Vertical Integration of Bioinformatics Tools and Information Processing on Analysis Outcome

Andigoni Malousi, Vassilis Koutkias, Ioanna Chouvarda, and Nicos Maglaveras

Lab. of Medical Informatics, Faculty of Medicine, Aristotle University
P.O. Box 323, 54124, Thessaloniki, Greece
{andigoni,bikout,ioanna,nicmag}@med.auth.gr

Abstract. Biological sources integration has been addressed in several frameworks, considering both information sources incompatibilities and data representation heterogeneities. Most of these frameworks are mainly focused on coping with interoperability constraints among distributed databases that contain diverse types of biological data. In this paper, we propose an XML-based architecture that extends integration efforts from the distributed data sources domain to heterogeneous Bioinformatics tools of similar functionalities ("vertical integration"). The proposed architecture is based on the mediator/wrapper integration paradigm and a set of prescribed definitions that associates the capabilities and functional constraints of each analysis tool. The resulting XML-formatted information is further exploited by a visualization module that generates comparative views of the analysis outcome and a query mechanism that handles multiple information sources. The applicability of the proposed integration architecture and the information handling mechanisms was tested and substantiated on widely-known ab-initio gene finders that are publicly accessible through Web interfaces.

1 Introduction

The distribution of massive amounts of biological data through several multi-institutional databanks has prompted the development of numerous decentralized Bioinformatics tools that analyze, predict and attempt to interpret raw data into meaningful knowledge [1]. Currently, a wide range of analysis tools are freely accessible through Web interfaces implementing diverse algorithmic approaches on multiple Bioinformatics problems. The efficiency of these methods is increasingly inviting and the exported outcomes are quite valuable, in order to decipher and model biological processes. An even more challenging issue is to orchestrate multiple resources and accordingly interrelate their functionalities/outcomes through integration frameworks that are built upon globally accepted data exchange protocols [2]. Despite the complexity of the domain, systematic efforts gave rise to the establishment of structured and semi-structured data formats that facilitate advanced querying and interpreting capabilities, basically addressing requirements of diverse data that are deposited in heterogeneous databanks [3].

In this work, we propose an XML-based architecture that a) integrates multiple, heterogeneous biological analysis tools of similar functionalities, i.e., vertical integration, b) structures the resulting outcomes in form of XML documents that follow common formatting instructions, and c) incorporates advanced information handling capabilities in terms of comparative graphical views and query processing on the analysis results. The applicability of the proposed approach was tested on a set of ab-initio gene prediction tools that are freely accessible through Web interfaces.

2 Technical Background

2.1 Bioinformatics Resources Profile

In general, the design and implementation requirements of an integration architecture are implied by physical and functional restrictions of the associated resources. In the Bioinformatics field most analysis tools share common characteristics as described below [4]:

- *Decentralization*: Typically, most Bioinformatics analysis tools are accessible through Web interfaces that span through geographically distributed servers. A specific functionality may be provided by multiple tools and the analysis outcome may diverse depending on the algorithmic approach followed.
- *Heterogeneity*: Diversities among Bioinformatics tools involve both their schematic and semantic representation. Interoperability issues are partially coped with the adoption of technologies such as XML, Web Services, etc. Currently, several biological data repositories have adopted XML and related technologies within their general data model, in order to offer structured views of non-structured data sources, facilitating this way flexible data handling [5]. Unlike databanks, most biological analysis tools do not offer machine-readable and processable data structures of their outcome.
- *Autonomy*: Most analysis tools are functionally autonomous, i.e., they perform analysis independently and their underlying design and implementation model may be changed/updated without prior public notification.
- *Multiple data formats*: The data types involved in an analysis range from simple, textual DNA sequences to complex 3D protein structures, according to domain-specific requirements.
- *Query capabilities, parameterization*: Analysis tools that typically offer similar functionalities may exhibit diverse query capabilities or even different configuration parameters, depending mostly on their underlying algorithmic approach.

Based on these remarks, it is evident that the development of integration architectures that interlink multiple types of biological analysis resources requires a thorough examination of the functional requirements, and the implied interoperability constraints.

