Implementation Of Mediator-Based Integration System For A Transparent Access To Multiple Biological Databases

Wahidah Husain
School of Computer Sciences
Universiti Sains Malaysia
11800 Minden, Penang
604-6573645
wahidah@cs.usm.my

Muhammad Fairuz Anwar
School of Computer Sciences
Universiti Sains Malaysia
11800 Minden, Penang
6012-4366511
aoih81@yahoo.com

ABSTRACT
Existing biological databases seem to function independently from each other. Different organizations and research groups build customised databases with different formats for collecting biological data through their research. The variety of incompatible data sources unfortunately produces some wastage in terms of reusable data and workforce. Therefore, the integration of these biological databases is a practical solution. Database integration will help biologists share and utilise data. Conceptually, an ideal database integration system to be built should have a common query interface. From this user interface, biologists may query data and as the result, the system will display the records found within an integrated view. Database integration with mediator-based approach is a suitable solution despite the problems and challenges faced. Appropriately overcoming the problems that might arise during the development of a database integration system is an important factor in developing efficient and reliable database integration systems. Efficient and reliable biological data retrieval system results from the proper approach chosen to handle heterogeneity of multiple databases.

Keywords
Heterogeneous Databases, Bioinformatics, Mediator, Wrapper, Mapping Tool, Translation Tool.

1. INTRODUCTION
Presently, there are more than 700 biological databases available around the world [1]. This situation will eventually produce an enormous amount of related records and concepts which should better be shared by distant biologists from distinct research groups. Information shared between different research groups may accelerate a better understanding of a particular problem domain. Unfortunately, there are some problems. The separation between biological databases might have been caused by the ownership of the databases. A number of educational institutions, research groups, government and non-government organisations may possess and develop their own database for specific internal use. Research and development division within these organisations conduct research experiments to find a better solution particularly on life sciences, e.g. for producing a new medicine. Typically, life science databases contain data about metabolic pathways, protein structures, DNA sequences, organisms, diseases, etc.

A biological database can be a web database or a traditional database, proprietary or public. The implementation database format includes flat files, proprietary databases, public databases, data marts, spreadsheets etc [2]. Regardless of the characteristics of the databases, almost all are increasing in size. Biologists are accessing the databases for their specific research area. With new data added, the databases become denser each day. Therefore, there is a need to integrate the databases in order to share the data. By combining the databases under one efficient system, biologists are able to query and obtain the data they need. Thus, data sharing and distribution can help improve their research. Biologists should be presented a computer system that will help them work more efficiently. A database integration system will provide a collective view of diverse biological databases.

The objectives of this research are to allow unified access to biological information by providing appropriate tools for retrieving, searching and sharing data. The system should also enable users to access data location, sources and structures transparently through a single interface.

2. RELATED WORK
Many bioinformatics integration systems use virtual view and materialized view approach to integrate multiple databases [2]. Virtual view conceptualizes integration by providing transparency to the databases' original structures. A user would not be exposed to the database formats, whether a database is a relational database, an object-oriented database, a flat file database, etc. Virtual view approach refers to the integration method where access data sources are located remotely. Data queries are done remotely too. Federated database is an example of virtual view approach. The examples of biological integration system that use this approach are P/FDM [3] and EMBL CORBA server [4]. EMBL CORBA server implements the object access and relational database query via CORBA server that allows object invoking in order to retrieve sequence information, location information and feature information. P/FDM system uses CORBA wrapper to provide transparent access to multiple databases that are implemented in various platform. The main advantage of the federated approach is that data redundancy does not exist. The participating databases are maintained by their owners. One of the drawbacks of this approach is that the speed to access data relies on the network traffic. Since the query is executed at the data

source, the query response time depends on the usage access of data sources.
For materialized view approach, the filtered data from remote data sources are stored into the local database [5]. This local database is accessed by the user and data queries are based on it. Data warehouse is an example of materialized view approach. An example of biological integration system that uses the data warehouse approach is Genome Information Management System (GIMS) [6]. GIMS is a scientific data warehouse that stores the replicated genomic data and allows users to perform complex analysis against the stored data.

