16 Oct 2006

A P2P Integration Architecture for Protein Resources

K. T. Claypool

Sanjay Kumar Madria

Missouri University of Science and Technology, madrias@mst.edu

A P2P Integration Architecture for Protein Resources

Kajal T. Claypool[†] and Sanjay Madria[‡]

(†)Department of Computer Science
University of Massachusetts
Lowell, MA 01854
kajal@cs.uml.edu

(‡) Department of Computer Science
University of Missouri
Rolla, MO 65409
madrias@umr.edu

Abstract

The availability of a direct pathway from a primary sequence (denovo or DNA derived) to macromolecular structure to biological function using computer-based tools is the ultimate goal for a protein scientist. Today's state of the art protein resources and on-going research and experiments provide the raw data that can enable protein scientists to achieve at least some steps of this goal. Thus, protein scientists are looking towards taking their bench-top research from the specific to a much broader base of using the large resources of available electronic information. However, currently the burden falls on the scientist to manually interface with each data resource, integrate the required information, and then finally interpret the results. Their discoveries are impeded by the lack of tools that can not only bring integrated information from several known data resources, but also weave in information as it is discovered and brought online by other research groups. We propose a novel peer-to-peer based architecture that allows protein scientists to share resources in the form of data and tools within their community, facilitating ad hoc, decentralized sharing of data. In this paper, we present an overview of this integration architecture and briefly describe the tools that are essential to this framework.

1 Introduction

Protein scientists today, especially in this post genomics age, expect to compliment their bench-top research with computer based discoveries. Until now, this bench-top and desk-top approach has had a heavy bias towards wet chemistry due to the lack of available digital resources. Now, however, there are several hundred large protein databases, each with distinct aims, shapes and usages. For example, some primary resources contain only data gathered on one specific organism (GDB on the Human Genome Project), others collect data on all biologically interesting concepts (SWISS-PROT on proteins for all organisms), while still others focus on storing literature (PubMed on scientific documents).

However, while there is this broad spectrum of information that is accessible over the Web, each data source comes with its own concepts, semantics, data formats, and access methods. Currently the burden falls on the scientist to manually (via programs) convert between the data formats, resolve conflicts, integrate data, and interpret results in order to make use of this information. Given the increasing number of protein data sources currently on-line (somewhere between 500 and 1000 [4]), such a manual approach is inevitably tedious, error-prone and consequently obsolete, leaving data under-exploited and under-utilized. Surveys have shown that due to the burden of manual integration, more often than not scientists use and limit their search for information to a select few (three to five out of a possible 500 or more) data sources [4].

Research in architectures and tools for data integration has been extensively investigated in the database community [19]. One approach that has been successfully used to develop integrated systems includes materialization in data warehouses [2]. Data warehouses [2] import the databases of interest into a single large database wherein information can be queried, retrieved and organized as a whole. However, while this architecture provides control over the contents of the warehouse and faster access to the information, it is not an ideal integration approach for the rapidly growing protein database field. Its disadvantages include a lack of scalability and the heavy burden of maintenance for local administrators in the face of updates to the local data sources.

Another approach uses middle-ware mediation based solutions [3] in which an administrator defines a global *mediated schema* for the application domain and specifies semantic mappings between the sources and the mediated

Proceedings of the 17th International Conference on Database and Expert Systems Applications (DEXA'06)
0-7695-2641-1/06 $20.00 © 2006 IEEE

schema. In more recent research [3], mediated schemas have often been replaced by ontologies that describe the concepts of the particular domain. Queries are made against the mediated schema/ontology and the results integrated locally to provide answers to the queries. It would appear that the mediated schema provides some degree of flexibility in that local data sources can evolve independently; and provides better scalability than the warehouse approach in terms of the number of sources that can be integrated. In reality however, the mediated schema is often the bottleneck and requires that the schema design be done carefully and globally [7]. Moreover, data sources cannot change significantly or they may violate the mappings to the mediated schema and concepts can only be added to the mediated schema by the central administrator. The ad hoc join-at-will extensibility of the Web that is closer to a *natural* fit for the biological domain is missing from these current approaches, making in some cases even small-scale information sharing tasks difficult to achieve.

