OBJECT IDENTITY

AND

LIFE SCIENCE RESEARCH

ROBERT J. ROBBINS

FRED HUTCHINSON CANCER RESEARCH CENTER

RROBBINS@FHCRC.ORG

TABLE OF CONTENTS

OBJECT IDENTITY AND LIFE SCIENCE RESEARCH

ROBERT J. ROBBINS

BIOLOGICAL IDENTITY

Identity is not a simple concept in biology. When I taught introductory genetics my first lecture always began:

> There is one word that will give you special trouble in this course. The meaning of that word will seem uncomfortably slippery. Every time you think you've learned what it means you will discover a new twist. The difficult word is: SAME.

The subject matter of biology is characterized by intense individuality coupled with historicity. No two biological objects are genuinely the "same". Even the "same" biological object does not remain the "same" over time. Although biologists often say things like, "We used the same methods to isolate the same gene from the same clone of the same organism," when pressed they might add, "but the clone isn't really the same as it was when we first did the isolation."

In talking about taxonomic databases, Frank Bisby has spoken of the "retail" and the "wholesale" side of biological information. On the retail side, a Virginia gardener wants to know the correct name for the plant she found in her back yard and whether or not the species is the "same" as the similar looking one she found in her cousin's yard in New York. On the wholesale side, a botanist wants to know all of the different ways that "same" species has been classified in the past and perhaps whether the Virginia population should truly be considered the same species as the New York population, or whether a closer molecular examination might show that the present species concept should be divided into two or more different species.

Two geneticists look at a map of human chromosome 21. A year later, they both want to look at the same map again. But, to one of the biologists, "same" means exactly the same map (same data, bit for bit), whereas to the other "same" means the current map of the "same" biological object, even if all of the data in that map have changed.

To a protein chemist, two molecules of beta-hemoglobin are the "same" because they are comprised of exactly the same sequence of amino acids. To a

biologist, the same two molecules might be considered different because one was isolated from a chimpanzee and the other from a human.

In comparing alleles to determine identity, biologists sometimes distinguish between "identical by state" and "identical by descent":

> **identity-by-state**: Two alleles are identical by state (IBS) when they are scored the same. for example, two unrelated individuals each with blood group AB share two alleles IBS. Alleles that are IBS are not always identical-by-descent.
>
> **identity-by-descen**t: Two alleles are identical by descent (IBD) when it can be determined with certainty that they have been inherited from a common ancestor. For instance, a mother with blood type O and father with blood type AB have two children, each with blood type A. Since the genotypes of the children are AO, the children share one allele IBD, the AO allele. Whether the maternally inherited O allele is IBD in the children is unclear since the mother is homozygous for the O allele. Alleles that are identical-by-descent are always identical-by-state.

That is, just because two alleles are base-for-base identical (i.e., they are IBS) does not prove that they are also IBD.

Provenance is often a key part of the concept of identity in biological thinking. In distinguishing IBD from IBS, identical provenance seems to provide a refinement of identity – a kind of historical identity that is added on to mere physical identity. But sometimes provenance can trump, not merely refine actual physical identity. For example, it is easy to find biological prose like:

> Homology forms the basis of organization for comparative biology. In 1843 Richard Owen defined homology as "the same organ in different animals under every variety of form and function." For example, reptiles, mammals, and birds have the same bones of the upper and lower arm…

But here "same" means merely that the bones are similar by descent, although wildly different in actual state.

In many cases the issue of identity cannot be resolved – experts differ. Much life-science literature consists of debates between opposing views as to whether two populations should be considered to belong to the "same" species, whether two different pathologies should just be considered different manifestations of the "same" disease, whether pathologies that historically had been considered the "same" should now be considered different, etc.

We could continue accumulating examples indefinitely, but the point is established:

Identity is not a simple concept in biology; therefore to meet all of the needs of the life-science community, digital life-science identifiers must have the subtlety (and flexibility) to match the multi-varied senses of sameness that pervade the life sciences.

DATABASE REQUIREMENTS

More than ten years ago, a report[1] from a human genome project workshop stated:

> Since users need to integrate findings from several different community databases, each community database should be designed as a component of a larger information infrastructure for computational biology. Specifically, community databases should recognize the biological interdependence of information in multiple databases and should provide support for integrated queries involving multiple databases.

The report went on to assert, "The goal must be the adoption of minimum interoperability standards, so that adding a new database to the federation would be no more difficult than adding another computer to the Internet," But then cautioned, "Individual community databases must be charged with collecting, maintaining, and distributing information that shows how data objects in their system relate to data objects in other systems. An embarrassment to the Human Genome Project is our inability to answer simple questions such as, 'How many genes on the long arm of chromosome 21 have been sequenced?'"

The problem then was that the various community databases provided no real support for inter-database referential integrity. The problem now is that there is still no systematic support for inter-database referential integrity.

Life Science Identifiers as Primary Keys

Ten years ago the technology to support a read-only federation of heterogeneous structured databases simply did not exist. The database research community had generally determined that the problem of maintaining a read-write federation of loosely couple databases was insoluble[2] and the WWW had not yet proven the immense value of a read-only loosely coupled federation of information systems.

Now the technology does exist. Individual data objects may be served up using a variety of technologies (WWW, SOAP, XML, etc) and it should be possible to devise consistent syntactic methods that are sufficiently flexible to represent (almost) arbitrarily complex data objects while still sufficiently constrained to allow the development of a closed query language to operate over them. In that environment, a truly federated information infrastructure for biology could finally be created – one that permits not only human-mediated traversal but also supports automated, set-oriented transactions. But this can only happen if

---

[1] Robbins, RJ. [Ed.] 1994. Genome informatics I: Community databases. *Journal of Computational Biology*, 3:173–190.

[2] Sheth, AP, and Larson, JA: 1990. Federated databases systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22:183–236.

there are appropriate life-science identifiers that can serve as primary keys in a distributed federation of loosely coupled heterogeneous databases.

Although there are many other needs for "life science identifiers" the need for truly distributed database support is so great that any identifier system that ultimately ignored the needs of the database community could hardly be called a true solution to the life-science identifier problem.

Understanding how the concepts of identity and how digital identifiers might be effectively used to support a distributed federation requires some background detail. Later we will turn our attention to an overview of how a federated object server model might work. Now we consider some challenges associated with biological data management.

