UBIC^2 – Towards Ubiquitous Bio-Information Computing: Data Protocols, Middleware, and Web Services for Heterogeneous Biological Information Integration and Retrieval

Pengyu Hong

BioX program, Department of Statistics, Stanford University, Stanford, CA 94305-4065.

Email: pengyuhong@stanford.edu

Sheng Zhong

BioX program, Department of Statistics, Stanford University, Stanford, CA 94305-4065.

Email: Zhong@stanford.edu

Wing H. Wong*

BioX program, Department of Statistics, Stanford University, Stanford, CA 94305-4065.

Email: whwong@stanford.edu

To whom correspondence should be addressed: whwong@stanford.edu

Abstract

The Ubiquitous Bio-Information Computing (UBIC^2) project aims to disseminate protocols and software packages to facilitate the development of heterogeneous bio-information computing units that are interoperable and may run distributedly. UBIC^2 specifies biological data in XML formats and queries data using XQuery. The UBIC^2 programming library provides interfaces for integrating, retrieving, and manipulating heterogeneous biological data. Interoperability is achieved via Simple Object Access Protocol (SOAP) based web services. The documents and software packages of UBIC^2 are available at http://www.ubic2.org.

Keywords: Biological Data Integration, Middleware, Interoperability, Ubiquitous Computing.

1 Introduction

The emergence of various high throughput biological experimental techniques has enabled researchers to monitor the activities of cellular molecules at system levels. Diverse biological data (e.g., sequence data, gene expression data, phenotype data, protein interaction data, etc.) is being generated distributedly at explosive rates. At the same time, numerous bio-information computing methods and applications are being developed daily. It has now been widely realized that deeper and more comprehensive biological knowledge can be discovered faster only by systematically utilizing heterogeneous bio-information and computational utilities. It is expected that future bio-information computing could be pervasive in our daily life (e.g., Personal Health Care System) and complex procedures of biological experiments, which generate inter-related heterogeneous data and require timely analyses to speed up the whole procedures. To this end, we propose to develop ubiquitous bio-information computing units that are capable of selectively integrating and computing heterogeneous bio-information and are interoperable.

Currently, there are substantial barriers for ubiquitous bio-information computing. First, data are widely distributed. There are more than 700 databases related to molecular biology [10]. Second, most databases are mainly designed for human users and require manually interaction with their web sites. The information retrieving procedure is non-programmable or requires non-trivial programming. Third, there has been a great diversity of data specification conventions, which are often not semantically sound and may be changed from time to time. Fourth, although there have been existing efforts for developing and sharing bio-information computation programming libraries, such undertakings are often passive without coordinating with the development of data specification protocols.

To speed up the evolution of ubiquitous bio-information computing, an integrated endeavor should be devoted to developing protocols for semantic data specification, software middleware, and interoperable computing units. In the rest of the paper, we first review related works. Then we introduce the concept of UBIC^2 and report the current status of UBIC^2.

2 Backgrounds

2.1 Data specifications

XML (the eXtensible Markup Language) is playing an increasingly important role in promoting the evolution and interoperability of the Internet. XML provides a mechanism to describe and exchange structure-rich data in computer understandable ways. Recently, the biological database research community began formatting biological data using XML DTDs (document-type definitions) or XML Schemas. Outstanding achievements include Distributed Annotation System (DAS) [8] for decentralized sequence annotation, XEMBL [39] for distributing EMBL data, RNAML [40] for exchanging RNA information, Systems Biology Markup Language [26] for representing biochemical reaction networks, and Microarray and Gene Expression Markup Language for microarray data [35]. Such efforts often excel in formatting a certain type of data and do little for integrating heterogeneous data and developing software middleware.

2.2 Data integration

Works on database middleware for large-scale integration of heterogeneous biological data mainly take the following two approaches: database federation and data warehousing. A federation approach builds a middle layer on tope of a collection of distributed databases and makes distributed databases as an integrated one to user. It allows users to access up-to-date

distributed data at run-time and requires only very small local space for storing information. On the other hand, the performance of distributed data retrieval at run-time depends mainly on the reliability and speed of the underlying network connectivity, which often turns out to be the bottleneck. Examples include BioKleisli [6], TINet [9], DiscoveryLink [11], SEMEDA [30], etc.