In this paper, we propose a peer-to-peer integration architecture that facilitates ad hoc, decentralized sharing and administration of data and defining of semantic relationships. Using this architecture, every user of the system can contribute new data by relating it to existing concepts and schemas, define new schemas that others can use as frames of reference for their queries, or establish new relationships between existing schemas, and query this "Web of Information" in an effective manner.

The rest of the paper is organized as follows. Section 2 presents the overall architecture of our proposed peer-to-peer integration system, Promethea while Section 3 briefly describes the key services offered by our system. We conclude in Section 4.

2 Promethea Architecture

Data management using the emerging class of peer-to-peer architectures offers advantages over both the data warehouse and the mediated schema approaches. A peer-to-peer (P2P) distributed system is one in which participants rely on one another for service, rather than solely relying on a dedicated and often centralized infrastructure. Instead of strictly decomposing the system into clients (which consume services) and servers (which provide services), peers in the system can elect to provide services as well as consume them. To accomplish information integration in a P2P based system, every participant need only define the mappings between their data and that of some of the peers[1], and the peers are not forced to map to a single mediated schema/ontology. Thus, a peer-based data management system provides not only the advantages of the mediated schema approach, but also the ad hoc de-centralized extensibility of the Web wherein every participant can define its own schema and data, and can join in by simply defining a mapping of their data to some other peers in the system.

The membership of a pure peer-to-peer system is, however, relatively unpredictable: service is provided by the peers that happen to be participating at any given time. To provide a basic level of quality of service, there have been proposals for a *hybrid* class of peer-to-peer architectures [11] where in one or more peers take on the role of a centralized server. To provide both the ad hoc extensibility of the Web and quality of service, we propose a hybrid P2P-based distributed system architecture for Promethea.

Figure 1 depicts the hybrid network topology used in Promethea. To provide a basic level of quality of service we divide the peers[2] in the system into two main categories: *permanent peers* and *transient peers*. A permanent peer is available at all times, while a transient peer participates and shares its data for some period of time, but provides no long term commitment on its availability. All permanent peers in the network share and conform to a *mediated ontology*, called *PrOnto*. A transient peer, however, shares its resources on Promethea by providing a local mapping of its ontology/schema[3], *peer ontology*, to the mediated ontology, *PrOnto*. That is, a transient peer provides its own ontology and data. Both permanent and transient peers can serve as *query portals*, i.e., queries can be issued at and against either the mediated ontology at the permanent peers, or against the peer ontology at the transient peers.

Figure 2 shows the architecture of a peer in Promethea. The **Wrapper** module, tuned based on the local data model, manages the interaction with the local database including the local database schema (LDS). The **P2P Communication** layer is responsible for all of the network activity of the peer, i.e., the communication with the other peers. We adopt the JXTA [8] peer-to-peer platform, a de-facto standard for P2P applications, as the implementation framework for the Promethea P2P Communication layer. JXTA provides the basic building blocks needed for development of a P2P system, including support for the definition of a peer on a network, creation of communication links (called pipes) between peers, and sending and receiving of messages. In addition JXTA also provides a set of advanced capabilities such as the creation of peer groups (useful for setting up permanent peers), specification of services based on SOAP [13] and WSDL [18] standards (useful for easy integration at the services level, and discovery of services in de-centralized environments that are especially useful in the context of Promethea. We note that while P2P and Web Services were originally designed to address differ-

[1] A mapping to all peers is not required, as the P2P network can enable the *friends of friends* protocol, wherein a query that cannot be answered by the peer is forwarded to its known peers.

[2] Each participating computer is termed as a peer.

[3] For the peer, we use the term ontology and schema interchangeably.

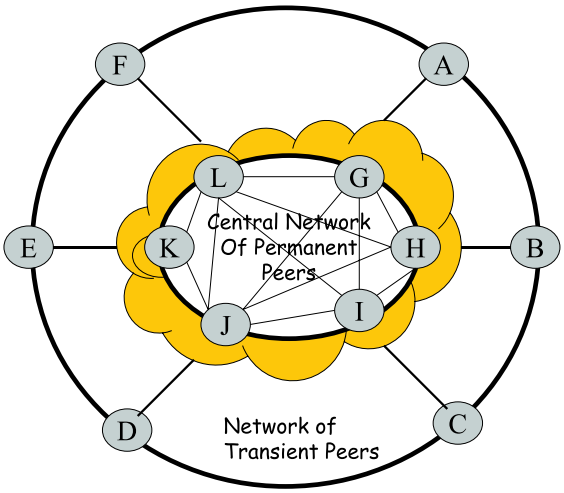

Figure 1. Hybrid Peer-to-Peer Network of Promethea.