The main advantage of data warehouse is that it saves queries response time. Since the data are query locally, the query execution can be performing faster. This is different with the federated database approach where network traffic can influence the queries respond time. Other than that, this approach also optimizes query efficiency because of the existence of the global data model. The global data model is normalized to integrate the different databases model. However, this approach is not suitable for biological databases. The data volume is large and it continues to grow exponentially. This will increase the cost for the maintainer in terms of data storage. Moreover, the data in bioinformatics requires frequent update. Thus, the biggest issue in this approach is the data currency. Data currency is questionable when the system does not update the database frequently. It is a common situation in the bioinformatics databases where newly experimental data are always discovered. Data warehouse need to be updated if there is any data update in the data sources.

3. SYSTEM DESIGN

Based on the requirements specified above, a mediator-based integration system is implemented to provide a virtual view of all integrated heterogeneous databases. The mechanism of this architecture carries out a high level of transparency to the user. Therefore, the user can query the mediated system without knowing the detail of the data sources' actual location and structure. The main purpose of the mediated schema is to translate the global query into the execution query for each wrapper to search data from databases, reconcile different data sources' schemas and integrate them into a coherent global schema.

Mediator-based integration approach consists of two main components – the mediator and a wrapper for each individual database. The mediator is responsible for receiving the formulated query from the global mediated schema. Next, the mediator will decompose the query into sub-queries that can suit each data source's execution based on individual database. The next step is to reformulate the queries into respective execution plan based on the local models and schemas. Later, the mediator will send the reformulated queries to the respective wrappers of the concerned databases. At this stage, the wrapper is responsible for converting these queries into expected execution format depending on the data source. However, the wrapper hides technical and local implementation detail from the mediator. Lastly, the queries are executed remotely on the remote databases. The result of each query is then returned to the wrapper, mediator and finally to the user. The existing mediator-based system has some limitation. For example, when there is a network connection disruption, user may not able to use the system to access remote data. As a solution, this mediator-based system combines data warehouse approach as well as virtual view approach to the system.

As shown in Figure 1, there are six components forming the mediator-based system: presentation layer, data warehouse, mediator, wrapper, translation tool, and mapping tool. Every component plays an important role and works with each other to ensure that the system performs its tasks efficiently.

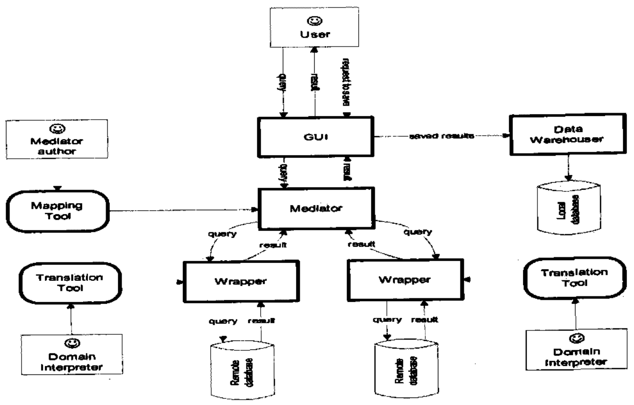


Figure 1. System Architecture.

3.1 Presentation Layer (GUI)

A dynamic and user-friendly interface is generated within the presentation layer. The presentation layer provides a mechanism for interaction between the user and the system. It is important to let users build queries as desired. Moreover, it is significant to make sure that users construct only meaningful queries. Thus, the presentation layer provides a graphical user interface to help users construct meaningful queries and avoid ambiguous ones.

In addition, the information fields contained in remote data sources might change along with time. Therefore, it is more suitable to generate a user interface dynamically instead of a static interface. This may avoid frequent manual changes to the user interface. Besides that, it can also support attachment of new data sources to the system. This dynamic user interface is generated by using the local schema and global schema. Further detail is discussed in section 4.

The user is given the choice to select only useful data to be viewed as results. To achieve this, filters are used. There are three inputs required to perform filtering:

- Restriction collection – keywords, accession number, species, author, etc.
- Target sources – the remote databases to be searched (GenBank, Swiss-Prot, etc.)
- Projection – output fields a user wants to be returned (controlled by the administrator).

3.2 Data Warehouse

One of the characteristics of the existing mediator-based system is that it does not store any data and does not require a copy of original database residing in the local storage [7]. However, it is a drawback whenever the network connection fails, disrupts or slows down.

