Journal of
Information Technology & Software Engineering

Malick et al., J Inform Tech Softw Eng 2013, 3:1
DOI: 10.4172/2165-7866.1000115

Case Report Open Access

A Generalized Framework for Biological Data Integration, Processing and Visualization

Rauf Ahmed Shams Malick[1]*, Muhammad Ussama[2] and M. Kamran Azim[3]

[1]Department of Computer Science, FAST NU, Shah Latif Town, Karachi, Pakistan
[2]Center of Excellence in Information Assurance (CoEIA), College of Computer and Information Sciences, Riyadh, Kingdom of Saudi Arabia
[3]International Center for Chemical and Biological Sciences (H.E.J. Research Institute of Chemistry; Dr. Panjwani Center for Molecular Medicine and Drug Research) University of Karachi, Karachi 75270, Pakistan

Abstract

Massive amount of present biological datasets and their current rate of expansion demand effective ways of data storage, processing and presentation. The ever increasing nature of biological data sets has tremendously increased the complexity in data integration and presentation processes. This paper presents an experience of a unified framework development which combines complex input module with massive processing methods in a collaborative visualization environment. Our experience of the development of IDGAAM framework is presented in this paper along with functional details of AINAAN application. AINAAN is being used for collaborative environment for data sharing and more specifically for visualization purpose. The present study is an effort towards seamless integration of heterogeneous data bases with a complexity of analysis and varieties in its presentation formats.

Keywords: Computer science; Information systems; Biological databases; Integrated system; Biological framework; AINAAN; IDGAAM

Introduction

The volume of biological data sets is increasing exponentially since the last decade. Not only, the volume is increasing, also the new types of data sets are being added on almost daily basis. This extra ordinary increment in biological data sets with numerous formats and data types has tremendously increased the complexity of information retrieval mechanism significantly for the biological scientists.

Integration of multiple biological databases provides seamless access to all of the databases which simplifies the complex biological analysis. The complexity of data retrieval from multiple sources increases when the heterogeneity, diversity and dispersion of underlying data increase. The volume of data is increasing with the diversity in the content type exponentially in biological context. For example in Biological databases, if a Bacterial or Archae genome is under observation it's an elementary step to do Sequence Alignment of genomes to outline potential 'gene' regions. Finally the gene expression level may lead to the evolution and reformation of a 'gene regulatory network'. In this case, information extraction from databases about potential genes with their expression levels and possible gene regulatory network indeed becomes a unit task in a pipeline. The integration of multiple information sources, for a unified access with effective views and processing capabilities is certainly an important issue for the biological community.

It has become a big challenge for the computational biology community to provide some easy to use and powerful tool for data integration, visualization and processing capabilities. In this paper a generalized framework is discussed which offers solution regarding data presentation, user interface, processing capabilities and issues regarding multiple database formats. The discussion on proposed framework is accompanied by our experience on implementing tools at different layers including presentation, data extraction, user interface and processing. AINAAN is discussed as a collaborative tool for network data sharing and more specifically as a visualization tool on presentation layer; IDGAAM is discussed under database integration tool with a query language for complex data processing with functional capabilities. In the following section a brief description is presented about the database integration efforts in modern biological database context. Section 2 discusses the motivation of the generalized framework and basic features with the core of the system. Following to the section 2 AINAAN, IDGAAM and query language modules are discussed with their features and architectural details. This work extended the idea of an integrated environment for biological community which could not be achieved by only integrating the biological databases itself without incorporate powerful toolset for information retrieval and presentation.

Generalized Framework for Biological Data Integration

This study is aimed at a scalable architecture for biological data management, processing and visualization. This architecture emphasizes over adequate updation at the presentation layer and processing layer when a new data model is added with a diverse content type. A unique feature of this architecture is its capability to update the query language (for user interaction) in coherence with an extension in views, processing and data resources.

In order to provide seamless access to multiple databases the development of federation of databases is not new in biological community. Among the federation of databases, the use of a mediator with data wrappers has been experienced by several studies like schema based integration for federation of databases is presented in [1]. Mediator with wrapper architecture gives global repository of the local schema. Then it becomes easier to evaluate a query for underlying heterogeneous data sources. Common data model representation is carried out by the wrappers and remains coherent with the supported views TAMBIS [2], and BioKleisli [3], Davidson et al. [4] are some of the mediator with wrapper architecture based systems. Some of the biological databases integration efforts are [5-9].