2.2 XML and Bioinformatics

XML is the standard markup language for describing structured and semi-structured data over the Internet [6]. Together with other proposed and accepted standards, supervised also by the W3C Consortium, XML is considered an ideal means to provide the infrastructure for integrating heterogeneous resources. Several biological databases such as Entrez [7] and EMBL [8], support implementations of external modules that export or view contents as XML documents and plug them to their general relational schema. In addition, various XML-based markup languages have been introduced as standard formats for describing biological data such as BioML (Biopolymer Markup Language) [9], MAGE-ML (Microarray Gene Expression Markup Language) [10], and BSML (Bioinformatic Sequence Markup Language) [11].

With the advent of more sophisticated XML-based technologies, including RDF (Resource Description Framework) and Web Services, XML is intended to play a more important role in the future. Moreover, related technologies, such as XQuery [12], offer enhanced capabilities in performing complex queries and retrieving combined data from multiple XML data sources.

2.3 Integration Architectures in Bioinformatics

Several integration frameworks have been developed aiming to facilitate reusability capabilities among the incorporated resources, addressing diverse descriptive data models and transparency levels. Apart from the warehousing approach [13], mediator-based integration schemes, followed in the proposed architecture, handle on demand integration requests through tailored query formulation [14], [15]. For example, BioKleisli is an integration architecture relying on an object-oriented database system that takes advantage of the expressiveness of CPL (Collection Programming Language), in order to perform queries against multiple heterogeneous resources in pipelines of domain-specific processing nodes [16]. TAMBIS (Transparent Access to Multiple Bioinformatics Information Sources) follows a mediator-based integration scheme aiming to formulate queries against ontology-driven descriptions of Bioinformatics sources [17]. In addition, DiscoveryLink offers intermediary modules available to applications that compose requests on retrieving data from multiple biological sources [18]. Unlike TAMBIS, DiscoveryLink is a wrapper-oriented system that is intended to be used by other applications rather than end-users.

Most of these integration architectures focus on biological databases, rather than on analysis tools. Moreover, integration in the aforementioned architectures involves primarily the orchestration of complementary resources in form of workflows that execute predefined tasks in sequential order, aiming to reveal hidden dependencies/relations among different types of biological data. Unlike these horizontal integration architectures [19], this work focuses on a vertical integration scheme, in which multiple Bioinformatics analysis tools of similar functionalities are integrated, so as to provide machine-readable and comparative views of their outcomes.

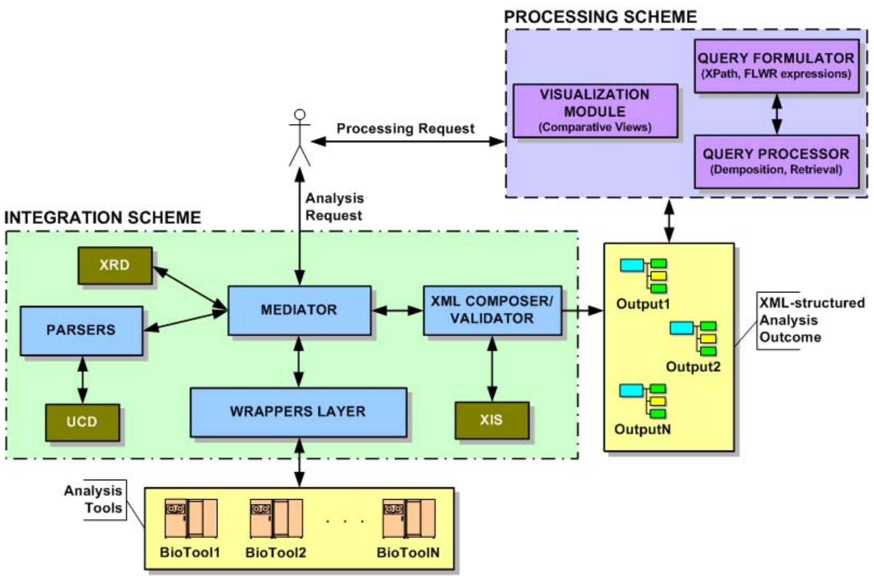

Fig. 1. The components of the XML-based integration architecture

3 Integration Scheme

As shown in Fig. 1, the proposed architecture involves two distinct parts: a) the Integration Scheme which follows the mediator/wrapper paradigm, and b) the Processing Scheme, which provides graphical representation and advanced query capabilities on the analysis outcome. It has to be noted that these two parts are not synchronized and may operate independently, since the XML documents generated by the former part are stored locally and, therefore, can be accessed by the processing modules on demand. The basic structural and functional components of the Integration Scheme are described in the following.