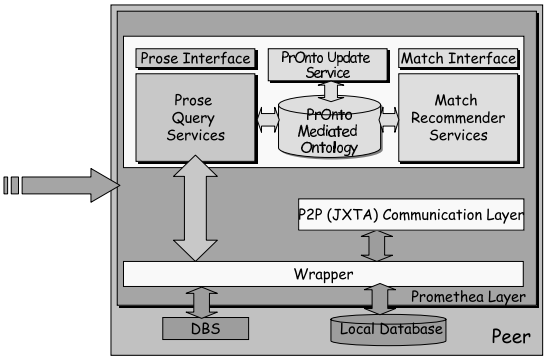

Figure 2. Architectural Overview of a Peer in Promethea.

ent problem domains, there are advantages to exposing the Promethea services as Web Services, and vice versa in integrating existing Web Services to be part of the Promethea services. An example of this is the possible integration of the BLAST Web Services interface with the Promethea query services.

The **PrOnto**, **Match Recommender**, **Ontology Update** and Prose are the main services provided by a peer in Promethea. These are described in Section 3.

3 The Promethea Modules

PrOnto. ***PrOnto***, the global mediated ontology, is a

```
<xsd:element name = "concept">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref = "hasRelationship"/>
         <xsd:element ref = "subClassOf"/>
         <xsd:element ref = "synonyms"/>
         <xsd:element ref = "location"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
<xsd:element name = "location">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element ref = "name"/>
         <xsd:element ref = "URI"/>
         <xsd:element ref = "translation"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
```

Figure 3. *PrOnto* **Concept Schema.**

key resource for all other services offered by a peer, and describes the information content of all permanent peers in the network. A transient peer, however, shares its resources in Promethea by providing a local mapping of its ontology/schema[4], *peer ontology*, to the mediated ontology, *PrOnto*, effectively providing its own ontology and data.

The past few years have seen a surge of activity in the development and use of ontologies in the bio-informatics field. Most of these ontologies [1, 14] are targeted towards facilitating specific tasks. For example, RiboWeb's [1] main aim is to facilitate the construction of three-dimensional models of ribosomal components and compare the results to existing studies. The TAMBIS [3] ontology, TaO [14], on the other hand is used as a primary source of information to enable biologists to ask questions over multiple external databases using a common query interface. The mediated ontology, *PrOnto*, plays a similar crucial role in the success of Promethea. *PrOnto* serves both as a repository of information for the concepts and their relationships described on the permanent peers in Promethea; and as a facilitators of mediation services such as (1) the schema matching process of the Match Recommender; and (2) the semantic query processing of Prose. Figures 3 and 4 show examples of the types of information that can be captured in *PrOnto*.

Match Recommender. The **Match Recommender** services automate (with user feedback) the process of discovering semantic mappings between the local content and the content available in Promethea as described by the mediated ontology, *PrOnto*. Figure 2 gives an overview of the internal architecture of the data sharing tool, the *Match Rec-*

[4]For the peer, we use the term ontology and schema interchangeably.

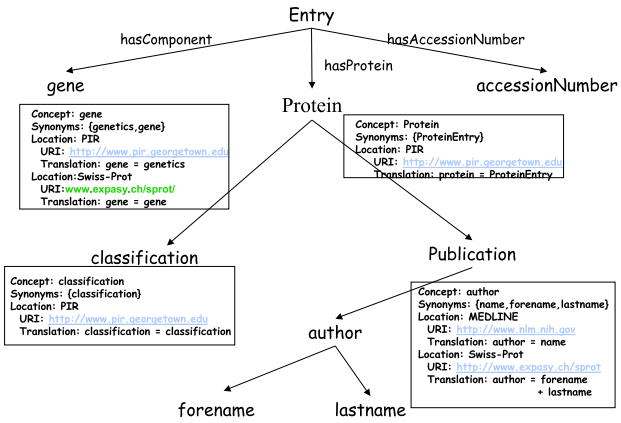

Figure 4. Example Concept in *PrOnto*.

ommender, and its interaction with the initiating peer. Here, the initiating peer, `Peer 1`, communicates with the Match Recommender providing its ontology as input. As a next step in the process, *match algorithms*, the crux of the Match Recommender, are deployed to semi-automatically detect the semantic matches between the provided peer ontology and *PrOnto*. The best matches, as decided by the *QoM evaluator*, are then presented to `Peer 1` who can either accept or decline the recommended matches.