A data warehousing approach periodically downloads data in batch from remote databases, then extracts, re-arranges, and manages data locally. This approach is more efficient for repeatedly retrieving a large amount of heterogeneous data. Usually, different data fetching programs should be written to serve different data sources. However, most of programs use a similar operation procedure. An inheritable and expandable template can be designed and implemented to reduce the cost of programming. Examples of warehousing systems include AnnBuilder [42], BioMolQuest [4], EnsMart [28], InterPro [31], JXP4BIGI [27], SLAD [34], SRS [41], SOURCE [7], TAMBIS [32], etc.

2.3 Software middleware

Most of data integration research projects provide limited software middleware supports for developing bioinformatics applications. A desired software middleware should at least provide basic classes (e.g., list, vector, hash table, etc.) for storing and manipulating bio-information elements (e.g., gene annotation, biological sequence, etc.). Existing open-source programming toolkits including BioPerl [36], BioPython [5], BioJava [15], BioRuby [17], and BioConductor [14]. These projects aim to facilitate the development of stand-alone applications.

2.4 Interoperability

Stein advocated web services as a solution to achieve interoperability among online biological databases [37]. Web services provide interfaces, which are described in a machine-processable format, as standard means of interoperating between different databases and computation utilities regardless of their platforms and programming languages. Applications can be developed to weave together web services from a variety of sources to create a distributed application. The BioMOBY project [16], the caCORE project [18], and the myGrid project [19] are three successful projects sharing a similar approach to achieve this goal. They define a set of schemas and provide a centralized registry of data and services, which are decentralized and can be accessed programmatically. The myGrid project emphasizes on developing high-level middleware to support bioinformatics research on a Grid. Objects in both caCORE and BioMOBY can be serialized into XML streams, which can be exchanged via SOAP web services or HTTP-XML interfaces. The BioMOBY objects are data-only elements and the caCORE objects have rich functions in addition to data.

3 UBIC^2

The UBIC^2 project advocates developing UBIC^2 units (see Figure 1), which can be a data consumer, a service provider, or the combination of two. As a data consumer, a UBIC^2 unit is capable of integrating heterogeneous data from selected sources via a mixture of federation and warehousing at run time. Some frequently used remote databases are warehoused. Other databases, which are used occasionally or too large to be managed locally, are integrated via federation. UBIC^2 units manage data in XML formats. UBIC^2 adopts several existing XML data specifications and initiates new ones when needed. As a service provider, a UBIC^2 unit uses the Web Services Description Language to define its data-oriented or procedure-oriented services, which are accessible on the Internet as SOAP web services and can be

queried. Different from BioMOBY, caCORE, and myGrid, UBIC^2 does not have a centralized registration but do require standardizing data specifications. The UBIC^2 programming library is designed and implemented using object-oriented technology. The library contains a set of inheritable generic classes, which are designed for biological objects sharing similar or common structures and functions.

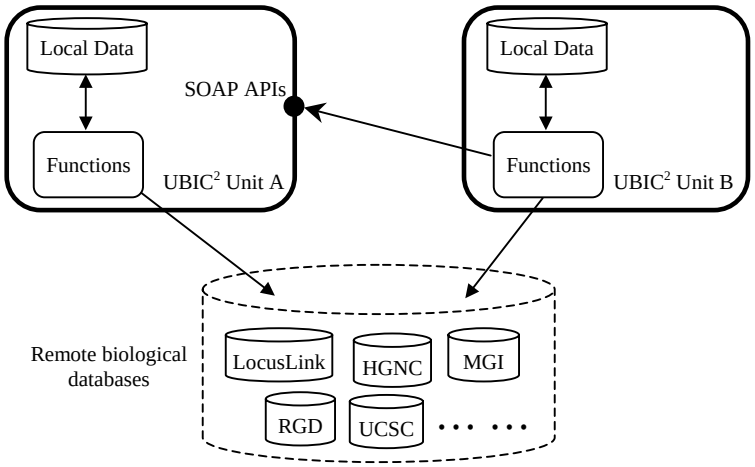

Figure 1. The UBIC^2 Concept. Unit A is a data consumer and a service provider. Unit B is a data consumer.

