

Lessons learned to boost a bioinformatics knowledge base reusability, the Bgee experience

Tarcisio Mendes de Farias^{1,2,*}, Julien Wollbrett^{1,2}, Marc Robinson-Rechavi^{1,2} and Frederic Bastian^{1,2}

¹SIB Swiss Institute of Bioinformatics, Lausanne 1015, Switzerland

²Department of Ecology and Evolution, University of Lausanne, Lausanne 1015, Switzerland

*Correspondence address. Tarcisio Mendes de Farias. Swiss Institute of Bioinformatics (SIB) - Quartier UNIL-Sorge - Bâtiment Biophore - Office : 3220 CH-1015 Lausanne - Switzerland. E-mail: tarcisio.mendes@sib.swiss

Abstract

Background: Enhancing interoperability of bioinformatics knowledge bases is a high-priority requirement to maximize data reusability and thus increase their utility such as the return on investment for biomedical research. A knowledge base may provide useful information for life scientists and other knowledge bases, but it only acquires exchange value once the knowledge base is (re)used, and without interoperability, the utility lies dormant.

Results: In this article, we discuss several approaches to boost interoperability depending on the interoperable parts. The findings are driven by several real-world scenario examples that were mostly implemented by Bgee, a well-established gene expression knowledge base. To better justify the findings are transferable, for each Bgee interoperability experience, we also highlight similar implementations by major bioinformatics knowledge bases. Moreover, we discuss ten general main lessons learned. These lessons can be applied in the context of any bioinformatics knowledge base to foster data reusability.

Conclusions: This work provides pragmatic methods and transferable skills to promote reusability of bioinformatics knowledge bases by focusing on interoperability.

Keywords: interoperability, databases, data reusability, ontologies

Introduction

Bioinformatics knowledge bases (KBs) are often built to serve a specific community of interest. By providing software tools, methods, services, and data, these KBs aim to facilitate and to provide the means for their users' work, such as scientific research. Therefore, the notion of reusability is an important aspect to be considered by any bioinformatics KB. Reusability is the capability of a resource to be used multiple times by distinct agents. This statement is a generalization of the data reusability definition in [1].

In applied computing, the relevance of data reusability by computer programs has been highlighted since the late 20th century [1]. More recently, leveraging an efficient discovery and reusability of digital research resources by both machines and humans has been endorsed by the Findable, Accessible, Interoperable, and Reusable (FAIR) principles since 2016 [2]. According to Jacobsen et al. [3], findability, accessibility, and interoperability together enable the final goal of trusted, effective, and sustained reuse of research resources. This goal is also emphasized by Mons et al. [4], who show that FAIR principles focus on ensuring that research objects are reusable.

In this article, among several similar and complementary definitions of interoperability as reported in [5], we consider the following IEEE standard definition of interoperability: "the ability of two or more systems or elements to exchange information and to use the information that has been exchanged" [6]. Based on this definition, and looking at bioinformatics KBs as systems, we will describe how to improve reusability through interoperability enhancement.

Moreover, achieving interoperability has been well recognized as a complex task by several researchers [7–10]. Since it is a hard task, it also presents an impediment to the exchange of information among independent bioinformatics KBs. Therefore, to mitigate this issue, in this article, we focus on pragmatic approaches for interoperability enhancement of bioinformatics KBs. We mainly illustrate these approaches with our experience with our development of Bgee as a more reusable KB.

Bgee is a well-established knowledge base to retrieve and compare gene expression patterns in multiple animal species [11]. It integrates and harmonizes multiple data sources that are based on heterogeneous techniques, namely, single-cell RNA sequencing (scRNA-Seq), bulk RNA-seq, Affymetrix, *in situ* hybridization, and expressed sequence tags (EST). It is based exclusively on curated healthy wild-type expression data (e.g., no gene knockout, no treatment, no disease), to provide a comparable reference of normal gene expression. Moreover, the usefulness of the Bgee interoperability practices has been recognized by several researchers such as in [12].

Although we center our work on the Bgee use case, the practices, methods, and lessons learned discussed here are transferable to other KBs such as those reported in the Nucleic Acids Research Molecular Biology Database Collection [13]. To further demonstrate they are transferable, for each Bgee interoperability experience, we will also highlight, when it is applicable, similar implementations by widely used bioinformatics KBs such as GeneCards (a human gene-centric KB) [14], UniProtKB (a protein-

Received: March 17, 2023. Revised: May 30, 2023. Accepted: July 7, 2023

© The Author(s) 2023. Published by Oxford University Press GigaScience. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

centric KB) [15], and Orthologous Matrix (OMA, an orthology resource) [16].

Furthermore, the lessons learned could be applied in part to other contexts such as low carbon energy databases (DB) mentioned in [17]. The latter report that energy DBs are significantly heterogeneous and can benefit from common data exchange formats and semantic representations to improve interoperability among these DBs.

Broad Aspects for Improving Interoperability

Enhancing data/metadata interoperability is the solution for solving data/metadata heterogeneities among the different parts (e.g., systems) between which we seek to exchange information [18, 19]. According to [20], we can categorize these heterogeneities as structural (schema), syntactic (format), and semantic (meaning) heterogeneity. As noted by Halevy [21], semantic heterogeneity appears whenever there is more than one way to structure a body of data.

To correctly exchange information between different systems, we have to solve the syntactic and semantic heterogeneities, if any, between producer and consumer of this information. By correctly, we mean the information is perceived by the consumer exactly as it is intended by the producer, and the opposite is also true where the information conceived/written by the producer is defined exactly as expected by the consumer. As a result, there is no need for guessing, heuristics, or machine learning methods by the consumer/producer to correctly process the exchanged information. To illustrate this, let us consider a semistructured format for data exchange such as a comma-separated value (CSV) file (i.e., a tabular data format). The 2005 technical standard RFC 4180 [22] formalizes the CSV file format, but there are still multiple syntactical ways to define a CSV file. For instance, the header line, appearing as the first line of the file, is an optional one, and there is no explicit manner to identify whether it is present or not; thus, care is required by the consumer when importing data. Therefore, the producer and the consumer have to come to an agreement to solve this syntactical heterogeneity to become interoperable, such as is the case for the Google Ads system [23] or the US National Center for Biotechnology Information (NCBI) LinkOut service [24].

More flexibility implies more heterogeneity. Choosing a more flexible information exchange solution often implies more heterogeneities to solve [21]. Although a highly constrained, formal, and accurate interoperability solution significantly reduces heterogeneity, its adoption may be compromised due to the difficulties of implementation, adaptability, and fitness for information being exchanged. For example, describing data from a model (e.g., the Bgee native data model) into another (e.g., the Wikidata [25] data model) can lead to data loss (i.e., partial interoperability) due to semantic heterogeneities. These heterogeneities exist whenever experts with several modeling practices and constraints (e.g., application scope, real-time capabilities, security) can produce different conceptual models to represent the same ensembles of data. Actually, even if an ontology is defined as “a formal, explicit specification of a shared conceptualization” [26], different ontologists can produce different ontologies for a same knowledge domain. For example, more than ten ontologies in the Ontology Lookup Service [27] report a “different” *Gene* concept. To address the latter issue, notably ontology matching and alignment have been recognized as interesting approaches [28, 29]. In general, data, metadata, and data schema

mappings between the different interoperable elements enable matching/alignment.

Nevertheless, even to define mappings and alignments, a language with a specific vocabulary, syntax, and semantics is chosen and applied. For example, we could define alignments with plain English—implying not machine-ready to be read; programming languages—specific-purpose adaptors/translators; the OWL—Web Ontology Language (e.g., owl:sameAs, owl:equivalentClass) [30]; SKOS—Simple Knowledge Organization System vocabulary (e.g., skos:closeMatch) [31]; SWRL—Semantic Web Rule Language (e.g., swrlb:matches, swrl:Imp) [18]; VoIDext—Extended Vocabulary of Interlinked Datasets (e.g., voidext:resourceMapping) [32]; and so on. As a result, the heterogeneity problem and, consequently, interoperability issues persist but at another level. When choosing an interoperability solution, which often includes data models, languages, standards for representing the metadata, and data, we have to consider different heterogeneity degrees to solve, as for the definition of mappings and alignments. Moreover, depending on the types of heterogeneity, the nature of elements we wish to interoperate, and the interoperability level we want to achieve, one language can be better than another one to declare mappings. For example, if we are dealing with ontology matching, OWL language is not expressive enough to define complex data schema alignments. For instance, if we suppose the existence of three attributes/properties genus, species, and scientific name, the concatenation of genus (e.g., *Homo*) and species (e.g., *sapiens*) implies the species’ scientific name (e.g., *Homo sapiens*). Then, to define these complex mappings, other languages such as SWRL are more appropriate. SWRL and OWL are both logic-based formalisms. Combining them to define complex mappings further allows us to automatically derive new alignments, thanks to inference engines supporting these languages [18]. In some context, to allow different levels of interoperability in terms of precision, it may be crucial to define the nature of the mappings such as reported in the SKOS vocabulary: `skos:closeMatch`, `skos:exactMatch`, `skos:broadMatch`, `skos:narrowMatch`, and `skos:relatedMatch`.

Depending on the interoperability aim we want to achieve, a semantic relaxation approach can be applied. Semantic relaxation is the capacity of ignoring semantic and data heterogeneities for the sake of interoperability [32]. For example, when interoperating with different orthology databases (i.e., containing information about corresponding genes in different species), the concepts of genes and proteins can be interchangeably used. This is because some algorithms infer orthologous genes using protein sequences. Hence, we can increase interoperability if some loss of information or of precision is admissible.

Knowledge Base Interoperability Approaches and Practices

We define three different types of interoperability approaches for KBs as follows:

Definition 1.

One-side interoperability: one side must strictly comply with the other’s procedure to interoperate. There is no or little possible negotiation between interoperable parts. If the other’s procedure to interoperate with is based on an independent interoperability procedure, it will be classified as a multi-side interoperability as defined in Definition 3.

Definition 2.

Two-side interoperability: both sides must reconcile with each other, that is, establish a common agreement to interoperate.

Definition 3.

Multi-side interoperability: the two or more sides that want to interoperate comply with an independent interoperability procedure. Improvements or changes in this procedure may be upon request and may or may not be accepted by the third-party organization or community that maintains the interoperability procedure. This procedure is usually composed of interoperability standards such as Schema.org. Two-side interoperability is considered a multi-side interoperability, if and only if the reconciliation is based on an independent interoperability solution that can be reused by others.

The classification of an interoperability approach based on those definitions highly depends on the context and timeline. For example, a one- or two-side interoperability can evolve to a multi-side interoperability if it becomes a standard or part of one that can be reused by others. In addition, a one-side interoperability can become a two-side one, for instance, if both sides want to improve and increase the information exchanged, which is not supported by the existing one-side approach. Another possible scenario is a hybrid approach where more than one interoperability approach is implemented. For example, a multi-side interoperability might not be sufficient or timely to establish the desired degree of interoperability between KBs. In this scenario, a two-side interoperability approach may complement the multi-side one.

Table 1 exemplifies the interoperability approaches defined in Definition 1, Definition 2, and Definition 3. Moreover, this table summarizes the use cases involving Bgee as a data producer that are fully described in the next section.

