact to get more information. For this reason, we decided to analyze how many variants submitted to the system are actually rare.

We define the rarity of variants using their minor allele frequency (MAF) in a common dataset often used as a reference for genomic studies – 1,000 Genomes Project (October 2014 release). The frequency is computed as a number between 0 and 1 with respect to all the individuals in the project rather than a specific population or cohort, such as African or European. MAF refers to the frequency at which the least common allele occurs in a given population, and rare variants are typically defined as variants with MAF lower than 0.05 [57].

We used ANNOVAR's filter-based annotation to generate allele frequencies. 1,000 Genomes Project contains 8,349 of our variants. 14.82% of them were classified as rare, and 85.18% were described as common. The complete distribution of allele frequencies

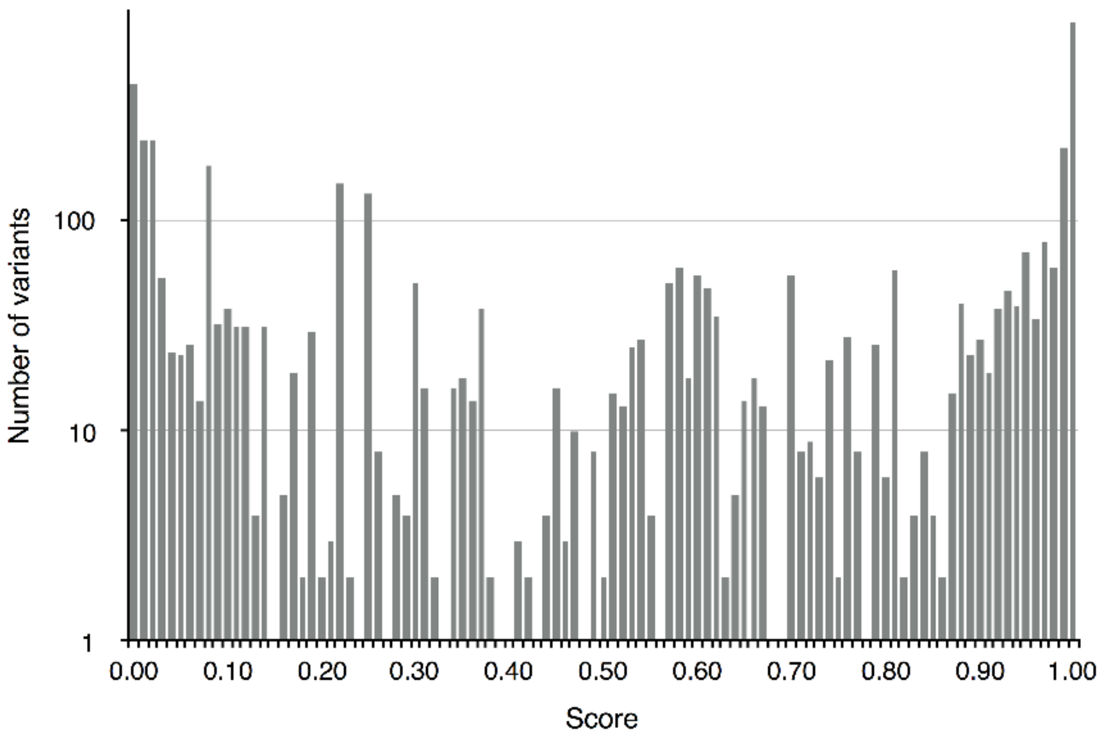


Figure 4.9: Distribution of PolyPhen-2 HVAR scores for variants submitted to the Beacon Network in 2015.

is shown in Figure 4.10.

The fraction of rare variants in our queries is interesting and suggests people do use the Beacon Network for rare disease research. As far as the 1,000 Genomes Project is concerned, over 76% of variants are rare, although only about 1-4% of variants in a single genome tend to be rare [4].

4.11 Genes

Variants, and particularly SNVs, are often found in the DNA between genes. They can act as biological markers and help locate genes associated with a disease. SNVs found in a gene are interesting as they might be directly related to a disease by affecting the gene's function. For this reason, we decided to compute what genes the queries are associated with.

We generated gene-based annotation through ANNOVAR using UCSC's assemblies and refGene, a collection of known human genes from the NCBI reference sequences collection, and aggregated the data across genes.

Overall, 35.69% of all alleles submitted as queries to the system have been classified by ANNOVAR as exonic, i.e. overlapping a coding. It should be noted that exonic here implies a coding exonic region, not the untranslated regions. Furthermore, 60.51% of the variants fall into genes. These percentages are significantly higher than the percentages for random SNVs would be, since only about 1.5% of the human genome consists of coding sequence and only one-third of the genome is transcribed in genes [1]. In other

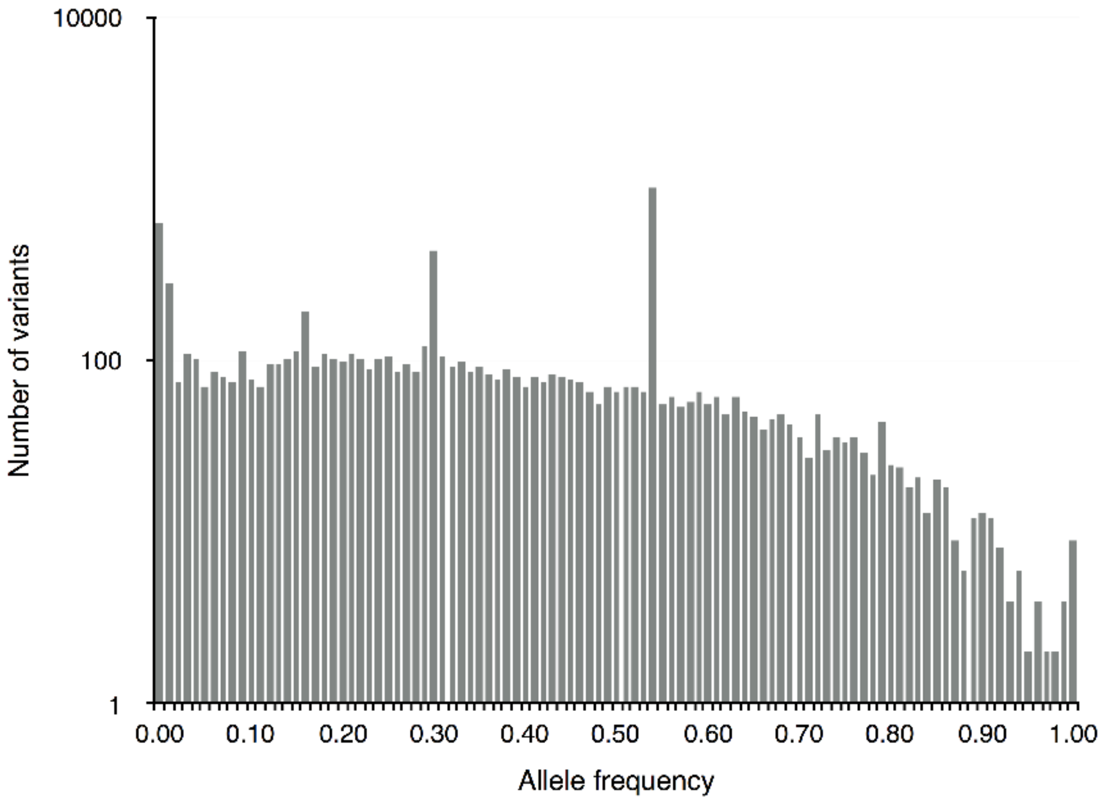


Figure 4.10: Distribution of allele frequencies of variants submitted to the Beacon Network in 2015 in 1,000 Genomes Project (October 2014 release).

words, people use the system to search for "interesting" variants.

