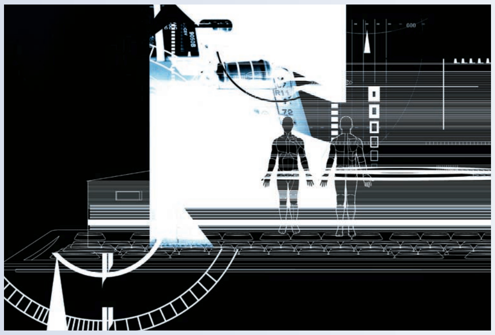
Integrating Biological Research through Web Services

A case study demonstrates that Web services could be key to coordinating and standardizing incompatible applications in bioinformatics, an effort that is becoming increasingly critical to meaningful biological research.



Hong Tina Gao
Jane Huffman Hayes
University of Kentucky

Henry Cai
Big Lots

No longer only a field of experimental science, biology now uses computer science and information technology extensively across its many research areas. This increased reliance on technology has motivated the creation of *bioinformatics*, a discipline that researches, develops, or applies computational tools and approaches for expanding the use of biological, medical, behavioral, or health data.[1]

Because tools and approaches cover how to acquire, store, organize, archive, analyze, and visualize data,[1] bioinformatics is a promising way to help researchers handle diverse data and applications more efficiently. Unfortunately, at present, bioinformatics applications are largely incompatible, which means that researchers cannot cooperate in using them to solve important biological problems. The “The Integration Challenge” sidebar explains this problem in detail.

Web services might be a way to solve the integration problem because Web services technology provides a higher layer of abstraction that hides implementation details from applications. Using this technology, applications invoke other applications’ functions through well-defined, easy-to-use interfaces. Each organization is free to concentrate on its own competence and still leverage the services that other research groups provide.

To test the potential of a Web services solution, we implemented a microarray data-mining system that uses Web services in *drug discovery*—a research process that attempts to identify new avenues for developing therapeutic drugs. Although our implementation focuses on a problem within the life sciences, we strongly believe that Web services could be a boon to any research field that requires analyzing volumes of data and conducting complex data mining.

WHY WEB SERVICES?

A Web service is a group of network-accessible operations that other systems can invoke through XML messages using the Simple Object Access Protocol (SOAP). The service can be a requester, provider, or registry. A service provider publishes its available services on a registry. A service requester looks through the registry to find the service it needs and consumes the service by binding with the corresponding service provider. The services are independent of environment and implementation language.

In biology research, these traits are advantageous because, as long as the interfaces remain unchanged, researchers need not modify the application or database or unify diverse schemas. Moreover, invoking a Web service can be as easy as checking an information directory and calling the right number. Given that data analysis is the most time-consuming step in many bioinformatics applications, this simplicity makes it tolerable to incur even the overhead of transmitting XML tags for explaining the data structures.

Web services also transform biology's current ad hoc software-development architecture into a component-based structure. Unlike technologies such as the common object request broker architecture (Corba), using Web services makes it easier to glue components together by exploiting existing standards and implementing underlying communication protocols instead of using a specifically defined transportation protocol for each technology. Corba assumes that its users will be competent programming professionals. Web services are oriented toward the less technical IT communities.

For biological researchers in highly specific subfields, the less technical solution is far better. A group annotating a human genome segment, for example, must precisely locate genes on genomes and assign genes to their protein products. To invoke services that implement the needed algorithms, the researchers simply acquire the services' descriptions from the registry and generate SOAP requests to those services. They don't have to know how to implement the algorithms.

More important, because integration occurs at the client instead of on the server side, service providers and requesters have more flexibility and autonomy. Each service provider can incrementally add value to the overall community by building Web services that integrate existing services.

WEB SERVICES IN DRUG DISCOVERY

Our microarray data-mining system uses Web services to identify potential drug targets—molecules with problematic biological effects that cause diseases in animal models. The drug targets then serve as a basis for developing therapeutic human drugs.

With a better understanding of human genes, scientists can identify more drug targets and design more effective drugs, but traditional techniques—those based on one gene in one experiment—discover gene functions too slowly. Many high-throughput genomics technologies, such as microarrays and gene chips, could speed up gene-function analysis. Arranging gene products in a microarray lets researchers monitor the entire genome's expression on a single glass slide[2] and gain insight into the interactions among thousands of genes simultaneously.