Enhancing interoperability: the experience of Bgee with other knowledge bases

The Bgee KB integrates and aggregates data from heterogeneous data sources by reconciling them and applying a data warehouse approach, resulting in a large relational database (8 TB at time of writing). Curation and quality control are at the core of the Bgee mission. Moreover, being licensed as a public domain database makes Bgee an interesting case study for boosting interoperability since no ownership restrictions exist when reusing its data. Figure 1 shows a simplified view of the Bgee interoperability network boost and of the technologies involved, and Fig. 2 illustrates an overview of the implemented Bgee data interoperability architecture, which are further detailed in the next subsections. Finally, for each Bgee experience, we mention similar implementations, if any, by other KBs (namely, UniProtKB, GeneCards, or OMA) and, when applicable, how they can benefit from our experience too (e.g., if they are not interoperable with a target KB of interest yet).

File-based data exchange

Exchanging data with computer files has been done since the advent of computer file systems. The advantages of this method for exchanging and reusing data among bioinformatics KBs include easy deployment and possible autonomy. For example, the data producer may impose the data format, which the data consumers can use without a previous agreement with them. In this example, the consumers have to adapt their tools (e.g., implement a file reader) to be able to interoperate with data producers. Similarly, a data consumer may also impose the data exchange format to be used by data producers. Nevertheless, this interoperation

mode often leads to misinterpretations, mainly due to the lack of data interoperability standards and data structure (i.e., unstructured or semistructured data), and it complicates interoperability because of syntactic and semantic heterogeneities between consumer and producer. Moreover, it does not necessarily provide access to the latest data because of asynchronous and independent exporting and importing data operations of static files.

Currently, Bgee interoperates with the following KBs by file-based data exchange: NCBI Gene database, GeneCards, UniProtKB, RIKEN MetaDataBase, and OncoMX. Moreover, for advanced users, Bgee provides highly structured data through two data dumps using relational and graph data models. In the next subsections, we discuss each Bgee file-based interoperation case and how we mitigate the aforementioned issues such as misinterpretations.

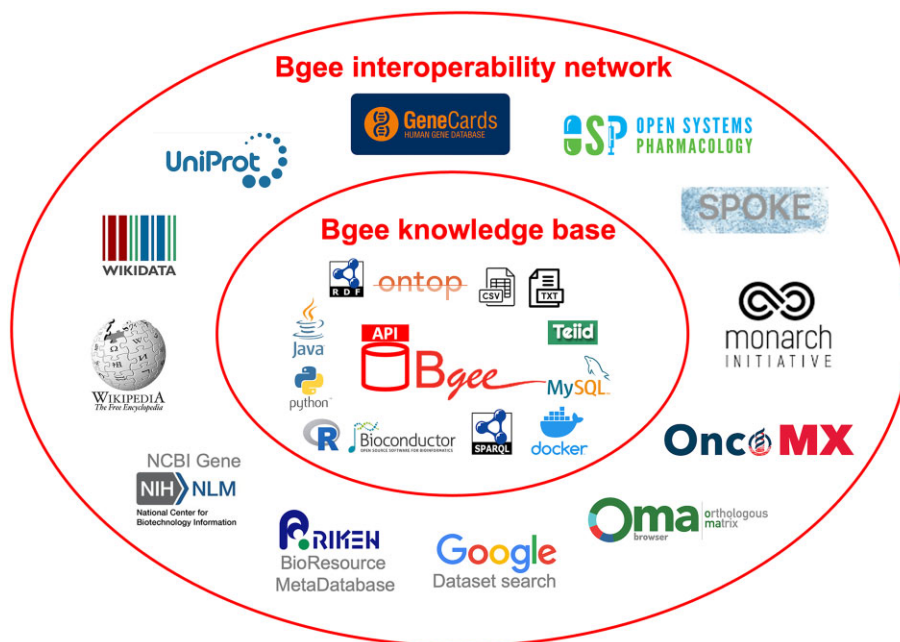
Bgee in the NCBI Gene database

NCBI Gene provides gene-centric information such as sequence, expression, structure, function, citation, and homology data. To be able to interoperate with NCBI Gene through the NCBI LinkOut service [43], we have to strictly comply with NCBI's own specifications to write the expected data exchange files, either an XML-based file or CSV-based files. As a result, its reader will be able to consume the provided data in an automatic way once deployed at a file transfer location. The LinkOut system has successfully enabled more than 250 data providers [44], including Bgee, GeneCards, and OMA, to link their resources to different NCBI databases such as the NCBI Gene. The data we provide are gene symbols and links to the Bgee gene pages that correspond to a NCBI gene page through the NCBI LinkOut section. Although an extensive documentation is provided for both CSV and XML file definitions [45], the lack of better semantic representations may result in noncompliant files for the NCBI LinkOut file reader by the data producer. Moreover, even though a Document Type Definition (DTD) [46] exists for the LinkOut XML file creation, it does not provide enough control on the XML structure, such as an XML Schema Definition (XSD) [47]. For example, with DTD, we are not able to define data types. Relevant XML element data types for the LinkOut XML file definition, such as `<LinkId>` data type, are unknown, and thus, we do not know if a `<LinkId>` can be any character or just integer values greater than zero. Even if the data types or a complete data schema are explicitly defined in the documentation, this schema will not be machine readable, hence complexifying tasks that could otherwise be automatized, such as the writing of a data exchange file.

To interoperate with the NCBI LinkOut system for publishing the Bgee gene links at NCBI Gene pages, we decided to be compliant with its CSV format. A portion of the generated CSV file is shown in Fig. 3. Although a CSV file provides less structure than an XML one, this decision was mainly dictated by the fact we can easily generate the Bgee tabular data output by simply writing a single Structured Query Language (SQL) query over the Bgee relational database. In addition, in this query, we also project the expected CSV header line by the NCBI LinkOut tool. Nevertheless, the flexibility provided with this file format comes with a high price, that is, the lack of semantics and data structure creating heterogeneities. First, at the syntactic level, the CSV file expected by the LinkOut tool is not fully aligned with most implementations as documented in the standard RFC 4180. This standard formalizes the CSV format and notably states that "each field may or may not be enclosed in double quotes." However, we cannot use quotation marks to enclose fields in the LinkOut CSV file; if we do so, it results into invalid files. Moreover, the LinkOut tool restricts an NCBI Gene entry to have at most three records by the same

Table 1: Use cases involving Bgee as a data producer along with the implemented interoperability approaches.

Target knowledge base (KB)	Interoperability approach	Description
NCBI Gene [33]	One-side	Bgee must comply with the NCBI LinkOut system exchange file format, which is either a CSV or XML file.
UniProtKB [15]	One-side	Bgee must comply with the UniProtKB exchange file format, which is a text file based on its own format.
GeneCards [14]	Two-side	Bgee and GeneCards defined from scratch a TSV-like exchange file format that is easy and quickly consumed by GeneCards and produced by Bgee.
OncoMX [34]	Two-side	At first, Bgee and OncoMX defined from scratch a TSV-like exchange file format that was easy and quickly consumed by OncoMX and produced by Bgee.
RIKEN Metadatabase [35]	One-side	RIKEN Metadatabase directly imports the downloadable Bgee RDF dump file into its triple store as a named graph.
Monarch Initiative [36]	One-side	The Monarch Initiative project uses the available Bgee download files as they are.
SPOKE [37]	One-side	The Bgee download files are used as they are to build a precision medicine knowledge graph.
Open Systems Pharmacology [38]	One-side	The Bgee download files are used as they are to build a KB of gene expression information for drug development.
Wikidata [25]	One-side	Bgee developed a bot using Wikidata Python APIs in order to automatically extract, transform, and load its data into Wikidata [39].
Wikipedia [40]	One-side	Bgee implemented and integrated a software component in the existing gene infobox module [41]. This allows Wikipedia to dynamically retrieve Bgee data in Wikidata.
Google Dataset Search [42]	Multi-side	Bgee provides Schema.org-based metadata embedded in its webpages. These metadata are automatically retrieved and consumed by other systems that support Schema.org such as Google Dataset Search tool.
OncoMX federation	One-side	OncoMX directly uses the available Bgee MySQL database, called EasyBgee (i.e., a one-side interoperability), to federate both KBs. To do so, the federated data schema is composed of the OncoMX native relational data schema and a view for the EasyBgee data schema is defined along with mappings.

**Figure 1:** A simplified illustration of the boost of the Bgee interoperability network mentioned in this article. The elements in the inner circle are examples of the ensemble of techniques used to implement the Bgee interoperability with diverse databases and systems, which are illustrated with their logos in the outer circle.