In total, 3640 distinct genes were covered by the variants submitted to the Beacon Network. We cross-referenced their symbols with the database of the HUGO Gene Nomenclature Committee (HGNC) responsible for providing unique and informative nomenclature for all genes within the human genome [50]. The genes hit most often are listed together with their HGNC names in Table 4.2.

As explained in Section 4.8, BRCA2 is leading the table primarily due to a sample query in the Beacon Network client. As the rest of testing queries target intergenic variants, the remaining genes in Table 4.2 seem to be of valid research interest.

4.12 Disorders

The Beacon Network can be used as a discovery tool for variants related to certain hereditary diseases, which is why we decided to investigate what phenotypic features variants submitted to the system are linked to.

We cross-referenced annotated variants submitted to our system with OMIM's Morbid Map, which contains cytogenetic locations of disorders. Only OMIM entries for which a cytogenetic location has been published in the cited references are represented in the map.

In total, 2,510 OMIM entries were associated with the queried variants. The ones

No.	**Gene symbol**	**HGNC name**	**Percentage**
1	BRCA2	breast cancer 2	33.58
2	NOC2L	NOC2-like nucleolar associated transcriptional repressor	12.75
3	OR4F5	olfactory receptor family 4 subfamily F member 5	4.71
4	OR10A2	olfactory receptor family 10 subfamily A member 2	2.66
5	CHL1	cell adhesion molecule L1-like	2.42
6	MECP2	methyl-CpG binding protein 2	1.70
7	DDX11L1	DEAD/H (Asp-Glu-Ala-Asp/His) box helicase 11 like 1	1.09
8	ACAP3	ArfGAP with coiled-coil, ankyrin repeat and PH domains 3	0.78
9	WASH7P	WAS protein family homolog 7 pseudogene	0.77
10	RNPEP	arginyl aminopeptidase (aminopeptidase B)	0.73
11	SMARCA2	SWI/SNF related, matrix associated, actin dependent regulator of chromatin, subfamily a, member 2	0.69
12	ASXL1	additional sex combs like 1, transcriptional regulator	0.54
13	NMNAT1	nicotinamide nucleotide adenylyltransferase 1	0.53
14	AKAP6	A-kinase anchoring protein 6	0.48
15	PMS2	PMS1 homolog 2, mismatch repair system component	0.44
16	TTN	titin	0.41
17	CHAMP1	chromosome alignment maintaining phosphoprotein 1	0.39
18	CNTN5	contactin 5	0.38
19	NLRP3	NLR family, pyrin domain containing 3	0.33
20	TRNT1	tRNA nucleotidyl transferase, CCA-adding, 1	0.33
21	TAB2	TGF-beta activated kinase 1/MAP3K7 binding protein 2	0.33
22	BRCA1	breast cancer 1	0.32
23	LZIC	leucine zipper and CTNNBIP1 domain containing	0.30
24	MLH1	mutL homolog 1	0.28
25	KMT2A	lysine (K)-specific methyltransferase 2A	0.28

Table 4.2: Top 25 genes according to the number of targeting variants submitted to the Beacon Network in 2015 together with the percentage of relevant gene-targeting queries.

No.	MIM No.	Disorder	Percentage
1	176807	Prostate cancer	33.58
2	613347	Pancreatic cancer 2	33.58
3	155255	Medulloblastoma	33.58
4	605724	Fanconi anemia, complementation group D1	33.58
5	613029	Glioblastoma 3	33.58
6	612555	Breast-ovarian cancer, familial, 2	33.58
7	114480	Breast cancer, male, susceptibility to	33.58
8	194070	Wilms tumor	33.58
9	613643	Parkinson disease 5, susceptibility to	2.45
10	615491	Neurodegeneration with optic atrophy, childhood onset	2.45
11	188580	Thyrotoxic periodic paralysis, susceptibility to, 1	2.42
12	601887	Malignant hyperthermia susceptibility 5	2.42
13	170400	Hypokalemic periodic paralysis, type 1	2.42
14	611875	Brugada syndrome 3	2.42
15	601005	Timothy syndrome	2.42
16	614896	Sinoatrial node dysfunction and deafness	2.42
17	615474	Primary aldosteronism, seizures, and neurologic abnormalities	2.42
18	312750	Rett syndrome, atypical	1.70
19	312750	Rett syndrome	1.70
20	300055	Mental retardation, X-linked, syndromic 13	1.70
21	300260	Mental retardation, X-linked syndromic, Lubs type	1.70
22	300673	Encephalopathy, neonatal severe	1.70
23	300496	Autism susceptibility, X-linked 3	1.70
24	312750	Rett syndrome, preserved speech variant	1.70
25	276300	Mismatch repair cancer syndrome	0.92

Table 4.3: Top 25 OMIM disorders according to the number of targeting variants submitted to the Beacon Network in 2015 together with the percentage of relevant gene-targeting queries.

with the most associations are listed in Table 4.3. As the entries were computed through gene annotation, a single variant query might be associated with multiple disorders. This is indeed the case of some entries in the table – disorders 1-9, for example, lead the list due to the popularity of BRCA2 gene (see Section 4.11).

4.13 Clinical abnormalities

There are many thousands of hereditary diseases in humans, each of which has a specific combination of phenotypes. For this reason, we summarize what phenotypic features variants submitted to the system are related to.

One of the main difficulties in using data from OMIM in computational analysis is that OMIM does not use a controlled vocabulary, and it can be difficult to recognize synonyms by computational means. Ontologies, i.e. structured, controlled vocabularies that formally describe the kinds of entities within a domain, such as the Human Phenotype Ontology (HPO), have been created in an attempt to provide a comprehensive controlled vocabulary and knowledge base describing the manifestations of human diseases, and these ontologies have been applied to characterize diseases in OMIM [53].

HPO itself does not describe disease entities, but rather the clinical abnormalities associated with them [89]. Each term is assigned to one of four ontologies – Phenotypic abnormality (the main ontology containing descriptions of clinical abnormalities), Clinical modifier (ontology containing classes that describe typical modifiers of clinical symptoms, such as the speed of progression), Mortality/aging (subontology describing the time of death), or Mode of inheritance (ontology describing the mode of inheritance, such as autosomal dominant) [56]. Most terms of the HPO belong to the Phenotypic abnormality ontology.

We cross-referenced annotated variants submitted to our system with HPO's ontology database. HPO is a system respecting the true-path rule [80] – if a disease is annotated to a term, then all of the ancestors of this term also apply. In order not to pollute our list with very general terms, we mapped each gene to the most specific HPO classes rather than all the ancestors.

In total, 3,517 HPO terms were associated with the queried variants. The ones with the most associations are listed in Table 4.4.

Terms 1, 2, 12, 13 are direct children of the Mode of inheritance ontology. As such, they are linked to a large number of genes, which explains the respective high numbers of their requests.