Drug discovery scenario

Drug discovery using a microarray involves a chain of large-scale data-processing modules and databases. In our implementation, we wrapped each module in the data-analysis chain into a Web service and integrated them. We then built a portal to make this aggregated functionality available to users.

The Integration Challenge

Because of the Human Genome Project's great success, the current research trend in the life sciences is to understand the systemic functions of cells and organisms. Not only has the project increased data on gene and protein sequences, it has further diversified biology itself. Many study processes now involve multistep research, with each step answering a specific question. Researchers in vastly different organizations design and develop computing algorithms, software applications, and data stores—often with no thought as to how other researchers are doing the same tasks. Consequently, one interdisciplinary research question might require interactions with many incompatible databases and applications.

The study of *E. coli* enzymes is a good example. Researchers must visit EcoCyc, Swiss-Prot, Eco2DBase, and PDB to obtain information about the enzymes' catalytic activities, amino acid sequences, expression levels, and three-dimensional structures.[1] This labor-intensive process can be even more tedious if the research requires studying thousands of genes.

An integrated process that follows a certain research pathway is thus critical, and its successful evolution depends heavily on the compatibility of the applications involved. The current incompatibility level of bioinformatics applications makes integration of data sources and programs a daunting hurdle. From cutting-edge genomic sequencing programs to high-throughput experimental data management and analysis platforms, computing is pervasive, yet individual groups do little to coordinate with one another. Instead, they develop programs and databases to meet their own needs, often with different languages and platforms and with tailored data formats that do not comply with other specifications.

Moreover, because biology research lacks a well-established resource registry, no one can share information efficiently. Users from diverse backgrounds repeatedly generate scripts for merging the boundaries between upstream and downstream applications, wasting considerable time and effort.

The integration challenge is not just for those in the life sciences. Any discipline that deals with massive amounts of data and computing loads and geographically distributed people and resources faces the same problem: economics, Earth sciences, astronomy, mechanical engineering, and aerospace, for example. Solving the integration problem in the life sciences will provide vital benefits to these fields as well.