The number and the quality of matches suggested by the Match Recommender are dependent on the match algorithms employed to discover the correspondences between the peer schema and the mediated ontology. The Match Recommender currently employs a *linguistic* algorithm that utilizes domain-specific dictionaries as well as domain-independent dictionaries such as Wordnet [10] to decide on a semantic match [6, 16], a *structure*-based match algorithm that relies on finding correspondences based solely on the schema structure [6, 16], and two unique hybrid algorithms that combine the structure and linguistic algorithms to determine correspondences between schema entities [15, 16, 12].

Ontology Update and Propagation. Acceptance of the semantic relationships produces a set of updates that can be applied to either *PrOnto* for a permanent peer, or the peer ontology for a transient peer. Additionally, annotations may be made on *PrOnto* or the peer ontology to capture the actual data translation. This dichotomy between the permanent and transient peers allows for scalability of the peer network in a safe manner, and enables transient peers to make their information available with little impact to the overall system, namely the mediated ontology. The actual updates to the ontology are handled by the **Ontology Update** services module.

Moreover, in most cases the biological data is highly dynamic changing at both the data and the schema level, and requires complex maintenance procedures to (1) update the peer's repository in light of the changes to the underlying data sources and (2) to detect changes across peers in order to keep same data consistent. This raises a number of practical problems (1) How to detect underlying data sources that have changed and detect those changes within and across peers? (2) How to detect changes in the schema across peers? (3) How can we automate the refresh process and propagate those to related peers? (4) How can we track the origins or the "provenance" of data?

Furthermore, as each peer can store data in many different formats, such as in text file, in relational data model, spreadsheets, or in a hierarchical format, change management techniques should consider all the different formats to address heterogeneous biological. The changes can also be of different types: changes in metadata information, changes in data format, or changes in data by means of versions along with timestamps of changes, or addition of data etc. The changes are either periodically uploaded for the users or are time stamped so that users can infer changes or keeping a list of corrections. However, none of these methods precisely describe the minimal changes that have been made to the data. The version management tools only detect changes as deletion and insertion, but can not relate changes semantically. Similarly, these tools fail to identify the positions in a protein sequence where a segment has been inserted, modified, or deleted. Therefore, there is a need to develop effective change management tools. Our approach here is to map the Biological data sources and meta data to XML and then store them in a relational model. Then apply change management techniques like [5] to detect changes. This approach is much more scalable which is very important as Biological data is huge and techniques such as [17] have problems with large data sets as they are main memory algorithms.

Similarly, Biological data sources are often represented as XML schema across data peers. Schema plays an important role in searching process and data integration across peers. However, XML schema can evolve. Therefore, there is a need to detect schema changes using techniques similar to proposed in [9] for detecting changes in DTDs.

Query Services. All peers in the system can also serve as query portals, i.e., queries can be issued at and against either *PrOnto* at the permanent peers, or against the peer ontology at the transient peers. The query services which include the query interface and processing are provided as part of the $\mathfrak{Prose}$ **Query** services module. Figure 5 gives an overview of the internal architecture of $\mathfrak{Prose}$. A query is submitted to the peer network via the $\mathfrak{Prose}$ Query Interface. The query is reformulated/unfolded based on the

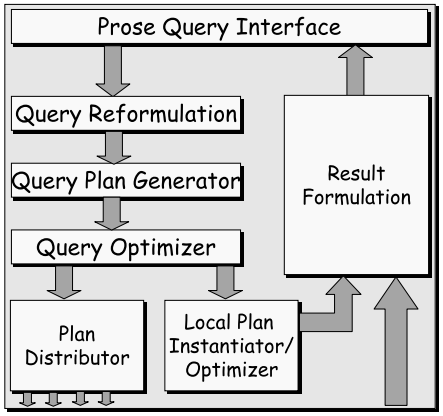

Figure 5. Internal Architecture of the Prose Query Engine.

mediated ontology, *PrOnto* in the **Query Reformulation** module, and a query plan is generated (**Query Plan Generator**). The **Query Optimizer**, **Plan Distributor** and **Local Plan Instantiator** represent the key components of the distributed query processing module that is responsible for the execution of the query. The results obtained by the evaluation of the query plan are combined locally by the **Results Formulator** and returned to the user after any needed local processing.