Therefore, this mediated-based system is combined with the data warehouse approach to handle the problem mentioned earlier. During a search, when users identify any useful data, they may choose to store those data in the local database. Eventually, the data saved in the local database will form a data warehouse. Users can have better response time searching data from local database and need not suffer from keeping on waiting to fetch data when a network connection problem occurs.

3.3 Mediator

Mediator as illustrated in Figure 2 is the core components of the system. When a user constructs source independent queries at the presentation layer, it will then be passed to the mediator. The mediator will rewrite the source independent queries in presentation layer into a series of source dependent queries.

Mediator performs its task by applying the mapping concept. The mapping process maps the independent queries with the specific dependent queries by using the local schema. Mediator then has to map these source-dependent queries with the specific data source information before passing all the relevant information to the concerning wrapper. These specific data source information are obtained from data source detail schema. The data source detail schema contains the name of the data source, URL of the data source, sequence of navigation queries that lead to the result, and a set of rules to interpret the result. After completing the query transformation process, the output of this process will be passed to the wrapper.

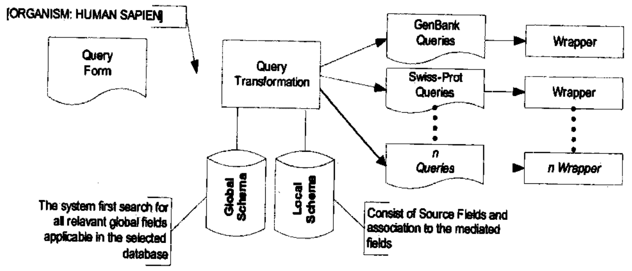

[ORGANISM: HUMAN SAPIEN]
Query Form
Query Transformation
GenBank Queries
Swiss-Prot Queries
n Queries
Wrapper
Wrapper
n Wrapper
Global Schema
Local Schema
The system first search for all relavant global fields applicable in the selected database
Consist of Source Fields and association to the mediated fields

Figure 2. Mediator Architecture.

3.4 Wrapper

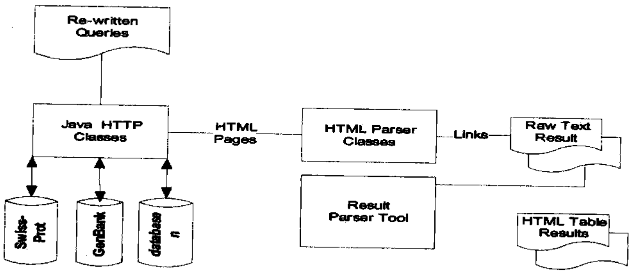


Figure 3. Wrapper Architecture.

Another core component of this mediator-based system is the wrapper. The functionality of a wrapper as shown in Figure 3 is to facilitate the physical execution of query and to handle the communication with the data source. The wrapper is responsible for fetching data and responses from a specific data source. Wrapper is used to hide the technical and data source heterogeneity from the mediator. A generic bioinformatics toolkit known as BioJava is used to remedy the tedious effort in maintaining different versions of wrappers for different data sources.

3.5 Translation Tool

The translation tool is used to maintain the local schema. The administrator uses this tool to map a remote data source's individual schema with the local schema that resides in local database. Maintenance is required purposely for the wrapper to access data from remote data sources. Prior to using this tool, the administrator is required to study the structure of the data source. For data source fields which cannot map with the existing fields, the administrator is allowed to add the field as a new field.

Without the translation tool, tedious maintenance will have to be done manually to map local fields with the global field in the local schema and to set the actual location and data source detail for the wrapper to access data source. This will eventually lead to higher probability of incorrect mappings. The functionality of the translation tool is to assist the administrator to manage the fields of a data source that the system already supports and to allow addition of a new data source for the system to support.

3.6 Mapping Tool

The mapping tool of the system maps a global field within the local schema with a field description within the global schema. The global schema is used to store the field name that is understandable by users. It also contains the content description for the specific field. The administrator utilises the tool to maintain the mediator.