References

1. P.D. Karp, "Database Links Are a Foundation for Interoperability," *Tibtech*, vol. 14, 1996, pp. 273-279.

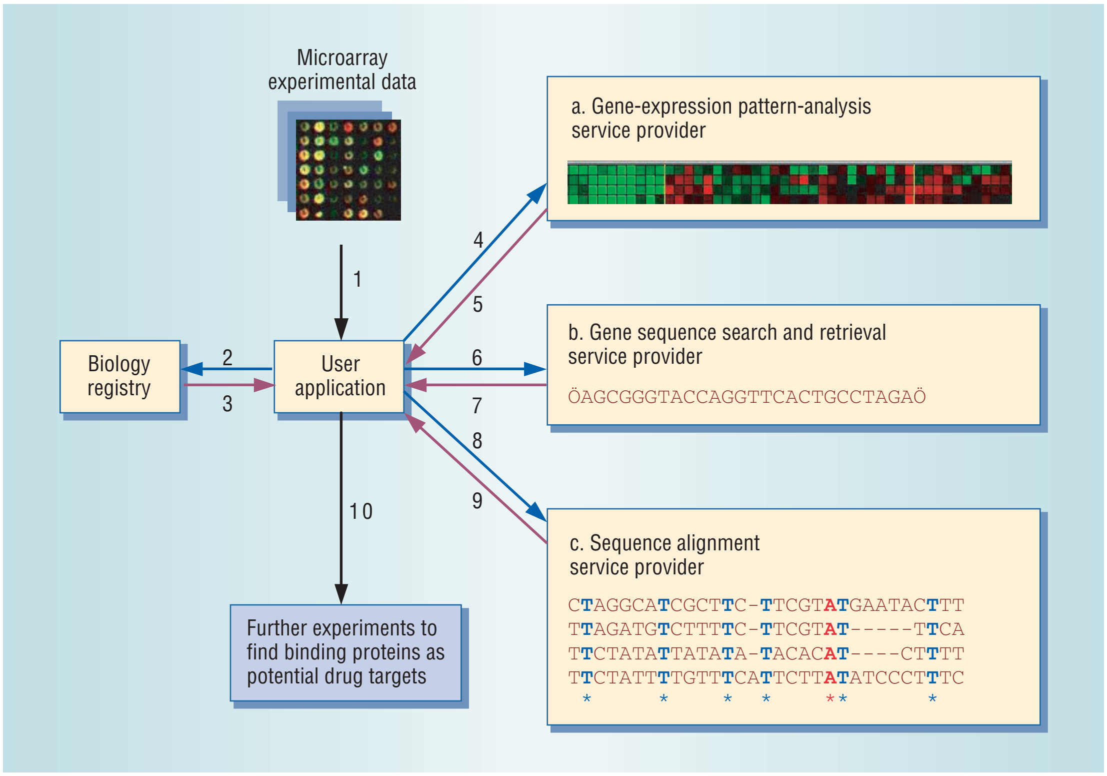

Microarray experimental data
a. Gene-expression pattern-analysis service provider
b. Gene sequence search and retrieval service provider
ÖAGCGGGTACCAGGTTCACTGCCTAGAÖ
c. Sequence alignment service provider
CTAGGCATCGCTTC-TTCGTATGAATACTTT
TTAGATGTCTTTTC-TTCGTAT-----TTCA
TTCTATATTATATA-TACACAT----CTTTT
TTCTATTTTGTTTCATTCTTATATCCCTTTC
Biology registry
User application
Further experiments to find binding proteins as potential drug targets
1
2
3
4
5
6
7
8
9
10

Figure 1. Microarray data-analysis scenario for identifying targets in drug discovery. Components a, b, and c are three service providers that provide Web services for the data analysis related to drug discovery. The numbered lines are the steps in the analysis path. A researcher passes the data collected from a microarray experiment to a user application (1), which queries a biology service registry for the locations of service providers (2 and 3). The user application invokes the Web services provided by the three service providers (4 to 9). The user application transmits the result of an upstream service as the input of the next downstream service. Finally, the researcher passes the result of the last queried Web service to other drug discovery experiments (10).

Figure 1 shows the three Web services in the scenario and the data-analysis path to discover drug targets using these services. The path begins with the user finding the URLs of the necessary Web services from a biology service registry. She then queries those remote services to find similar fragments from the gene sequences that have similar expression patterns in the microarray experiments. Finally, she uses the fragments in additional experiments to identify drug targets.

Scenario implementation

We decided to implement scenario steps in three applications that use different, largely incompatible algorithms or databases to accomplish their tasks. We reviewed only applications we felt we could easily translate into Web services. The candidate applications had to have

- good encapsulation of implementation details,
- clear interface definitions, and
- simple input and output data structures.

Table 1 lists the three applications we selected: IBM's Genes@Work,[3] the National Center for Biotechnology Information's Entrez Databases,[4] and the Baylor College of Medicine's Search Launcher.[5]

We then built a Web service for each scenario step using a mix of green-field and bottom-up strategies.[6] The green-field strategy is a from-scratch implementation of both the Web service's description and its functionality. The bottom-up approach is similar except that the functionality it exposes as a Web service already exists.

Next we rewrote the interfaces for each application. For Genes@Work, the interface takes as its input the gene-expression data set file and the corresponding phenotype file for each microarray experiment and returns an expression pattern file. We adopted SOAP with attachment technology to transfer the files.

Finally, we wrote the service interface and implementation descriptions. Many tools are available to help generate these definitions, but we used the Java2 Web Services Description Language (WSDL) tool in IBM's Web services toolkit.[6]

We published the service interface and implementation in a local registry suitable for testing and for restricting user access to services. In some cases, the service provider might want to make the services available to the entire community. If so, a public registry, such as universal description discovery and integration (UDDI) or a special biological registry would be more appropriate.

We also built a client platform to consume the services. Users can invoke the three services independently as network-accessible stand-alone applications or as a group, to perform the tasks in our scenario. This system provides more flexibility for researchers to use the functionality in the three applications we chose, while data integration through Web services streamlines the entire analysis process.