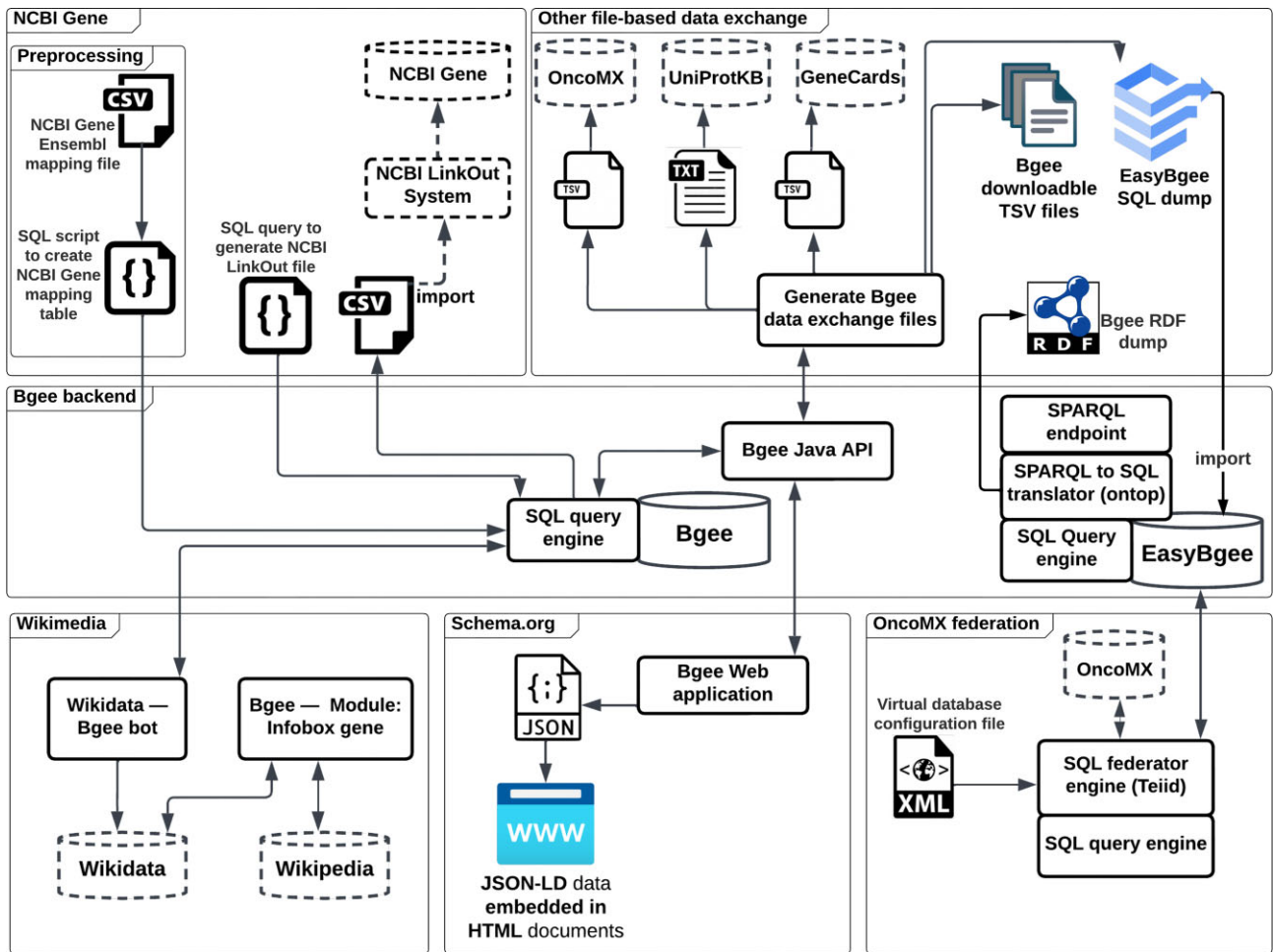


Figure 2: The architecture schema of the Bgee’s interoperability with other major knowledge bases. Dashed cylinders represent external KBs. All rectangles are software components implemented by Bgee except the dashed one that is a third-party system component. In general, outgoing arrows from the “SQL query engine” and “Bgee Java API” are output data (e.g., retrieved results) and ingoing arrows to them are received SQL statements or API calls, respectively. All outgoing arrows from “Generate Bgee data exchange files” are file exports. The “NCBI Gene” container illustrates the components implemented and used to interoperate with NCBI Gene KB. “Other file-based data exchange” container groups all software components developed to generate Bgee exchange files specific to other KBs, non-KB-specific TSV files containing views of the Bgee data, a MySQL database dump, and a RDF dump of the Bgee data. To simplify the schema, not all KBs interoperating with Bgee through files are shown—notably, RIKEN Metadatabase, which imports the generated Bgee RDF dump, and SPOKE, Open Systems Pharmacology, and Monarch Initiative project, which directly use the downloadable Bgee TSV files. “Bgee backend” container shows the data store layer and data access modes (i.e., Bgee Java APIs, SPARQL, and SQL query languages) of the Bgee KB. “Wikimedia” container depicts the implemented software components to exchange information with Wikidata and Wikipedia. In the “Wikimedia” container, the arrow to Wikidata from the bot means data insertion from Bgee to Wikidata, and the outgoing and ingoing arrows from/to “Module: Infobox Gene” represent querying and retrieving data, respectively. “Schema.org” container illustrates metadata are embedded in the Bgee webpages with the Schema.org vocabulary, which are consumed by systems supporting this vocabulary such as Google Dataset Search. Finally, “OncoMX federation” container depicts a dynamic interoperability approach between two independent KBs (i.e., Bgee and OncoMX) that can replace file-based approaches (i.e., interoperability via static exchange files), as illustrated in the “Other file-based data exchange” container.

```

1 PrId,DB,UID,URL,IconUrl,UrlName,SubjectType,Attribute
2 10418, Gene,103476274,https://bgee.org/bgee15_0/gene/ENSPREG00000013759,,rbm41 gene expression,,
3 10418, Gene,103465305,https://bgee.org/bgee15_0/gene/ENSPREG00000013760,,kdm5ba gene expression,,
4 10418, Gene,103474996,https://bgee.org/bgee15_0/gene/ENSPREG00000013761,,BACH1 gene expression,,
    
```

Figure 3: A portion of the data exchange file used by the NCBI LinkOut system.

data provider. Second, at the semantic level, the required third field illustrated in Fig. 3 can have multiple interpretations: either a unique identifier (UID) or a query based on a custom syntax. This field is critical because it enables one to intersect the Bgee and NCBI Gene data, in other words, to correctly publish the Bgee links and gene symbols in the NCBI Gene pages. A UID for our use case means the NCBI Gene identifier. In the Bgee relational database, we do not have the NCBI Gene identifiers; hence, we first sought to define a NCBI-like query instead of a UID as a third CSV field

by providing common IDs between Bgee and NCBI Gene, such as the Ensembl IDs. However, this may result in inconsistencies between Bgee and NCBI gene entries, such as an Ensembl ID that is not present at NCBI or that does not retrieve the same gene entry. The latter case may occur because the query is a keyword matching any indexed word of a NCBI Gene page. The Bgee team was instructed by the LinkOut service to only consider Ensembl IDs present in NCBI Gene, which implies that the LinkOut tool may not work properly if no result is retrieved by a given query (i.e., an

Ensembl ID keyword). Therefore, in order to avoid ambiguity and to solve this semantic heterogeneity, we decided to use the NCBI Gene IDs and to include them in the Bgee database. This was possible thanks to the NCBI Gene ID mapping file [48] we imported into our database. It is notable that none of these issues or solutions are reported in the LinkOut documentation or formally defined in any related data schema. We would also like to stress that since it is a CSV, a semistructured data format, there is no explicit, complete, and formal data schema. Therefore, to address those issues, we had recourse to directly contacting the LinkOut service providers via email, to exactly clarify the expected data exchange format by the LinkOut system. This is a time-consuming process, and while in this case, it is notable that NCBI was very responsive, it is less reliable than an available documentation.

Finally, with this use case, we can see that although the exchanged information is simple [3], it is not straightforward to achieve interoperation between independent resources such as Bgee and NCBI Gene. The aforementioned issues would be significantly worse if the exchanged information were more complex, for example, including gene expression levels, anatomical structures, and developmental stages, that are not expected by the LinkOut system. However, this relevant information would enrich, for instance, the existing “Expression” section of some NCBI Gene pages and allow to further include this section for NCBI Gene pages that do not have any expression information at time of writing, such as the chimpanzee’s hemoglobin subunit beta gene page [49].

Other KBs’ experiences

GeneCards and OMA developed a similar procedure to interoperate with the NCBI LinkOut system by adopting its one-side interoperability approach. In addition, UniProtKB is not exchanging information with the LinkOut system. Nevertheless, UniProtKB accession numbers are integrated and part of the NCBI Gene KB by applying a two-side interoperability approach as reported in [50]. For instance, the “human HBB” NCBI Gene page refers to UniProtKB accession numbers in the “NCBI Reference Sequences (RefSeq)” and “Related Sequences” sections [51].

Bgee in the UniProt knowledge base

To interoperate with UniProt, we had to adopt its one-side interoperability approach that relies on a specific syntax and a text file format.

This file format can be interpreted as a CSV-like file where the separator is a semicolon followed with a space character (;) and without a header row. Nevertheless, the first cell contains values defined with a specific syntax, which is a “[UniProt identifier] [internal code] [external resource name]” where white spaces are actually three times the space character, the character-set encoding is us-ascii, and the internal code is composed of two letters. An example of a row (entry) in the UniProt information exchange format is shown below. In this example, the internal code used is “DR,” which is the two-letter code used by UniProt for cross-references.

```
A0A3B1E4W9 DR Bgee; WBGene00304181; Expressed
in pharyngeal muscle cell (C elegans) and 1
other tissue.
```

Mainly thanks to the identifiers in the first and second columns, we are able to establish an interoperation between Bgee and

UniProt. Consequently, for each UniProt protein entry, a corresponding Bgee gene entry is assigned with a link to a Bgee gene page. This link is built by prefixing the Bgee-related identifier (e.g., WBGene00304181) with the Bgee gene page web address at <https://bgee.org/gene/>. Moreover, the third column contains the description we defined for each Bgee entry. This description is composed of the cell or tissue where the gene is expressed the highest and of the number of tissues for which Bgee has expression data for this gene.

To generate the UniProt information exchange file, we developed a file writer that is part of the Bgee software and workflow. A new file is generated for each Bgee release. To ensure that UniProt has access to the latest Bgee data, we provide a persistent URL [52].

Other KBs’ experiences

OMA and GeneCards are also interoperating with UniProtKB by implementing the same approach as Bgee except that they do not use the description field to provide additional information as Bgee does.

Bgee in the GeneCards knowledge base

GeneCards is a KB that automatically integrates human gene-centric data from about 150 web sources, including genomic, transcriptomic, proteomic, genetic, clinical, and functional information. Unlike NCBI Gene, GeneCards does not provide guidelines or a specific file format for information exchange. The absence of a predefined data format for interoperability gave us the freedom to define one, as well as more flexibility about the information to exchange. First, before contacting GeneCards, we drafted a tab-separated value (TSV) file containing basic information from the Bgee database such as values that could be used as intersections between Bgee and GeneCards, Bgee gene page links, and short summaries about expression per gene. Moreover, to leverage our interoperability, we reused existing data exchange workflows between Bgee and other KBs. For instance, the TSV generated for GeneCards contains similar information as the file exchanged with UniProtKB. The strategy we adopted was to first present a simple file with minimal information about Bgee entries that could be easily understood and included in the GeneCards’ gene expression sections. With this strategy, our goal was to facilitate the discussions and to convince them to reuse our data. To further convince them, we also demonstrated our engagement and interest to publish links to their corresponding gene pages (or other data, if interested) on the Bgee website. Second, we contacted GeneCards (i.e., our potential data consumer) and presented them our solution for interoperability. Thanks to the simplicity, ease, and benefits of adopting our proposed interoperability solution, GeneCards’ maintainers quickly agreed with it, while suggesting a few changes. They then implemented a reader for this Bgee TSV file, resulting in the integration of gene expression information from Bgee as illustrated in Fig. 4. As a good practice, we also agreed to provide a persistent link pointing out the TSV file containing the latest Bgee data [53]. Therefore, each new GeneCards release has access to the latest Bgee data.

Although we had to define how we would perform the information exchange between Bgee and GeneCards from scratch, interoperating these two KBs did not require major efforts. For example, it required from us the presentation of a solution to an interoperability agreement, the exchange of five emails in total. The fact that the information being exchanged was minimal and simple was critical to the ease and rapidity of implementing an interoperation from scratch. Having established a prior data for-

mRNA Expression by UniProt/SwissProt for HBB Gene: P68871-HBB_HUMAN

Tissue specificity: Red blood cells.

Evidence on tissue expression from [TISSUES](#) for HBB Gene 

Heart(5), Blood(5), Muscle(5), Spleen(4.9), Liver(4.7) See all 19 »

Bgee gene expression patterns for HBB gene:

Expressed in trabecular bone tissue, vena cava, periodontal ligament and 202 other tissues.

Figure 4: An example of a Bgee link and a gene expression summary in the GeneCards “Expression” tab.

mat exchange such as the NCBI LinkOut system enables one to promote interoperability between KBs without requiring the implementation of a new data file reader each time a new resource appears to interoperate with it. As a result, the interoperation is straightforward for the data consumer, once the data producer complies with the consumer’s procedure and expectations. Therefore, the burden to perform interoperability is mostly put on the data producer side (i.e., one-side interoperability). The main drawback of the one-side interoperability is the lack of flexibility to add new information. For instance, with the LinkOut system, we are not able to exchange a description such as a summary of gene expression, as is the case with GeneCards, where we established a two-side interoperability.

Other KBs’ experiences

UniProtKB information is present in GeneCards because GeneCards unilaterally extracts data from UniProtKB by using UniProtKB’s one-side interoperability methods such as web application programming interfaces (APIs) [54]. On the other hand, OMA is not in GeneCards. However, OMA could establish a two-side interoperability approach similar to Bgee and be part of the “Orthologs” and “Paralogs” sections for each GeneCards entry.

Bgee in the OncoMX knowledge base

The OncoMX is a KB that integrates relevant datasets to support the research of cancer biomarkers [34]. Bgee provides OncoMX with healthy gene expression data thanks to a two-side interoperability approach. As a result, the Bgee dataset is available in the OncoMX web portal [55]. We, the Bgee team, have defined TSV data files that contain human and mouse gene expression present and absent calls per experiment condition (i.e., anatomical structures and developmental stages present in OncoMX) along with expression scores. Moreover, we reuse ontologies such as UBERON [56] for anatomical structures to avoid ambiguities and improve semantic interoperability. This facilitates the integration with other cancer biomarker-related data such as the differential expression dataset that OncoMX integrates. In addition, work toward a federated and automatic interoperability between Bgee and OncoMX has been done in the context of the Intelligent Open Data Exploration (INODE) project [57]. By applying this federated approach, we address most of the issues mentioned in the first paragraph of subsection File-based data exchange.