Mediator based architectures are discussed more often in three

***Corresponding author:** Rauf Ahmed Shams Malick, Department of Computer Science, FAST NU, Shah Latif Town, Karachi, Pakistan, E-mail: raufmalick@yahoo.com

Received December 06, 2012; **Accepted** December 28, 2012; **Published** December 31, 2012

Citation: Shams Malick RA, Ussama M, Azim MK (2013) A Generalized Framework for Biological Data Integration, Processing and Visualization. J Inform Tech Softw Eng 3:115. doi:10.4172/2165-7866.1000115

layers corresponding to presentation layer, processing layer and data wrappers. The real problem occurs when new types of views become the requirement of the existing system especially when a new database, with some specific new data model, is added. An update in the current function repository with an extension in existing view capabilities for effective analysis and visualization is indispensable. In a federation of biological databases, if a new data model is integrated, like protein-protein interaction data DIP or BIND [10] databases, with the existing one, certainly the view and processing capabilities demand an update. Apart from several benefits of structured based retrieval systems for federation of databases, there exist many issues inherent to them including lack of extendibility in view and data retrieval methods as biological data sets are continuously growing and expanding in multiple dimensions. On the other hand ontology based retrieval systems interprets the user requirement based on provided semantic information which might not desired on many occasions by the target users [11,12]. The major reason of avoiding ontology based retrieval or natural language processing based retrieval is their limit of effective usage. Such systems could be very useful when user have vague idea about what has to be searched or what particular data has to be retrieved. Our motivation for the development of yet another query language is for 'well aware' users, those already know what data has to be retrieved from which database. Through this framework and following implementation intermediate processing will be reduced and results will be provided in an integrated environment with network based access features.

The presented framework consists on certain important components including query processing, global schema repository, visualization, function repository and mediator. The interaction between these components is accomplished by 'messages' which makes it a flexible and loosely coupled architecture. A user interface for a 'query interface' is discussed separately from the presentation layer which is dedicated for view support. Queries and outputs of queries all pass in the form of XML documents.

The proposed architecture is based on three layers with four components for mediator based integration system. Presentation layer for visualization, processing layer encapsulates functions (to be performed over data) and data extraction layer (or wrappers).

Each layer consists on a respective component whereas the processing layer interacts with the query component. The key characteristics of the architecture are as following:

The user interface is separated into two separate components. The user interface for visualization is placed over the presentation layer, whereas, the user interface for input is placed at the processing layer within the query component. The separation of two interfaces is due to the difference of concerns. The user interface for visualization is dedicated for the presentation of views of the underlying data. This framework proposes that a query language should be used for data extraction in place of fixed parameter based forms. The same query language should have the ability of parametric view representations for complex visualization.

In figure 1 it is shown clearly that the UI for visualization is separate from the UI for querying. The detailed features of the query language will be discussed in the similar section.

A global repository of local schemas has to be maintained. This global repository of global schema is a part of query component. The local schema is used to evaluate query semantics, query execution plans, optimization methods and finally the cost of the query. All of these schemas in the global repository represent local data model and are tightly coupled.

Query processing through query component

The query component encapsulates all of the phases regarding query evaluation till query execution. As shown in figure 1 the query component interacts with the global query repository to evaluate the semantics of the query. After validity of the syntax of a query, query decomposition takes place and if possible query decomposition phase leads to the generation of the query execution order.

The architecture supports the query language not only to extract data from underlying database but rather it should be the way of incorporating analysis tools with the user interface for visualization. The user should have the power and feel easy using the query language. The suggested query language comprises SQL type syntax for multi databases, with an extension of using data analysis methods within it.

Data extraction layer with wrapper

The wrapper component is in fact the file execution. The wrappers are responsible to extract data from the respective file according to its data mode. In the mean time the wrapper component understands the common query and generates the respective output in the form of XML document. It makes the design flexible while placing data source over multiple workstations.

Dynamic integration issues

The dynamic integration issues arise in many ways for e.g. in terms of new views, new data models, extension in query language, updation of query execution mechanism and addition in current repository for analysis tools. The dynamic integration of each module causes separate issues of component interactions:

The capabilities of updating the current view implies that either the query language is capable to visualize the new view type or that an extension is necessary in query language. Like, in biological databases new query language features have been proposed while dealing with databases that contain biological networks related data. Capabilities and current view support have to be upgraded to coup up with the current changes.