All the other terms on the list are descendants of the Phenotypic abnormality ontology. As for their basic classification, i.e. first-level children of the main ontology, terms 3, 4, and 25 represent abnormalities of the nervous system. Terms 5, 6, and 8 are related to growth, 7 and 19 are problems with the skeletal system, 9, 15, and 18 are linked to blood and blood-forming tissues. Term 10 is an abnormality of head or neck, and 11, 20, as well as 21 are abnormalities of the integument. Terms 14 and 17 are tied to the immune system, 16 is related to neoplasm, 22 and 23 are abnormalities of metabolism/homeostasis. Terms 24 represents an abnormality of the musculature.

No.	Term ID	Term	Percentage
1	HP:0000007	Autosomal recessive inheritance	40.85
2	HP:0000006	Autosomal dominant inheritance	39.64
3	HP:0100543	Cognitive impairment	38.62
4	HP:0000252	Microcephaly	38.23
5	HP:0004322	Short stature	38.16
6	HP:0001508	Failure to thrive	37.56
7	HP:0002650	Scoliosis	37.06
8	HP:0001511	Intrauterine growth retardation	35.39
9	HP:0001903	Anemia	35.03
10	HP:0000581	Blepharophimosis	34.95
11	HP:0000957	Cafe-au-lait spot	34.58
12	HP:0001428	Somatic mutation	34.40
13	HP:0001425	Heterogeneous	33.98
14	HP:0004808	Acute myeloid leukemia	33.94
15	HP:0001873	Thrombocytopenia	33.87
16	HP:0002667	Nephroblastoma (Wilms tumor)	33.71
17	HP:0001882	Leukopenia	33.70
18	HP:0005528	Bone marrow hypocellularity	33.66
19	HP:0006501	Aplasia/Hypoplasia of the radius	33.66
20	HP:0001053	Hypopigmented skin patches	33.65
21	HP:0007400	Irregular hyperpigmentation	33.65
22	HP:0003220	Abnormality of chromosome stability	33.60
23	HP:0003221	Chromosomal breakage induced by crosslinking agents	33.59
24	HP:0001252	Muscular hypotonia	6.51
25	HP:0001250	Seizures	6.40

Table 4.4: Top 25 HPO terms according to the number of targeting variants submitted to the Beacon Network in 2015 together with the percentage of relevant gene-targeting queries.

5 Future work

"The only way to deal with variants of unknown significance is to create a large-scale genomic knowledge network that includes all the genetic variants observed worldwide, and their associated health observations. This requires a global network of shared data."

– David Haussler, Director, UC Santa Cruz Genomics Institute. Source: [82].

As a test of institutions' willingness to share their data publicly, beacons have basically fulfilled their role, and the demonstration project has been a success. In order to remain useful, however, the API should be extended to support a much more sophisticated queriy language involving patient and variant metadata. Instead of asking if a beacon has observed a particular allele, one should be able to ask if the beacon has observed the allele in patients with a specific disease. In collaboration with the Metadata team in GA4GH, the Beacon Network should keep driving the development of the API in this direction, and implement it as soon as it is finalized. The work on incorporating metadata into beacons is already in progress.

This document contains a proposal for the Beacon 0.3 specification, which is currently under review. Once the specification is accepted, the Beacon Network should be updated to support it. This includes wrapping all the beacons in 0.3 API as well as dealing with cross-dataset queries.

Currently, our system only integrates public beacons. As the specifications for additional access levels are getting finalized and early implementations are starting to appear, the system should start supporting authentication and authorization. Federation of queries across authenticated beacons is hard, especially since they do not recognize the same identity providers, or even the same protocols. The Security working group in GA4GH has been developing a solution to this problem for some time, and the Beacon Network should implement it, once a proposal is made.

Once the system supports authentication and authorization, the operators should be allowed to manage their beacons programmatically. This includes programmatic registration of beacons, which is currently very restricted.

In order to keep the implementation of individual beacons light-weight, our system should take care of hard tasks all the beacons could benefit from. This includes, for example, conversions of coordinates between assemblies, or setting of per-beacon limits on query frequency.

The Beacon Network should keep supporting the beacon developers and keep getting enhanced in a way that would attract new organizations and users.

6 Conclusions

"There's going to be an enormous change in how science is done, and it's only because the signal-to-noise ratio necessitates it. You can't get your result with just 10,000 patients – you are going to need more. Scientists will share now because they have to."

– Arthur Toga, Director, USC Institute For Neuroimaging and Informatics. Source: [87].

In this work, we analyzed the problem of federating queries and aggregating responses over a network of public beacons, web services sharing genetic datasets. We introduced the Beacon Project run by the Global Alliance for Genomics and Health, analyzed it from security and privacy perspectives, and put it in context by briefly describing other systems facilitating sharing of genomic data. The status of the project created a demand for a global search engine across the world's public beacons and standardization of the API.

We designed and implemented the Beacon Network, a system enabling global discovery of genetic mutations, federated across a large and growing network of shared genetic datasets. Many requirements on the system's functionality were specified – the system's responsibilities include federation of queries across beacons, integration of publicly available beacons, aggregation of data from multiple sources, online distribution of queries without the need to store genomic data, registry of public beacons, unified API, and easy programmatic access. Constraints on performance, scalability, extensibility, logging, engineering practices, and monitoring were also in place. The system was developed under the umbrella of GA4GH, with the barrier of entry for beacon developers in mind. It also unified the Beacon API and became the driving force behind the new version of the standard.

The Beacon Network has been running for over a year and collected some interesting information. We analyzed the data to determine the size, adoption rate, and traffic in the system. We focused on the queries the system received in order to determine what kind of alleles, chromosomes, assemblies, genes, disorders, and clinical abnormalities people are searching for. We also looked into harmfulness and rarity of the queried variants.

Our evaluation concluded that the system met all our requirements. Integrating 64 beacons from 23 organizations, searching estimated 2 million samples and 2 billion variants, and having facilitated over 800,000 queries in 2015, it is truly a global search engine accessing a lot of interesting information. In fact, to our knowledge, it is the largest aggregator of human genomic variation in the world. The usage pattern of the network indicates that people are looking for interesting data, including many harmful variants, unusual phenotypes, and rare diseases.

Bibliography

[1] Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, February 2001.

[2] ISO/IEC 27001:2013: Information technology – Security techniques – Information security management systems – Requirements. Available at `http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=54534`, October 2013. Edition 2. Accessed on December 30, 2015.

[3] Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.*, 44(D1):D7–D19, Jan 2016.

[4] 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature*, 526(7571):68–74, October 2015.

[5] I. A. Adzhubei et al. A method and server for predicting damaging missense mutations. *Nature methods*, 7(4):248–249, April 2010.

[6] D. Allen, A. Knutsen, P. Muir, A. Rubinger, and K. Piwko. Arquillian: An integration testing framework for containers. `https://docs.jboss.org/arquillian/reference/1.0.0.Alpha5/en-US/html_single/`, 2011. Reference guide. Version 1.0.0.Alpha5. Accessed on December 22, 2015.

[7] D. Altshuler et al. A map of human genome variation from population-scale sequencing. *NATURE*, 467:1061–1073, 2010.

[8] D. Altshuler, J. Bell, et al. Creating a global alliance to enable responsible sharing of genomic and clinical data. 2013.

[9] Amazon Web Services. Amazon Web Services: Overview of security processes. 2015. Accessed on December 22, 2015.

[10] J. Amberger, C. Bocchini, and A. Hamosh. A new face and new challenges for Online Mendelian Inheritance in Man (OMIMÂŽ)., May 2011.

[11] S. Bamford et al. The COSMIC (Catalogue of Somatic Mutations in Cancer) database and website. *Br. J. Cancer*, 91(2):355–358, Jul 2004.