3.1 Protocols for data specifications and query

Ideally, data specifications can be unified by major biological databases. Many data sources have specified data in XML formats, such as DAS XML specification [8], XEMBL [39], NCBI Document Type Definitions [20], and Swiss-Prot XML Schema [24]. However, there are still many databases formatting data in semantically unsound ways. UBIC^2 XML data specification has four categories: Molecule annotation information, Sequence, Ontology, and Literature. Molecule information schemas cover annotation information of genes and proteins. Sequence schemas define the formats to represent nucleotide sequences and amino

acid sequences. We designed XML schemas for gene annotation information (including name, symbol, synonyms, products, chromosome locations, Gene Ontology annotation, literature links, etc.), gene homology information, and gene upstream sequences. The Ontology category only includes specification of Gene Ontology [1]. We adopt Swiss-Prot XML Schema [24] for protein information and PubMed DTD [21] for literature data.

As for information query, UBIC^2 uses a query language XQuery [25], which is designed to

```
<UBIC2_GeneInfoQuery>
{
  for $gene in doc('GeneInfo.xml')/Gene
      where $gene/LocusLinkID = "1" or $gene/LocusLinkID = "16999"
  return
      <Gene>
         { $gene/LocusLinkID }
         { $gene/Name }
         { $gene/Symbol }
         { $gene/UniGeneID }
         { $gene/RelatedPapers }
      </Gene>
}
</UBIC2_GeneInfoQuery>
```

```
<GeneInfo>
  <Gene>
    <LocusLinkID>1</LocusLinkID>
    <Name>alpha-1-B glycoprotein</Name>
    <Symbol>A1BG</Symbol>
    <UniGeneID>Hs.390608</UniGeneID>
    <RelatedPapers>
      <PubMedID>2591067</PubMedID>
      <PubMedID>3458201</PubMedID>
      <PubMedID>8889549</PubMedID>
      <PubMedID>12477932</PubMedID>
    </RelatedPapers>
  </Gene>
  <Gene>
    <LocusLinkID>16999</LocusLinkID>
    <Name>latent transforming growth factor beta binding protein 4</Name>
    <Symbol>Ltbp4</Symbol>
    <UniGeneID>Mm.272251</UniGeneID>
    <RelatedPapers>
      <PubMedID>10349636</PubMedID>
      <PubMedID>11042159</PubMedID>
      <PubMedID>11076861</PubMedID>
      <PubMedID>11217851</PubMedID>
      <PubMedID>12208849</PubMedID>
      <PubMedID>12477932</PubMedID>
    </RelatedPapers>
  </Gene>
</GeneInfo>
```

Figure 2. A query example. Top: query. Bottom: results.

be broadly applicable to XML data sources[1]. This is one of the most significant differences between UBIC2 and other projects (e.g., BioMOBY, caCORE, myGrid, etc.). Query results are returned in XML formats defined by UBIC^2 XML data specifications. Figure 2 shows a query example, which use LocusLink IDs to retrieve the following information of two genes: official names, official symbols, UniGene IDs, and PubMed IDs of papers related to the genes.

3.2 Software Architecture

The UBIC^2 software architecture contains six components: (1) Internet interfaces, (2) remote database conductors, (3) species-specific data integrators, (4) basic data structures, (5) applications, and (6) web services. Their relations are illustrated in Figure 3. The components and their relations are explained as the following.

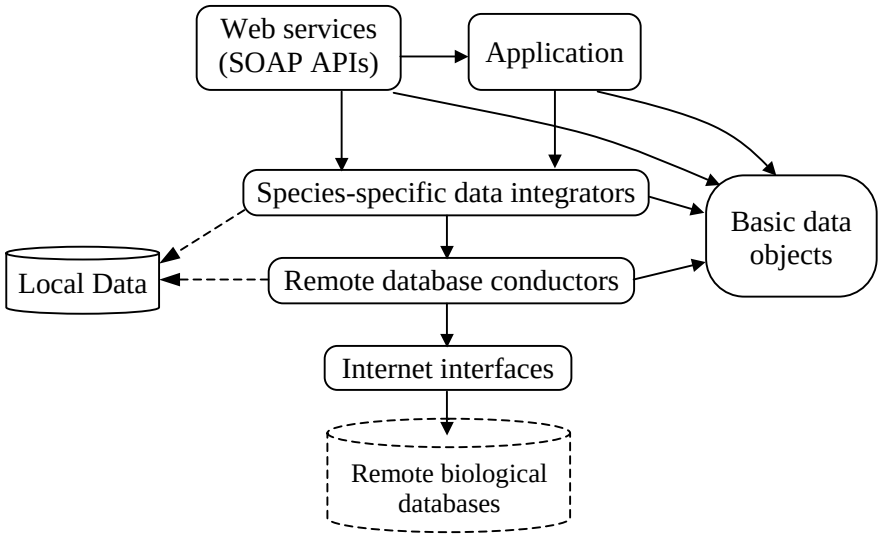

Figure 3. The UBIC^2 software architecture.