Novel methods of interpreting biological data and novel tools for learning and inferring useful results encapsulated in the form of analysis tools. If an update in the analysis tools is committed then there may be a need of updating in the query and view capabilities as well.

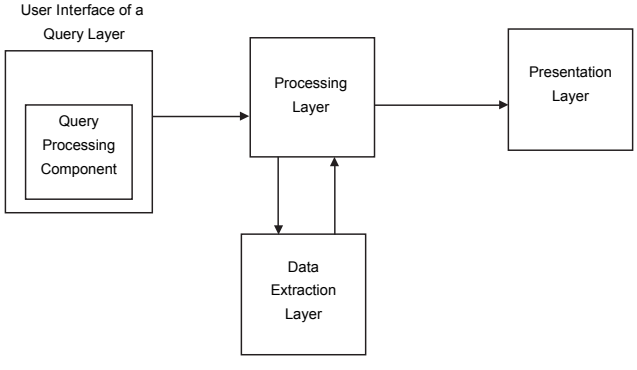

Figure 1: Architecture of the integrated framework.

Presentation layer

Biological databases contain avalanches of data in different file formats with different data models. The Protein/DNA related data visualization and extraction from multiple database demands huge amount of scripting and tedious jobs to be performed. Certainly, new visualization applications are needed to integrate with existing front ends with the addition of new databases. Plenty of application environments exist to represent 3D structures for visualization. Rasmol [13] is one of the earliest tools for protein visualization that supports different views of proteins, FPV [14] is designed and implemented for protein visualization with fast scene graph based architecture. MOLVIE [15], WebMol [16], Tripos Java3D Molecule Viewer, VMD [17], JMV (Java Molecular Viewer) are some of the frequently used available protein visualization tools. Most of the visualization applications are developed over the Java3D platform.

Addition of new visualization methods always required in integrated database management systems. In order to keep supporting new biological views, visualization is encapsulated into presentation layer within the proposed framework. The presentation visualization layer allows addition of new views dynamically, in the mean time, it requires a smooth communication with the query interface and the processing layer as well. In context of integrated biological databases, complex 'analysis' methods perform their job under the processing layer (Figure 2).

In addition, in the context of an integrated biological database environment, it is not an easy task to decouple the user interface for the input and output of data and results respectively.

Initially AINAAN [18] application supports rich graphic user interfaces for protein 3D structure representations with Wire frame, Sticks, Ball-sticks, Space Fill, Backbone, Ribbon, Strands and Cartoon type views (Figure 3). Other supported features of AINAAN are the recording of events (movie), maintenance of network repository of proteins sequences, structures, aligned results, and movies. Collaborative sessions could be managed for the combined study of a protein with a set of other proteins.

AINAAN: A Collaborative Environment for Protein Visualization and Analysis

AINAAN is updated to become the presentation layer component i.e. user interface for visualization in IDGAAM framework.

IDGAAM framework allows integration of new methods in the processing layer, new view methods at the presentation layer (user interface for visualization), and extensions in the existing query language (user interface for query) in order to manage effective data retrieval while adding new databases in the system.

Important AINAAN features

AINAAN application provides protein visualization, analysis interfaces, collaborative sessions, network based storage mechanism and a highly interactive user interface. Following are some important features and components of AINAAN:

Rich graphic user interface: AINAAN is supports rich graphic user interfaces and in fact is an integrated environment. Users may visualize multiple proteins at a time in multiple windows. Multiple collaborative sessions may be run by the user in parallel in multiple windows. Zoom and rendering may help the user to study big and complex molecule.

Data sharing through network repositories: In general collaborative applications offer only the sharing of views, however AINAAN provides collaborative analysis in parallel of protein views at the network level. An individual may maintain network storage with the name of 'Network Repository'. Network Repositories may contain separate folders for, structural alignment results, sequence alignment results, PDB structures, and collaborative session movies. A user can remotely access permitted data after establishment of collaborative session upon granted rights from the data owner.

Underlying distributed architecture: Collaborative applications like MICE [19,20] and other applications which offer session sharing are client-server in nature. Client-server architectures could be helpful when the numbers of sessions are limited and participating users are large in number. On the other hand, when participants are interested in running their own collaborative sessions, thereby increasing the total number of sessions on the network will significantly cause a decrement in system performance.