Other KBs’ experiences

UniProtKB accession numbers (a.k.a. identifiers) are also assigned per biomarker or gene entry in OncoMX. This was done by the OncoMX developers when integrating different datasets and using mappings between gene names and UniProtKB identifiers. Therefore, although it is limited, UniProtKB and OncoMX are interoperating with a one-side interoperability approach. This could be improved by also querying UniProtKB to retrieve relevant information for OncoMX such as associated diseases to a given biomarker.

Moreover, OncoMX could retrieve the GeneCards gene list and refer to the GeneCards entries (i.e., adding GeneCards cross-references) as was done for UniProtKB. Alternatively, UniProtKB and GeneCards could build a ready-to-use dataset for OncoMX by applying a two-side interoperability method as Bgee did. Therefore, it would be more informative for the OncoMX users rather than a simple cross-referencing between KBs. Finally, OMA could provide human–mouse orthologs to relate the integrated human and mouse gene expression data from Bgee in OncoMX and directly give further insights for biomarker researchers in the OncoMX portal.

Bgee database dumps and the RIKEN Metadatabase use case

The Bgee data dumps that contain the main processed information are a simplified version of the entire Bgee relational database. As a result, we provide a simplified view that excludes the complexity of the integrated raw data by providing explicit and processed gene expression information. Without this view, it would be difficult for the end user (including third-party computer tools) to understand and deal with a massive amount of data and the writing of complex queries to extract the needed information. These data dumps contain highly structured data based on a relational data model and another one based on the Resource Description Framework (RDF) [58] data model. The relational database dump is called EasyBgee, and Fig. 5 shows a portion of its data schema. We defined declarative mappings and applied the Ontop tool [59] to the EasyBgee database to generate the Bgee RDF dump [59, 60]. Therefore, the EasyBgee data are also available as RDF triple patterns, more specifically, using Turtle, the Terse RDF Triple Language, a concrete syntax for RDF [61].

We provide EasyBgee with both data models as a good practice to reach more users, to facilitate interoperability, and, consequently, to make the Bgee data more reusable. For example, having the RDF dump available enabled the Japanese Institute of Physical and Chemical Research (RIKEN) to directly import the Bgee knowledge graph into the RIKEN Bioresource Metadatabase as a named graph. This RIKEN database integrates several life science datasets to support researchers in making a comprehensive use of RIKEN’s research results. Thanks to this interoperation, RIKEN can now reuse Bgee gene expression data to support researchers when searching for a bioresource such as the use case for the Alzheimer disease study described in [62].

Other KBs’ experiences

Similarly to the Bgee use case, the RIKEN Metadatabase directly imports the available OMA RDF dump that is composed of fewer triples than the Bgee RDF dump. This cannot be the case of GeneCards, because it does not provide any RDF dump of its data. Furthermore, although importing the UniProtKB RDF dump to the RIKEN Metadatabase may be prohibitive because of its size (>100

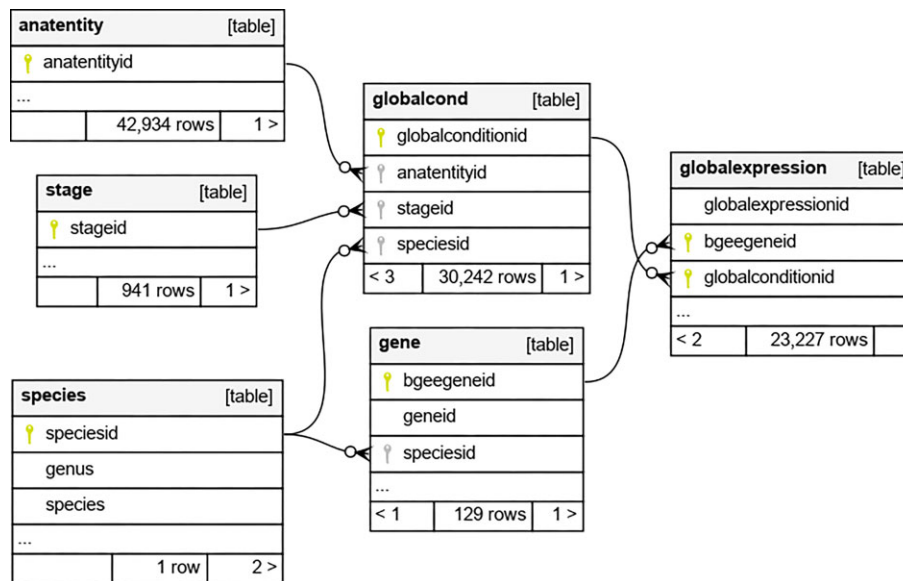


Figure 5: A portion of the EasyBgee relational data schema.

billion of triples), UniProtKB provides views of its RDF data that could be reused by RIKEN, such as human diseases datasets.

Bgee download files: a one-side interoperability approach

As a one-side interoperability approach, Bgee provides views of its data as per-species TSV files. These files contain gene expression calls of presence/absence of expression and processed expression values that are currently used as the interoperability method with Bgee in different use cases. For example, the Monarch Initiative project aims at connecting phenotypes (e.g., diseases) to genotypes (e.g., genes causing a disease) [36] and uses the Bgee download files to retrieve associations between genes and the anatomical entities they are expressed in. This information is then displayed on their website in an “Anatomy” section of each gene page entry. Similarly, the Bgee download files are used to build a precision medicine open knowledge graph for a system called SPOKE [37] by retrieving association between genes and the anatomical entities where they are up- or downregulated to generate new edges in the graph. The downloaded files are also used to build a knowledge base of gene expression information relevant to drug development in the context of the Open Systems Pharmacology suite [38], where Bgee provides a reference of normal gene expression in healthy conditions for multiple species, including human.

Other KBs’ experiences

As a one-side interoperability approach, OMA, UniProtKB, and GeneCards provide downloaded files to exchange information, including views of their data, respectively, in [63], [64], and [65].

Programmatic interfaces

Providing several ways to programmatically interoperate and work with the data and information contained in a KB facilitates its reusability. This is because an interoperation method may be more suitable than another one, depending on the user skills and use cases. In this regard, Bgee provides three distinct programmatic interfaces to query and to manipulate its data: a SPARQL endpoint, R packages, and a web API. For the latter one, although the web API is already available [66], we are still working on provid-

ing a documentation and to be fully compliant with the OpenAPI [67] specification and standard to improve interoperability.

SPARQL is a structured query language and protocol for RDF-based data. Based on the Bgee RDF data dump depicted in subsection File-based data exchange, a SPARQL endpoint is available in [68]. The query results can be retrieved in different formats such as JSON or CSV. Queries can be also executed through application programming interfaces (APIs) in several programming languages (e.g., Python via SPARQLWrapper [69]). Moreover, SPARQL also enables one to perform federated join queries to combine various KBs that also provide the SPARQL 1.1 endpoint such as Wikidata and UniProtKB. This capability of performing federated queries with Bgee is extensively demonstrated in [60].

The BgeeCall R package allows the user to generate present/absent gene expression calls without using an arbitrary cutoff (e.g., 1 TPM) by estimating background transcriptional noise based on nonexpressed genomic features (i.e., intergenic sequences). We also provide the BgeeDB package for the annotation and gene expression data download from the Bgee database (i.e., interoperation) and for TopAnat analysis, a GO-like enrichment of anatomical terms mapped to genes by expression patterns. Both packages along with their documentations are accessible at the Bgee website [70] and from Bioconductor [71]. Therefore, we reach Bioconductor’s users, who are interested in reusing gene expression-related data. To facilitate the reuse of these packages, we make available a Docker container [72], a lightweight, standalone, executable package of software that includes all dependencies. In addition, other package repositories and systems interoperate with Bioconductor such as BioContainers [73]. As a result, we are able to attain a wider R user community. For example, so far, the BgeeDB package was downloaded around 325,000 times from BioContainers [74], which is significantly more than the 14,000 downloads from Bioconductor in the past 7 years.

Other KBs’ experiences

UniProtKB and OMA provide SPARQL endpoints to access their data too. On the other hand, GeneCards does not have one. Having a SPARQL endpoint would facilitate, for example, the GeneCards

interoperation with UniProtKB, OMA, and Bgee as extensively demonstrated in [60]. In addition, OMA provides a R package in Bioconductor and REST APIs. Moreover, the OMA R package (OmaDB) was downloaded more than 67,000 times from BioContainers in the past 5 years. On the other hand, UniProtKB and GeneCards do not provide a R package authored and maintained by them, but UniProtKB makes available REST APIs. Both KBs could benefit bioinformaticians by providing a ready-to-use R package. We highlight that R language is one of the most used languages in bioinformatics. To illustrate the relevance of doing this for reusability, a third-party R package that is not maintained by UniProtKB, called “UniProt.ws: R Interface to UniProt Web Services,” is available in Bioconductor [75] and has been downloaded more than 280,000 times from BioContainers.

Automatizing interoperability

The ideal interoperability of KBs is the one that allows seamless information exchange between them, in a way that looks like a unique system. To achieve this smooth and continuous information exchange, we aim for automatizing interoperability. For example, we apply this approach to interoperate Bgee with Wikidata, Wikipedia, OMA [16], Google Dataset search engine [76], and OncoMX [77]. This approach addresses several issues of the file-based interoperability mentioned in subsection File-based data exchange, such as asynchronous and independent exporting and importing data operations.

To automatize interoperability between KBs, one of the steps in common between different solutions is to provide structured data, often by using interoperability standards to solve syntax and semantic heterogeneities. To do so, Bgee applies one-side, two-side, or multi-side interoperability depending on the use case, as described in the next paragraphs.

Bgee in the Wikidata knowledge base

Wikidata is an open and free KB that can be read and edited by any agent (i.e., both humans and machines). It contains and acts as a central storage of structured data related to other Wikimedia projects, including Wikipedia. The Wikidata contents are available under a free license (i.e., CC0 [78]), can be exported using data standard formats, and can be interlinked to other data sets on the web of linked data. Its contents include highly relevant life science data. Because Bgee data are also licensed under CC0, there is no restriction to reuse them in Wikidata. Figure 6 shows a part of the Wikidata graph, including Bgee data. To interoperate with Wikidata, we developed a bot that automatically extracts data from the Bgee relational database, structures them with the Wikidata data model [79], and loads them into the Wikidata KB. The bot is written in Python with the WikidataIntegrator library [80] and is available in our GitHub repository [39]. This bot inserts to Wikidata gene entries’ “expressed in” statements. For example, see the “expressed in” statements on the INS gene Wikidata page [81] and Fig. 7, where 1 insertion is illustrated. Note that we defined versioned and persistent links as references to the “expressed in” statements, which is a good practice in order to track information provenance. Currently, only existing Wikidata gene entries from Ensembl and Wikidata anatomic entities (e.g., stomach) with a stated corresponding UBERON ontology term are considered (including cell ontology). Thus, not all data in Bgee are inserted into Wikidata. The Bgee gene entries for the species in common with Wikidata are identified with Ensembl gene IDs. We do so to avoid ambiguities and to accurately include gene expres-

sion calls in Wikidata. Thus, the UBERON ontology and Ensembl gene identifiers allow us to address semantic heterogeneities.