[12] Bean Validation 1.1 Expert Group. JSR 349: Bean Validation. `http://jcp.org/en/jsr/detail?id=349`, 2013. Version 1.1, Final Release. Accessed on December 22, 2015.

[13] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. GenBank. *Nucleic Acids Res.*, 33(Database issue):D34–38, Jan 2005.

[14] E. Birney et al. An overview of Ensembl. *Genome Res.*, 14(5):925–928, May 2004.

[15] O. J. Buske et al. *Human Mutation*, 36(10):922–927, 2015.

[16] O. J. Buske et al. PhenomeCentral: A Portal for Phenotypic and Genotypic Matchmaking of Patients with Rare Genetic Diseases. *Human mutation*, August 2015.

[17] E. A. Chatzimichali et al. Facilitating Collaboration in Rare Genetic Disorders Through Effective Matchmaking in DECIPHER. *Human mutation*, July 2015.

[18] G. M. Church. The Personal Genome Project. *Molecular Systems Biology*, 1(1):msb4100040–E1–msb4100040–E3, December 2005.

[19] S. Company. *Maven: The Definitive Guide*. O'Reilly Media, 1 edition, 10 2008.

[20] T. U. Consortium. The UK10K project identifies rare variants in health and disease. *Nature*, 526(7571):82–90, September 2015.

[21] Contexts and Dependency Injection 1.1 Expert Group. JSR 346: Contexts and Dependency Injection for Java EE. `http://jcp.org/en/jsr/detail?id=346`, 2013. Version 1.1, Final Release. Accessed on December 22, 2015.

[22] C. E. Cook et al. The european bioinformatics institute in 2016: Data growth and integration. *Nucleic Acids Research*, 2015.

[23] M. Cupak. *Parallelization of Queries on Genomic Data*. LAP Lambert Academic Publishing, 2015.

[24] E. D. Hardt. The OAuth 2.0 Authorization Framework. `https://tools.ietf.org/html/rfc6749`, October 2012. RFC 6749. Accessed on December 22, 2015.

[25] R. Dalgleish et al. Locus Reference Genomic sequences: an improved basis for describing human DNA variants. *Genome medicine*, 2(4):24+, April 2010.

[26] P. Danecek et al. The variant call format and VCFtools. *Bioinformatics*, 27(15):2156–2158, 2011.

[27] P. Danecek et al. The variant call format and VCFtools poster. 2011.

[28] K. Davies. *The $1,000 Genome: The Revolution in DNA Sequencing and the New Era of Personalized Medicine*. Free Press, 2010.

[29] Docker. The Docker user guide. `https://docs.docker.com/engine/userguide/`, 2015. Accessed on December 22, 2015.

[30] R. D. Dowell, R. M. Jokerst, A. Day, S. R. Eddy, and L. Stein. The distributed annotation system. *BMC Bioinformatics*, 2:7, 2001.

[31] A. Dunham et al. The DNA sequence and analysis of human chromosome 13. *Nature*, 428(6982):522–528, Apr 2004.

[32] S. O. Dyke et al. Consent codes: Upholding standard data use conditions. *PLoS Genet*, 1 2016.

[33] EJB 3.2 Expert Group. JSR 345: Enterprise JavaBeans. `http://jcp.org/en/jsr/detail?id=345`, 2013. Version 3.2, Final Release. Accessed on December 22, 2015.

[34] Exome Aggregation Consortium et al. Analysis of protein-coding genetic variation in 60,706 humans. *bioRxiv*, 2015.

[35] S. Fayzullina et al. Genetic associations with traits in 23andMe customers, December 2014.

[36] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, 2000. AAI9980887.

[37] H. V. Firth et al. DECIPHER: Database of Chromosomal Imbalance and Phenotype in Humans Using Ensembl Resources. *American journal of human genetics*, 84(4):524–533, April 2009.

[38] M. Fiume et al. Savant Genome Browser 2: visualization and analysis for population-scale genomics. *Nucleic Acids Res.*, 40(Web Server issue):W615–621, Jul 2012.

[39] M. Fiume. System for interpretation of personal genomes. 2015.

[40] I. F. Fokkema, J. T. den Dunnen, and P. E. Taschner. LOVD: easy creation of a locus-specific sequence variation database using an "LSDB-in-a-box" approach. *Human mutation*, 26(2):63–68, August 2005.

[41] J. Frezal. Genatlas database, genes and development defects. *C. R. Acad. Sci. III, Sci. Vie*, 321(10):805–817, Oct 1998.

[42] W. Fujibuchi et al. DBGET/LinkDB: an integrated database retrieval system. *Pac Symp Biocomput*, pages 683–694, 1998.

[43] Global Alliance for Genomics and Health. About the Global Alliance. Available at `http://genomicsandhealth.org/about-global-alliance`. Accessed on August 26, 2015.

[44] Global Alliance for Genomics and Health. Beacon project. Available at `http://ga4gh.org/#/beacon`. Accessed on December 11, 2015.

[45] Global Alliance for Genomics and Health. Working groups. Available at `https://genomicsandhealth.org/working-groups`. Accessed on December 11, 2015.

[46] Global Alliance for Genomics and Health. Beacon project mitigates privacy risks while maximizing value of responsible data sharing. Available at `https://genomicsandhealth.org/files/public/GA4GH_BeaconRelease_2015.pdf`, October 2015. Accessed on December 30, 2015.

[47] Global Alliance for Genomics and Health. Security infrastructure: Standards and implementation practices for protecting the privacy and security of shared genomic and clinical data. Available at `https://genomicsandhealth.org/files/public/SecurityFramework-v1.1-2015-03-12-FINAL.pdf`, 2015. Version 1.1. Accessed on December 30, 2015.

[48] Global Alliance for Genomics and Health. Terms of use. `https://beacon-network.org/#/terms`, July 2015. Accessed on December 22, 2015.

[49] G. Glusman, J. Caballero, D. E. Mauldin, L. Hood, and J. C. Roach. Kaviar: an accessible system for testing snv novelty. *Bioinformatics*, 27(22):3216–3217, 2011.

[50] K. A. Gray, B. Yates, R. L. Seal, M. W. Wright, and E. A. Bruford. Genenames.org: the HGNC resources in 2015. *Nucleic acids research*, 43(Database issue), January 2015.

[51] S. G. Gregory et al. The DNA sequence and biological annotation of human chromosome 1. *Nature*, 441(7091):315–321, May 2006.

[52] G. R. Harp et al. A new class of seti beacons that contain information (22-aug-2010). *CoRR*, abs/1211.6470, 2012.

[53] R. Hoehndorf, P. N. Schofield, and G. V. Gkoutos. Analysis of the human diseasome using phenotype similarity between common, genetic, and infectious diseases. *Scientific reports*, 5, 2015.

[54] N. Homer et al. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genet.*, 4(8):e1000167, Aug 2008.

[55] P. Hublitz et al. NIR is a novel INHAT repressor that modulates the transcriptional activity of p53. *Genes Dev.*, 19(23):2912–2924, Dec 2005.

[56] Human Phenotype Ontology. Documentation | human phenotype ontology. Available at `http://human-phenotype-ontology.github.io/documentation.html`. Accessed on December 30, 2015.

[57] International HapMap Consortium. A haplotype map of the human genome. *Nature*, 437(7063):1299–1320, October 2005.