4. IMPLEMENTATION AND RESULTS

4.1 Implementation Environment

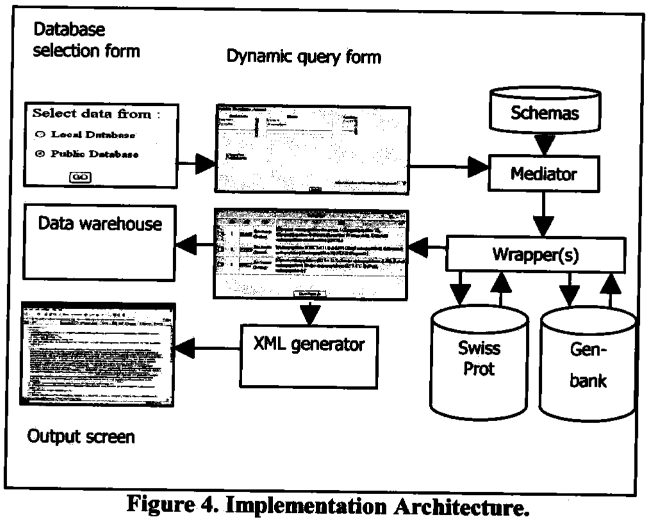


Figure 4. Implementation Architecture.

In the implementation of the mediator-based system to access biological databases, J2SE (Java 2 Platform Standard Edition) is adopted. As illustrated in Figure 4, the mediator-based system is implemented in Java Server Pages (JSP) and Java Servlet which execute on Apache Tomcat web server. It uses JSP to generate GUI forms based on the global schema and to display results fetch from local databases. On the other hand, Servlet is used to perform mediation of queries submitted by user. JSP and Servlet are chosen because they are able to perform dynamic generation of web pages, process requests and construct responses. Furthermore, Java is a programming language that supports various computer platforms and architectures.

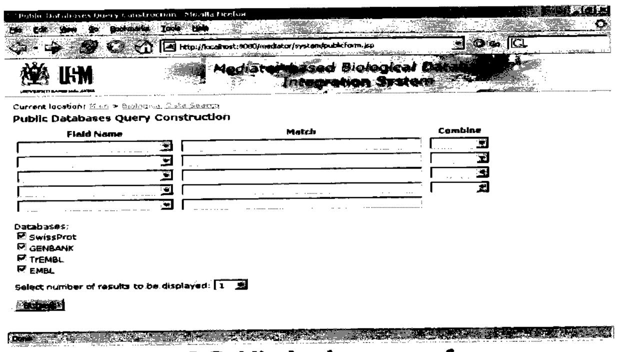
XML format is used to display a record since it is platform and system-independent i.e. may work on any computer. It also increases speed of user access to data and provides dynamic user-configurable views. Besides that, it is fully extensible i.e. it has no tag limitations. It adopts the ISO 10646 standard (also known as Unicode) which is a framework that encodes characters and supports most human languages. Thus, it does not force people to use English for coding.

Linux is chosen as the implementation platform because of its stability and reliability to perform as a web server. Although the system is set up on Linux, it is also possible to implement the system on other operating systems such as Windows and Macintosh since the technologies employed e.g. Java, Tomcat, MySQL and XML are virtually platform-independent.

The system provides an option for users to retrieve data either from local or remote databases. By default, the user interface is designed to access biological data from public data sources i.e. remote databases. A good understanding of the databases is very important in designing a dynamic GUI to support heterogeneous databases since different databases have different database structures. This system focuses primarily on protein and genome databases. Thus, Swiss-Prot and GenBank are chosen as the main data sources to experiment and implement the mediation approach. It is essential to determine the similar data fields of Swiss-Prot and GenBank that are suitable for generating an interface which may perform searches from different data sources. As mentioned earlier, similar fields are grouped as global fields for displaying standardised field names to the user. Field descriptions from the global schema are used for this purpose on the GUI. This ensures that the integrated system is totally hiding the internal searching process from the user.

Besides that, users are provided with filtering. As there are some limitations to implement a complete filtering from remote databases, only the combine options e.g. AND and OR, are provided to let users construct queries. However, for searching the data warehouse, users are allowed to use both combine options and logical operators such as =, >, <, <>, <= and >=. These options are provided to let users filter the data more specifically. The filtering process is done by the wrapper. Figure 5 shows the query form provided by the system to search from public databases.