3.1 Structural Components

The specifications of the proposed integration architecture rely on particular features that describe Bioinformatics tools and distinguish them from the general physical and technical characteristics of the biological data sources. Unlike distributed databases, most Web-accessible biological analysis tools present their outcomes in unstructured or semi-structured forms, focusing more on human-readability, rather than machine-manageable formats. To cope with representational heterogeneities and potential accessibility incompatibilities of the associated resources, a set of structural components is included in the Integration Scheme as described below.

Uniform Concept Description. Even within the same class of Bioinformatics tools, there may be variations in the type of information provided, different metrics to encode quantitative parameters (e.g., probabilities and reliability scores), and grammatically diverse representations of common concepts. To increase the expressiveness of the extracted information, a Uniform Concept Description (UCD) is incorporated, which describes similar concepts, related to both input and output data, using unified terms that are stored in a global data index.

Resource Description. The XML Resource Description (XRD) module contains descriptions related to the accessibility parameters of each tool in machine-processable way that facilitates interoperation among relevant functional components. So far, most Bioinformatics tools that perform computational data analysis do not offer self-descriptive capabilities to allow direct access to their sources, thus, it is necessary to associate resource descriptions within the integration scheme. In addition, due to heterogeneities in the schematic representation, it is not feasible to incorporate a global schema that describes resources. Therefore, for each analysis tool the corresponding descriptive elements are defined in terms of XML-formatted tuples.

Information Schema. The information extracted from heterogeneous Bioinformatics analysis tools is structured in XML-formatted documents that are validated against a predefined XML-based Information Schema (XIS). The XIS provides a hierarchical view of the associated tagged elements, corresponding to the extracted data along with their relations. The exported XML-formatted outcome must comply with the formatting instructions of the XIS, in order to enable data handling by the query and visualization modules included in the Processing Scheme. The XIS elements are defined according to problem-specific requirements, and the selection is primarily based on the significance and relevance with the problem to be solved.

3.2 Functional Components

The functional components of the Integration Scheme provide transparent and coordinated access to the appropriate analysis tools, according to the requested analysis [15]. This way, the inherent complexities of submitting a request to the appropriate resource(s) and retrieving the analysis outcome(s) are hidden from the end-user. Furthermore, information extraction is applied on the obtained results, in order to select features that are relevant with the structural specifications, and, finally, compose XML-structured documents containing the information of interest. This functionality is supported by the following modules.

Mediator. Given a user request, the mediator decomposes the query, passes the relevant parameters to each one of the wrappers and coordinates the overall submission/retrieval procedure. To avoid inconsistencies among user requirements and the tools' specifications, the mediator excludes non-matching resources. The tools' compliance is validated by the mediator that matches the query parameters to the XRDs and simultaneously performs CGI and/or Java servlet requests on the matching resources, through the corresponding wrapping modules [15].

Wrappers/Parsers. For a set of analysis tools, a wrapping layer is incorporated, i.e., each tool corresponds to a relevant wrapper. Each wrapper accesses the XRD module of its assigned tool, through the mediator module, submits requests, and ships back the unstructured information obtained to the mediator [20]. After that, the mediator initiates the corresponding parsing modules. The outcome that is obtained by each wrapper is passed through the corresponding parsing module that filters the extracted information according to the XIS descriptions. Moreover, to overcome representational heterogeneities coming from semantically similar concepts that are encoded in different terms, the global XML schematic descriptions of the incorporated outcome are used to transform each tool's terminology into annotations of the predefined UCD.

Document Composer/Validator. The extracted information is processed by the Document Composer, which structures data within appropriate tagged elements, following the formatting rules of the predefined XIS. Accordingly, the Document Validator confirms the validity of the extracted XML documents and reports potential inconsistencies with the prescribed XIS definition. This process is iteratively applied and the resulting information is stored in XML documents.

4 Processing Scheme

In the proposed architecture, information handling is related with visualization of the generated XML documents, in terms of comparative views of the obtained outcomes, and an advanced query formulation/execution mechanism that enables feature extraction and interrelation among the information obtained from vertically integrated analysis tools. The basic modules contained in the Processing Scheme are shown in Fig. 1 and described below.