The design and architecture of AINAAN is purely distributed in nature. Any system may act as a server in a session (while running and owning the session) and may participate as a client at the same time. The server machine will be responsible to delegate the views to all the participants of the session.

Rapid data access: Delegation of responsibilities results in the distribution of load over the network which may cause bottlenecks or congestion over the network. Session initiator bares the load of the session as an owner. The AINAAN architecture allows network

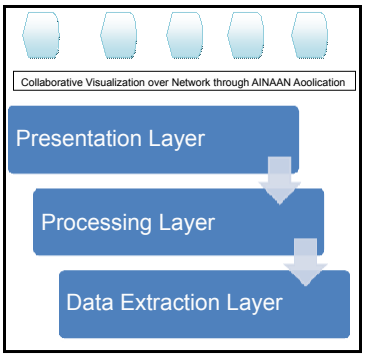

Figure 2: Higher level presentation of proposed framework.

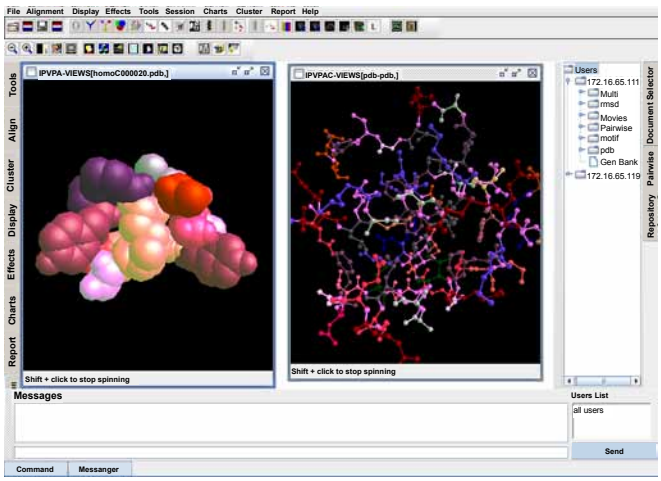

Figure 3: Snapshot of an instance of AINAAN application.

repositories to host a collaborative session in a way that all the participants may keep the molecular data to themselves. In contrast to the centralized server approach, the data is distributed over the network which minimizes the latency time of each information access. In client server architecture, the server is responsible for hosting all the data. This results in a significant delay when large numbers of requests have to be entertained by a single server and specially about data sharing.

Modifiability and programmability: AINAAN applications support various paradigms of collaborative sessions. With the growing rate of technology development and the rapid increment in biological datasets complex data analysis requirement also increased indeed. An AINAAN application is designed with consideration of extensibility in particular. Addition in the existing protein view capabilities or extensions in the analysis methods will not cause any integration problem at run time. The usage of the Command Pattern [21] made it possible to shape AINAAN as a flexible architecture, with the help of 'event' broadcasting during a session, makes the application even more flexible.

Event recoding: The user has freedom to record their own activity or a collaborative session of activities. Each performed action like protein view change or usage of any analysis methods, rotations, will be recorded in our own file format 'USWA'. This recorded file can be revisited later in the form of a movie. Network repositories support 'USWA' files over network as well.

Application development environment

AINAAN is built on the Java platform by using JDK 1.5 with the Open GL library for graphical representation of protein views, which also has a variety of implementations over numerous platforms.

IDGAAM: An Implementation of the Proposed Framework for Biological Databases

In this section architecture of the IDGAAM framework is discussed. IDGAAM is based on the mediator architecture which is the most flexible design for data integration. It decouples the processing layer with client layer and underlying heterogeneous data sources. The IDGAAM framework discusses the integration problem in biological databases under these layers. The presentation layer concerns about the visualization of complex biological data, the processing layer incorporates functions for data manipulation and data extraction layer over multiple biological data sources.

The three layers actually are the wrappers for the underlying complex mechanism at different levels. Fig. 4 presents a high level conceptual organization of the IDGAAM framework.

Presentation layer

Presentation layer in the IDGAAM framework abstracts the mechanism of a user interface.

User interface for visualization: AINAAN application is designed and implemented for collaborative visualization of proteins and complex analysis results, this is to address first issue. Initially AINAN was designed to visualize 3D views of protein structures on the desktop and collaborative environments. Now AINAN is updated to a scalable front end application. New views can be incorporated in the existing visualization capabilities without affecting the integrity of the system. Abstract factory pattern is used in AINAN to add new views and dynamic visualizations. AINAN is presented in IDGAAM framework to be used as a visualization interface over the presentation layer.