The Challenge of Biological Data Management

The challenge of integrating information resources has been publicly recognized for more than a decade. The failure to meet that challenge has been a continuing annoyance, even embarrassment. Removing this embarrassment will require several interoperability improvements:

- *Technical interoperability* must be achieved, so that minimum functional connectivity can be assumed among participating information resources.
- *Semantic interoperability* must be developed, so that meaningful associations *can* be made between data objects in different databases.
- *Social interoperability* must occur, so that meaningful associations *are* made between data objects in different databases. Each asserted link is an act of scientific creativity, not merely the result of computations on existing data. Therefore, social changes must occur to stimulate the creation and entry of this information.

These three advances will likely occur in the order given. Without semantic interoperability, it is difficult to define, much less enter links between objects. Without technical interoperability, the motivation for providing semantic interoperability is lacking.

All of the advances will depend is some way upon the development and deployment of a consistent system of distributed primary keys and foreign keys (identifiers) and a system of tools and technology for supporting resource discovery, for facilitating automated multi-database joins, and for maintaining referential integrity in a distributed federation of loosely coupled heterogeneous databases.

Biological Databases as Publishing

Databases within commercial enterprises are information resources that determine the behavior of the organization. Paychecks are issued, products manufactured, shipments made, and invoices sent, according to the contents of the enterprise's databases. Since acting on the basis of inconsistent data would lead to

chaos, both within the enterprise and with its external interactions, commercial database management systems have emphasized the maintenance of internal data consistency and data integrity. Not unexpectedly, this emphasis has carried over into research efforts to develop multidatabase systems.

Scientific community databases, however, have more in common with scientific publishing than they do with the deterministic databases of business enterprises. Projects such as the Genome Data Base, or GenBank, or Swiss-Prot offer communication channels through which observations, sometimes inconsistent observations, may be shared among researchers. It is not required that scientific findings be rendered consistent with all previous work before they can be published in the print literature. Similarly, new observations submitted to a database need not be consistent with all other data before they can be entered.

These information resources, as seen by users, are better conceived as database *publishing* systems (DBPSs), not as database management systems (DBMSs). Of course, formal DBMSs will likely be used to create and manage individual information resources. But, when the data are made available to users, they are "published" in a sense, and it is interoperability among the resulting DBPSs that is greatly needed by the broad scientific community.

Achieving interoperability among loosely coupled read-only DBPSs is much easier than doing so with read-write DBMSs. With DBPSs, the notions of "loosely coupled" and "tightly coupled" are better considered as naming the ends of a continuum of relationships, rather than designating two mutually exclusive states. Figure 1 illustrates some possible points along the continuum.

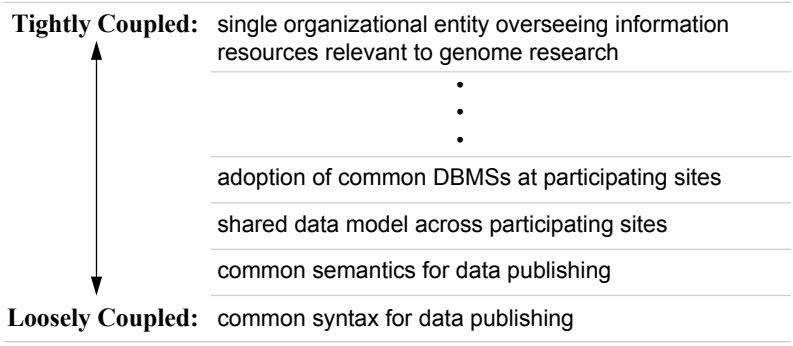

Figure 1. The distinction between tightly coupled and loosely coupled systems, seen as designating the ends of a continuum of relationships among database publishing systems. The tightest level of coupling yields a completely integrated, single management structure. The loosest level of coupling involves merely a collection of wholly independent organizations that publish their data in a common syntax.

Efforts to create federated information infrastructures are being made in many research fields, within and without biology. Together these constitute a global information infrastructure (GII) for science and technology. Boundaries between

GII components are not well defined, since there will always be a need to access information across different GII subcomponents. For example, understanding the genome will ultimately require integrating genome findings with protein structure (structural biology) and metabolic information (physiology), and comparative genomics must involve systematics and other areas of comparative biology.

Stand-alone database management systems provide robust local functionality, but low interoperability. Loosely coupled generic, read-only systems, such as the web, provide wide interoperability, but with low local functionality. Because the cost of mounting WWW servers was initially very small for those already building large local databases, many biological information resources began using WWW to supplement, not replace, existing services.

The value of participation in widely available generic systems, especially to users, can be astoundingly high, since the overall value of an interoperable network of cross-referencing information systems increases non-linearly with the number of participants. Thus, for life science research, attaining increasing generic ***database*** interoperability among all relevant information resources must be a continuing goal.

Database Interoperability

Today's crisis of data integration cannot be resolved through data consolidation (the collection of all relevant data in one facility), since the number of relevant information resources is large and growing.[1] Nor can it be solved by creating a distinct, officially sanctioned subset of data resources relevant to genome research, since it is simply impossible to identify a set of information resources that are all relevant to one, and only one, biological community.

Biological information resources dynamically group and regroup into transient overlapping collections of resources, with each collection being of special interest for some research discipline, or some individual researcher, at some time. As certain key databases (e.g., nucleotide sequence collections) play crucial roles in many such dynamic groups, physical or even administrative consolidation holds little prospect as a solution. Rather, advances will be required to allow autonomous data resources to interoperate productively. The challenge will be creating collections of data resources that are perceived by users to be functionally integrated, yet with each resource maintaining its autonomy, especially in the basic creation and maintenance of its data resources.

---

[1] The problems are as much social as technical: Would a scientific community tolerate the requirement that all publication in a given field must occur in only one journal? As electronic biological publications become easier to build, we can expect a general increase, not a reduction, in their number.

Integrating Distributed Information Systems

Structured databases are essential for life-science research, with the long-term success of the field depending upon significant functional integration among them. The absence of relevant generic tools renders the development of even small tightly coupled solutions difficult and *ad hoc*, with correspondingly short life expectancies and general utility. Although such systems may be needed for some identifiable subcomponents of the information infrastructure for life-science research, such development should be undertaken with care and with a clear recognition that when more generic tools become available much of the development effort expended on specific solutions may need to be repeated.