Wikidata defines a specific data model to organize data and provides relations between properties in Wikidata and in RDF [79]. To interoperate with Wikidata, we must reuse existing Wikidata schemas or propose new ones based on the Wikidata model. Further instructions are available in [82]. Moreover, massive data insertion through a Wikidata bot such as the Bgee bot requires granted permissions by the Wikidata community [83]. Once this authorization was granted, we were able to automatically insert and update the Bgee data in Wikidata entries. Permissions are often granted based on Wikidata contributors’ support; currently, the Bgee bot is supported by three different Wikidata users [84]. Therefore, we performed a one-side interoperability because we had to strictly comply with the Wikidata procedure for interoperability. In addition, running the Bgee Wikidata bot is part of the Bgee pipeline final steps for each new release.

Other KBs’ experiences

GeneCards is a commercialized KB, and UniProtKB and OMA are not CC0 licensed KBs; hence, they are not compatible with Wikidata’s copyright requirements. Therefore, in principle, these KBs cannot interoperate with Wikidata. Nevertheless, non-CC0 KBs can donate part of their data under the CC0 license. Consequently, they can interoperate with Wikidata by improving open knowledge reuse and increasing the traffic to these KBs. For example, currently, the presence of UniProtKB is limited to identifiers in Wikidata protein entries (i.e., as cross-references). These cross-references are fed to Wikidata with a third-party bot, *ProteinBoxBot* [85]; thus, it is not maintained by UniProtKB.

Bgee in Wikipedia

The interoperation between Bgee and Wikipedia is fully automatic. From the end-user perspective, the anatomical structures such as the pancreas where a gene is expressed, along with links to the corresponding Bgee gene pages, are included in the information box (infobox) of each Wikipedia gene article in English, as illustrated in Fig. 8. To do so, we implemented a Lua [86] script that is defined in the Wikipedia gene infobox module [41], which retrieves structured data from Wikidata. Thus, this script queries Wikidata to fetch Bgee gene expression information and display it in the infobox. Since Bgee data were added by the Wikidata bot described above, the interoperation with Bgee is done indirectly through Wikidata. A highly relevant benefit of doing this is that changes in Wikidata are promptly available in the Wikipedia gene articles. Similar to the Wikidata use case, changes in the Wikipedia infobox module code require permissions granted by the Wikipedia community and full compliance with their interoperability procedure, hence, a one-side interoperability approach. Nevertheless, a test environment so-called sandbox, for a given infobox module, is provided where, in principle, anyone can edit it [87].

Other KBs’ experiences

OMA is not present in Wikidata; hence, its data are not accessible by the Wikipedia Gene infobox module as in the Bgee use case. UniProtKB identifiers are referred in the Wikipedia gene pages’ infobox thanks to their availability in Wikidata. GeneCards is also mentioned and links are built based on the gene names retrieved from Wikidata. However, none of these KBs (namely, OMA, GeneCards, and UniProtKB) provide meaningful ready-to-use information for the Wikipedia users that is more than a simple external KB link. Moreover, the UniProtKB and GeneCards links in

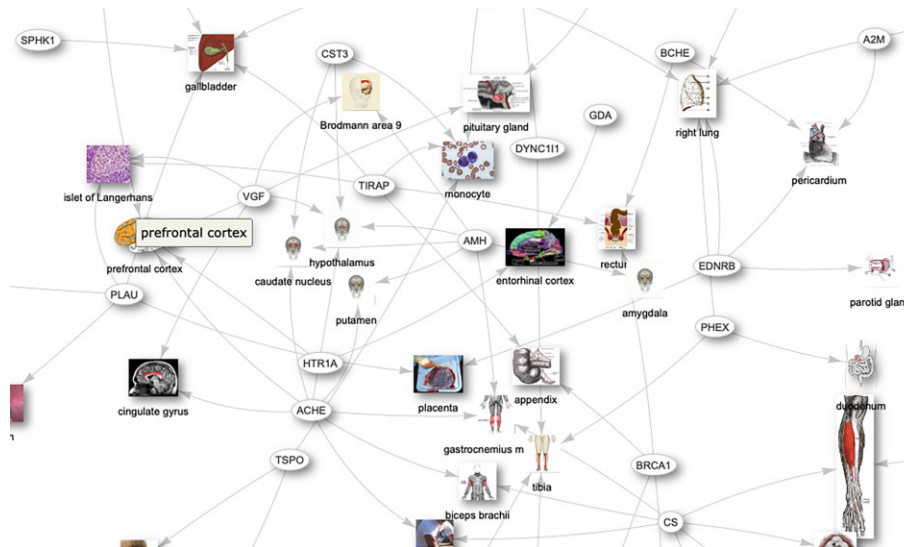


Figure 6: A portion of the Bgee data integrated into the Wikidata knowledge graph. It illustrates genes expression calls, where edges represent “expressed in” statements.

expressed in by BgeeDB-bot and Tarsmf	beta cell
	series ordinal 1
	1 reference
	stated in Bgee
	retrieved 20 July 2022
	reference URL https://bgee.org/bgee15_0/gene/ENSG00000254647

Figure 7: An “expressed in” statement entry in Wikidata by Bgee including provenance via a Bgee versioned URL. This image was extracted from the following Wikidata page, <https://www.wikidata.org/wiki/Q21163221>.

the Wikipedia Gene infoboxes are maintained by third-party contributors (i.e., nonauthoritative source). Therefore, these KBs and other bioinformatics KBs can learn from this Bgee experience to improve their knowledge reuse.

Bgee in the Google Dataset Search engine

This use case is an example of a multi-side interoperability approach. Google Dataset Search engine fully automatizes the process to index and to retrieve metadata from webpages that contain Schema.org structured data. Figure 9 depicts a search of “Homo sapiens gene expression” datasets in this Google tool. Notice that Schema.org is not exclusively under the authority of Google or Bgee. Therefore, producing and consuming Schema.org structured data is in principle independent of the interoperable parts. This further allows other data consumers to reuse the data once they comply with the Schema.org approach. The compliance with a global data schema, data model (e.g., RDF graph), and syntax (e.g., JSON for Linked Data—JSON-LD [88]) intrinsically solves semantic and syntactical heterogeneities among interoperable parts. Founded by Google, Microsoft, Yahoo, and Yandex, Schema.org vocabularies are developed by an open community process, using a mailing list [89] and through GitHub. Drawbacks of this approach include a lack of flexibility and that reaching an


agreement for changes is not straightforward. Hence, it greatly limits the information we are able to exchange. For instance, as of 2 March 2023, in the Schema.org GitHub, there are more than 700 issues open, some of them since 2014, and about 1,300 closed [90].

Implementation details

Google Dataset solely considers the Dataset, Datacatalog, and Download concepts and their properties from Schema.org [91]. Therefore, interoperation with Bgee is restricted to these concepts. Although Taxon, Gene, and “Anatomical structure” Schema.org concepts are not considered by Google Dataset Search, we also provide them via a JSON-LD embedded script at each Bgee gene page, and they can be consumed by any tool implementing this multi-side interoperability approach. Figure 10 shows “expressed in” statements structured with Schema.org and included as a script in the human insulin Bgee gene page.

Other KBs’ experiences

GeneCards and UniProKB do not use Schema.org to describe their datasets with metadata embedded in their webpages. Therefore, datasets such as those in [64] are not directly available with Google Dataset Search. Similar to Bgee, OMA implements



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute

Help
Learn to edit
Community portal
Recent changes
Upload file

Tools

What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export

Download as PDF
Printable version

In other projects

Wikimedia Commons

Languages ⚙

Deutsch
Español
Français
हिन्दी
Italiano
Norsk bokmål
Português

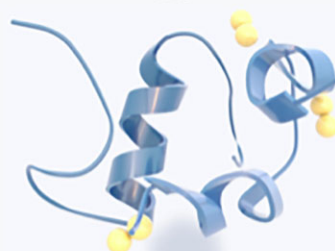
Insulin

From Wikipedia, the free encyclopedia

*This article is about the naturally occurring protein. For uses of insulin in treating diabetes, see [insulin \(medication\)](#).
Not to be confused with [Inulin](#).*

Insulin
(/ˈɪn.sʊ.lɪn/^[5]^[6] from Latin *insula*, 'island') is a **peptide hormone** produced by **beta cells** of the **pancreatic islets** encoded in humans by the *INS* gene. It is considered to be the main **anabolic hormone** of the body.^[7] It regulates the **metabolism** of **carbohydrates**, **fats** and **protein** by promoting the absorption of **glucose** from the blood into **liver**, **fat** and **skeletal muscle** cells.^[8] In these tissues the absorbed glucose is converted into either **glycogen** via **glycogenesis** or **fats** (triglycerides) via **lipogenesis**, or, in the case of the liver, into both.^[8]

INS



Available structures					
PDB	Ortholog search: PDBe RCSB				
	List of PDB id codes [show]				
Identifiers					
Aliases	INS, IDDM, IDDM1, IDDM2, ILPR, IRDN, MODY10, insulin, PNDM4				
External IDs	OMIM: 176730 MGI: 96573 HomoloGene: 173 GeneCards: INS				
	Gene location (Human) [show]				
	Gene location (Mouse) [show]				
RNA expression pattern [hide]					
Bgee ↗	<table style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f2f2f2;"> <th style="width: 50%; padding: 2px;">Human</th> <th style="width: 50%; padding: 2px;">Mouse (ortholog)</th> </tr> </thead> <tbody> <tr> <td style="padding: 2px;"> Top expressed in <ul style="list-style-type: none"> beta cell body of pancreas right lobe of liver Right adrenal gland Left adrenal </td> <td style="padding: 2px;"> Top expressed in <ul style="list-style-type: none"> Islet of Langerhans pyloric antrum yolk sac retinal pigment epithelium secondary oocyte quadriceps femoris muscle </td> </tr> </tbody> </table>	Human	Mouse (ortholog)	Top expressed in <ul style="list-style-type: none"> beta cell body of pancreas right lobe of liver Right adrenal gland Left adrenal 	Top expressed in <ul style="list-style-type: none"> Islet of Langerhans pyloric antrum yolk sac retinal pigment epithelium secondary oocyte quadriceps femoris muscle
Human	Mouse (ortholog)				
Top expressed in <ul style="list-style-type: none"> beta cell body of pancreas right lobe of liver Right adrenal gland Left adrenal 	Top expressed in <ul style="list-style-type: none"> Islet of Langerhans pyloric antrum yolk sac retinal pigment epithelium secondary oocyte quadriceps femoris muscle 				

Figure 8: A Wikipedia gene article contained gene expression information from Bgee. It was extracted from <https://en.wikipedia.org/wiki/Insulin>.

Schema.org for datasets, but only a few of those listed in [63] are described. Moreover, OMA could provide richer metadata as Bgee has done, for instance, by assigning the downloadable forms that would allow users and software tools to directly access and download the files.

Bgee and OncoMX: a knowledge base federation

Federating data sources is the capacity of uniformly accessing data from distinct and potentially heterogeneous data sources without needing to physically move the data from them (i.e., data virtualization). Thus, the evaluation of queries is directly and real-time performed on the original data sources. By uniformly, we mean users see the data as if they were available in a single data source.

To address drawbacks of the file-based interoperability between Bgee and OncoMX as discussed in subsection File-based data exchange, we federate both KBs in the context of the IN-ODE project [57] by applying data virtualization and one-side interoperability approaches. To implement this federation, we use

Teiid [92], a real-time integration engine that supports a high query volume and transactions. The information to be exchanged with OncoMX is covered by EasyBgee, a materialized view of the Bgee relational database as described in subsection File-based data exchange and its data schema illustrated Fig. 5. Thus, in practice, we interoperate OncoMX with EasyBgee. This is done in order to optimize query performance, because the native Bgee relational database does not explicitly provide the information needed in this federation. To define the virtual database (VDB) (i.e., the OncoMX and EasyBgee federation) with Teiid, we have to write an XML file that captures information about the VDB, the sources it integrates, and preferences for importing metadata [93]. The XML can also embed Data Definition Language (DDL) statements (i.e., some SQL commands). Figure 11 depicts a portion of the XML file to set up a VDB based on OncoMX and EasyBgee. The OncoMX SQL database dump and the VDB XML file are available in [94].