User interface for data extraction from multi database: Fixed forms based user interfaces or menu driven interfaces are not the first choice in biological data extraction. It is not possible to use some common form based user interface to extract data from multiple sources when the sources contain different data models. In addition to this, the extracted data from heterogeneous databases has to be redirected to respective processing tools to convert raw data to useful information.

A new query language is introduced in IDGAAM framework which is capable to extract data from multiple underlying databases.

The structure of this new query language is close to SQL. Some examples of this query language are shown in later section. The application keeps the query in an XML document and during the query execution, the XML document is sent to the host workstation. The results of the queries are then saved in the same XML document. The basic architecture of the application is presented in figure 4. The framework comprises a parser, global schema repository, path evaluation, query optimization, network topology and result fuser components.

Adding new database in the existing system

Database integration according to IDGAAM framework is a 3 fold process. Adding a schema of the new database, updating the graph and writing the executer for the database are the requirements of database integration. Following are the steps to integrating a database:

Step 1: Managing the global schema: The parser component of the application checks the syntax and parameters within the schema component. In the integration of a database, first its file format should be added in the schema component. Once the format of a database is added into the schema repository, the query parser will be able to validate the entities of new database.

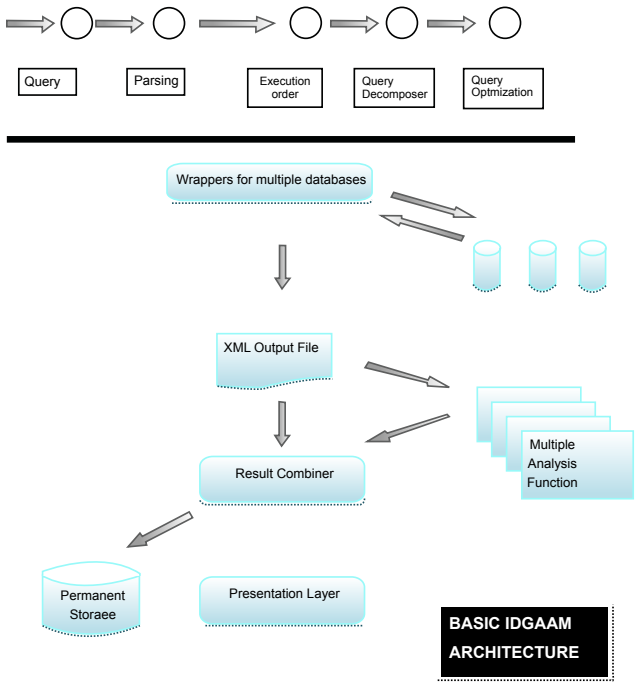

Figure 4: Basic architecture of IDGAAM.

Step 2: Updating the directed link graph: The query execution order and discovering paths between databases will traverse a graph consisting of database cross references. The notation for the graph is *G(V,E)* where the vertices (V) is the set of all databases and edges (E) is the set of cross references within these databases. Therefore, the references present in the new database should be entered into the graph (G) component (for Path Evaluation). It helps in the navigation from one database to another. When the database graph (G) is updated, the system will be able to recognize the new database, now the path evaluation phase will consider the new database for query execution sequence generation.

Step 3: Providing Local File Executer: The federation of biological databases is implemented in a distributed environment. The requirement of the framework is that the executer of the database should be co-located where the database is stored. Finally, the query starts execution, the execution plan consists of what to search & which database to access. The query execution plan is in the form of an XML document, the document will transfer from one database host machine to other database host machines. The implementation of IDGAAM uses a tree like interconnection of PC machines. Each node (computer) in the tree is a single database. The tree is implemented by using Java SDK 1.5 with the help of the TCP sockets.

The result of each query is written in the same XML query file, so when this XML file returns to the user (query imitator) the query interface displays the results. Currently extracted results from Uniprot and PDB databases can be visualized at the user interface. The XML document (the output file) can be saved at any desired location. The user interface is an open architecture and open source application; therefore, one can enhance visualization capabilities or add multiple result viewers for other databases.