4.1 Visualization Module

The Visualization Module serves the need of combining and comparing evidence derived from multiple vertically integrated resources, in the form of user-friendly depictions that may help researchers get a more comprehensive view of the extracted information and even facilitate further investigation of the underlying biological processes. Apart from the XML documents' content, graphical representations may be applied on the resulting XML views generated by specific XQuery statements. In both cases, the information depicted may be configured with respect to the query criteria and the technical constraints implied by each analysis problem.

4.2 Query Processing

Query Requirements. The query mechanism of the Processing Scheme addresses comparative management and analysis of the information enclosed in the tuples of the validated XML documents. The query processing capabilities conform with specific structural and functional requirements such as:

– The XIS-compliant data that are dynamically extracted from the integration scheme are stored in valid and machine-readable documents.
– Query processing may involve more than one information sources.
– The tree nodes of the XML-formatted documents are a priori known, unlike the number of instances of each tuple which is dynamically determined.
– The resulting documents follow same schema definitions and incorporate semantically identical concepts.
– Information retrieval is performed using both simple XPath expressions, against a single source, and complex statements targeting multiple sources.
– The query processing may involve the construction of new tagged elements, generating aggregated views of single or multiple information sources.
– The query mechanism does not need to address error-handling, since the outcomes' well-formedness and validity are confirmed by the XML Validator.

The Query Processor is built on the XQuery data model [12]. XQuery considers sources as nesting structures of nodes, attributes and atomic values, and assigns XML-oriented descriptions in both query formulation and information retrieval. The implemented Query Processor constitutes the intermediary connecting the client interface with the XML data sources in an effort to further manage information derived from the vertically integrated tools and reveal hidden dependencies/relations among the incorporated I/O features.

Types of Information Retrieval. Three types of queries are supported:

1. *Join*: Generate aggregated views of node elements encoding similar concepts that span through multiple information sources. This type of query serves the vertical integration requirements, supporting customized views of selected tuples that are stored in same-schema information sources.
2. *Select-on-One*: Navigate through the tagged elements of a single source, in order to select node(s)/atomic value(s) that match user-defined criteria. This type of information is obtained by simple XPath/XQuery expressions.
3. *Select-on-Many*: Comparative exploration by defining selection rules against multiple information sources. To address this type of requests, the query mechanism has to formulate and execute complex FLWR (For-Let-Where-Return) expressions in the generic form of:

```
[LET <$src_i$> :=doc(<$URI_i$>) <$p_i$>]
FOR [<$vr_i$> in <$sp_i$>]
WHERE [<$f(vr_i)$ >]
RETURN <atomic value(s)/tuple(s)> of <$vr_i$>
```

Assuming N XML-formatted information sources, the `LET` clause locates the source documents by their $\{URI_i | i = 1 \dots N\}$ definitions and binds each identifier to a variable $\{src_i | i = 1 \dots N\}$. Expressions in square brackets may appear more than once, thus, the `LET` clause may be similarly repeated to include additional sources. p_i and sp_i correspond to simple path expressions that bind the target tuples to variables src_i and vr_i respectively (`FOR` clause). The `WHERE` clause contains the condition declarations that meet end-user requirements using logical, numerical operators and other functions among the predefined variables vr_i. The `RETURN` clause defines the resulting atomic values/tuples that match selection criteria.

5 Test Case: Ab-initio Gene Prediction

5.1 Problem Description

Computational gene prediction in eukaryotic organisms is performed by identifying coding features (*exons*) within usually larger fragments of non-coding regions (*introns*) [21]. Ab-initio methods rely exclusively on the intrinsic compositional and structural features of the query DNA sequence in order to build the most probable gene structure. Technically, these approaches implement various statistical methods, applied on species-specific gene models, and the analysis outcome is usually associated with probabilities/scores that reflect the reliability of the prediction [22]. The sensitivity/specificity of each prediction tool exhibits significant variations, depending mostly on the trained gene model and the underlying algorithmic approach. Moreover, various exhaustive assessments have shown that most ab-initio gene finders exhibit high accuracy prediction levels at nucleotide level; however the overall accuracy at gene level is still disputable, basically due to alternative expression patterns that change coding boundaries and therefore the identified gene assembly [23], [24].