Some generic tools (e.g., WWW) have appeared that facilitate the development of loosely coupled systems and the ease with which some interoperability can be achieved using these tools suggests that most data providers should consider their adoption as *part* of their local interoperability strategy.

However, these tools are not designed for manipulating structured data and this limits their utility. Several gateways between these systems and local structured databases have been developed, but none provides for intersite connectivity at the data element level, an essential requirement with cross-referenced structured data. Even the current net-services models are missing several key features that will be necessary for delivering real interoperability.

DATA PUBLISHING IN A LOOSELY COUPLED FEDERATION

Although the web has been employed to tremendous use in the distribution of structured data by several major biological databases, present deployments are not capable of meeting all of the needs of the biological database community. Historically, the WWW had greater intellectual ties with information retrieval (IR) than with database development, and many differences exist between the needs of database users and the services delivered by IR systems:

- IR query systems are designed to support ambiguous queries and to resolve them using probabilistic retrieval systems. Databases, on the other hand, hold structured data and provide exact answers to well-formed, structured queries.
- Hypertext supports flexible linkages between objects, but more structured linkages, with defined semantics (such as a foreign key to primary key reference), are required for structured data.
- WWW servers often present their data objects one at a time. Many web systems that provide set-like queries generally produce a single listing of a set of data objects, then allow the user to retrieve the objects from the

listing one at a time. In database queries, users frequently want to obtain *sets* of objects that match their request.

- Hypertext links are available as paths the user may or may not choose to follow. Active steps must be taken to follow any particular step. Database queries frequently involve requested "joins" among data objects, in which the user wants to specify in advance what related objects are to be retrieved and in what connected configuration. Single database queries should be capable of returning large sets of joined objects, not merely the "option" of following what might be hundreds or thousands of hypertext links one mouse click at a time.
- Hypertext browsers are intended for human usability, with the assumption that they will present multiple navigation options to a human user. Database users frequently need a computational application programming interface with which to interact, so that they can direct an application program to extract and analyze data sets, then return the analytical results.

The list could be extended. But, the goal here is to offer neither the definitive characterization of the problem nor the definitive solution. Instead, we wish to establish that, *in their present form*, the widely available tools for easily fetching text and hypertext do not adequately meet the needs of those who desire integrated access into structured databases.

Many groups are working to extend WWW: technologies to handle more structured data, in varying degrees of generality. A good solution would do for databases what WWW has done for hypertext: provide an easy way to deliver transparent navigation through the holdings of information resources. The WWW approach involved a new data model (HTML documents), new protocols (e.g., HTTP), and, most importantly, a new vision for how information should be represented, organized, and delivered. It is presently an open question whether all the needs of structured database users can be met through clever additions to the WWW system, or whether substantial new database equivalents of HTML and HTTP will need to be developed.

Reference Architecture for a Federated Object-Server Model

Ten years ago, Robbins[1] offered a reference architecture[2] for a *Federated Object Server Model* (FOSM) as a "robust straw man" to stimulate discussion. FOSM was presented as a straw man in the sense that it was freely admitted not to be *the* (or even necessarily *a*) solution. But FOSM was also robust, in that it provided a

[1] In a keynote address at the Third International Conference on Bioinformatics and Genome Research in 1994.

[2] A reference architecture summarizes a system's basic functional elements and the interfaces between them. It identifies needed protocols and suggests groupings of functionality, but it does not imply a physical implementation.

focus around which requirements for interoperating structured databases may be considered. At the time, many of the ideas behind FOSM had no obvious technical solutions. Now SOAP, XML, and other systems provide ready models for implementing FOSM. However, many FOSM concepts still have no ready solution, including the concepts of FOSM resource discovery and FOSM data identifiers.

A review of the FOSM concept, emphasizing some aspects of the data model, is presented here. Because this is a review of ideas first presented more than ten years ago, the reader should pay less attention to the many aspects of the FOSM model for which solutions are now at hand and should instead ask which, if any, components of FOSM might still offer challenges to those interested in developing the identifiers and other tools necessary to support real data integration and coordinated retrieval.

FOSM Overview

Like WWW, the FOSM approach derives data structures and protocols from a vision of how a networked information space might operate. In FOSM, servers provide access to richly structured data objects that can contain semantically well-defined cross references to other data objects, allowing the rough equivalent of distributed joins in a relational database. The FOSM concept entails a strong commitment to resource discovery and resource filtering. Resource filtering, the deliberate restriction of queries to "trusted" sources, is essential if retrieved data are to be passed directly to other software for analysis.

As a robust straw man, FOSM provides a framework for continuing discussion. Whatever the architecture that ultimately supports global interoperability among database publishing systems, the quality of that design will benefit from significant input from biologists.[1]

Biology's claim to special relevance in driving information-management advances is real. In chemistry and physics all things of interest in a particular class (hydrogen atoms, electrons, quarks, etc.) are held to be genuinely, not metaphorically interchangeable. All living things, on the other hand, are truly unique, and the properties of individual living things are determined in significant part by the unique, frequently contingent historical events that happened to each of their unique ancestors.

The number of living things that now exist, that have existed, or that ever will exist is sufficiently small in relation to their information content, that we will never be able to apply some sort of law of large numbers so that they could be described in all interesting ways as essentially, if not actually, interchangeable items. Understanding biology will depend in part on managing information in a

[1] Many widely used statistics tests were invented specifically to solve genetics problems. For example, Galton devised regression analysis to compare the phenotypes of parents and progeny, Pearson developed the X^2 test to study the occurrence of different morphs in snail populations, and Fisher implemented analysis of variance to partition inherited variation. Efforts to devise better biological information systems will likely make similar contributions to the development of a global information infrastructure.

way that preserves the individuality of the subjects, and this makes solutions to biological information-management problems highly relevant in the commercial sector.

FOSM Applicability

The FOSM approach is generally applicable to any set of information resources involving structured data. Examples would certainly include scientific data resources and also many types of commercial information, either to be published externally for customers or as an internal resource within an enterprise.

FOSM Assumptions and Requirements

The FOSM approach begins with the collection of basic assumptions and requirements. Some examples are offered here.