Results

With the Web services, service registry, and Web portal we built, we were able to smoothly pass the experimental data from microarray experiments to individual service providers and perform the analysis in Figure 1.

Although we were the only users who performed pilot tests with this system, we believe anyone doing drug discovery research could easily use this system with very little computer training. Users must understand only generic operations, such as loading data files, entering the index of the gene expression patterns of interest, selecting the genes of the sequences to be retrieved, and so on. They need not worry about writing their own patches of scripts to transform data among incompatible programs from various versions.

Our system handles the many time-consuming and tedious transformations between data formats. The only time left is the time it takes for each service provider's analysis program to execute and the delays from network traffic.

Using the traditional approach, it could take a user one hour to set up the Genes@Work standalone application and a few more hours to manually transform results to the legible input for gene identification. This doesn't include the mechanics of cutting and pasting, which can take 5 to 10 minutes per operation, depending on how many patterns a user must query.

With our system, typically it takes approximately 10 minutes from uploading the microarray data to finally clustering the gene sequences of the expression patterns of interest. Considering that a microarray experiment usually includes a few hundred to thousands of genes, our system saves significant time, most of which is otherwise spent in tedious tasks.

Lessons learned

In conducting this project, we discovered two keystones to the successful and widespread use of Web services in biological research.

Table 1. Selected applications for the drug discovery scenario.

Application	Scenario component	Description
Genes@Work	a	A package that automatically analyzes gene expression patterns from the data microarray that technologies obtain
Entrez Databases	b	A search and retrieval system that stores nucleotide sequences, protein sequences, and other sequences
Search Launcher	c	A project that aids in clustering gene and protein sequences

Well-defined interfaces. To support a services-oriented architecture, each software component must have a well-defined function and interface. If functions for different components are orthogonal, software coupling will be minimal, which will make it more convenient to transform these components into Web services that many kinds of researchers find acceptable.

One way to achieve clean functions and a decoupled software architecture is to use an object-oriented design with systematic analysis in conjunction with design patterns. The bioinformatics software we worked with, for example, required much refactoring to separate the calculation logic from its Java Swing interface. Had its implementers followed the model-view controller design pattern instead of coupling the presentation logic and the business logic, we might have been able to extract a clear interface much more easily. Then the remaining work would have been to simply wrap the interface with the Web service.

Standardization. Only by using a standard and widely agreed-on vocabulary can a given service requester and provider understand each other. If the biological research community is to realize the full benefit of Web services, it will have to make more progress in standardizing data formats and ontologies. Many researchers have already taken steps toward accomplishing this, such as conducting the Minimum Information about a Microarray Experiment for microarray data[7] and developing ontologies that can apply to all life sciences and accommodate the growth and change in knowledge about gene and protein roles in cells.[8]

Standardization can also aid in creating corresponding data serializers and deserializers more systematically. Although this could take time, researchers need not wait until the community has defined every standard in detail. With Web services, they can transmit highly complicated data as attachments to SOAP messages, which can save the bandwidth taken by sending XML tags.

In addition to working on vocabularies and data formats, standardization must formalize service descriptions so that registries can assign all Web services that address the same problems to the same category. A service requester can then easily identify all the available services that can solve a problem. And it can choose to invoke different services that provide the same interface without modifying the client-side programs.

Registry standardization is also critical. The data objects and service descriptions in a registry can give software developers clues about how others have defined services. The problem for biology researchers is that, although registries such as UDDI store many services, most are unrelated to biology research. To avoid wasting time sifting through irrelevant services, biologists need registries built specifically for biology and its subfields. These registries should have a hierarchical structure, with the top-level registry mirroring the registries of other scientific fields.

Finally, help from widely coordinated organizations can be invaluable. The Web Services Interoperability Organization (www.ws-i.org), for example, provides guidance, best practices, and resources for developing Web services solutions across standards organizations. Its first release of WS-I Basic Profile, a set of nonproprietary Web services specifications, represents a milestone for Web services interoperability.

WORK IN PROGRESS

Evolving standardization will not be trivial, but the adoption of Web services technology is a solid first step because such a move can have a snowball effect: The more people are willing to provide their resources in Web services format, the more attractive this strategy becomes for others—and the more favorably users and providers will view standardization in general.