Currently, there are 25 ab-initio gene prediction tools that are freely accessible through Web interfaces [22]. The applicability of the presented architecture was tested and substantiated on a subset of 5 widely known gene finders, namely, *Augustus*, *Fgenes*, *Fgenesh*, *Genscan*, and *HMMgene*. The following sections describe the implemented architectural components and illustrate the virtue of the approach adopted through examples of use.

5.2 Integrating Ab-initio Gene Finders

The gene finding integration scheme was implemented in accordance with the structural and the functional modules of the Integration Scheme shown in Fig. 1. Specifically, for each gene finder an XRD module was incorporated addressing individual accessibility requirements and descriptions of the associated I/O and configuration parameters. In addition, to cope with the representational heterogeneities of the extracted information, a common vocabulary was introduced that encodes the predicted features along with their positional descriptions into common terms, following the UCD specifications.

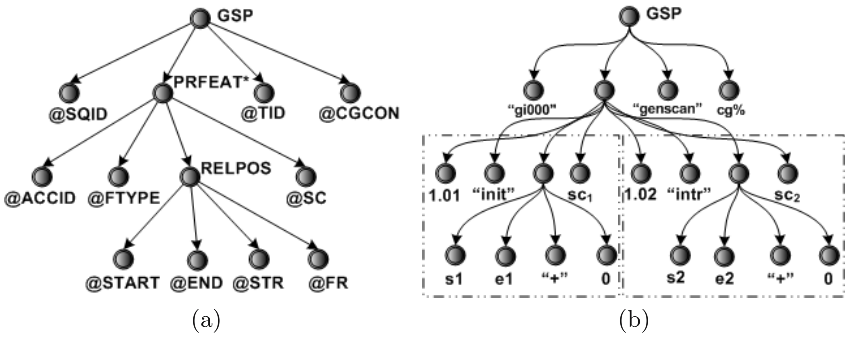

Fig. 2. (a) Hierarchical view of the XIS (* indicates potential repeated elements and @ denotes valued schema elements) and (b) an example well-formed XML document containing sample values

The associated information types along with their relations and accepted terminology were finally captured in an XIS description, as shown in Fig. 2(a). Each exonic feature ($PRFEAT$) identified within the query DNA sequence (@$SQID$) by a gene finder (@TID) is further described by its type (@$FTYPE$), its positional features ($RELPOS$@[$START, END, STR, FR$]), and the associated probability/score (@SC). An example XML document that conforms to the XIS formatting instructions is shown in Fig. 2(b). The overall coordination of the processes involved in the integration scheme is taken by a mediator that was implemented according to the specifications of the generic Integration Scheme. Similarly, for each gene finder the corresponding wrapper and parser modules were developed resulting in uniform outcome descriptions that are finally structured in XML-formatted documents that comply with the predefined XIS rules.

5.3 Comparative Graphical Representation

In general, the Visualization Module is configured based on problem-specific requirements aiming to generate user-friendly and comparative depictions of the analysis outcome. In this case, the Visualization Module was used to dynamically formulate graphical representations of the identified coding features in horizontal panels that facilitate comparisons of the evidence derived from the incorporated gene finders. Figure 3 illustrates a comparative depiction of the exons identified within a query genomic sequence that is known to contain exons 2-9 of the human *p53* tumor suppressor gene. It is noteworthy that the identified exonic features are not identical (especially at 5' end region) in all predictions, resulting in alternative transcript forms.

5.4 Query Processing

According to the types of information retrieval described in Section 4.2, the query processing mechanism may be applied either on a single source, serving simple navigation/filtering requirements, or on multiple XML-based sources in

