

A Comparison of Federated Databases with Web Services for the Integration of Bioinformatics Data

ZHIMING WANG¹, XIN GAO², CONGZHOU HE¹, JOHN A. MILLER¹, JESSICA C. KISSINGER^{3,4},
MARK HEIGES⁴, CRISTINA AURRECOECHEA⁴, EILEEN T. KRAEMER¹ AND CARY PENNINGTON⁴

Abstract - *Between technological breakthroughs and new computational approaches, the amount of biological data is increasing explosively. Currently, there are many data integration approaches that can provide biologists central and uniform access to all kinds of biological data. To minimize the disruption of current operations, maintain local autonomy and handle heterogeneities, Federated Databases and Web Services have been proposed as candidate solutions. The Web Services approach offers the most flexibility. However, the performance of Web Services has been a concern for many developers. This paper describes a comparison based on our experience with both approaches. It discusses the trade-offs among performance, support for heterogeneity, robustness and scalability. Of significance, is the discovery that the Web Services approach performs very competitively.*

Keywords: Bioinformatics, Biological Data Integration, Web Services and Federated Databases.

1 Introduction

Currently, the amount of biological data is increasing explosively; most of these data are stored in databases including the International Nucleotide Sequence Database Collaboration (DDBJ, EMBL and GenBank) and numerous other specialized databases, such as PROSITE, EC-ENZYME, GDB, Reactome, UniProt, PIR, DIP, Pfam, PDB. There were 968 biological databases by 2007 [5]. This tremendous diversity of biological data greatly improved biologists' ability to study the interactions between the components of a biological system, and how these interactions give rise to the function and behavior of that system. Such studies need to access multiple types of data which are likely to be stored in different, geographically distributed, databases. However, providing biologists with central, uniform access to all kinds of data is not a trivial task. It would be a poor solution, if not impossible, to make a single database to include all biological data [2]. To provide better query facilities and expedite the research process in an automatic way, data integration is essential, and is one of the most important bioinformatics research areas [8].

From our point of view, there are three main concerns about such data integration: heterogeneity, autonomy, and scalability/maintainability.

- **Heterogeneity** includes structural heterogeneity and semantic heterogeneity. Both heterogeneities are important issues in biological data integration. Many different data models including relational and object-oriented are used in biological databases. Some data are even stored in flat files. Semantic heterogeneity is more severe and is caused by a lack of a consistency in naming conventions in the life sciences.
- **Autonomy** includes design autonomy, communication autonomy, execution autonomy and association autonomy [12]. It would be neither feasible, nor desirable, to establish total centralized control of all the biological data sources existing around the world.
- **Scalability/maintainability** means the system can be easily enlarged to fit new requirements, which also aid the maintainability. Considering the rapid increase in biological data and databases, scalability/maintainability is also an important issue for biological data integration.

Based on the four approaches (Hypertext Navigation, Data Warehouses, Unmediated Multi-Database Queries and Federated Databases) outlined by Peter Karp [1] and those outlined by Lincoln Stein (Link Integration/Web Services, View Integration, Data Warehousing) [2], one could argue that there are five basic data integration approaches in use: Link integration, Query-based integration, Data Warehouses, Federated Databases and Web Services. Among them, Federated Databases and Web Services are prominent approaches because each minimizes the disruption of current operations, maintains local autonomy, and handles heterogeneities as well as scalability.

The principle intent of this paper is to compare the more traditional approach of Federated Databases with the newer approach of Web Services. We have implemented and evaluated both approaches for the ApiDB project (a member of the National Institute of Allergy and Infectious Disease's (NIAID) Bioinformatics Resource Centers (BRC)

¹ Department of Computer Science, University of Georgia, Athens, GA, USA

² Department of Genetics and Center for Bioinformatics, University of Pennsylvania, Philadelphia, PA, USA

³ Department of Genetics, and ⁴ Center for Tropical and Emerging Global Diseases, University of Georgia, Athens, GA, USA

<http://www.niaid.nih.gov/dmid/genomes/brc/>). The goal of the ApiDB project (see: <http://www.apidb.org>) is to provide the end user with uniform, integrated and centralized Web access to *apicomplexan* genome resources that currently are located in three distinct databases, CryptoDB, PlasmoDB and ToxoDB, which represent the *apicomplexan* parasites *Cryptosporidium*, *Plasmodium* and *Toxoplasma*, respectively. ApiDB and CryptoDB are hosted by the University of Georgia, while PlasmoDB and ToxoDB are hosted by the University of Pennsylvania. The ApiDB project uses two main software modules separated by functionality, a front-end module that provides uniform Web access for end users via the Web Development Kit (WDK) (see: <http://www.gusdb.org/wdk/>) and a data storage module based on the Genomics Unified Schema (GUS), Figure 1 [14].