Some health agencies have already taken this step. The National Cancer Institute, for example, provides a group of legacy Web services for direct access to information and has a list of applications already wrapped into Web services (http://cabio.nci.nih.gov/soap/services/index.html). In 2003, for the DNA Data Bank of Japan (DDBJ), Hideaki Sugawara and colleagues defined DDBJ-XML and developed a DDBJ-SOAP server. Their work is Japan's earliest published effort using Web services in the life sciences.[9]

Some organizations that have already invested in an integration technology can use Web services as an enhancement. The EMBL Nucleotide Sequence Database provides an extended version of EMBL (http://www.ebi.ac.uk/xembl/) that can run as a Web service using SOAP and WSDL. XEMBL users can keep their original Corba framework.

As our case study shows, Web services have great potential for solving the data- and application-integration problems in biology, particularly for time-consuming data analysis. The wider application of this technology depends greatly on the willingness of the biological research community to pursue standardization, including building ontologies, developing biology-specific registries, and defining the service interfaces for well-known functions. The community will also need to develop more frequently used services and address the concerns of security and quality of service. Fortunately, there is a huge volume of existing applications and modules on which to base these efforts, and the successful implementation of Web services will further it.

Clearly much work lies ahead, but the efficiency payoff should be well worth the effort. In the interim, researchers who spend even a short time becoming familiar with the service descriptions will benefit. This familiarity will expedite the spread of technology, increase the number of services provided, and eventually raise the quality and quantity of available Web services. ■

References

1. M. Huerta et al., "NIH Working Definition of Bioinformatics and Computational Biology," The Biomedical Information Science and Technology Initiative Consortium (BISTIC) Definition Committee of National Institutes of Health (NIH), 17 July 2000; www.bisti.nih.gov/CompuBioDef.pdf.
2. M. Schena et al., "Quantitative Monitoring of Gene Expression Patterns with a Complementary DNA Microarray," *Science*, vol. 270, no. 5232, 1995, pp. 467-470.
3. A. Califano, G. Stolovitzky, and Y. Tu, "Analysis of Gene Expression Microarrays for Phenotype Classification," *Proc. Int'l Conf. Intelligent Systems for Molecular Biology*, vol. 8, AAAI Press, 2000, pp. 75-85.
4. D.L. Wheeler et al., "Databases Resources of the National Center for Biotechnology," *Nucleic Acids Research*, vol. 31, no. 1, 2003, pp. 28-33.
5. R.F. Smith et al., "BCM Search Launcher—An Integrated Interface to Molecular Biology Database Search and Analysis Services Available on the World Wide Web," *Genome Research*, May 1996, pp. 454-462.

6. J. Snell, "Implementing Web Services with the WSTK v3.3: Part 1," *IBM DeveloperWorks*, Dec. 2002, pp. 5-6.
7. A. Brazma et al., "Minimum Information about a Microarray Experiment (MIAME)—Toward Standards for Microarray Data," *Nature Genetics*, vol. 29, 2001, pp. 365-371.
8. M. Ashburner et al., "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, vol. 25, 2000, pp. 25-29.
9. H. Sugawara and S. Miyazaki, "Biological SOAP Servers and Web Services Provided by the Public Sequence Data Bank," *Nucleic Acids Research*, vol. 31, 2003, pp. 3836-3839.

***Hong Tina Gao** is a software engineer at Lexmark. Her research interests include bioinformatics, software maintenance, testing, software architecture, and Web engineering. She received an MS in computer science from the University of Kentucky and an MS in molecular biology from Shanghai Jiao Tong University in China. Contact her at tgao@lexmark.com.*

***Jane Huffman Hayes** is an assistant professor of computer science at the University of Kentucky. Her research interests include software verification and validation, requirements engineering, and software maintenance. Huffman Hayes received a PhD in information technology from George Mason University. Contact her at hayes@cs.uky.edu.*

***Henry Cai** is a senior application analyst at Big Lots. His research interests include software engineering, supply chain management, and e-commerce. Cai received an MS in computer science from the University of Kentucky. Contact him at hcai@hotmail.com.*