Basic Assumptions

The FOSM system will follow a generic client-server design, emphasizing autonomy of local sites and enabling structured queries into structured data.

- Participating sites will maintain their databases in whatever manner they choose, but to participate in a federation they will make their data available in a read-only format via a standard object-server system.
- Participating databases will "publish" their data as objects, represented in a standard data model.
- Generic client software will be used to obtain data from the read-only federation.
- With a single query, users will be able to obtain *sets* of related data objects from multiple independent data resources.

General Requirements

- The system should be relatively impervious to changes in data volume or in the number of participating sites—i.e., scalability is essential.
- The system must be designed to facilitate value-adding activities by third-party developers.
- The system should be easy to install and operate. Ideally, it would approach plug-and-play simplicity.
- The system must be robust in a dynamic, heterogeneous environment.
- The system must be data driven and self configuring. This means that a naive client should be able to contact a server for the first time and, as a result of transactions with the server, produce a usable user interface and initiate a query dialogue.
- The system should provide a local (i.e., client side) API, as well as the networked API into the server.

- The system should permit "subscription" to user-constructed queries. That is, there should be some way for users to capture the steps necessary to execute a query, then request the system to execute that same query on regular timed intervals, returning data to the user via some specified route (email, ftp, etc.).
- The system must support data retrieval both in human readable and in computable format.
- The system must provide support for multiple concepts of object identity.
- The system must provide support for resource discovery in a manner at least loosely equivalent to that offered by the data dictionary in a stand-alone database.
- The system must support the equivalent of foreign key to primary connectivity between objects in different databases.
- The system must be able to provide query operators more or less equivalent with the SELECT, PROJECT, and JOIN operators of relational databases.
- The system must provide some minimal support for domain and referential integrity across entries in multiple data resources.
- The system must support both outer and true equi-joins across distributed object servers. Semantically well-defined cross-referencing (equivalent to foreign key to primary key references in a relational database) must be representable in the data structures and traversable by the system software. It must be possible to traverse such links without human intervention (mouse clicking, as is done to traverse hypertext links, cannot be a requirement).

Server Requirements

FOSM servers will need to provide both data to satisfy queries and metadata to support building and operating the client interface. They will also need to provide some server-to-server information to help maintain external references.

- FOSM servers must provide full-function anonymous data serving.
- FOSM servers must support negotiation with clients regarding protocols, data, and formats.
- FOSM servers must support both value-based queries and identity-based queries.
- FOSM servers must serve several different kinds of objects: "type objects" that document the structure of the data objects so that the client software can produce an appropriate query and retrieval interface; "data objects" that contain the actual data of interest; "help objects" that contain help messages to be used by the client to provide context-sensitive help messages.

- FOSM servers should provide support for remote domain and referential integrity in external servers. That is, if one FOSM servers references another server, the second server should provide specific support to assist in maintaining the integrity of references towards it.

Client Requirements

To support the needs of database users, the FOSM client will need to be able to maintain more *customizable* functionality than does a typical WWW browser. Some specific requirements of the FOSM client are:

- FOSM clients must be able to build dynamically custom forms-based graphical interfaces to allow the interrogation of any FOSM server. To do this, FOSM clients will obtain metadata describing the structure of objects served by a particular FOSM server.
- FOSM clients must allow users to manipulate the structure of data objects from one server, or combine structure objects from different servers, to build single, virtual objects against which unified queries may be dispatched.
- FOSM clients will need to "negotiate" with FOSM servers regarding the format and structure of objects requested.
- The client must support "batch" as well as interactive, retrieval operation. That is, users must be able to execute large, complex queries automatically during off-hours by writing scripts or otherwise storing commands which can then be executed later.
- The client software must allow significant user customization both in the configuration of the local software, but also in the configuration of interfaces into particular databases.
- FOSM clients should be able to direct unified queries to multiple data resources simultaneously.

Resource-Discovery Requirements

The FOSM approach assumes that users will need assistance in identifying relevant FOSM objects and servers. It also assumes that a key part of resource discovery is resource filtering—i.e., the explicit rejection of data objects from undesirable sources. Therefore, the FOSM approach supports the free development of "editorial" activities, so that editorial bodies may indicate approval for individual FOSM objects, or for individual FOSM servers, or for sets of objects or servers. Editorial annotations could be hierarchical. That is, an editorial board might wish to assign its approval to all of those objects already approved by editorial boards A, B, C, and D.

Resource discovery tools are of no use, if they are difficult to locate. Therefore, access to FOSM resource-discovery tools should be a built-in component of the FOSM client. Whether this should be accomplished via a

central, known source, via distributed search engines or via some significant extensions to self-propagating name systems (like DNS) is an open question.

Data-Structure Requirements

Just as the HTML data structure is the key to WWW functionality, so an appropriate data structure will be required for handling structured data. Here are some preliminary notions of what the basic FOSM data structure should do.

- The data structure must be able to represent considerable (arbitrary?) complexity.
- The structure must be able to offer meaningful representations of data objects extracted from different underlying DBMSs (e.g., RDBMS, OODBMS, etc.)
- The structure must be readily parsable.
- The structure, or some consistent representation of it, must be reasonably easy to understand. (This would facilitate the development of virtual objects by users and/or third-party developers.)
- The structure should be closed under basic retrieval and manipulation operations.
- The structure must robustly and unambiguously support the ability of data objects to contain, as attributes, references to data objects published elsewhere.
- In its most basic form, the data must be *self-describing*, so that almost anything can be represented, yet *constrained*, so that generic client tools can be developed.

FOSM Architecture

FOSM architecture is based on a generic client-server approach, with explicit support for middleware development by third-party developers. A registry of FOSM information supports both direct queries and resource discovery activities. Whether the registry should be a central database, or a system that supports duplicated information propagation (such as domain name servers) is an open question. The registry would hold information about FOSM servers, FOSM objects (and versions), FOSM links, FOSM subfederations, FOSM editorial records, FOSM methods, FOSM names, FOSM cataloging, etc.

An overview of the FOSM architecture is given in Figure 2.