[58] Java Architecture for XML Binding 2.2 Expert Group. JSR 222: Java Architecture for XML Binding (JAXB). `http://jcp.org/en/jsr/detail?id=222`, 2009. Version 2.2, Maintenance Release. Accessed on December 22, 2015.

[59] Java Persistence 2.1 Expert Group. JSR 338: Java Persistence API. `http://jcp.org/en/jsr/detail?id=338`, 2013. Version 2.1, Final Release. Accessed on December 22, 2015.

[60] JAX-RS 2.0 Expert Group. JSR 339: JAX-RS 2.0: The Java API for RESTful Web Services. `http://jcp.org/en/jsr/detail?id=339`, 2013. Version 2.0, Final Release. Accessed on December 22, 2015.

[61] W. J. Kent et al. The human genome browser at UCSC. *Genome Res.*, 12(6):996–1006, Jun 2002.

[62] B. E. Kirkpatrick et al. GenomeConnect: matchmaking between patients, clinical laboratories, and researchers to improve genomic knowledge. *Hum. Mutat.*, 36(10):974–978, Oct 2015.

[63] P. A. Kitts et al. Assembly: a resource for assembled genomes at NCBI. *Nucleic Acids Res.*, 44(D1):73–80, Jan 2016.

[64] B. M. Knoppers. Framework for responsible sharing of genomic and health-related data. *The HUGO Journal*, 8, October 2014.

[65] M. Kolchagova. DNAdigest interviews GA4GH. DNAdigest, April 29, 2015. Available at `http://dnadigest.org/dnadigest-interviews-ga4gh/`. Accessed on August 27, 2015.

[66] D. E. Krane and M. L. Raymer. *Fundamental concepts of Bioinformatics*. Benjamin Cummings, first edition, September 2002.

[67] M. Krawczak et al. Human gene mutation database: A biomedical information and research resource. *Human Mutation*, 15(1):45–51, 2000.

[68] A. Krol. Beacon project cracks the door for genomic data sharing. Bio-IT World, August 14, 2015. Available at `http://www.bio-itworld.com/2015/8/14/beacon-project-cracks-door-genomic-data-sharing.html`. Accessed on August 27, 2015.

[69] P. Kumar, S. Henikoff, and P. C. Ng. Predicting the effects of coding non-synonymous variants on protein function using the SIFT algorithm. *Nat Protoc*, 4(7):1073–1081, 2009.

[70] O. Lancaster et al. Cafe Variome: general-purpose software for making genotype-phenotype data discoverable in restricted or open access contexts. *Hum. Mutat.*, 36(10):957–964, Oct 2015.

[71] M. J. Landrum et al. ClinVar: public archive of relationships among sequence variation and human phenotype. *Nucleic acids research*, 42(Database issue):D980–D985, January 2014.

[72] G. Laurie et al. Global Alliance for Genomics and Health: Privacy and Security Policy. Available at `https://genomicsandhealth.org/work-products-demonstration-projects/privacy-and-security-policy`. Version 26 May 2015. Accessed on December 30, 2015.

[73] P. H. Lee and H. Shatkay. F-SNP: computationally predicted functional SNPs for disease association studies. *Nucleic Acids Res.*, 36(Database issue):D820–824, Jan 2008.

[74] R. Leinonen, H. Sugawara, and M. Shumway. The Sequence Read Archive, November 2010.

[75] X. Liu, X. Jian, and E. Boerwinkle. dbNSFP: a lightweight database of human nonsynonymous SNPs and their functional predictions. *Human mutation*, 32(8):894–899, August 2011.

[76] T. Madej et al. MMDB: 3D structures and macromolecular interactions. *Nucleic Acids Res.*, 40(Database issue):D461–464, Jan 2012.

[77] M. D. Mailman et al. The NCBI dbGaP database of genotypes and phenotypes. *Nat. Genet.*, 39(10):1181–1186, Oct 2007.

[78] S. Mathew. Overview of Amazon Web Services. 2014. Accessed on December 22, 2015.

[79] V. A. McKusick. Mendelian Inheritance in Man and its online version, OMIM. *American journal of human genetics*, 80(4):588–604, April 2007.

[80] C. A. Michael Ashburner. Creating the Gene Ontology Resource: Design and Implementation. *Genome Research*, 11(8):1425–1433, August 2001.

[81] Oracle. JSR 907: Java Transaction API. `http://jcp.org/en/jsr/detail?id=907`, 2013. Version 1.2, Final Release. Accessed on December 22, 2015.

[82] Personalized Medicine World Conference. Interview with david haussler. November 20, 2015. Available at `https://genomics.soe.ucsc.edu/news/article/172`. Accessed on December 30, 2015.

[83] A. A. Philippakis et al. The matchmaker exchange: A platform for rare disease gene discovery. *Human Mutation*, 36(10):915–921, 2015.

[84] A. Pizarro and C. Whalley. Architecting for genomic data security and compliance in AWS. 2014. Accessed on December 22, 2015.

[85] J. U. Pontius, L. Wagner, and G. D. Schuler. *The NCBI Handbook, Bethesda (MD).* The NCBI Handbook, 2003.

[86] K. D. Pruitt, T. Tatusova, and D. R. Maglott. NCBI Reference Sequence (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Res.*, 33(Database issue):D501–504, Jan 2005.

[87] A. Regalado. Internet of dna – a global network of millions of genomes could be medicineâs next great advance. MIT Technology Review, February 18, 2015. Available at `http://www.technologyreview.com/featuredstory/535016/internet-of-dna/`. Accessed on December 30, 2015.

[88] J. T. Robinson et al. Integrative genomics viewer. *Nature Biotechnology*, 29(1):24–26, January 2011.

[89] P. N. Robinson et al. The Human Phenotype Ontology: a tool for annotating and analyzing human hereditary disease. *Am. J. Hum. Genet.*, 83(5):610–615, Nov 2008.

[90] P. W. Rose et al. The RCSB Protein Data Bank: redesigned web site and web services. *Nucleic Acids Res.*, 39(Database issue):392–401, Jan 2011.

[91] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore. OpenID Connect Core. `http://openid.net/specs/openid-connect-core-1_0.html`, November 2014. Version 1.0, Final Release. Accessed on December 22, 2015.

[92] K. P. Seiler et al. ChemBank: a small-molecule screening and cheminformatics resource database. *Nucleic Acids Res.*, 36(Database issue):D351–359, Jan 2008.

[93] S. S. Shringarpure and C. D. Bustamante. Privacy risks from genomic data-sharing beacons. *The American Journal of Human Genetics*, 97(5):631–646, 2015/12/19.

[94] M. E. Skinner, A. V. Uzilov, L. D. Stein, C. J. Mungall, and I. H. Holmes. JBrowse: a next-generation genome browser. *Genome Res.*, 19(9):1630–1638, Sep 2009.

[95] J. F. Smart. *Jenkins: The Definitive Guide*. O'Reilly Media, 1 edition, 7 2011.

[96] D. Smedley, S. Haider, B. Ballester, R. Holland, D. London, G. Thorisson, and A. Kasprzyk. Biomart - biological queries made easy. *BMC Genomics*, 10(1):1–12, 2009.

[97] E. M. Smigielski, K. Sirotkin, M. Ward, and S. T. Sherry. dbSNP: a database of single nucleotide polymorphisms. *Nucleic Acids Res.*, 28(1):352–355, Jan 2000.

[98] A. D. Smith et al. *Oxford dictionary of biochemistry and molecular biology*. Oxford University Press, USA, revised edition, February 1997.