[1] Although the World Wide Web Consortium has not finalized the recommendation of XQuery, UBIC^2 can easily accommodate future updates of XQuery.

3.2.1 Internet interfaces

The Internet interfaces provide programmable means for retrieving data from remote databases. Currently, two network protocols, HTTP and FTP, are supported. Other network protocols can be easily accommodated. If the remote database supports, a program can choose to download a chunk of data instead of the whole data file using the HTTP protocol. It requires a network protocol, an Internet address (e.g., IP address, URI, etc.), and/or the name of a remote data file. The interfaces are capable of recovering broken download sessions.

3.2.2 Remote database conductors

Each remote database conductor deals with one remote biological database. Remote database conductors are derived from a generic class, which contains common member variables and functions (e.g., download data, re-format data in XML, index XML data, retrieve XML data, etc.). To make information retrieval efficient, the XML data files are indexed by major IDs (e.g., LocusLink ID, RefSeq ID, UniGene ID, etc.) at the level of genes/proteins. To retrieve information about genes/proteins, a conductor first uses the indexing information to locate the positions of the corresponding information segments in files, and then load and hash the information segments. To date, the UBIC^2 software package supports a number of major biological databases including LocusLink [23], HUGO Gene Nomenclature Committee (HGNC) [33], Mouse Genome Database (MGD) [2], Rat Genome Database (RGD) [38], Swiss-Prot [3], HomoloGene [22], and sequence databases from UCSC Genome Bioinformatics [29], and so on.

3.2.3 Species-specific data integrators

Different from BioMOBY, caCORE, and myGrid, UBIC^2 has species-specific data integrators. Since remote biological databases are developed and maintained cooperatively as well as independently, the information in those databases could be complementary, overlapping, and inconsistent. For example, both LocusLink and HGNC contain annotations for human genes, but with different ranges of coverage. To integrate those data, we design an integrator for each species. A species-specific data integrator invokes appropriate remote database conductors to obtain combines various information of one species from different databases. It then combine those information. This layer makes miscellaneous distributed data of a species as coming from one unified source to users. We have implemented integrators for three organisms: *Homo sapiens*, *Mus musculus* and *Rattus norvegicus*. The integrators are inherited from a generic class, which can be derived to support other species.

Currently, the information is categorized into four types: gene annotation, protein annotation, gene homology annotation, and upstream sequence. Gene annotation include the following information of a gene: LocusLink ID, RefSeq ID, UniGene ID, GenBank ID, Swiss-Prot ID, Gene Ontology annotation, official name, official symbol, synonyms, PubMed IDs of related literature. Protein annotation includes: Swiss-Prot ID, protein name, EMBL IDs, HSSP IDs, InterPro IDs, Pfam IDs, PIR IDs, PRINTS IDs, ProDom IDs, PROSITE IDs, SMART IDs, and TRANSFAC IDs. Each type of information is stored in a species-specific XML data file, which is indexed by major IDs (e.g., LocusLink ID, Swiss-Prot ID, UniGene ID, etc.).

In addition to information integration, the integrators also detect data inconsistency across databases, reconcile data discrepancies, and track obsolete information. The detected inconsistency is stored and reported to users. Currently, the integrators invoke a simple

method to automatically reconcile such discrepancies. This method uses a designated database as the correct information source. Developers can replace this method by simply overriding the discrepancy reconciliation function of an integrator based on their own experience with a remote database.

Integrators also handle species-specific query of heterogeneous data. When a query is received, an integrator first tries to retrieve information from local species-specific XML data files. If the required information cannot be found locally, it will invoke appropriate remote database conductor to retrieve information from remote databases. Online query of remote databases is useful when a remote database (e.g., PubMed) is too big for a local machine. The following case exemplifies how integrators retrieve information. Given a set of LocusLink IDs, the user want to retrieve related gene annotations, upstream sequences, protein annotations, and PubMed abstracts. The integrator will first retrieve gene annotation information, which contains RefSeq IDs, Swiss-Prot IDs of genes' protein products, and PubMed IDs of related papers. Then, the integrators will retrieve upstream sequences from upstream sequence files using RefSeq IDs, protein annotations from protein annotation files using Swiss-Prot IDs, PubMed abstracts from PubMed database via the PubMed conductor using PubMed IDs.