In section 2, we consider common database integration approaches, such as data warehouses and federated databases and demonstrate why we initially chose Federated Databases for our project. Two implementations (Oracle DB Link and JDBC) of the Federated Databases approach are discussed. In section 3, the Web Services approach is discussed including flexibility and performance issues. Then, we introduce two implementations of the Web Services approach (Web Services with XML and Web Services with binary) in our project. The performance of all implementations will be compared and discussed in section 4. Section 5 discusses our conclusions and future work.

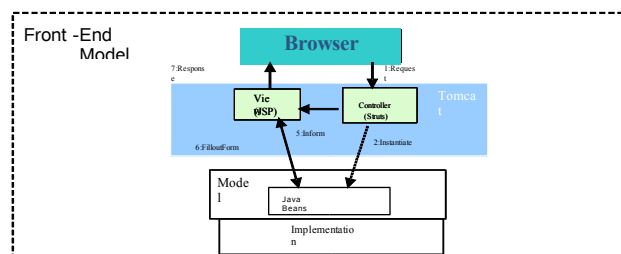
5 Federated Databases Approach

Data Warehouses and Federated Databases are the most two common approaches for database integration in the biological domain. Data Warehouses deal with data integration through conversion according to predefined rules, and then centralize all the data to one data warehouse. This system can retrieve data from any resource to the central repository as long as there is a suitable data conversion/wrapper available. Data Warehouses provide several benefits such as increasing data consistency, improving performance, supporting centralized and integrated access to data from multiple databases. One issue however, is how to keep the data in the central repository up to date, especially when the data in component sites is updated frequently. In addition, great effort is required to deal with data conversion and database management [2; 7; 9]. Since Data Warehouses require centralized data from component databases in a central database, it can cause problems such as scalability and data privacy. The Integrated Genome Database (IGD) project was a Data Warehouse approach that tried to integrate more than a dozen genomic databases (including GenBand and the Genome Database (GDB)) and experimental resources. Since each source database changed the data model twice a year on average, this meant that IGD must stop service and rewrite the data conversion program every two weeks. Such a schedule is nearly unmanageable and finally lead to the failure of IGD [2].

Compared with Data Warehouses, Federated Databases [12] reduce the problems such as keeping central data up to date, scalability and data privacy because there is no need to centralize data from component databases. A Federated Database consists of multiple component databases that are capable of sharing data through a unified interface, thus to end users, it appears as one logical database. According to Casey, Federated Databases are becoming a mainstream approach for biological database integration and there are more and more federated databases in the life sciences community, such as Comparative Mouse Genomics Centers Consortium (CMGCC), Cell Centered Database (CCDB) and iProClass [3].

The mandate for the ApiDB project [16] is to integrate multiple existing component databases, while maintaining local autonomy of each component. The Data Warehouse approach would have made maintaining local autonomy difficult. With the Federated Database approach (Figure 1), we need not copy the data from component databases as in a Data Warehouse. The federation approach provides us the most flexibility. In practice, we sometimes combine Federated Databases with Data Warehouses to boost performance through centralizing sequence data. In order to improve the performance for a single database, the WDK will cache primary key tables in the component database. This causes a performance problem for ApiDB. The original implementation for our ApiDB Federated Databases was Oracle Database Links which offers a mechanism to retrieve remote Oracle database tables for users. However, Oracle can not always generate an optimal execution plan for distributed SQL queries, the large tables in the component databases will be copied to ApiDB to join with the small primary key cache table, which can severely degrade performance. Although we can optimize the distributed query through decomposing the one complex SQL query into several smaller ones combined with an Oracle “hint”. Unfortunately, two problems remain: First, it is only applicable to Oracle Database Links which makes it too specific. Second, Oracle “hints” may be ignored when the simple queries are united together. It is a very risky task trying to change Oracle’s execution plan. Based upon the above reasons, we explored Java Database Connection (JDBC) for the implementation of Federated Databases. Our JDBC implementation first copies the primary key cache table from ApiDB to component databases, then connects from ApiDB to component databases through JDBC instead of using Oracle Database Links.