The Query Language

The problem faced by the users in life sciences domain is the unawareness of conceptual models, database schema, and file formats of the large number of databases. To access multiple biological databases in order to perform some sequential tasks to gather related data, the query language is a preferred choice by [22,23] whereas [24] and [25] are some query based biological database integration systems). Most of the query languages proposed or implemented for biological systems are either for web based information portals or based on some special kinds like OSQL in Bio-Kelisi/K2. SQL based query language proposed by [26], suggested functions within a query for further analysis on query results.

The query language designed for the application framework is scalable. In this application, the query can be asked by using the name of the database and its required field. The query language supports 'function calls' providing arguments by redirecting results from nested queries. New functions can be added in the query model as the query language is open for extension. The results of queries can be redirected to some destination file by using the insert keyword. The generated output is in the form of XML document, which can be further used for more results. Representation of a simple query in XML format is shown in figure 5.

The query model supports nested queries as well, the inner query executes first and its output becomes input of the outer query. Its significance is evident if accessing databases in sequence is needed.

>Select <database name.database entity>* where database. entity==""";

```
<?xml version="1.0" ? >
<Query>
- <Required>
  - <subquery src =" DQServer"  Destination=" pdb" condition=" true" >
    - <select>
        - <atom_siteCategory type =" Parameter" >structure</atom_siteCategory>
      </select>
    - </where>
        <dbReference type ="pdb ">1A37,1A38,1A40</dbReference>
      </where>
    </subquery>
  </Required>
</Query>
```

Figure 5: Simple query in XML form.

>Select <database name.database entity>* where database.entry == (Select <database name.database entity>* where database.entity==""")*;

The following query (Query no. 1) visualizes a protein structure, where its 3D information is being extracted from PDB databases whereas Uniprot ID is provided in the query.

1> Select Molecule Visualization (pdb.structure) from pdb where uniprot_id ="1433Z_BOVIN";

Query no. 2 is performing pair wise protein sequence global alignment where one sequence is provided in the function argument and the other sequence has to be retrieved from the Uniprot database.

2>Select Global Alignment ("MASPREENVYMAKLAEQAERY EEMVEFMEKVVAAADGAEELTVEERNLLS", uniprot.seq) from uniprot where uniprot.id ="1433Z_BOVIN";

The third query (Query no. 3) is saving all protein sequences from Uniprot database into a file where their exist a respective PDB structures.

3> Insert into user1 (Select uniprot.seq, uniprot.id from uniprot where uniprot.type = "pdb");

Conclusion and Future Work

The paper presented a framework for dynamic database integration, which can be a need of BioMetric, GIS and Bioinformatics domains. The framework comprises over three layers including presentation, processing and data extraction. Each of these layers encapsulates separate components with a query component. Presentation layer corresponds to visualization issues of complex structures, processing layers concerned with analysis tools over data extracted from heterogeneous databases by the data extraction layer. Effectiveness of a query language is discussed for complex visualization and usage of processing tools. Moreover, the same query language gives interfaces for extensions whenever any of the new views, functions or data models are added dynamically. An application of this framework is also discussed for federation of biological databases. IDGAAM application comprises over the front end which is based on AINAN, repository of analysis tools like sequence and structural alignments and data wrappers over PDB and Uniprot databases with a implementation of a new query language.

In the future, spatial and multimedia databases are aimed to be integrated with the existing system. Spatial and multimedia databases within the biological community are in their infancy. In fact we are more interested to integrate new databases with new content types which will require new ways to query, analyze and visualization.

Consequently, new methods will be needed to analyze underlying data with possible extensions in the existing query and view capabilities.

An important aspect of Biometrics, GIS, and Bioinformatics domains is the rapid evolution of new databases based on parent data extracted from heterogeneous databases.

We are more interested to devise some technique to maintain stages of 'database versions' like software versions. The novelty of idea will come up with some methodology and concrete implementation experience about managing intermediate stages of a database from its evolutionary period to maturity. The idea about 'database versions' or 'intermediate stages' for a database may restricts the rapid growth of databases in the above mentioned field in the future.