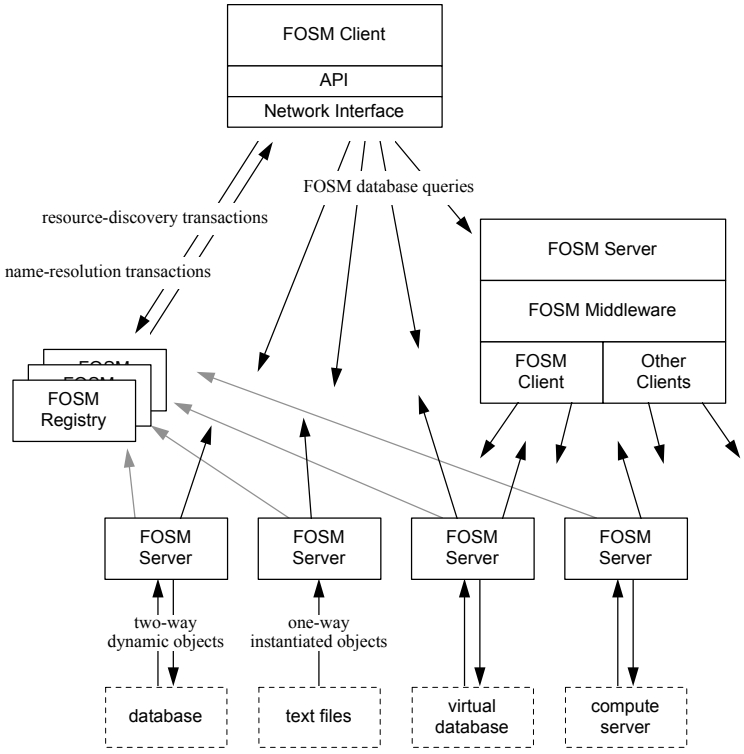

Figure 2. FOSM clients interact with FOSM servers and with a FOSM resource registry. Servers publish holding information to the registry (gray arrows) and respond directly to client queries (black arrows). Explicit support for n^{th}-party developers is provided, through the encouragement of middleware development.

The FOSM client (Figure 3) is built around a central kernel, the FOSM user-interface manager (UIM), which interacts with various local programs and remote servers. The UIM would probably be some kind of script interpreter, possibly a generic script interpreter so that more than one scripting language could be used. The UIM core is surrounded by a variety of other programs, which are invoked to call the local execution of "methods" associated with remote data objects, and files, which provide appropriate metadata and caches.

FOSM views will allow users to create local views on FOSM objects or to build virtual FOSM objects. To build a FOSM interface, the client must first query a server to obtain necessary type and format information. This, and other FOSM metadata, should be storable in a local cache. The size of the cache should be under user control. Normally, the cache would be first-in, first-out, but the user should be able to specify certain cached elements that are never to be flushed.

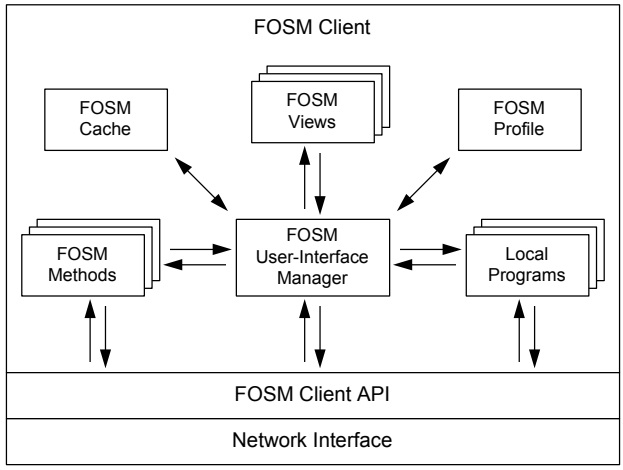

Figure 3. The FOSM client provides much of its functionality through its component-based design. All aspects of the FOSM system are intended to facilitate the value-adding activities of third-party developers. That is, it should be easy for users to install locally FOSM methods or views or profile components created elsewhere.

FOSM methods are local, hardware-specific[1] software packages that are invoked to "view" objects obtained from FOSM servers. For example, one of the standard local methods would display and operate HTML documents; another would build, display, and operate query interfaces for FOSM objects.

A FOSM profile will allow users to customize the behavior both of the local client and of remote servers without requiring servers to maintain registries of users and preferences. The FOSM API should allow easy development of local programs that interact directly with the client API, without requiring assistance from the UIM, thus facilitating the development of third-party bulk-data-transaction modules for special markets: DNA sequences, finance, etc.

FOSM Data Model

A generic tree data structure provides a fundamental data representation that meets FOSM requirements. Any data model that be represented in an extended entity relationship (EER) schema can have read-only data objects extracted from it into tree-shaped configurations. Each type of tree represents one class of real-world objects and each individual tree corresponds with one member of that class.

Figure 4 shows how a portion of an EER schema could be converted into a tree-shaped data object. The occurrence of the same entity from the EER diagram at different nodes in a tree indicates participation of different members of that

[1] This is the ten-year-old FOSM approach. Now we would expect the hardware-specific aspects to be handled in some kind of local virtual machine (e.g., Java interpreter) so that the methods could, ideally, be write-once, run-anywhere.

entity class in different roles. For example, the faculty data-object tree also includes "faculty" at two sublocations, one corresponding to the role of "departmental colleagues" and the other of "departmental chair." Individual FOSM trees are one-to-many downward, and lower nodes can be considered as sets of sub-objects, associated in some role as attributes of an object at the next higher node.

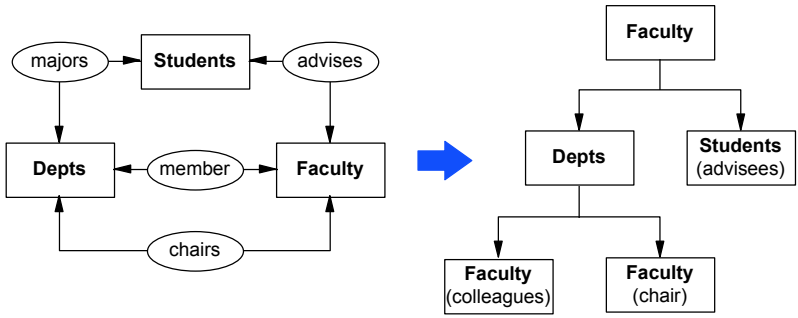

Figure 4. Tree data objects can be easily extracted from EER schemas. Here a "faculty" object is extracted from a portion of a university database schema.