3.2.4 Basic data objects and Applications

The basic data types of UBIC^2 include gene annotation, protein annotation, biological sequence, gene expression, etc. Basic data objects, e.g., lists, vectors, hash-tables, etc., are designed and implemented for each data type. Remote database conductors and species-specific data integrators use basic data objects to manage information in computer memory.

Built on top of basic data objects and species-specific data integrators, UBIC^2 applications can retrieve local information and selectively integrate remote data at run time.

3.2.5 Web services

The web services of UBIC^2 units are described in the Web Services Description Language and hence are searchable on the Internet. A client invokes UBIC^2 Web services using SOAP-messages, which are typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. Upon receiging a SOAP message, a UBIC^2 unit can use species-specific data integrators to access local and remote data or use TCP/IP to communicate with UBIC^2 applications, which run on the same server and provide complicate computational functions. Data retrieval services can be unified because UBIC^2 standardizes data specifications and uses XQuery to retrieve data. If a data service provider uses different data specification protocols, a client can still query data from that provider as long as it can transform its data into the data formats of UBIC^2 using Extensible Stylesheet Language Transformations.

4 UBIC^2 Version 1.0

The current release of UBIC^2 (version 1.0) was implemented using Microsoft's C# programming language and .Net technology, which supports FreeBSD OS and Mac OS X 10.2. The UBIC^2 concept can be implemented using other programming languages and platforms, e.g., Java and Apache/SOAP. UBIC^2 1.0 is distributed with a demo application that can integrate several remote biological databases and perform batch retrievals of heterogeneous biological data. UBIC^2 1.0 releases a web service for querying gene annotations [13]. The web service is deployed on a Window Server 2003 installed with

Microsoft Internet Information Service 6.0. Documents and manuals of the UBIC^2 programming library and the demo are available at http://www.ubic2.org.

5 Conclusions

The current version of UBIC^2 is designed to favor individual researchers and small research groups. We have chosen an economical way for maintaining and managing data as local XML files, which do not require a database management system to manage. The UBIC^2 programming library provides a set of objects and APIs to allow quick development of bioinformatics applications. We have used the library to develop GeneNotes [12], which already has a substantial number of users. A UBIC^2 unit can provides its data-oriented and procedure-oriented services on the Internet via SOAP-based web-services. We hope UBIC^2 will contribute to solving data comparability and accessibility and computational interoperability in bioinformatics research.

Acknowledgement

The work of Pengyu Hong is supported by NIHGM67250. The work of Wing H. Wong is supported by NIH-HG02341.

References

[1] M. C. Ashburner, A. Ball, et al., Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nature Genetics. 25(1), 2000, 25-29.

[2] J. A. Blake, J. E. Richardson, et al., MGD: the Mouse Genome Database. Nucleic Acids Res. 31(1), 2003, 193-195.

[3] B. Boeckmann, A. Bairoch, et al., The SWISS-PROT protein knowledge base and its supplement TrEMBL in 2003. Nucleic Acids Research 31(1), 2003, 365-370.

[4] Y. V. Bukhman and J. Skolnick, BioMolQuest: integrated database-based retrieval of protein structural and functional information. Bioinformatics 17(5), 2001, 468-478.

[5] B. Chapman and J. Chang, Biopython: Python tools for computation biology. ACM SIG-BIO Newsletter, 2000.

[6] S. B. Davidson, C. Overton, et al., BioKleisli: A Digital Library for Bio-medical Researchers. International Journal of Digital Libraries 1(1), 1997, 36-53.

[7] M. Diehn, G. Sherlock, et al., SOURCE: a unified genomic resource of functional annotations, ontologies, and gene expression data. Nucleic Acids Res. 31(1), 2003, 219-223.

[8] R. D. Dowell, R. M. Jokerst, et al., The distributed annotation system. BMC Bioinformatics 2(1), 2001, 7.

[9] B. A. Eckman, A. S. Kosky, et al. Extending traditional query-based integration approaches for functional characterization of post-genomic data. Bioinformatics 17(7), 2001, 587-601.

[10] M. Y. Galperin, The molecular biology database collection: 2005 update. Nucleic Acids Research, 2005, Vol. 33, Database issue D5-D24.