As a result, the integration of those two data sources is no longer a manual effort, nor does it involve data duplication. Hence, the main advantage of federating by setting up a VDB is that

The screenshot shows a Google search for 'homo sapiens gene expression'. The search results page displays '100+ datasets found'. Three datasets are listed:

- Homo sapiens : Homo sapiens Transcriptome or Gene...** from datamed.org, updated Jul 7, 2021.
- Homo sapiens gene expression simple** from bgee.org, updated Jul 1, 2021. This dataset is highlighted with a blue border.
- Homo sapiens gene expression advanced** from bgee.org.

The right-hand side of the page provides details for the selected dataset, 'Homo sapiens gene expression simple':

- Explore at:** bgee.org
- Dataset updated:** Jul 1, 2021
- Dataset authored and provided by:** The Bgee Team
- License:** CC0 1.0 Universal Public Domain Dedication
- Description:** Anatomical entities only, file without advanced columns.

Figure 9: Searching for human gene expression datasets and retrieving Bgee datasets via the Google Dataset Search engine.

Gene	
0 ERRORS 0 WARNINGS ^	
ID: https://bgee.org/gene/ENSG00000254647/	
@type	Gene
@id	https://bgee.org/gene/ENSG00000254647/
description	insulin [Source:HGNC Symbol;Acc:HGNC:6081]
alternateName	iddm2
alternateName	iddm1
identifier	ENSG00000254647
name	INS
expressedIn	
@type	AnatomicalStructure
@id	http://purl.obolibrary.org/obo/UBERON_0002167
identifier	UBERON:0002167
name	right lung
expressedIn	

Figure 10: An example of the Bgee gene expression data structured with Schema.org.

whenever either Bgee or OncoMX gets updated, the new data are immediately available, as opposed to manual interoperation

via TSV files. Moreover, it facilitates interoperability maintenance mainly because only relevant data modifications or changes at the data schema of data sources are required. It is often possible to perform needed changes by editing the VDB configuration, for instance, by creating SQL views as illustrated in Fig. 11. For example, currently, only human and mouse gene expression data are exchanged with OncoMX; nonetheless, if another species available in Bgee is of interest, an easy fix is to change the SELECT statement by adding the species taxon identifier in the listing (9606, 10090). Semantic and data heterogeneities may also be addressed by creating views and using SQL built-in functions to perform data transformations (e.g., concatenation of columns) during query evaluation. Therefore, keeping interoperation properly functional becomes an easier task, with no need to execute massive data export and import operations to deploy changes. In addition, we solve data storage heterogeneity, as shown in Fig. 11, since EasyBgee is a native MySQL database and OncoMX a PostgreSQL one.

Finally, in the context of the INODE project [57], we provide an ontology-based data access to the OncoMX-Bgee federation with the Ontop tool. This is done mainly to achieve two aims: to improve the semantics of the data sources by applying ontologies to reduce ambiguities and to provide the possibility of performing federated SPARQL queries, hence the capacity to interoperate with other knowledge graphs through its SPARQL endpoint [95].

Other KBs' experiences

To the best of our knowledge, none of the KBs discussed in this article except from Bgee and OncoMX implements a federation over relational databases that are owned and managed by different organizations.


```

<vdb name="oncomx_federation" version="1">
  <model visible="false" name="easybgee_v15">
    <source name="easybgee_v15" translator-name="mysql5"
      connection-jndi-name="java:/easybgee_v15"/>
    <metadata type="NATIVE"></metadata>
  </model>
  <model visible="true" type="VIRTUAL" name="oncomx_bgee">
    <source name="easybgee_v15" translator-name="mysql5"
      connection-jndi-name="java:/easybgee_v15"/>
    <metadata type="DDL"><![CDATA[
      CREATE VIEW species (
        speciesId long NOT NULL,
        genus string(70) NOT NULL,
        species string(70) NOT NULL,
        speciesCommonName string(70) DEFAULT '' )
      AS SELECT speciesId, genus, species, speciesCommonName
      FROM easybgee_v15.species WHERE speciesId in (9606,10090);
      ...]]>
    </metadata>
  </model>
  <model visible="true" name="oncomx_v1_0_36">
    <property name="importer.tableTypes" value="TABLE,VIEW"/>
    <property name="importer.schemaName" value="oncomx_v1_0_36"/>
    <source name="oncomx_v1_0_36" translator-name="postgresql"
      connection-jndi-name="java:/oncomx_v1_0_36"/>
    <metadata type="NATIVE"></metadata>
  </model>
</vdb>

```

Figure 11: A portion of the virtual database (VDB) configuration file set up to federate OncoMX and Bgee KBs. With the XML element `<model>`, we define the data source and metadata (e.g., data schema) to be considered. Some property/attribute values can also be assigned, for example, to set if a model should be visible or not in the federation. The metadata type asserted as `NATIVE` (i.e., `<metadata type="NATIVE"></metadata>`) means the database metadata (e.g., data schema) will be considered exactly as it is originally defined in the data source. It is also with the `<metadata>` XML element we can create views and, consequently, new data schemas to structure the underlined data. This can be done through Data Definition Language (DDL) statements as shown above in the second `<metadata>` tag definition. The VDB XML file is fully documented in [93].

OMA in Bgee

Although we are focusing on the data provider side when considering two interoperable parts, in this subsection, we want to illustrate the interoperation from a data consumer perspective and to further justify the applicability of the interoperability approaches discussed here by another KB too. OMA is a KB that contains information about evolutionary relationships among genes across species such as orthologs. Orthologs are genes in different species that evolved from a common ancestral gene by speciation. Bgee integrates OMA evolutionary relationships to further enable gene expression comparison among species.

To do so, Bgee interoperates with OMA through the OMA SPARQL endpoint. A tool along with several SPARQL queries based on the ORTHology ontology was developed to extract from the hierarchical orthologous group the pairwise orthology and paralogy relations [96]. The code source is available in the Bgee pipeline GitHub repository [97]. This tool is currently part of the Bgee pipeline for each new release, ensuring a semiautomatic procedure to exchange OMA information with Bgee. It is semiautomatic mainly because our one-side interoperability approach to include OMA data in Bgee is independent of OMA KB management, and it does not allow real-time updates. Therefore, significant changes in OMA will not be fully automatically considered by our specialized tool. However, most of the relevant changes can be done by editing the SPARQL queries or configuration-related files, for example, modifications in the OMA data schema (that does not happen often; last time was more than 2 years ago) may require modifying

the SPARQL queries. Therefore, we facilitate interoperability maintenance and perform a quasi-seamless interoperability with OMA.

Finally, with this use case, we want to emphasize that in a one-side interoperability approach, either data producer or consumer may impose the way the interoperation is done. However, the fact of providing different ways to exchange information, bioinformatics KBs like OMA and Bgee, facilitates data reusability because it gives the possibility to choose which method is the more suitable by the data consumer. For example, OMA provides different programmatic interfaces such as Python libraries, REST APIs, and SPARQL, which further allows automatizing interoperability. Therefore, the interoperability approach is imposed but with choices.

Final implementation considerations

As a disclosure, when multiple options to interoperate with a target KB were available, we chose the one that was easier and faster for us to implement and did not compromise the minimal information we wanted to exchange. Otherwise, we implemented the only viable option for Bgee to be present in a target KB at that moment. For example, NCBI LinkOut system supports two data syntax, CSV and XML; we chose CSV because it was easier for us to execute a single SQL query over our relational database and directly get the Bgee data in the expected tabular format for NCBI. Another example is related to our presence in Wikipedia that was first limited to including a few top organs where a human gene was expressed that were already meaningful and valuable infor-

mation for Wikipedia Gene pages. We did this to simplify discussion with Wikipedia moderators at first. Once a minimal interoperability was thus established between Bgee and Wikipedia, improvements were easier to perform and to deploy. Currently, Wikipedia also reuses Bgee mouse gene expression data, and cells are considered in addition to organs and tissues.

Ten lessons learned on improving the reusability of bioinformatics KBs

Lesson 1—Partial interoperability is better than none

We argue that it is better to have some extent of interoperability rather than none in order to increase (re)usability of a KB. A perfect and automatic interoperability between independent KBs is often hard to achieve due to issues such as legacy systems and practices, lack of resources (e.g., human resource allocation and skills, technology acquisition), technical incompatibilities, or reconciliation difficulties. Thus, aiming for partial interoperability when full interoperability is not feasible in the short term allows for delivering at least minimal data reuse along with the possibility to improve it over time.

Lesson 2—Iteratively improving the information exchange is better than trying to achieve full interoperability at once

Having less information to exchange can significantly simplify discussions between independent KB delegates and interoperability. A KB delegate will be more keen to perform an information exchange that is simple and easy to understand and implement than a complex one. This simplification usually results in partial interoperability but with the great benefit of being present and interoperating with a target KB. This target KB will be potentially more prone to accept and implement improvements later. Thus, once a minimal interoperability is established between KBs, improvements are easier to perform and to deploy.

Lesson 3—Reusability implies better visibility and vice versa

KB dissemination through information exchange with other KBs fosters better visibility and, consequently, more reusability. This is because data are reused not only by an external KB but also, potentially, by its own user community and related software tools. Moreover, by providing provenance of the reused data, users may access and discover the original data source and enable more interoperability and data reuse.

Lesson 4—Interoperability requires maintenance

Similar to a software development life cycle that includes maintenance, ensuring long-term interoperability needs maintenance too. Therefore, interoperability efforts among KBs should ideally continue as long as they reuse data from each other. This also improves chances of reusing the latest data and of better quality and quantity of information exchanged. To reduce maintenance efforts, it is important to consider as a first step, before implementation, which interoperability approach is the most suitable given the requirements and constraints of the interoperable partners.

Lesson 5—Automatize interoperability as much as possible

There are various benefits of automatizing interoperability, notably (a) reducing maintenance efforts, (b) providing real-time processing, and (c) facilitating scalability. An automatic approach significantly reduces maintenance efforts because once the data are up-to-date in a KB, the changes are propagated to other in-

teroperable KBs and potentially at real time, for instance, via the execution of a bot. Moreover, real-time processing is an interoperability feature that the information received is processed by the data consumer almost immediately. To further explain (c), we can highlight KB federation approaches. By federating KBs, information exchange is easily scalable because adding a new KB or new type of information and data is done by editing configuration files and defining mappings between data sources (e.g., data schema alignments, data transformation functions).

Lesson 6—Be flexible when choosing and providing interoperability approaches

Defining how to establish interoperability depends on which methods are possible and available to the KBs. Moreover, different technical and resource constraints and KB delegates' skills may favor different approaches. For example, automatizing interoperability requires KB delegates' technical skills that are not necessarily available. This can be alleviated by documentation (Lesson 10). In addition, a KB that makes one-side interoperability available should provide, if possible, distinct ways to exchange information. By doing so, a KB increases its reusability because it eventually matches users' skills and addresses third-party KB constraints to achieve interoperability. Moreover, by having reusability as a main KB goal, the data producer or consumer should be prepared to make concessions, and the target KB delegates are also less prone to collaborate and, consequently, to implement an interoperability if they have to do more work. Thus, it is important to drastically reduce their workload when performing interoperability (e.g., by providing ready-to-use data according to the target KB practices).