Within the same interface, users may specify which data sources they prefer to access. The databases supported by the system are dynamically displayed on the GUI using data from the local schema. For that reason, no modification to the interface (presentation layer) is required if a new database is added to the system. If the user does not specify any database to search for, the system will automatically access all databases.



Public Databases Query Construction - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:8080/mediator/system/publicform.jsp
Go
USM
Mediator-based Biological Data Integration System

Current location: Main > Biological Data Search

Public Databases Query Construction

Field Name	Match	Combine

Databases:
- [✓] SwissProt
- [✓] GENBANK
- [✓] TrEMBL
- [✓] EMBL

Select number of results to be displayed: 1

Figure 5. Public database query form.

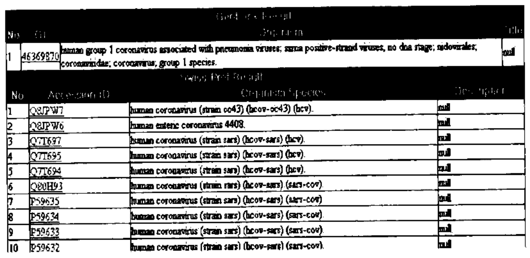
The query submitted by the user will then be passed to the mediator. By using the mapping method, the mediator later translates the declarative query (based on the global schema) into an executable query (from local schema) for individual databases.

Next, the wrapper submits the executable query to remote databases via Java's HTTP class. At this stage, the wrapper integrates with HTML Parser and BioJava tools to parse the result. The wrapper filters the parsed results according to the user's search criteria. The result will be presented as HTML tables. The user may select records from the result to be displayed in XML format and decide which records he/she wants to store locally for later customised processing.

Without a GUI, users are required to specify the database query statement manually. Therefore, with the aid of the GUI, users do not need to learn how to write the query language. Users are only required to select and submit search criteria through the GUI form. Queries will be constructed by the system automatically and are transparent to the users.

4.2 Results

Upon accessing the application, the system will prompt the user to select the database(s), either Swiss-Prot, GenBank or any listed databases. The system then generates the query form according to the database(s) selected by the user. A dynamic query form is constructed by identifying the union between the global schema and the local schemas. Figure 6 shows the result of accessing "human coronavirus" data from Genbank and Swiss-Prot databases.



GenBank Result

No	GI	Organism	Title
1	46369870	human group 1 coronavirus associated with pneumonia viruses; ssrna positive-strand viruses, no dna stage; nidovirales; coronaviridae; coronavirus; group 1 species.	null

Swiss-Prot Result

No	Accession ID	Organism Species	Description
1	Q8JPW7	human coronavirus (strain oc43) (hcov-oc43) (hcv).	null
2	Q8JPW6	human enteric coronavirus 4408.	null
3	Q7T697	human coronavirus (strain sars) (hcov-sars) (hcv).	null
4	Q7T695	human coronavirus (strain sars) (hcov-sars) (hcv).	null
5	Q7T694	human coronavirus (strain sars) (hcov-sars) (hcv).	null
6	Q80H93	human coronavirus (strain sars) (hcov-sars) (sars-cov).	null
7	P59635	human coronavirus (strain sars) (hcov-sars) (sars-cov)	null
8	P59634	human coronavirus (strain sars) (hcov-sars) (sars-cov).	null
9	P59633	human coronavirus (strain sars) (hcov-sars) (sars-cov).	null
10	P59632	human coronavirus (strain sars) (hcov-sars) (sars-cov).	null

Figure 6. Accessing data from GenBank and Swiss-Prot.

In terms of achieving up-to-date result, mediator-based system provides the most up-to-date query result. This is because the system querying the data source directly and the owner of the data source do the updates autonomously. However, update delay is a common problem in data warehouse because the update propagation from data source does not occur instantly to the local data replica of the data warehouse.

An evaluation has been carried out to examine the accuracy of the process to retrieve data from remote databases. The data accuracy increase when the variables in a query increase as shown in Figure 7. This is significant because the number of results retrieved decrease as more search criteria exist in a query. The accuracy rate is high in the mediator-base system because the search results return by a data source is being parsed and filtered in the wrapper before it is returned to the presentation layer.