[99] B. D. Solomon, A. D. Nguyen, K. A. Bear, and T. G. Wolfsberg. Clinical genomic database. *Proc. Natl. Acad. Sci. U.S.A.*, 110(24):9851–9855, Jun 2013.

[100] M. L. Speir et al. The UCSC Genome Browser database: 2016 update. *Nucleic Acids Res.*, 44(D1):D717–725, Jan 2016.

[101] L. Stein. Generic Feature Format version 3 (GFF3). `http://www.sequenceontology.org/gff3.shtml`, 2013. Version 1.21.

[102] P. D. Stenson et al. The Human Gene Mutation Database: building a comprehensive mutation repository for clinical and molecular genetics, diagnostic testing and personalized genomic medicine. *Human genetics*, 133(1):1–9, January 2014.

[103] Z. D. Stephens et al. Big Data: Astronomical or Genomical? *PLoS Biol.*, 13(7):e1002195, Jul 2015.

[104] P. Tahchiev, F. Leme, V. Massol, and G. Gregory. *JUnit in Action, Second Edition*. Manning Publications, second edition edition, 8 2010.

[105] The UniProt Consortium. UniProt: a hub for protein information. *Nucleic Acids Research*, 43(D1):D204–D212, January 2015.

[106] R. Thompson et al. RD-Connect: an integrated platform connecting databases, registries, biobanks and clinical bioinformatics for rare disease research. *J Gen Intern Med*, 29 Suppl 3:S780–787, Aug 2014.

[107] D. Todorov and Y. Ozkan. AWS security best practices. 2013. Accessed on December 22, 2015.

[108] S. Wallace et al. Global Alliance for Genomics and Health: Consent Policy. Available at `https://genomicsandhealth.org/work-products-demonstration-projects/consent-policy`. Version 27 May 2015. Accessed on December 30, 2015.

[109] K. Wang, M. Li, and H. Hakonarson. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. 38(16):e164+, 2010.

[110] Y. Wang et al. PubChem: a public information system for analyzing bioactivities of small molecules. *Nucleic Acids Res.*, 37(Web Server issue):W623–633, Jul 2009.

[111] J. N. Weinstein et al. The Cancer Genome Atlas Pan-Cancer analysis project. *Nat. Genet.*, 45(10):1113–1120, Oct 2013.

[112] D. Welter et al. The NHGRI GWAS Catalog, a curated resource of SNP-trait associations. *Nucleic Acids Res.*, 42(Database issue):D1001–1006, Jan 2014.

[113] A. Yates et al. Ensembl 2016. *Nucleic Acids Res.*, 44(D1):D710–716, Jan 2016.

[114] J. M. Young and B. J. Trask. The sense of smell: genomics of vertebrate odorant receptors. *Human Molecular Genetics*, 11(10):1153–1160, 2002.

[115] R. K. Yuen et al. Whole-genome sequencing of quartet families with autism spectrum disorder. *Nat. Med.*, 21(2):185–191, Feb 2015.

[116] E. A. Zerhouni and E. G. Nabel. Protecting Aggregate Genomic Data. *Science*, 322(5898):44a+, October 2008.

[117] J. Zhang et al. International Cancer Genome Consortium Data Portal–a one-stop shop for cancer genomics data. *Database (Oxford)*, 2011:bar026, 2011.

A Beacon 0.1 specification

```
@namespace("org.ga4gh")

/**
A Beacon is a web service for genetic data sharing that can be queried for
information about specific alleles.
*/
protocol BEACON {

/**
A request for information about a specific allele.
*/
record BEACONRequest {
  /** The name of the targeted population */
  union{ null, string } populationId = null;

  /** The version of the reference */
  union{ null, string } referenceVersion = null;

  /** The chromosome of the request */
  union{ null, string } chromosome = null;

  /** 0-based allele locus */
  union{ null, long } coordinate = null;

  /** allele */
  union{ null, string } allele = null;
}

/**
A response containing information about the specified allele.
*/
record BEACONResponse {
  /** outcome of the existence test */
  union{ null, boolean } exists = null;

  /** allele frequency */
  union{ null, long } frequency = null;
}

}
```

B Beacon 0.2 specification

```
@namespace("org.ga4gh.beacon")

/**
A Beacon is a web service for genetic data sharing that can be queried for
information about specific alleles.
*/
protocol BEACON {

/**
A request for information about a specific site
*/
record QueryResource {
  /**
  The reference bases for this variant, starting from ‘position‘, in the
  genome described by the field ‘reference‘.
  */
  string referenceBases;

  /** The bases that appear instead of the reference bases. */
  string alternateBases;

  /** The chromosome of the request */
  string chromosome;

  /** 0-based allele locus */
  long position;

  /** The version of the reference */
  string reference;

  /** The name of the targeted population */
  union{ null, string } dataset = null;
}

/**
ErrorResource
*/
record ErrorResource {
  /** Error name/code, e.g. "bad_request" or "unauthorized". */
  string name;

  /** Error message. */
  union{ null, string } description = null;
}