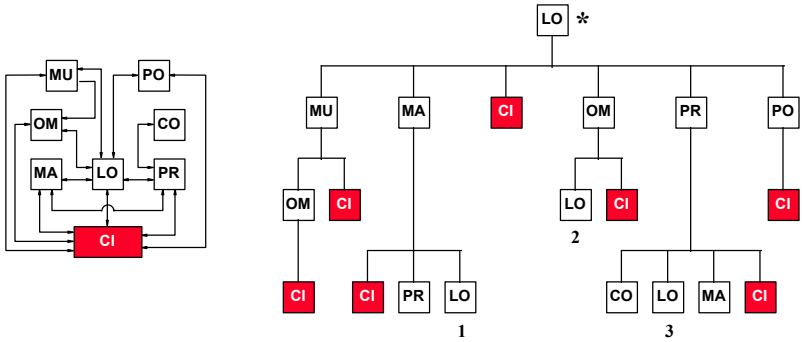

Figure 5. A "locus" object is extracted from a portion of the Genome Data Base schema. (LO = locus, MU = mutation, MA = map, CI = citation, OM = OMIM, PR = probe, PO = polymorphism, CO = contact.)

Figure 5 shows a similar extraction performed upon a real biological database. In this tree, the "citation" node appears seven times, in each case representing a semantically different type of citation. Even the root node (*) of "locus" appears several times. In a particular instance of a locus tree, the root node would contain all of the single-valued attributes associated with a particular locus. The locus node marked with "1" would contain a set of other loci, corresponding to loci known to co-map (i.e., be carried on the same maps) with the root locus. The node marked "2" would contain those other loci known to cause the same

OMIM phenotype as the root locus. The node marked "3" would contain those loci recognized by the same probes as the root.

Individual tree-shaped data objects could be "selected" from a data server either through value-based or key-based queries. Once obtained, the data objects can be manipulated using operators such as "prune" and "graft" (Figure 6). These operators produce results similar to those of the "project" and "join" operations in relational databases. The operations are "closed" in that they are defined to have well-formed trees as inputs and to produce well-formed trees as outputs.

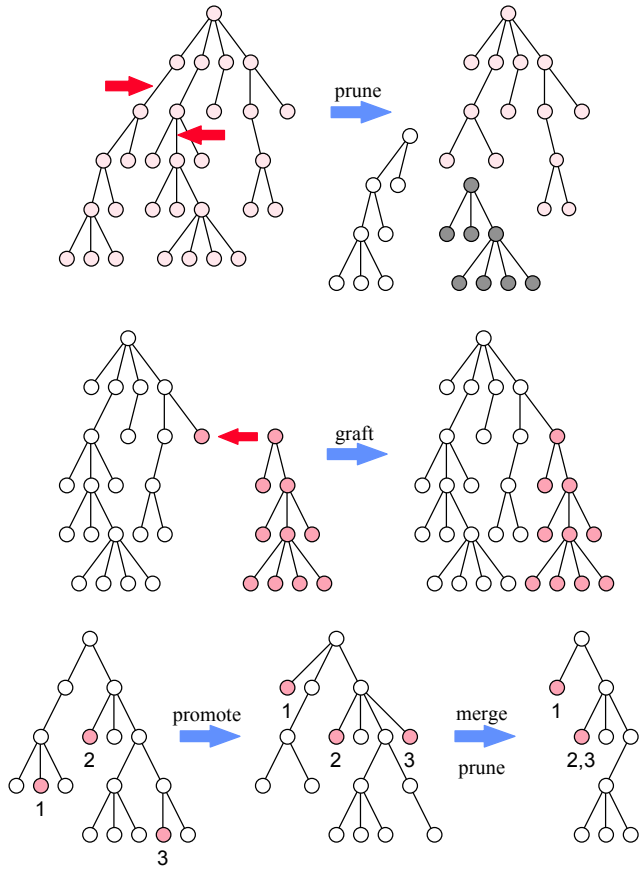

Figure 6. The "prune" operator is similar to the relational "project" operation. The "graft" operator is similar to the relational "join" operation. The "promote" operator allows the movement of nodes to higher positions in a tree, through a combination of pruning and grafting. If promotion results in multiple nodes defined over the same domain being attached at the same point in the tree, the "merge" operator combines them.

Prune and graft could be combined to give a "promote" operation that could move nodes higher up the tree, eliminating intermediate nodes (and requiring

some role definition refinements). The FOSM client would allow the user to create such custom trees, then store them locally to be used in driving queries to underlying data resources. This would give the ability to operate within a custom-tailored environment, while sparing servers from the need to maintain profile information on individual users.

FOSM Data Identifiers

To be "federation ready" a FOSM server would have to provide absolutely stable, unambiguous identifiers for every rooted object in its published collection. Similarly, every external reference in a FOSM server would be in the standard format for global FOSM names. All rooted FOSM objects must be unambiguously identifiable in a global FOSM name space of arbitrary identifiers. Although biological names are too volatile to serve as FOSM identifiers, value-based queries of FOSM objects must be supported so that researchers can interrogate the system using familiar terms.

In a single copy of a stand-alone database, object identity is a fairly simple concept. However, in a FOSM system, *copies* of objects will be distributed from servers to clients where they may be stored for local use. Occasionally, then, clients will need to compare object copies to determine their equivalence. This raises some slightly more complex notions of identity. For example, each FOSM object can be subdivided into three components: (i) an object identifier, (ii) an associated type tree, that specifies what attributes the object *could* have, and (iii) an associated data-value tree that specifies what attributes the object *does* have and gives their values. This allows at least three different concepts of identity, which we will call *semantic identity*, *computational identity*, and *true identity*:

- Two copies of FOSM objects are said to be *semantically identical* if they have the same object identifier. This is the most fundamental component of identity and it persists across value updates to the object's attributes and even across schema updates to the object's type tree.
- Two copies of FOSM objects are said to be *computationally identical* if they are semantically identical *and* they have identical type trees. However, computationally identical objects could have different values stored for the object's attributes.
- Two copies of FOSM objects are said to be *truly identical* if they are computationally identical *and* they have exactly the same values for all of their attributes.