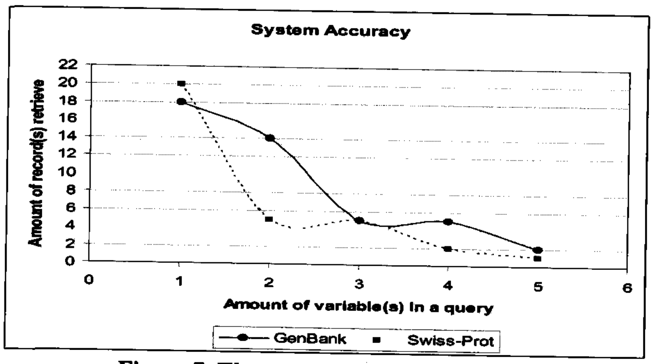


Figure 7. The amount of data retrieved.

5. CONCLUSION AND FUTURE WORK

There are hundreds of biological databases available today with different formats and sizes. Yet, most of them utilise the same technology i.e. flat files. As the number and size increase, there is a need to integrate the databases to help biologists search for data more efficiently. The mediator approach is chosen for the integration purpose as it eases maintenance and supports scalability. New databases may be added and existing ones may be modified or removed more easily.

A web-based implementation is a better solution since Internet is widely used nowadays. This allows users to access the system anywhere. A user is required to decide which databases to access and provide some keywords to perform a search. However, the inner workings of the system are transparent to the user. Although the system is currently working, there is still room for improvements. Below is a list of proposed future enhancements.

5.1 Increased Wrapper Reusability

Currently, wrapper reusability is limited due to the bounds of BioJava's capabilities. Although BioJava provides standard APIs for performing various bioinformatics-related functions, it does not allow the system developer to specify which fields to be searched in the database. For this reason, a wrapper is required to be customised with a data filter module to filter the results returned. Even though this increases the accuracy of query results, it reduces the reusability of a wrapper. It would be more favourable to develop a generic wrapper that is suitable for all biological data sources.

5.2 Intelligent Database Updater

The implementation of this mediator-based system provides users the ability to store useful search results into the local databases. Due to the implementation approach employed, data updates in the remote data sources will not occur automatically in the local databases. Synchronisation issue arises when remote data sources are updated. It is more appropriate if the local databases keep up-to-date remote data. This may be handled by an intelligent database updater. The intelligent database updater should have an agent that scans the data in the local database periodically. Comparisons will be done between local databases and remote data sources. If the agent identifies any differences, the intelligent database updater would update the local data accordingly.

6. ACKNOWLEDGMENTS

This work is supported by Universiti Sains Malaysia as a short term grant project named: Development of a Web-based System for Integrated Access to Heterogeneous Biological Databases.

7. REFERENCES

[1] Galperin MY. The molecular biology database collection: 2005 update. Nucleic Res 2005:33:D4-D25.

[2] Peter D. Karp. A Strategy for Database Interoperation, J Comput Biol 2(4): 573-86 (1995).

[3] Graham J.L. Kemp, Chris J. Robertson, Peter M.D. Gray, Nicos Angelopoulos. CORBA and XML:Design Choices for Database Federations, 17th British National Conference on Databases, Exeter. (2000).

[4] Lichun Wang, Patricia Rodriguez-Tome, Nicole Redaschi, Phil McNeil, Alan Robinson, Philip Lijnzaad. Accessing and distributing EMBL data using CORBA (Common Object Request Broker Architecture), Genome Biology 2000, 1(5):research0010.1–0010.10. (2000).

[5] Lincoln D. Stein. Integrating Biological Databases. Nature Reviews, Genetic, Vol 4, 337-345. (2003).

[6] Mike Cornell, Norman W. Paton, Shengli Wu, Carole A. Goble, Crispin J. Miller, Paul Kirby. GIMS – A Data Warehouse for Storage and Analysis of Genome Sequence and Functional Data, 2nd IEEE International Symposium on Bioinformatics and Bioengineering, Maryland. (2001). W.-K. Chen, Linear Networks and Systems (Book style).Belmont, CA: Wadsworth, 1993, pp. 123–135.

[7] The Mediagrid Project, A Mediation Framework for a Transparent Access to Biological Data Sources, In Proceedings of the ECCB 2003 Conference Poster Session, Paris (2003).