[11] L. M. Haas, P. M. Schwarz, et al., DiscoveryLink: A system for integrated access to life sciences data sources. IBM Systems Journal 40(2), 2001, 489-511.

[12] P. Hong and W. H. Wong, GeneNotes: A novel information management software for biologists, *BMC Bioinformatics* (to appear).

[13] http://bayes.fas.harvard.edu/Webservice/BioWebservice/BioWebservice.asmx

[14] http://www.bioconductor.org/

[15] http://biojava.org

[16] http://biomoby.org

[17] http://bioruby.org/

[18] P. A. Covitz, et al., caCORE: A common infrastructure for cancer informatics. Bioinformatics, vol. 19, no. 18, 2003, pp. 2404 – 4412.

[19] http://www.mygrid.org.uk

[20] http://www.ncbi.nih.gov/dtd/

[21] http://www.ncbi.nlm.nih.gov/entrez/query/static/PubMed.dtd

[22] http://www.ncbi.nlm.nih.gov/HomoloGene/

[23] http://www.ncbi.nlm.nih.gov/LocusLink/

[24] http://www.ebi.ac.uk/swissprot/SP-ML/

[25] http://www.w3.org/TR/xquery/

[26] M. Hucka, A. Finney, et al., The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19(4), 2003, 524-531.

[27] Y. Huang, T. Ni, et al., JXP4BIGI: a generalized, Java XML-based approach for biological information gathering and integration. Bioinformatics 19(18), 2003, 2351-2358.

[28] A. Kasprzyk, D. Keefe, et al., EnsMart: a generic system for fast and flexible access to biological data. Genome Research 14(1), 2004, 160-169.

[29] W. J. Kent, C. W. Sugnet, et al., The human genome browser at UCSC. Genome Research 12(6), 2002, 996-1006.

[30] J. Kohler and S. Schulze-Kremer, The semantic metadatabase (SEMEDA): ontology based integration of federated molecular biological data sources. In Silico. Biol. 2(3), 2002, 219-231.

[31] N. J. Mulder, R. Apweiler, et al., InterPro: an integrated documentation resource for protein families, domains and functional sites." Briefings in Bioinformatics, 3(3), 2002, 225-235.

[32] N. W. Paton, R. Stevens, et al., Query Processing in the TAMBIS Bio-informatics Source Integration System. Proceedings of the 11th International Conference on Scientific and Statistical Database Management, New York, IEEE, 1999.

[33] S. Povey, R. Lovering, et al., The HUGO Gene Nomenclature Committee (HGNC). Hum. Genet. 109(6), 2001, 678-680.

[34] C. Schonbach, P. Kowalski-Saunders, and V. Brusic. Data warehousing in molecular biology. Briefings in Bioinformatics 1(2), 2000, 190-198.

[35] P. T. Spellman, et al., Design and implementation of microarray gene expression markup language (MAGE-ML). Genome Biology 3(9), 2002, research0046.1-0046.9.

[36] J. E. Stajich, D. Block, et al. (2002). "The Bioperl toolkit: Perl modules for the life sciences." Genome Research 12(10), 2002, 1611-1618.

[37] L. Stein, Creating a Bioinformatics Nation. Nature 417, 2002, 119-120.

[38] S. Twigger, J. Lu, et al., Rat Genome Database (RGD): mapping disease onto the genome. Nucleic Acids Research 30(1), 2002, 125-128.

[39] L. Wang, J. J. Riethoven, et al., XEMBL: distributing EMBL data in XML format. Bioinformatics 18(8), 2002, 1147-1148.

[40] A. Waugh, P. Gendron, et al., RNAML: a standard syntax for ex-changing RNA information. RNA 8(6), 2002, 707-717.

[41] E. M. Zdobnov, R. Lopez, et al., The EBI SRS server-new features. Bioinformatics 18(8), 2002, 1149-1150.

[42] J. Zhang, V. Carey, et al., An extensible application for assembling annotation for genomic data. Bioinformatics 19(1), 2003, 155-156.

Figure captions

Figure 1. The UBIC^2 Concept. Unit A is a data consumer and a service provider. Unit B is a data consumer.

Figure 2. A query example. Top: query. Bottom: results.

Figure 3. The UBIC^2 software architecture.