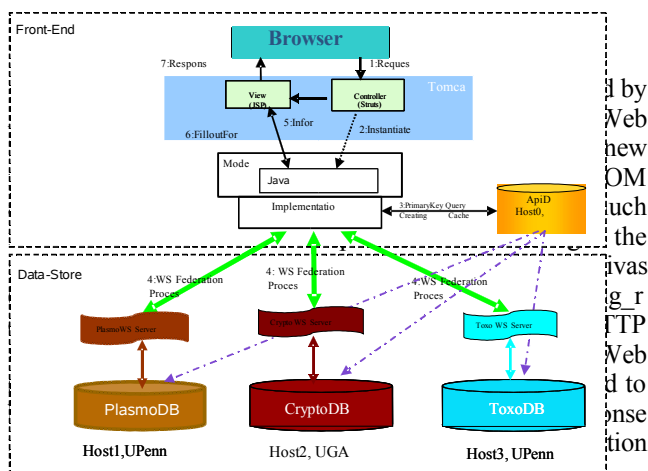
Although a Federated Database is the most flexible approach at the database integration level, it still has many limitations compared with Service-Oriented Architectures such as Web Services. For example, the Oracle Database Link implementation is too specific and the JDBC implementation needs to handle the structural heterogeneity caused by different data models or database schemas, which can be reduced by a Web Services approach.



6 Web Services Approach

Web Services (WS) technology is the most popular implementation of the Service-Oriented Architecture (SOA), which provides flexibility through decoupling the service provider from the service consumer. That is, consumers can choose any service from any provider as long as the interface is compatible, no matter what language the service is written in, or what platform the software service is running on. This decoupling brings two important benefits; abstraction and extensibility. Because of abstraction, both the client and server can evolve separately as long as they conform to the interface of the services. WS uses XML as the data exchange format. XML can facilitate solutions to heterogeneity issues, the main obstacle in data integration. For example, XML is the ideal candidate to provide a unifying data model in data integration systems to deal with structural heterogeneity². Because of the flexibility and extensibility provided by Web Services, more biological databases/projects are providing Web Services, such as the National Center for Biotechnology Information (NCBI) [11], the European Molecular Biology Laboratory (EMBL) [10], the DNA Data Bank of Japan (DDBJ) [5] and BioMoby [4].

Web Services performance is a concern for any project. One result discovered by our implementation is that the performance of Web Services is very competitive (see the performance evaluation section). The performance overhead of Web Services involves the server's hardware capacity, SOAP message size and complexity, XML parser, costs of serialization and deserialization, costs of connection establishment, security validation, UDDI registration and querying and binary XML vs. text. There are several approaches towards improving the performance of Web Services. We needed to minimize communication delays, so the compression offered by binary encoding provided the greatest impact on performance. Figure 2 represents ApiDB's Web Services architecture.



² For | Figure 2: Architecture of Web Services Approach ty in the context of Web Services, there is an emerging W3C standard called Semantic Annotations for WSDL (SAWSDL) (<http://www.W3.org/TR/sawSDL>) that we plan to use in our next implementation.

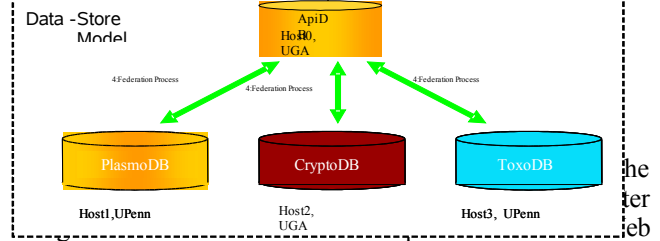


Figure 1. Architecture of Federated Databases Approach and binary attachment. For the sake of compatibility, the first implementation uses XML which is serialized from WebRowSet. For better performance, the second implementation uses CachedRowSet as a binary attachment.

7 Performance Evaluation and Discussion

There is no doubt that the system and architectural designs are paramount for the success of any software system, yet at the same time, performance is also vitally important. Table 1 lists the four different implementations that are evaluated in this section.

Table 1: Implementations evaluated

IMPLEMENTATION	TECHNOLOGY DESCRIPTION
Oracle DB Link	Use DB Link directly without decomposing the SQL query. A striped bar in Fig. 3
JDBC	Decompose SQL query in ApiDB and connect to component site directly with JDBC, setting fetch size of ResultSet to 10. A white bar in Fig. 3 & 4.
WS/XML	Decompose SQL query in ApiDB and communicate with remote sites through Web Services, where the result (WebRowSet) is encoded in XML format. A dark bar in Fig. 3 & 4.
WS/Binary	Decompose SQL query in ApiDB and communicate with remote sites through Web Services, where the result (CachedRowSet) is transferred as binary attachment. A grey bar in Fig. 3 & 4.