References

1. Kemp G, Angelopoulos N, Gray P (2000) A schema-based approach to building a bioinformatics database federation. Proceeding of the IEEE International Symposium on Bioinformatics and Biomedical Engineering, USA.
2. Paton NW, Stevens R, Baker PG, Globe CG, Bechofer S, et al. (1999) Query processing in the Tambis bioinformatics source integration system. Proc of the IEEE Intl Conf on Scientific and Statistical Databases(SSDBM).
3. Davidson SB, Overton C, Tannen V, Wong L (1997) BioKliesli: A digital library for biomedical researchers. International Journal of Digital Libraries 1: 36-53.
4. Davidson S, Crabtree J, Brunk B, Schug J, Tannen V, et al. (2001) K2/Kleiskli and GUS: Experiments in Integrated Access to Genomic Data Sources. IBM Systems Journal 40: 512-531.
5. Rimon G, Orengo C, Martin N, Poulovassilis A (2004) BioMap: gene family based integration of heterogeneous biological databases using AutoMed metadata. Proceedings of Database and Expert System Application.
6. Radrich K, Tsurokoa Y, Dobson P, Gevorgyan A, Swainston N, et al. (2010) Integration of metabolic databases for the reconstruction of genome-scale metabolic networks. BMC System Biology 4: 1-114.
7. Wang Y, Juan L, Teng M, Cheng L, Qiu S (2011) Bioinformatics Database Integration Based on Biomedical Ontology. International Journal of Advancements in Computing Technology 3: 2-66.
8. Hanifa GM (2006) GeneONE Portal Framework for integrating Genome Databases, tools, techniques and methods. ICBPE.
9. Markowitz MV, Ivanova NN, Szeto E, Palaniappan K, Chu K, et al. (2008) IMG/M: a data management and analysis system for metagenomes. Nucleic Acis Res 36: D534-D538.
10. Bader G, Betel D, Hogue WVC (2003) BIND: the Biomolecular Interaction Network Database. Nuclic Acid Res 31:242-245.
11. Muller H, Kenny EE, Sternberg PW (2004) Textpresso: An Ontology-Based Information Retrieval and Extraction System for Biological Literature. Plos Biol 2: e309.
12. Francois M, Ranwez S, Montmain J, Regnault A, Crampes M, et al. (2012) User centered and ontology based information retrieval system for life sciences. BMC Bioinformatics 13: S4.
13. Sayle R., Milner-White EJ (1995) RasMol: Biomolecular graphics for all. Trends Biochem Sci 20: 374.
14. Can T, Wang Y, Wang YF, SU J (2003) FPV: Fast Protein Visualization Using Java 3D. Bioinformatics 19: 913-922.
15. Sun H, Li M, Xu Y (2002) MOLVIE: An interactive visualization environment for molecular structures. Comput Methods Programs Biomed 71: 85-90.
16. Walther D (1997) WebMol-A Java based PDB viewer. Trends Biochem Sci 22: 274-275.
17. Humphrey W, Dalke A, Schulten K (1996) VMD-Visual Molecular Dynamics. J Mol Graph 14: 33-38.
18. Malick R, Khanzada U, Abbas A, Iqbal W, Azim K (2008) AINAN: Collaborative Application for Protein Visualization & Analyses. International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/ Distributed Computing.
19. Bourne P, Gribskov M, Johnson G, Moreland J, Wavra S, et al. (1998) A Prototype Molecular Interactive Collaborative Environment (MICE). Pac Symp Biocomput 118-129.
20. Simard J (2012) Collaboration haptique étroitement couplée pour la déformation moléculaire interactive. PhD Thesis 13: 25.
21. Jhonson R, Gamma E, Vlissides J, Helm R (1994) Design Patterns Elements of Reusable Object-Oriented Software. Addison Wesley.
22. Eckman B, Deutsch K, Janer M, Lacroix Z, Raschid L (2003) A Query Language to Support Scientific Discovery. IEEE Computer Society Bioinformatics Conference.
23. Jake YC, John VC, Ning G (2005) A Complex Biological Database Querying Method. Proceedings of the 2005 ACM symposium on Applied computing, USA.
24. Miled BZ, Li N, Baumgartner M, Liu Y (2003) A Decentralized Approach to the Integration of Life Science Web Databases. Informatics 27: 3-14.
25. Hernandez T, Kambhampati S (2004) Integration of Biological Sources: Current Systems and Challenges Ahead. ACM SIGMOD Record 33: 51-60.
26. Oinn T, Greenwood M, Addis M., Alpdemir N, Ferris J, et al. (2000) Taverna: Lessons in creating a workflow environment for the life sciences: Research articles. Concurrency Computat: Pract. Exper 18:1067-1100.