```

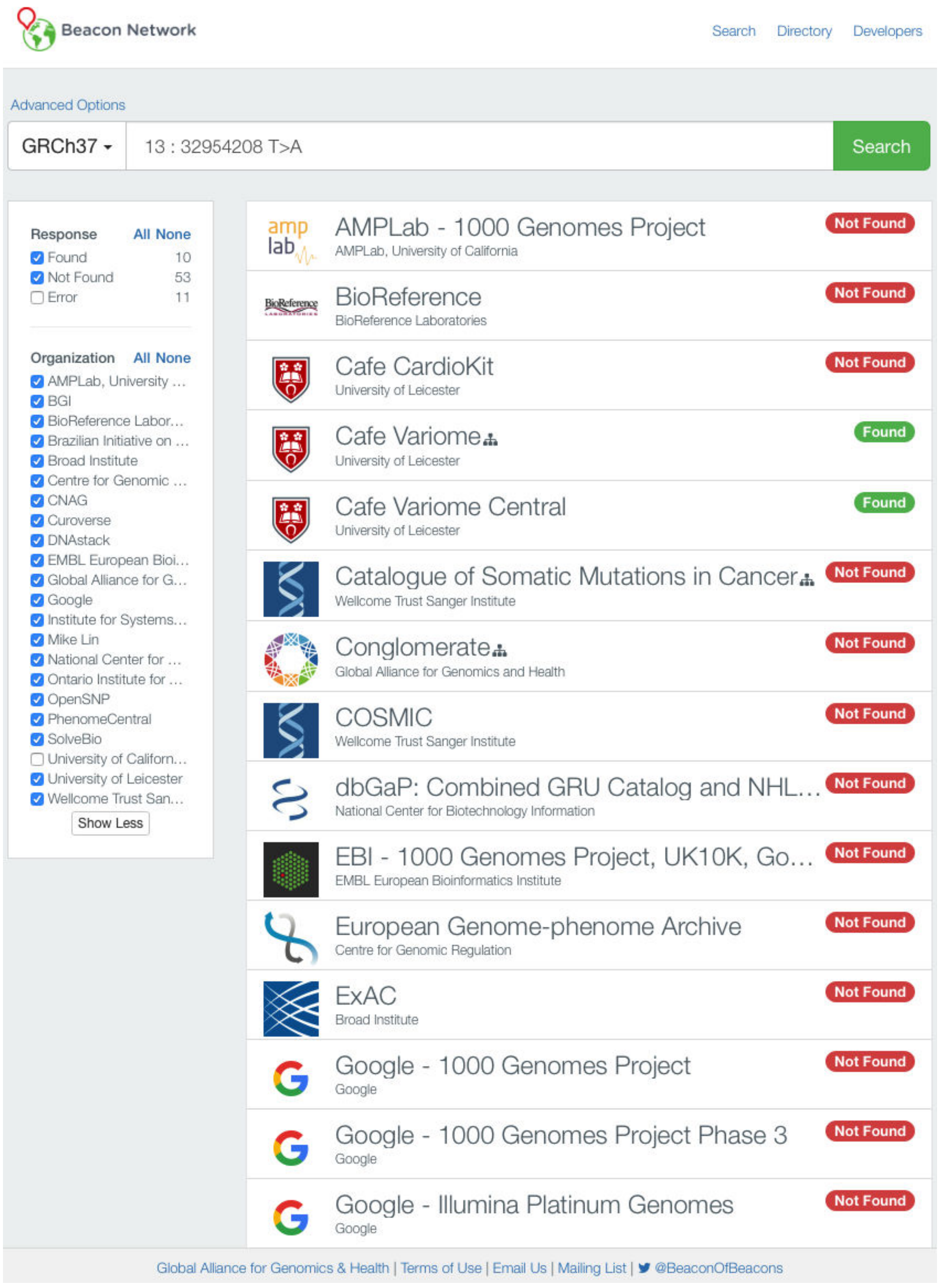
```
/**
DataUseRequirementResource
*/
record DataUseRequirementResource {
  /** Data Use requirement */
  string name;

  /** Description of Data Use requirement. */
  union{ null, string } description = null;
}

/**
DataUseResource
*/
record DataUseResource {
  /** Data Use category.*/
  string category;

  /** Description of Data Use category. */
  union{ null, string } description = null;

  /** Data Use requirements. */
  array<DataUseRequirementResource> requirements = [];
}

/**
DataSetSizeResource
*/
record DataSizeResource {
  /** Total number of variant positions in the data set */
  int variants;

  /** Total number of samples in the data set */
  int samples;
}

/**
DataSetResource
*/
record DataSetResource {
  /** Dataset name */
  string id;

  /** Reference genome */
  string reference;

  /** Dataset description */
  union{ null, string } description = null;

  /**
  Dimensions of the data set. Should be provided if the beacon reports
  allele frequencies.
  */
  union{ null, DataSizeResource } size = null;

  /** True if this dataset contains data from 2 or more other datasets. */
  boolean multiple;

  /**
  List of names of each of the datasets that comprises this aggregated
  dataset. Should be provided if ‘multiple‘ is true.
  */
  array<string> datasets = [];

  /** Data use limitations, specified as a set of DataUseResource. */
  array<DataUseResource> data_use = [];
}

/**
BeaconInformationResource
*/
record BeaconInformationResource {
  /**
  (Unique) beacon ID. Recommended pattern: [organization]-[beacon]
  (no special characters).
  */
  string id;

  /** Name of the owning organization. */
  string organization;

  /** Beacon description. */
  string description;

  /** Datasets served by the beacon. */
  array<DataSetResource> datasets = [];

  /** Beacon API version supported. */
  string api;

  /** URL to the homepage for this beacon. */
  union{ null, string } homepage = null;

  /** An email address for contact. */
  union{ null, string } email = null;

  /** Auth type. Expected value is OAUTH2. Defaults to NONE. */
  union{ null, string } auth = null;

  /**
  Examples of interesting queries, e.g. a few queries demonstrating
  different types of responses.
  */
  union{ null, string } queries = null;
}

/**
The response to the Beacon query
*/
record ResponseResource {
  /**
  Whether the beacon has observed variants. True if an observation
  exactly matches request. Overlap if an observation overlaps request,
  but not exactly, as in the case of indels or if the query used wildcard
  for allele. False if data are present at the requested position but no
  observations exactly match or overlap. Null otherwise.
  */
  string exists;

  /**
  Frequency of this allele in the dataset. Between 0 and 1, inclusive.
  */
  union{ null, double } frequency;

  /** Number of observations of this allele in the dataset. */
  union{ null, int } observed = null;

  /** Additional message. OK if request succeeded. */
  union{ null, string } info = null;

  /** Error details. Provided if a beacon encountered an error. */
  union{ null, ErrorResource } err = null;
}

/**
The response from the Beacon
*/
record BeaconResponseResource {
  /** Beacon ID */
  string beacon;

  /** Query */
  QueryResource query;

  /** Response */
  ResponseResource response;
}

}
```