In order to compare the performance among the four implementations, a shell script was made to save the output and measure the query time. It calls a class in the WDK to retrieve the answer to a question defined in the XML model file and print the result out to the screen. The most time-consuming question among all of the ApiDB questions—the “Annotated Keyword” question was chosen for evaluation. To test the scalability of performance, seven different parameters (phosphoenolpyruvate, da*, de*, ed*, e*, hypo*, *) were chosen for the “Annotated Keyword” question in order to return different result sizes (18, 598, 2,334, 10,508, 17,191, 42,905, 62,443 rows, respectively). The detailed results of the timing analysis are shown in Figures 3 and 4. The only difference between these two figures is that Figure 4 removes the Oracle DB Link implementation due to its poor performance. It is so slow that the data from the other approaches is not visible in Figure 3; especially, when the number of returned results is very large. For example, when the number of returned rows is 62,443, it takes more than two hours to finish answering the question which is unacceptable. This is because Oracle can not generate optimal execution plans. The sub-optimized execution plan causes the transfer of too much data between the central database and the component databases via the network which is the main reason for the inefficiency. In Figure 4, we can

see the performance of JDBC and Web Services is much better than the Oracle DB Link implementation. For example, when the parameter is “*” (62,443 rows), the JDBC implementation is 55 times faster than the Oracle DB Link implementation and WS/Binary is 48 times faster than the Oracle DB Link implementation. The most interesting discovery is that Web Service technology is very competitive. The performance of the Web Services approach is nearly the same as that of JDBC. Part of the reason is that Web Services returns all of the results at once instead of returning multiple times like JDBC as it iteratively transfers the ResultSet. With increasing numbers of returned results, Web Services become slower than JDBC because Axis2 needs more time to prepare the result to transfer and convert back after receiving the result. WS/Binary is always faster than WS/XML because there is no need to serialize WebRowSet to XML and vice versa and the size of binary attachment is smaller than that of XML.

5 Conclusions and Future Work

With the rapid increase in biological data and databases, data integration has been and is becoming one of ever increasing need for bioinformatics research, and is also the focus of this paper. Our findings are that federated databases and Web services are the best approaches for biological data integration when maintaining local autonomy is an important factor. Compared with other common approaches, including data warehouses, link integration and query-based integration, federated databases and Web Services can minimize the disruption of current operations, maintain local autonomy, and handle heterogeneities as well as scalability, the most important concerns in data integration. We compared the pros and cons of four implementations: Oracle Database Links, JDBC, Web Services with XML and Web Services with binary attachments to determine the most advantageous approach to pursue. Furthermore, based on industrial-strength technologies, the performance of multiple implementations of the federated databases and Web service approaches based on the ApiDB project are evaluated. Considering the need for the integration of biological data, our study provides practical insights. In particular, Web Services provide the most flexibility with good performance, which helped us to make the decision to eliminate all the primary key SQL queries in our current ApiDB model by using Web Services. This will greatly facilitate the maintenance of ApiDB considering the frequency of modifications to the component WDK models. Our study also laid a solid foundation for future integration projects that could combine two or more of the individual BRC resources. Integrating BRC resources, including genomic and proteomic information, for 134 Bacteria, 550 Viruses and 12 Eukaryotes would be one of the largest projects in information/data integration in the biological community. We believe that providing centralized and uniform access to such information will greatly help biologists to obtain a deeper understanding of the fundamental biology of sets of pathogenic organisms in order to counter the threats posed by these pathogens.