Additional identity concepts can be derived from these. For example, we might want to say that two objects are *apparently identical* if they have identical type and value trees, but different identifiers. To facilitate identity comparisons, FOSM objects could carry two computed identifiers, a *type identifier* one defined over the object's type tree and a *value identifier* defined over its value tree. These computed identifiers would calculated on the fly, when an object is provided by a FOSM server, much as check sums are calculated anew each type an IP packet is

placed on a physical medium. These calculated identifiers would also be useful for detecting corruption in local copies of FOSM objects.

Type identifiers could also be used to associate particular computational methods with FOSM objects. For example, semantically identical DNA sequence objects could be represented in computationally different FOSM trees that are equivalent to flat-file, ASN.1, BLAST, etc., formats. Each format would have a specific type identifier and this could be used automatically by software to determine the appropriate parser to be used in analyzing the data.

Schema version changes would also be reflected in type identifier changes. To allow ready detection of specific versions, perhaps the type identifier should contain two parts: one specifically giving the version number and the other a computed value derived automatically from the contents of the type tree itself.

A major goal of the FOSM approach is providing a scalable, automatable system for delivering structured data objects across a federation of autonomous resources. Achieving this will *require* that type identifiers contain a computed component so that software can check automatically to determine if it knows how to read and process the data. Data resource developers will differ in their personal notions of what changes are sufficiently significant to constitute a change in the designated version of the database. However, some third-party software may rely upon the precise configuration of data from a particular resource and would break in the face of even tiny changes in the schema. The only way to ensure that type identity is genuinely preserved is to eliminate human judgment (is this a big enough change to warrant a version change?) and to replace it with the use of check-sum-like computed identifiers (any change, even one bit, warrants a version change).

In the short term, care must be given toward the specification of appropriate global naming conventions to enable a global information infrastructure for biology. In the longer term, efforts by the overall networking community to modify network protocols to support transparent interactions among networked information resources, not just networked hosts, will likely provide a more complete solution. Until such functionality is delivered, those developing federated biological systems should take care to communicate their naming requirements to the appropriate organizations and developers.

Data-Level Integration Across Multiple FOSM Servers

The FOSM system would support data-level integration across data objects from multiple servers. For example, information on mammalian genes could be published by several different FOSM data servers (Figure 7).

Each server would have the local responsibility and autonomy[1] for formatting and publishing its own holdings in the form of trees. Leaves on the trees published

[1] Social pressures might exist on data resources to provide physically similar trees for semantically similar objects. However, these pressures would be external to FOSM itself, which only requires that servers adhere to the FOSM tree syntax.

by one data server could contain "tokens" that represent the roots of specific data trees available from other servers (Figure 8).

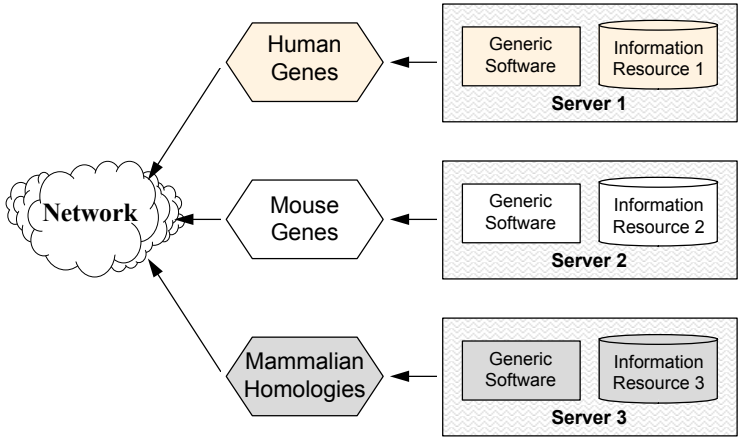

Figure 7. In a FOSM environment, individual data resources would publish their holdings to the network in a standard format, according to standard protocols.

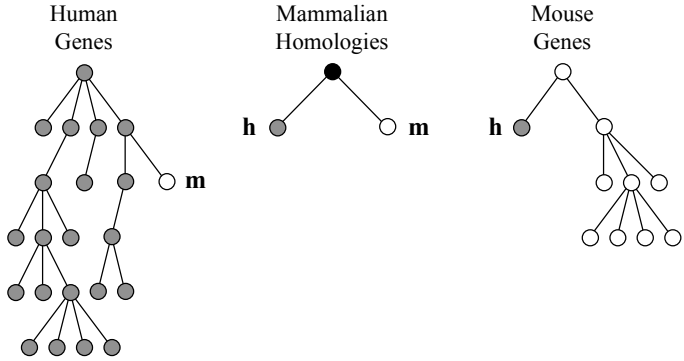

Figure 8. Possible tree structures for data objects published by FOSM servers. Nodes marked with "m" and "h" represent sets of tokens that would correspond to the root nodes for mouse-gene and human-gene objects respectively. The inclusion of these external references as leaf nodes indicates that the designer of the local database believes that these external objects are related to the database's primary objects in some role (which is defined in the local database). The decision to include such references, and the populating of them with values, would be the responsibility of the local FOSM server.

As long as all participating data servers followed these simple guidelines, and providing that a global naming system offered access into a stable, unambiguous naming space for FOSM objects, generic client software could allow users to navigate easily among related data items from different servers.

If data from different servers are combined using the "graft" operator, new trees are produced. For example, Figure 9 shows human-gene objects extended to include mouse genes as attributes, and vice versa. Mammalian-homology data objects could be extended to include both human and mouse genes as attributes.

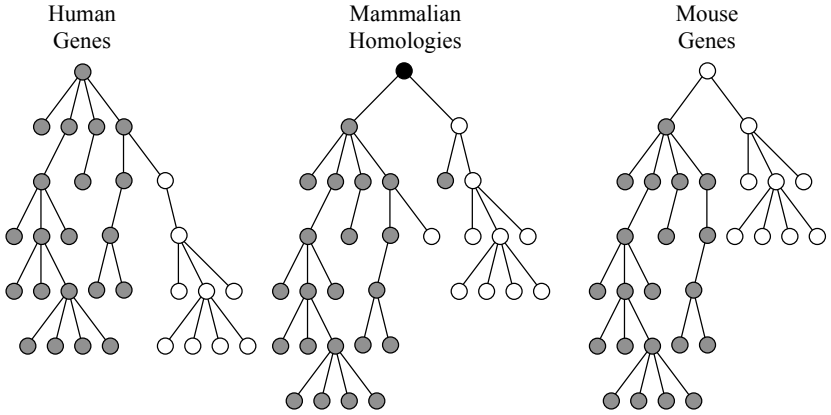

Figure 9. Related data objects may be obtained from different FOSM servers, then grafted together to give new, compound objects.