4 Conclusions and Future Work

In this paper we present the architecture of Promethea, a peer data management system that allows protein scientists to share resources in the form of data and tools within their community and have them available for searching in an integrated querying environment. In particular, we contribute: (a) a novel architecture for the management and integration of protein resources that provides the ad hoc extensibility of today's Web; (b) a set of match algorithms that semi-automate the detection of correspondences between the user's and the global schema; (c) ontology update and propagation tools that detect, propagate and modify the changes made to the global ontology; and (d) a novel query answering system that reaches across the available peers to answer user queries.

References

[1] R. B. Altman, M. Bada, X. J. Chai, M. W. Carillo, R. O. Chen, N. F. Abernethy, and S. M. Informatics. RiboWeb: An Ontology-Based System for Collaborative Molecular Biology. *Intelligent Systems and Their Applications*, 14(5):68–76, 1999.

[2] J. An, T. Nakama, Y. Kubota, and A. Sarai. 3DInsight: An Integrated Relational Database and Search Tool for Structure, Function and Property of Biomolecules. In *Bioinformatics*, pages 188–195. ACM Press, 1998.

[3] P. Baker, A. Brass, S. Bechhofer, C. Goble, N. Paton, and R. Stevens. TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources: An Overview. In *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology, (ISMB)*, 1998.

[4] F. Bry and P. Kroger. A Computational Biology Database Digest: Data, Data Analysis, and Data Management. *International Journal on Distributed and Parallel Databases, special issue on Bioinformatics*, 2002.

[5] Y. Chen, S. Madria, and S. Bhowmick. DiffXML: Change Detection for XML Data. In *Int. Conference on Database Systems for Advanced Applications (DASFAA)*, 2004.

[6] K. T. Claypool, V. Hegde, and N. Tansalarak. QMatch: A Hybrid Match Algorithm for XML Schemas. In *Proceedings of the 2nd International Workshop on XML Schema and Data Management (to appear)*, April 2005.

[7] A. Halevy, Z. Ives, P. Mork, and I. Tatarinov. Piazza: Data Management Infrastructure for Semantic Web Applications. In *Proceedings of the World Wide Web Conference (WWW)*, pages 20–24, 2003.

[8] JXTA Technology. *http://www.jxta.org/*.

[9] E. Leonardi, T. T. Hoai, S. Bhowmick, and S. Madria. DTD-Diff: A Change Detection Algorithm for DTDs. In *Int. Conference on Database Systems for Advanced Applications (DASFAA)*, 2006.

[10] G. Miller. Wordnet: A Lexical Database for English Language. cogsci.princeton.edu/ wn/, 2002.

[11] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, et al. Peer-to-Peer Computing. In *Technical Report HPL-2002-57, HP Laboratories, Palo Alto*, March 2002.

[12] K. Patel and K. T. Claypool. SUSAX: Context-specific searching in xml documents using sequence alignment techniques. In *Proceedings of the Third International Workshop on XML Schema and Data Management (XSDM)*, April 2006.

[13] Simple Object Access Protocol (SOAP) Version 1.2. *http://www.w3.org/TR/soap12-part1/*.

[14] TAMBIS. The tambis ontology (tao). http://imgproj.cs.man.ac.uk/tambis/.

[15] N. Tansalarak and K. T. Claypool. QoM: Qualitative and Quantitative Schema Match Measure. In *International Conference on Conceptual Modeling (ER)*, October 2003.

[16] N. Tansalarak and K. T. Claypoool. QMatch - Using Paths to Match XML Schemas. *Elsevier's Data and Knowledge Engineering*, 2006. to appear.

[17] Y. Wang, D. DeWitt, and J.-Y. Cai. X-Diff: A Fast Change Detection Algorithm for XML Documents. In *IEEE International Conference on Data Engineering (ICDE)*, 2003.

[18] Web Services Description Language Version 1.1. *http://www.w3.org/TR/wsdl*.

[19] G. Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, 25(2):38–49, 1992.