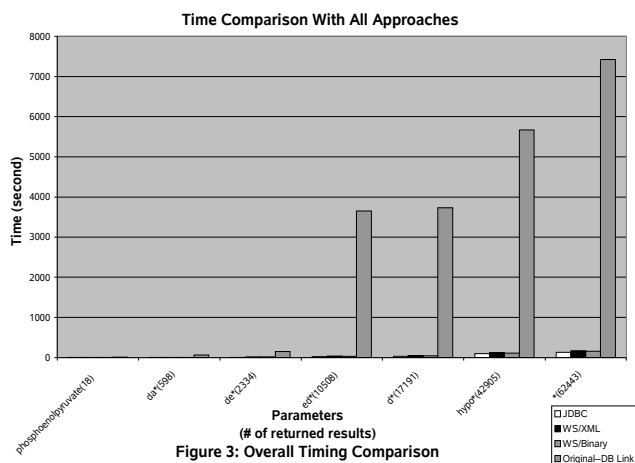


Figure 3: Overall Timing Comparison

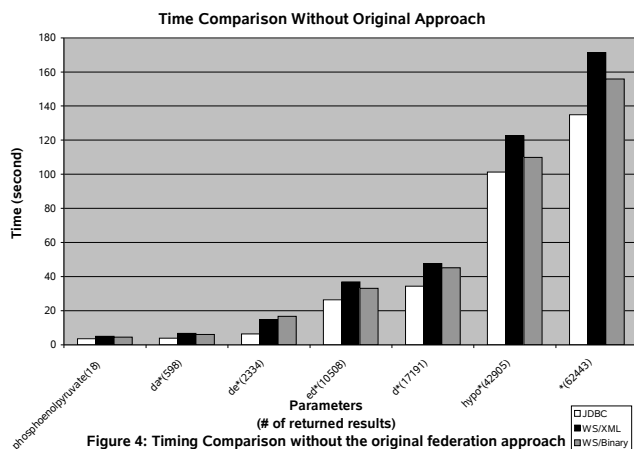


Figure 4: Timing Comparison without the original federation approach

8 Acknowledgements

We would like to thank all the ApiDB project team members for their help in this study. This project has been funded in whole or in part with Federal funds from the National Institute of Allergy and Infectious Diseases, National Institutes of Health, Department of Health and Human Services, under Contract No. HHSN266200400037C.

9 References

- [1] Karp, P. “A strategy for database interoperability”; *J. COMPUT. BIOL.*, 2, 573–586, 1995.
- [2] Stein, L. “Integrating biological databases”; *NAT. REV. GEN.* 4, 337–345, 2003.

- [3] Casey, R.M. “How Federated Databases Benefit Bioinformatics Research Business Intelligence Network: the Vision for BI and Beyond”; 2006.
- [4] Wilkinson, MD, Links, M. “BioMOBY: An open source biological web services proposal”; *BRIEFINGS IN BIOINFORMATICS* 3(4), 331-341, 2002.
- [5] Michael Y. Galperin. “The Molecular Biology Database Collection: 2007 update”; *NUCLEIC ACIDS RES.*, 35, D3–D4, 2007.
- [6] Robert, S., Robinson, A., and Goble, C. “myGrid: personalised bioinformatics on the information grid”; *BIOINFORMATICS*, 19:1302–1304, 2003.
- [7] Philippi, S. “Light-weight integration of molecular biological databases”; *BIOINFORMATICS* 20,51-57, 2004.
- [8] Stevens, R., Goble, C.A., Baker, P. and Brass, A. “A classification of tasks in bioinformatics”; *BIOINFORMATICS* 17, 180–188, 2001.
- [9] Schonbach C., Kowalski-Saunders, P., Brusica V. “Data warehousing in molecular biology”; *BRIEFINGS IN BIOINFORMATICS*, 1 (2), 190-198(9), 2000.
- [10] Wang, L., Riethoven, J.-J. and Robinson, A. “XEMBL: distributing EMBL data in XML format” *BIOINFORMATICS*, 18, 1147–1148, 2002.
- [11] Wheeler, D.L., Chappey, C., Lash, A.E., Leipe, D.D., Madden, T.L., Schuler, G.D. Tatusova, T.A. and Rapp. B.A. “Database resources of the National Center for Biotechnology Information”; *Nucleic Acids Res.*, 28, 10–14, 2000.
- [12] Sheth, A. and Larson, J. A. “Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases”; *ACM Computing Surveys*, Vol 22 (3), pp 183-236, 1990.
- [13] Pennington, C., Wang, Z., Aurrecochea, C., Gao, X., Kissinger, J., Miller, J. “Building Federated Bioinformatics Databases out of Web Services” (2007)
- [14] Davidson, S., Crabtree, J., Brunk, B.P., Schug, J., Tannen, V., Overton, G.C. and Stoeckert, C.J. “K2/Klesli and GUS: Experiments in integrated access to genomic data sources”; *IBM Systems J.* 40, 512-531, 2001.
- [15] Galperin, M. “The molecular biology database collection: 2006 update”; *NUCLEIC ACIDS RES* 34(DATABASE ISSUE),D3-D5, 2006.
- [16] Aurrecochea, C., Heiges, M., Wang, H., Wang, Z., Fischer, S., Rhodes, P., Miller, J, Kraemer, E, Stoeckert, C., Roos, D. and Kissinger, J. “ApiDB: integrated resources for the apicomplexan bioinformatics resource center”; *NUCLEIC ACIDS RESEARCH*, 2007, Vol. 35, Database issue D427-D430, 2006.
- [17] Wilkinson, M. and Links, M. “BioMOBY: An open source biological web services proposal”; *BRIEFINGS IN BIOINFORMATICS*. VOL 3. NO 4. 331–341, 2002.