Lesson 7—Focus on knowledge base delegates

When a fully automated interoperability is not possible, a KB interoperability may only be possible, if it is preceded by communication between the delegates of KBs. This should lead them to collaborate with each other to interoperate their KBs. It is thus important to focus on establishing a good work relation between representatives of the interoperable KBs. Without this human reconciliation aspect, it can be difficult to get any technical interoperability mechanism to work. Moreover, any application of two-side interoperability approaches would not be possible without representatives' reconciliation.

Lesson 8—There is a positive domino effect of knowledge base interoperability

Interoperating with another KB can lead to a more and more complete network of information exchange among KBs, which is a positive "domino effect" for interoperability. This is thanks to 3 main reasons: (i) potential transitivity of interoperability (i.e., A interoperates with B that interoperates with C then A interoperates with C too), (ii) providing a positive example to convince further KB delegates to interoperate, and (iii) possible reuse of interoperability procedures. To further illustrate the latter one, having already available interoperability solutions, such as a data virtualization solution for federating KBs, leverages scalability for including new KBs in a KB interoperability network. Moreover, actively participating or collaborating with multiside interoperability initiatives contributes to this positive domino effect. In summary, a continuous alignment with external resources propels data reusability.

Lesson 9—Adopt the most appropriated license

Although this lesson subject is already discussed in many distinct contexts including in the FAIR principles (i.e., "(Meta)data are re-

leased with a clear and accessible data usage license” [2]), we want to reinforce the importance of adopting as much as possible the least restrictive license for a KB. As a result, we eliminate possible legal interoperability issues, and hence, we can focus on addressing technical and human aspect issues to exchange information. Nevertheless, we recognize the fact that some KBs need to apply highly restrictive licenses. A reason for this might be to protect the authors’ data ownership, but in fact, it hinders many possibilities for interoperability and open science [98].

Lesson 10—Provide documentation, training, and tutorials for interoperability

To leverage one-side interoperability approaches, a KB should provide well-documented technical solutions, in-practice tutorials, and training to facilitate reusability. Moreover, having communication channels with prompt responses is extremely important to ensure a continuous user engagement, which includes external KB representatives.

Conclusion

To conclude, we argue that the best interoperability approach is the one that gets implemented despite partial interoperability or dissent between KB representatives. Therefore, one-, two- and multi-side interoperability methods are all relevant to promote KB data (re)use. Nevertheless, providing and implementing one or more of these methods should not be considered final solutions for interoperability. This is because KBs, information exchange technologies, and practices evolve, in addition to the apparition of new KBs. Finally, we illustrated with the Bgee KB several interoperability approaches and how we implemented them. We further illustrated how these approaches are transferable to other KBs by highlighting similar implementations by major bioinformatics KBs such as UniProtKB and what could be improved, when it is applicable. This allowed us to provide guidelines through pragmatic examples of how to interoperate with a variety of biological and general-purpose KBs such as Wikipedia.

Authors’ Contributions

Conceptualization: T.M.F. Software: T.M.F., J.W., F.B. Investigation: T.M.F. Supervision: self-organization. Writing, original draft: T.M.F. Writing, review and editing: all authors. Funding acquisition: T.M.F., F.B., M.R.R.

Data Availability

For more on the Bgee database see <https://bgee.org/>, the Bgee-UniProtKB cross-reference file <https://purl.org/bgee-uniprotkb>, the Bgee-GeneCards cross-reference file <https://purl.org/bgee-geneCards> and the Bgee-NCBI Gene database cross-reference file <https://purl.org/bgee-ncbigene>

Abbreviations

API: application programming interface; CSV: comma-separated value; DB: database; DDL: Data Definition Language; DOI: Digital Object Identifier; DTD: Document Type Definition; FAIR: Findable, Accessible, Interoperable, and Reusable; infobox: information box; INODE: Intelligent Open Data Exploration; JSON: JavaScript Object Notation; JSON-LD: JavaScript Object Notation for Linked Data; KB: knowledge base; NCBI: National Center for Biotechnology Information; OMA: Orthologous Matrix; OWL: Web Ontology Lan-

guage; RDF: Resource Description Framework; RIKEN: Japanese Institute of Physical and Chemical Research; SKOS: Simple Knowledge Organization System Reference; SPARQL: SPARQL Protocol and RDF Query Language; SWRL: Semantic Web Rule Language; SQL: Structured Query Language; TB: terabytes; TPM: transcripts per million; TSV: tab-separated value; Turtle: The Terse RDF Triple Language; UID: Unique Identifier; VDB: Virtual Database; VoIDext: Extended Vocabulary of Interlinked Datasets; XML: Extensible Markup Language; XSD: XML Schema Definition.

Competing Interests

The authors declare that they have no competing interests.

Funding

SIB Swiss Institute of Bioinformatics, Canton de Vaud, Swiss National Science Foundation [173048, 207853, 167149], National Institutes of Health (NIH) Award [U01CA215010 (OncoMX)], European Union’s Horizon 2020 research and innovation program [863410], and State Secretariat for Education, Research and Innovation (SERI) via ETHZ grant BG 02-072020. Funding for open access charge: Swiss National Science Foundation.

Acknowledgments

We thank the representatives of all knowledge bases we are in contact with for continuously supporting the Bgee interoperability network. We also acknowledge the work of all past and present members of the Bgee team that made it possible for Bgee to be a successful knowledge base.

References

1. Ancona M, Cagetti P, Castagna P, et al. Reusable distributed “data environments.” In: Proceedings of the 1992 ACM/SIGAPP Symposium on Applied Computing Technological Challenges of the 1990’s—SAC ’92. Kansas City, MO: ACM Press; 1992;1083–90. <http://portal.acm.org/citation.cfm?doid=130069.130134>.
2. Wilkinson MD, Dumontier M, Jan Aalbersberg I, et al. Addendum: the FAIR guiding principles for scientific data management and stewardship. *Sci Data* 2019;6(1):6.
3. Jacobsen A, de Miranda Azevedo R, Juty N, et al. FAIR principles: interpretations and implementation considerations. *Data Intell* 2020;2(1–2):10–29.
4. Mons B, Neylon C, Velterop J, et al. Cloudy, increasingly FAIR; revisiting the FAIR data guiding principles for the European Open Science Cloud. *Inform Serv Use* 2017;37(1):49–56.
5. Beránková M, Kvasnička R, Houška M. Towards the definition of knowledge interoperability. In: 2010 2nd International Conference on Software Technology and Engineering, Vol. 1. San Juan, PR, USA. 2010; V1–232–V1–236. <https://doi.org/10.1109/ICSTE.2010.5608843>.
6. IEC 62243:2012(E) (IEEE Std 1232-2010): Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE). In: IEC 62243 Second Edition 2012-06 IEEE Std 1232. 2012; 1–172. <https://doi.org/10.1109/IEEESTD.2012.6228492>.
7. Edmunds M, Peddicord D, Frisse ME. Ten reasons why interoperability is difficult. 127–37. In: C. Weaver, M. Ball, G Kim, J. Kiel(eds.) Healthcare Information Management Systems: Cases, Strategies, and Solutions. Health Informatics, Springer, Cham. 2016. https://doi.org/10.1007/978-3-319-20765-0_7.

8. Benson T, Grieve G. Why interoperability is hard. In: Principles of Health Interoperability: FHIR, HL7 and SNOMED CT. Health Information Technology Standards, Springer, Cham. 2016; 21–40. https://doi.org/10.1007/978-3-319-30370-3_2.
9. Diallo SY. On the complexity of interoperability. In: Proceedings of the Modeling and Simulation of Complexity in Intelligent, Adaptive and Autonomous Systems 2016 (MSCIAAS 2016) and Space Simulation for Planetary Space Exploration (SPACE 2016) MSCIAAS '16. San Diego, CA: Society for Computer Simulation International; 2016.
10. Kadadi A, Agrawal R, Nyamful C, et al. Challenges of data integration and interoperability in big data. In: 2014 IEEE International Conference on Big Data (Big Data). Washington, DC, USA. 2014; 38–40. <https://doi.org/10.1109/BigData.2014.7004486>
11. Bastian FB, Roux J, Niknejad A, et al. The Bgee suite: integrated curated expression atlas and comparative transcriptomics in animals. *Nucleic Acids Res* 20 20;49(D1):D831–47.
12. Oza VH, Whitlock JH, Wilk EJ, et al. Ten simple rules for using public biological data for your research. *PLoS Comput Biol* 2023;19(1):e1010749.
13. Rigden DJ, Fernández XM. The 2023 Nucleic Acids Research Database Issue and the online molecular biology database collection. *Nucleic Acids Res* 20 23;51(D1):D1–D8.
14. Safran M, Rosen N, Twik M, et al. The Gene Cards Suite. In: I. Abugessaisa, T. Kasukawa(eds.) Practical Guide to Life Science Databases. Springer, Singapore. 2021; 27–56. https://doi.org/10.1007/978-981-16-5812-9_2.
15. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Res* 20 21;49(D1):D480–9.
16. Altenhoff AM, Train CM, Gilbert KJ, et al. OMA orthology in 2021: website overhaul, conserved isoforms, ancestral gene order and more. *Nucleic Acids Res* 20 21;49(D1):D373–9.
17. Schwanitz VJ, Wierling A, Biresselioglu ME, et al. Current state and call for action to accomplish findability, accessibility, interoperability, and reusability of low carbon energy data. *Sci Rep* 20 22;12(1):5208.
18. de Farias TM, Roxin A, Nicolle C. SWRL rule-selection methodology for ontology interoperability. *Data Knowl Eng* 2016;105:53–72.
19. Farias TM, Roxin A, Nicolle C. FOWLA, a federated architecture for ontologies. In: N Bassiliades, G Gottlob, F Sadri, A Paschke, D Roman(eds.) Rule Technologies: Foundations, Tools, and Applications RuleML 2015 9202: 97–111. Lecture Notes in Computer Science, Springer, Cham. 2015; https://doi.org/10.1007/978-3-319-21542-6_7.
20. George D. Understanding structural and semantic heterogeneity in the context of database schema integration. *J Dept Comput UCLAN* 2005;4(1):29–44.
21. Halevy A. Why your data won't mix: new tools and techniques can help ease the pain of reconciling schemas. *Queue* 2005;3(8):50–8.
22. Shafranovich Y. Common format and MIME type for comma-separated values (CSV) files. RFC 4180. 2005; <https://doi.org/10.17487/RFC4180>.
23. Import a CSV file in Google Ads Editor. <https://support.google.com/google-ads/editor/answer/56368>. Accessed 17 March 2023.
24. File Preparation: Resource CSV File. https://www.ncbi.nlm.nih.gov/books/NBK3812/#ft.File_Preparation_Resource_CSV_File. Accessed 17 March 2023.
25. Vrandečić D, Krötzsch M. Wikidata: a free collaborative knowledgebase. *Commun ACM* 2014;57(10):78–85.
26. Guarino N, Oberle D, Staab S. What is an ontology? Springer; In: S Staab, R Studer(eds.) Handbook on Ontologies International Handbooks on Information Systems. Springer, Berlin, Heidelberg. 2009; 1–17. https://doi.org/10.1007/978-3-540-92673-3_0.
27. The Ontology Lookup Service (OLS): Search results for Gene. <https://www.ebi.ac.uk/ols/search?q=Gene&groupField=iri&exact=on&start=0&type=class>. Accessed 17 March 2023.
28. Otero-Cerdeira L, Rodríguez-Martínez FJ, Gómez-Rodríguez A. Ontology matching: a literature review. *Exp Syst Appl* 2015;42(2):949–71.
29. Thiéblin E, Haemmerlé O, Hernandez N, et al. Survey on complex ontology matching. *Semantic Web* 2020;11(4):689–727.
30. Hitzler P, Krötzsch M, Parsia B, et al. OWL 2 Web Ontology Language Primer (Second Edition). W3C recommendation. 2012; <https://www.w3.org/TR/owl2-primer/>. Accessed 23 July 2023.
31. Miles A, Bechhofer S. SKOS simple knowledge organization system reference. W3C Recommendation. 2009. <https://www.w3.org/TR/2009/REC-skos-reference-20090818/>. Accessed 23 July 2023.
32. Mendes de Farias T, Stockinger K, Dessimoz C. et al. VoIDext: Vocabulary and patterns for enhancing interoperable datasets with virtual links. In: H. Panetto, C. Debruyne, M. Hepp, et al. (eds.) OTM Confederated International Conferences “On the Move to Meaningful Internet Systems. Springer, Springer, Cham. 2019; 11877: 607–25. https://doi.org/10.1007/978-3-030-33246-4_38.
33. Brown GR, Hem V, Katz KS, et al. Gene: a gene-centered information resource at NCBI. *Nucleic Acids Res* 20 14;43(D1): D36–42.
34. Dingerdissen HM, Bastian F, Vijay-Shanker K, et al. OncoMX: a knowledgebase for exploring cancer biomarkers in the context of related cancer and healthy data. *JCO Clin Cancer Inf* 20 20;4:210–20.
35. Kobayashi N, Kume S, Lenz K, et al. RIKEN MetaDatabase: a database platform for health care and life sciences as a microcosm of linked open data cloud. *Int Semantic Web Inform Syst* 2018;14(1):140–64.
36. Shefchek KA, Harris NL, Gargano M, et al. The Monarch Initiative in 2019: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Res* 20 20;48(D1):D704–15.
37. Morris JH, Soman K, Akbas RE, et al. The scalable precision medicine open knowledge engine (SPOKE): a massive knowledge graph of biomedical information. *Bioinformatics* 2023;39(2):btad080.
38. Cordes H, Rapp H. Gene expression databases for physiologically based pharmacokinetic modeling of humans and animal species. *CPT Pharmacometrics Syst Pharmacol.* 2023.12: 311–9. <https://doi.org/10.1002/psp4.12904>.
39. The Wikidata BgeeDB-bot GitHub repository. https://github.com/BgeeDB/Wikidata_BgeeDB-bot. Accessed 17 March 2023.
40. Wikipedia, The Free Encyclopedia. <https://www.wikipedia.org>. Accessed 17 May 2023.
41. The Wikipedia infobox gene module. https://en.wikipedia.org/wiki/Module:Infobox_gene. Accessed 17 March 2023.
42. Brickley D, Burgess M, Noy N. Google Dataset Search: building a search engine for datasets in an open Web ecosystem. In: The World Wide Web Conference (WWW '19). Association for Computing Machinery, New York, NY, USA. 1365–75. 2019; <https://doi.org/10.1145/3308558.3313685>.
43. NCBI LinkOut service. <https://www.ncbi.nlm.nih.gov/projects/linkout/>. Accessed 17 March 2023.
44. Other LinkOut Resources: datasets, databases and more. https://www.ncbi.nlm.nih.gov/projects/linkout/journals/htmllists.cgi?type_id=9. Accessed 17 March 2023.