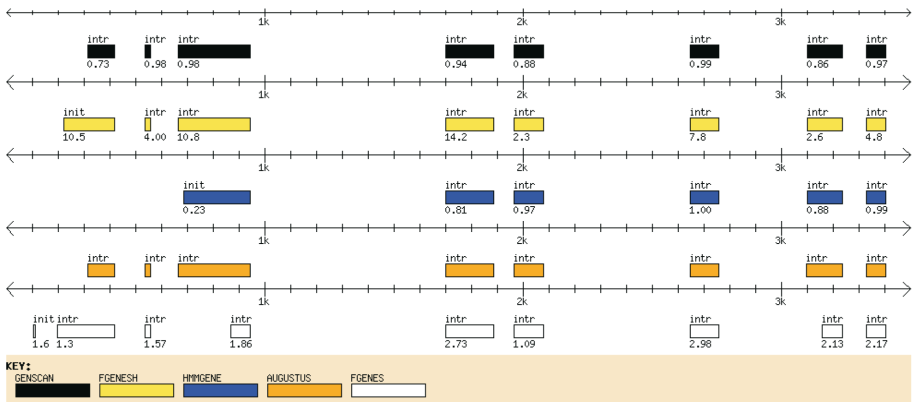

Fig. 3. Comparative graphical view of the exons identified in human chromosome 17, within 7,512,465:7,520,980 genomic region that is known to contain exons 2-9 of the *p53* tumor suppressor gene. The labels *init* and *intr* indicate an initial and internal exon respectively. Each exon (except those identified by *Augustus*) is associated with a probability (*Genscan*, *HMMgene*), or score (*Fgenesh*, *Fgenes*)

order to join, compare and/or select specific segments of the included tagged elements. An example "*Select-on-Many*" request that the query processor can answer is:

"*On the overall prediction outcome obtained for p53 human genomic sequence, identify exonic features that:*
1. *are predicted by both Genscan and Fgenesh, and*
2. *exhibit high probability scores* (*Genscan probability* > 0.80), *and*
3. *their relative start site is located within 100:1000 genomic region*".

The corresponding XQuery statement that retrieves the matching features of the declared XML sources is presented in the left column of Table 1. The tagged-elements in the right column contain the results obtained by executing the XQuery expression. As shown, the query is configured to retrieve specific elements (`RETURN` clause), providing a quantitative view of the matching features.

6 Conclusion

The architecture described in this paper addresses both vertical integration of biological analysis tools and information management on the resulting outcome. The mediator/wrapper approach was adopted for transparent and coordinated access to the incorporated resources and XML has been used to describe the structural components of the Integration Scheme, as well as the evidence derived from the analysis outcome. The information handling mechanisms are applied on the results obtained from the Integration Scheme, aiming to further exploit the information captured within the resulting XML documents. These handling capabilities are supported by a visualization module that generates comparative

Table 1. An example FLWR expression (left column) and the resulting tagged elements that match selection criteria (right column)

Query	**Results**
`<QUERY>`	`<QUERY>`
`{`	`<VIEW>`
`let $h :=doc(<fgenesh source>)//RELPOS`	`<TYPE>intr</TYPE>`
`for $g in doc(<genscan source>)//PRFEAT`	`<BEGIN>535</BEGIN>`
`where (($g/RELPOS/START=$h/START)`	`<END>556</END>`
`and ($g/RELPOS/END=$h/END)`	`<SCORE>0.983</SCORE>`
`and ($g/RELPOS/BEGIN>100)`	`</VIEW>`
`and ($g/RELPOS/BEGIN<1000)`	`<VIEW>`
`and ($g/SC>0.80))`	`<TYPE>intr</TYPE>`
`return`	`<BEGIN>666</BEGIN>`
`<VIEW>`	`<END>944</END>`
`  <TYPE>{$g/FTYPE/text()}</TYPE>`	`<SCORE>0.982</SCORE>`
`  <BEGIN>{$g/RELPOS/START/text()}</BEGIN>`	`</VIEW>`
`  <END>{$g/RELPOS/END/text()}</END>`	`</QUERY>`
`  <SCORE>{$g/SC/text()}</SCORE>`	
`</VIEW>`	
`}`	
`</QUERY>`	

graphical descriptions of the analysis outcome and a query processing mechanism that formulates and executes complex query expressions against multiple information sources, based on the expressiveness of the XQuery data model.

Following the need for interoperable biological resources, the proposed architecture may be adopted in various applications that address vertical integration problems. Furthermore, the modularity and extendability of the associated components enables incorporation of complementary resources that could help interlink diverse forms of biological data. Being able to co-relate different types of biological evidence and orchestrate distributed and heterogeneous resources is necessary, in order to increase our understanding and knowledge on the underlying biological processes.