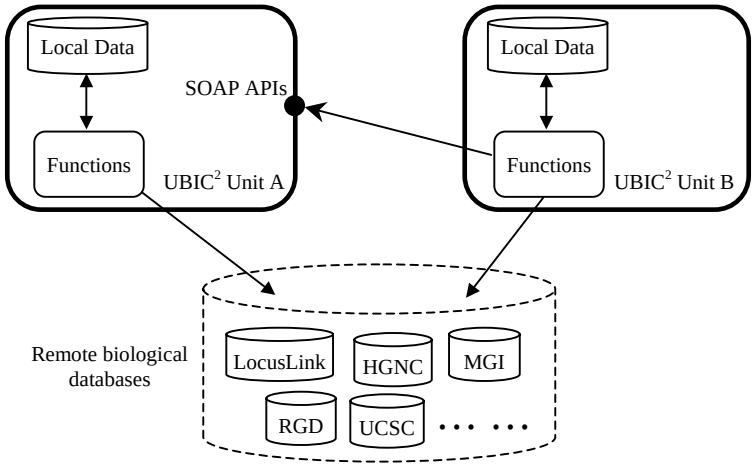

Figure 1. The UBIC^2 Concept. Unit A is a data consumer and a service provider. Unit B is a data consumer.

```
<UBIC2_GeneInfoQuery>
{
  for $gene in doc('GeneInfo.xml')/Gene
      where $gene/LocusLinkID = "1" or $gene/LocusLinkID = "16999"
  return
      <Gene>
         { $gene/LocusLinkID }
         { $gene/Name }
         { $gene/Symbol }
         { $gene/UniGeneID }
         { $gene/RelatedPapers }
      </Gene>
}
</UBIC2_GeneInfoQuery>
```

```
<GeneInfo>
  <Gene>
    <LocusLinkID>1</LocusLinkID>
    <Name>alpha-1-B glycoprotein</Name>
    <Symbol>A1BG</Symbol>
    <UniGeneID>Hs.390608</UniGeneID>
    <RelatedPapers>
      <PubMedID>2591067</PubMedID>
      <PubMedID>3458201</PubMedID>
      <PubMedID>8889549</PubMedID>
      <PubMedID>12477932</PubMedID>
    </RelatedPapers>
  </Gene>
  <Gene>
    <LocusLinkID>16999</LocusLinkID>
    <Name>latent transforming growth factor beta binding protein 4</Name>
    <Symbol>Ltbp4</Symbol>
    <UniGeneID>Mm.272251</UniGeneID>
    <RelatedPapers>
      <PubMedID>10349636</PubMedID>
      <PubMedID>11042159</PubMedID>
      <PubMedID>11076861</PubMedID>
      <PubMedID>11217851</PubMedID>
      <PubMedID>12208849</PubMedID>
      <PubMedID>12477932</PubMedID>
    </RelatedPapers>
  </Gene>
</GeneInfo>
```

Figure 2. A query example. Top: query. Bottom: results.

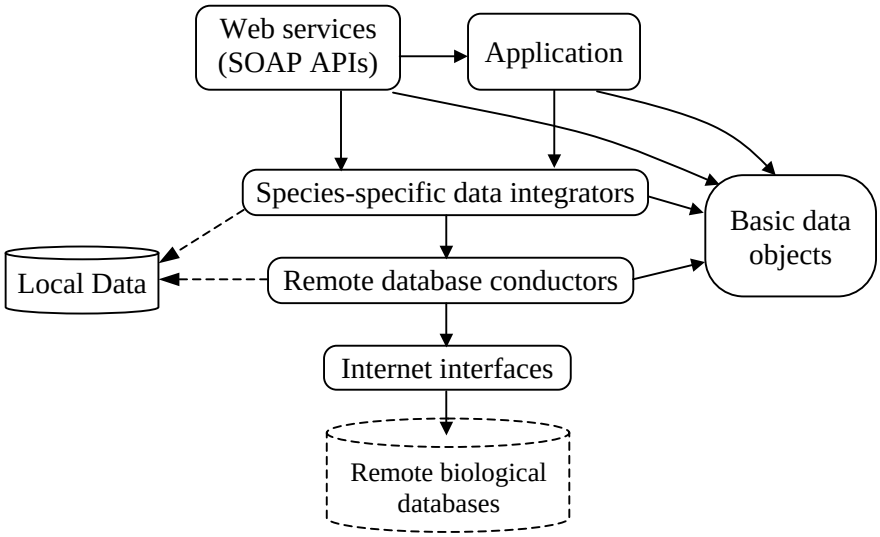

Figure 3. The UBIC^2 software architecture.