If data about human genes, mouse genes, and their possible homologous relationships were contained in a single database, obtaining the set of asserted homologous gene pairs would involve a simple, unambiguous join. In the FOSM model, however, individual data providers may offer data objects that reference objects in other databases. Different data providers would be free to publish logically equivalent, but not necessarily content-identical linkages among data objects, as there would be no formal requirement of identity. This freedom to diverge is necessary to allow the information resources to act as scientific literature, which must be able to support differences of opinion

SUMMARY

Biological databases, having survived a crisis of data acquisition, now face a crisis of data integration. Meeting this challenge will require the development of technical and sociological processes that will allow multiple databases to interoperate functionally, while still maintaining much of their individual managerial autonomy. Horizontal partitioning[1] of data, as is the case across some

[1] Some genome databases, such as sequence databases, involve a *horizontal partitioning* of data, in that data about similar objects are maintained in different locations. *Vertical partitioning* describes situations where data of different types (e.g., sequences and genes) are maintained at different locations.

genome data resources, makes the challenge of interoperability especially acute, since achieving good interoperability under these circumstances will require the development of considerable semantic consistency among participating sites.

Computer solutions that, from initial design onwards, are aimed at meeting the specific needs of some particular problem rarely evolve into generic interoperable systems. Solutions that are based on minimal generic components are more likely to evolve gracefully into specific systems, especially if the specificity is added as layers on top of the underlying generic foundation. Networking architectures have followed this pattern and the evolution of database systems from file-based approaches to cutting edge object-oriented databases show a similar trend.

To be truly useful to the widest range of potential users, on-line genome information systems should be capable of functionally interoperating, at some minimum basic level, with many different information systems (such as nucleotide sequence databases, clinical phenotype information systems, metabolic databases, systematics databases, etc.). Successful interoperation among a large, diverse, and autonomous set of independent data sites can only occur if all sites use equivalent, generic tools to publish their holdings according to common protocols and syntaxes. The web offers examples of the power in this generic client-server approach to information distribution, but it does not yet meet all of the needs of those interested in publishing structured data, especially those interested in published into a loosely coupled federation of heterogeneous systems across which anonymous interoperability is likely to be more common than organized collaborations..

An extended data publishing model, perhaps related to the FOSM concept discussed here, will be required if these needs are to be met in a generic fashion. In such a model, local sites are still free to manage their data internally according to whatever methods seem best. More importantly, collections of sites are free to react to scientific needs for convergence upon similar methods for internal data management, as well as upon common consensus data models and semantics for external data publication, while at the same time using generic methods, protocols, and syntaxes for data publication. The adoption of generic client-server methods for data distribution is purely an enabling technology. By not requiring common semantics of anyone, it allows for unrestricted syntactic interoperability. By permitting the adoption of common semantics by some, it facilitates unrestricted semantic interoperability.

Individual life-science communities could achieve the best of both worlds if they achieve interoperability by sandwiching generic data-distribution methods between converging internal data-management systems on one hand and common public consensus data models and semantics on the other. This would yield a unified conceptual model for their community's data, delivered in a system capable of generic interoperation with other communities' resources.

All of this assumes a truly adequate approach to managing the various life-science concepts of identity. Care must be taken to distinguish among the various different kinds of identity, their origins and their implications.
Technically different kinds of identity might include:

Asserted identity: the keys are the same but the data values may differ;

Computed identity: the keys and all[1] of the data values are the same;

Inferred identity: the keys are different but all of the values are the same.

Conceptually different kinds of identity might include:

Identity by state: the components are identical but the descent may differ

Identity by descent: the components are identical AND the provenance is identical

Identity by homology: the evolutionary provenance is identical but the state may differ

Identity by analogy: the state is identical (really just similar) but the evolutionary provenance does differ.

Semantically different kinds of identity might include:

Homologous genes: To traditional biologists, "homologous genes" means the evolutionary provenance is identical but the state may differ.

Homologous genes: To molecular biologists, "homologous genes" means the sequence (i.e., state) is similar, but the evolutionary provenance may differ.

Historically different kinds of identity might occur when biological concepts evolve over time but no one bothers to explore formally all of the implications of those changes. For example, fifteen years ago I asked a room full of microbiologists a simple question:

> Suppose a strain of *E. coli* is found in which the entire *lac* operon has been translocated to a different part of the chromosome. How should we describe the new location of the gene for ß-galactosidase? Should we say that the gene has moved to a different locus, or should we say that the locus is in a new position?

In other words, I was asking them whether or not gene and locus referred to the "same" concept. The problem, of course is that the concept of "locus" was originally developed when genes were assumed to be beads on a string. The gene was the bead and the locus was the position on the string where the bead occurred. A molecular understanding of the gene requires the recognition that there is no string that can be separated from the gene. So, does molecular understanding

[1] Or perhaps most of the values are the same, or some of the values are the same, or even just a few ‚critical values are very similar.

require that we associate "locus" with position in the genome or with the gene? That is, the word "locus" now means the "same" as _____?

Everyone in the room responded that the question was trivial and that the answer was obvious. When asked to specify their answers, the room split 50:50 in their opinions. As each half of the room tried to figure out how the other half could manage to be that wrong, I walked away contemplating the challenges that would go with efforts to build databases that depended upon such divergently interpreted biological concepts for their data models.

The problem of devising truly workable life-science identifiers is no easier. What, exactly, are we identifying? How do we deal with different concepts of identity? Whose definition of "same" do we use, or if we cannot find an authoritative source, how do we manage different concepts of sameness. In many cases, the concept of identity in biology is context dependent. How do we deliver context-dependent identifiers and avoid chaos? And the list goes on.

Who ever said this was going to be easy…