References

1. Stevens, R., Goble, C., Baker, P., Brass, A.: A Classification of Tasks in Bioinformatics. Bioinformatics 17(2) (2001) 180–188
2. Wong, R.K., Shui, W.: Utilizing Multiple Bioinformatics Information Sources: An XML Databases Approach. Proceedings of IEEE Symposium on Bioinformatics and Bioengineering IEEE Computer Society (2001) 73–80
3. Ng, S.-K., Wong, L.: Accomplishments and Challenges in Bioinformatics. IEEE IT Professional 6(1) (2004) 44–50
4. Hernandez, T., Kambhampati, S.: Integration of Biological Sources: Current Systems and Challenges. ACM SIGMOD Record 33(3) (2004) 51–60

5. Achard, F., Vaysseix, G. Barillot, E.: XML, Bioinformatics and Data Integration. Bioinformatics 17(2) (2001) 115–125
6. World Wide Web Consortium: Extensible Markup Language (XML) 1.0 W3C Recommendation (Third Edition) (2004) http://www.w3.org/TR/2004/REC-xml-20040204/
7. Entrez - Search and Retrieval System. http://www.ncbi.nlm.nih.gov/Entrez (2003)
8. Wang, L., Riethoven, J.-J., Robinson, A.: XEMBL: Distributing EMBL Data in XML Format. Bioinformatics 18(8) (2002) 1147–1148
9. Fenyö, D.: The Biopolymer Markup Language. Bioinformatics 15(4) (1999) 339–340
10. Spellman, P.T. et al.: Design and Implementation of Microarray Gene Expression Markup Language (MAGE-ML). Genome Biology 3(9) (2002)
11. BSML, XML Data Standard for Genomics: The Bioinformatic Sequence Markup Language http://www.bsml.org/
12. World Wide Web Consortium: XQuery 1.0: An XML Query Language (W3C Working Draft) (2005) http://www.w3.org/TR/xquery/
13. Hammer, J., Schneider, M.: Genomics Algebra: A New, Integrating Data Model, Language, and Tool for Processing and Querying Genomic Information. Proceedings of the 2003 CIDR Conference (2003)
14. Tork Roth, M., Schwarz, P.: Don't Scrap it, Wrap it! A Wrapper Architecture for Legacy Sources. Proceedings of the 23rd VLDB Conference Greece (1997) 266–275
15. Koutkias, V., Malousi, A., Maglaveras, N.: Performing Ontology-driven Gene Prediction Queries in a Multi-Agent Environment. In: Barreiro J. M., Martin-Sanchez, F., Maojo, V., Sanz, F. (eds.): Biological and Medical Data Analysis. LNCS, Vol. 3337. Springer-Verlag, Berlin Heidelberg New York (2004) 378–387
16. Davidson, S.B., Buneman, O.P., Crabtree, J., Tannen, V., Overton, G. C., Wong, L.: BioKleisli: Integrating Biomedical Data and Analysis Packages. In: Letovsky S. (Ed.): Bioinformatics: Databases and Systems. Kluwer Academic Publishers, Norwell, MA (1999) 201–211
17. Baker, P., Brass, A., Bechhofer, S., Goble, C., Paton, N., Stevens R.: TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. Proceedings of the 6th International Conference on Intelligent Systems for Molecular Biology (1998)
18. Haas, L., Schwarz, P., Kodali, P., Kotlar, E., Rice, J., Swope, W.: DiscoveryLink: A System for Integrated Access to Life Sciences Data Sources. IBM Systems Journal 40(2) (2001) 489–511
19. Sujansky, W.: Heterogeneous Database Integration in Biomedicine. Journal of Biomedical Informatics 34 (2001) 285–298
20. Aldana, J.F., Roldan, M., Navas, I., Perez, A.J., Trelles, O.: Integrating Biological Data Sources and Data Analysis Tools through Mediators. Proceedings of the 2004 ACM Symposium on Applied Computing Nicosia Cyprus (2004) 127
21. Zhang, M.Q.: Computational Prediction of Eukaryotic Protein-coding Genes. Nature 3 (2002) 698–710
22. Mathé, C., Sagot, M.F., Schiex, T., Rouzé, P.: Current Methods of Gene Prediction, their Strengths and Weaknesses. Nucleic Acids Research 30 (2002) 4103–4117
23. Rogic, S., Mackworth, A.K., Ouellette, F.: Evaluation of Gene-finding Programs on Mammalian Sequences. Genomic Research 11 (2001) 817–832
24. Mironov, A.A., Fickett, J.W., Gelfand, M.S.: Frequent Alternative Splicing of Human Genes. Genome Research 9 (1999) 1288–1293