C Beacon 0.3 specification (draft)

```
@namespace("org.ga4gh.beacon")

/**
A Beacon is a web service for genetic data sharing that can be queried for
information about specific alleles.
*/
protocol Beacons {

import idl "datause.avdl";

/** Query for information about a specific allele. */
record BeaconAlleleRequest {
  /** Chromosome identifier. Accepted values: 1-22, X, Y. */
  string chrom;

  /** Position, allele locus (0-based). */
  long pos;

  /** Reference bases for this variant, starting from ‘position‘. */
  string ref;

  /** The bases that appear instead of the reference bases. */
  string alt;

  /** Assembly identifier, e.g. ‘GRCh37‘. */
  string assembly;

  /**
  Identifiers of datasets, as defined in ‘BeaconDataset‘.

  If this field is null/not specified, all datasets should be queried.
  */
  union{ null, array<string> } datasetIds = null;
}

/** Dataset of a beacon. */
record BeaconDataset {
  /** Unique identifier of the dataset. */
  string id;

  /** Name of the dataset. */
  string name;

  /** Description of the dataset. */
  union{ null, string } description = null;

  /** Assembly identifier, e.g. ‘GRCh37‘. */
  string assembly;

  /**
  Data use conditions for this dataset based on consent codes.
  */
  DataUse dataUse;

  /**
  The time the dataset was created in the beacon in ms from the epoch.
  */
  long created = null;

  /**
  The time the dataset was last updated in the beacon in ms from the
  epoch.
  */
  long updated = null;

  /** Version of the dataset. */
  union{ null, string } version = null;

  /** Total number of variants in the dataset. */
  union{ null, long } variantCount = null;

  /** Total number of calls in the dataset. */
  union{ null, long } callCount = null;

  /** Total number of samples in the dataset. */
  union{ null, long } sampleCount = null;

  /** URL to an external system providing more dataset information. */
  union{ null, string } externalUrl = null;

  /** Additional structured metadata, key-value pairs. */
  union{ null, map<string> } info = null;
}

/** Organization owning a beacon. */
record BeaconOrganization {
  /** Unique identifier of the organization. */
  string id;

  /** Name of the organization. */
  string name;

  /** Description of the organization. */
  union{ null, string } description = null;

  /** Address of the organization. */
  union{ null, string } address = null;

  /** Web of the organization (URL). */
  union{ null, string } welcomeUrl = null;

  /**
  URL with the contact for the beacon operator/maintainer, e.g. link to
  a contact form or an email in RFC 2368 scheme.
  */
  union{ null, string } contactUrl = null;

  /** URL to the logo of the organization (image). */
  union{ null, string } logoUrl = null;

  /** Additional structured metadata, key-value pairs. */
  union{ null, map<string> } info = null;
}

/** Beacon. */
record Beacon {
  /** Unique identifier of the beacon. */
  string id;

  /** Name of the beacon. */
  string name;

  /** Version of the API provided by the beacon. */
  string apiVersion;

  /** Organization owning the beacon. */
  BeaconOrganization organization;

  /** Description of the beacon. */
  union{ null, string } description = null;

  /** Version of the beacon. */
  union{ null, string } version = null;

  /** URL to the welcome page/UI for this beacon. */
  union{ null, string } welcomeUrl = null;

  /**
  Alternative URL to the API, e.g. a restricted version of this beacon.
  */
  union{ null, string } alternativeUrl = null;

  /** The time this beacon was created in ms from the epoch. */
  union { null, long } created = null;

  /** The time this beacon was last updated in ms from the epoch. */
  union { null, long } updated = null;
  /**
  Datasets served by the beacon. Any beacon should specify at least one
  dataset.
  */
  array<BeaconDataset> datasets = [];

  /**
  Examples of interesting queries, e.g. a few queries demonstrating
  different responses.
  */
  union{ null, array<BeaconAlleleRequest> } sampleAlleleRequests = null;

  /** Additional structured metadata, key-value pairs. */
  union{ null, map<string> } info = null;
}

/** Beacon-specific error representing an unexpected problem. */
record BeaconError {
  /** Numeric status code. */
  int status;

  /** Short name or cause of the error. */
  string reason;

  /** Error message. */
  union{ null, string } message = null;
}

/**
Dataset's response to a query for information about a specific allele.
*/
record BeaconDatasetAlleleResponse {
  /** Identifier of the dataset, as defined in `BeaconDataset`. */
  string datasetId;

  /**
  Indicator of whether the given allele was observed in the dataset.

  This should be non-null, unless there was an error, in which case
  `error` has to be null.
  */
  union{ null, boolean } exists;

  /**
  Dataset-specific error.

  This should be non-null in exceptional situations only, in which case
  `exists` has to be null.
  */
  union{ null, BeaconError } `error` = null;

  /**
  Frequency of this allele in the dataset. Between 0 and 1, inclusive.
  */
  union{ null, double } frequency = null;

  /** Number of variants matching the allele request in the dataset. */
  union{ null, long } variantCount = null;

  /** Number of calls matching the allele request in the dataset. */
  union{ null, long } callCount = null;

  /** Number of samples matching the allele request in the dataset. */
  union{ null, long } sampleCount = null;

  /** Additional note or description of the response. */
  union{ null, string } note = null;

  /**
  URL to an external system, such as a secured beacon or a system
  providing more information about a given allele.
  */
  union{ null, string } externalUrl = null;

  /** Additional structured metadata, key-value pairs. */
  union{ null, map<string> } info = null;
}

/**
Beacon's response to a query for information about a specific allele.
*/
record BeaconAlleleResponse {
  /** Identifier of the beacon, as defined in `Beacon`. */
  string beaconId;

  /**
  Indicator of whether the beacon observed the given allele.

  The value of this field should be null if `error` is not null and true
  if and only if at least one of the datasets responded true, i.e. at
  least one of the `exists` fields nested in `datasetAlleleResponses` is
  true.
  */
  union{ null, boolean } exists = null;

  /**
  Beacon-specific error.

  This should be non-null in exceptional situations only, in which case
  `exists` has to be null.
  */
  union{ null, BeaconError } `error` = null;

  /** Allele request as interpreted by the beacon. */
  union{ null, BeaconAlleleRequest } alleleRequest;

  /** Indicator of whether the beacon has observed the allele. */
  array<BeaconDatasetAlleleResponse> datasetAlleleResponses = [];
}

}

@namespace("org.ga4gh.beacon")

/**
Data use conditions based on consent codes.
*/
protocol DataUseConditions {

/** Data use condition. */
record DataUseCondition {
  /**
  Consent code abbreviation, e.g. ‘NRES‘ for no restrictions primary
  category.
  */
  string code;

  /** Description of the condition. */
  union{ null, string } description = null;
}

/** Data use of a resource based on consent codes. */
record DataUse {
  /**
  Primary data use category.
  */
  DataUseCondition primaryCategory;

  /** Secondary data use categories. */
  array<DataUseCondition> secondaryCategories = [];

  /** Data use requirements. */
  array<DataUseCondition> requirements = [];
}

}

@namespace("org.ga4gh.beacon")

protocol BeaconMethods {

import idl "beacon.avdl";
```

```
/**************** / *******************/
/**
Gets beacon information.

‘GET /‘ (root of the beacon API) returns a representation of ‘Beacon‘.
*/
Beacon getBeacon();

/**************** /alleles *******************/
/**
Gets response to a beacon query for information about a specific allele.

‘GET /alleles‘ uses ‘BeaconAlleleRequest‘ for parameters and returns
a representation of ‘BeaconAlleleResponse‘. Example: ‘GET /alleles?
chrom=1&pos=1000&ref=A&alt=C&assembly=GRCh37&datasetIds=d1&datasetIds=d2‘
*/
BeaconAlleleResponse getBeaconAlleleResponse(
  BeaconAlleleRequest beaconAlleleRequest);

}
```

D Screenshot of the client's query interface


Beacon Network

Search | Directory | Developers

Advanced Options

GRCh37 ▾ | 13 : 32954208 T>A | Search

Response	All None
✓ Found	10
✓ Not Found	53
☐ Error	11

Organization — All None

- ✓ AMPLab, University …
- ✓ BGI
- ✓ BioReference Labor…
- ✓ Brazilian Initiative on …
- ✓ Broad Institute
- ✓ Centre for Genomic …
- ✓ CNAG
- ✓ Curoverse
- ✓ DNAstack
- ✓ EMBL European Bioi…
- ✓ Global Alliance for G…
- ✓ Google
- ✓ Institute for Systems…
- ✓ Mike Lin
- ✓ National Center for …
- ✓ Ontario Institute for …
- ✓ OpenSNP
- ✓ PhenomeCentral
- ✓ SolveBio
- ☐ University of Californ…
- ✓ University of Leicester
- ✓ Wellcome Trust San…

Show Less

Beacon	Organization	Response
AMPLab - 1000 Genomes Project	AMPLab, University of California	Not Found
BioReference	BioReference Laboratories	Not Found
Cafe CardioKit	University of Leicester	Not Found
Cafe Variome	University of Leicester	Found
Cafe Variome Central	University of Leicester	Found
Catalogue of Somatic Mutations in Cancer	Wellcome Trust Sanger Institute	Not Found
Conglomerate	Global Alliance for Genomics and Health	Not Found
COSMIC	Wellcome Trust Sanger Institute	Not Found
dbGaP: Combined GRU Catalog and NHL…	National Center for Biotechnology Information	Not Found
EBI - 1000 Genomes Project, UK10K, Go…	EMBL European Bioinformatics Institute	Not Found
European Genome-phenome Archive	Centre for Genomic Regulation	Not Found
ExAC	Broad Institute	Not Found
Google - 1000 Genomes Project	Google	Not Found
Google - 1000 Genomes Project Phase 3	Google	Not Found
Google - Illumina Platinum Genomes	Google	Not Found

Global Alliance for Genomics & Health | Terms of Use | Email Us | Mailing List | @BeaconOfBeacons

E Attachments

The attached archive contains:

- `beacon-network-src.zip`.

  The source code of the Beacon Network.

- `beacon-network-doc.zip`.

  The documentation of the Beacon Network.

- `thesis.pdf`.

  The text of this thesis in PDF format.

Index