45. LinkOut Help. <https://www.ncbi.nlm.nih.gov/books/NBK3812/>. Accessed 17 March 2023.
46. DTD Tutorial. https://www.w3schools.com/xml/xml_dtd_intro.asp. Accessed 17 March 2023.
47. XML Schema Tutorial. https://www.w3schools.com/xml/schema_intro.asp. Accessed 17 March 2023.
48. The NCBI Gene ID to Ensembl ID mapping file. <https://ftp.ncbi.nlm.nih.gov/gene/DATA/gene2ensembl.gz>. Accessed 17 March 2023.
49. HBB hemoglobin subunit beta [Pan troglodytes (chimpanzee)]. <https://www.ncbi.nlm.nih.gov/gene/?term=ENSPTRG0000040047>. Accessed 17 March 2023.
50. Mapping between UniProtKB and NCBI resources (GeneID, RefSeq): how does it work? https://www.uniprot.org/help/ncbi_mappings. Accessed 30 May 2023.
51. NCBI Human HBB gene page. <https://www.ncbi.nlm.nih.gov/gene/3043>. Accessed 11 May 2023.
52. The Bgee-UniProtKB cross-reference file. <https://bgee.org/ftp/current/XRefBgee.txt>. Accessed 17 March 2023.
53. The Bgee-GeneCards cross-reference file. https://bgee.org/ftp/current/geneCards_XRefBgee.tsv. Accessed 17 March 2023.
54. GeneCards Sources and External Links. <https://www.genecards.org/Guide/Sources>. Accessed 11 May 2023.
55. The OncoMX database. <https://oncomx.org>. Accessed 17 March 2023.
56. Mungall CJ, Torniai C, Gkoutos GV, et al. Uberon, an integrative multi-species anatomy ontology. *Genome Biol* 2012;13(1):1–20.
57. Amer-Yahia S, Koutrika G, Braschler M, et al. INODE: building an end-to-end data exploration system in practice [extended vision]. *ACM SIGMOD Record*. 2022; 50 (4):23–29. <https://doi.org/10.1145/3516431.3516436>.
58. RDF 1.1 Concepts and Abstract Syntax. <https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/Overview.html>. Accessed 17 March 2023.
59. Calvanese D, Lanti D, De Farias TM, et al. Accessing scientific data through knowledge graphs with Ontop. *Patterns* 2021;2(10):100346.
60. Sima AC, Mendes de Farias T, Zbinden E, et al. Enabling semantic queries across federated bioinformatics databases. *Database* 2019;2019. <https://doi.org/10.1093/database/baz106>.
61. RDF 1.1 Turtle. <https://www.w3.org/TR/turtle/>. Accessed 17 March 2023.
62. Mendes de Farias T, Kushida T, Sima AC, et al. Data in use for Alzheimer disease study: combining gene expression, orthology, biosource and disease datasets. In: A. Yamaguchi, A. Splendiani, M. Scott, et al. (eds.) 14th International Semantic Web Applications and Tools for Healthcare and Life Sciences (SWAT4HCLS) Conference. Basel, Switzerland. February 13-16 2023; 3415: CEUR-WS. 2023; <https://ceur-ws.org/Vol-3415/paper-47.pdf>
63. OMA — Download Current release. <https://omabrowser.org/oma/current/>. Accessed 11 May 2023.
64. UniProtKB — Downloads. <https://www.uniprot.org/help/downloads>. Accessed 11 May 2023.
65. GeneCards Suite data requests. <https://www.genecards.org/Guide/Datasets>. Accessed 11 May 2023.
66. The Bgee API. <https://bgee.org/api/>. Accessed 17 March 2023.
67. OpenAPI Specification v3.1.0. <https://spec.openapis.org/oas/v3.1.0>. Accessed 17 March 2023.
68. The Bgee SPARQL 1.1 endpoint. <https://bgee.org/sparql/>. Accessed 16 May 2023.
69. SPARQL Endpoint interface to Python. <https://sparqlwrapper.readthedocs.io>. Accessed 17 March 2023.
70. The Bgee R packages. <https://bgee.org/resources/r-packages>. Accessed 17 March 2023.
71. Bioconductor. <https://bioconductor.org/>. Accessed 17 March 2023.
72. The BgeeDB docker container. https://hub.docker.com/r/bgeedb/bgee_r. Accessed 17 March 2023.
73. da Veiga Leprevost F, Grüning BA, Alves Aflitos S, et al. BioContainers: an open-source and community-driven framework for software standardization. *Bioinformatics* 2017;33(16):2580–2.
74. BioContainers: bioconductor-bgeedb. <https://biocontainers.pro/tools/bioconductor-bgeedb>. Accessed 17 March 2023.
75. Carlson M, Ramos M. UniProt.ws: R Interface to UniProt Web Services. R package version 2.40.0. <https://github.com/Bioconductor/UniProt.ws>. Accessed 11 May 2023.
76. Google Dataset Search. <https://datasetsearch.research.google.com>. Accessed 17 March 2023.
77. Dingerdissen HM, Bastian F, Vijay-Shanker K, et al. OncoMX: a knowledgebase for exploring cancer biomarkers in the context of related cancer and healthy data. *JCO Clin Cancer Inf* 20 20;4:210–20.
78. CC0 1.0 Universal (CC0 1.0) Public Domain Dedication. <http://creativecommons.org/publicdomain/zero/1.0/>. Accessed 17 March 2023.
79. The Wikibase data model. <https://www.mediawiki.org/wiki/Wikibase/DataModel>. Accessed 17 March 2023.
80. The Wikidata Integrator GitHub repository. <https://github.com/SuLab/WikidataIntegrator>. Accessed 17 March 2023.
81. The INS gene Wikidata entry. <https://www.wikidata.org/wiki/Q21163221>. Accessed 17 March 2023.
82. Wikidata:Schema proposals. https://www.wikidata.org/wiki/Wikidata:Schema_proposals. Accessed 17 March 2023.
83. Wikidata:Requests for permissions/Bot. https://www.wikidata.org/wiki/Wikidata:Requests_for_permissions/Bot. Accessed 17 March 2023.
84. Wikidata:Requests for permissions/Bot/BgeeDB-bot. https://www.wikidata.org/wiki/Wikidata:Requests_for_permissions/Bot/BgeeDB-bot. Accessed 17 March 2023.
85. User:ProteinBoxBot. <https://www.wikidata.org/wiki/User:ProteinBoxBot>. Accessed 11 May 2023.
86. LUA language. <https://www.lua.org>. Accessed 17 March 2023.
87. The Wikipedia infobox gene module sandbox. https://en.wikipedia.org/wiki/Module:Infobox_gene/sandbox. Accessed 17 March 2023.
88. JSON for Linking Data. <https://json-ld.org>. Accessed 17 March 2023.
89. The public Schema.org e-mail. public-schemaorg@w3.org. Accessed 17 March 2023.
90. The Schema.org GitHub open issues. <https://github.com/schemaorg/schemaorg/issues?q=is%3Aopen+is%3Aissue+sort%3Acreated-asc>. Accessed 17 March 2023.
91. The Google Dataset documentation. <https://developers.google.com/search/docs/appearance/structured-data/dataset>. Accessed 17 March 2023.
92. Teiid: Cloud-native data virtualization. <https://teiid.io>. Accessed 17 March 2023.
93. Teiid documentation: XML VDB. http://teiid.github.io/teiid-documents/16.0.x/content/reference/r_xml-deployment-mode.html. Accessed 17 March 2023.
94. OncoMX database dump and Teiid virtual database configuration file. <https://purl.org/inode/oncomx-federation>. Accessed 17 June 2023.

95. The OncoMX SPARQL portal. <http://testbed.inode.igd.fraunhofer.de:18005>. Accessed 17 March 2023.
96. de Farias TM, Chiba H, Fernández-Breis JT. Leveraging logical rules for efficacious representation of large orthology datasets. In: A. Paschke, A. Burger, A. Splendiani, et al.(eds.) 10th International Semantic Web Applications and Tools for Healthcare and Life Sciences (SWAT4HCLS) Conference CEUR-WS, Rome, Italy, December 4-7 2017; 2042: CEUR-WS. <http://ceur-ws.org/Vol-2042/paper36.pdf>.
97. OMA-Bgee homologs tool. https://github.com/BgeeDB/bgee_pipeline/tree/develop/pipeline/OMA_homologs. Accessed 11 May 2023.
98. Lenharo M. GISAID in crisis: can the controversial COVID genome database survive? *Nature* 2023;617(7961):455–7.