BioFuice: Mapping-Based Data Integration in Bioinformatics

Toralf Kirsten and Erhard Rahm

University of Leipzig, Germany
tkirsten@izbi.uni-leipzig.de, rahm@informatik.uni-leipzig.de

Abstract. We introduce the BioFuice approach for integrating data from different private and public data sources and ontologies. BioFuice follows a peer-to-peer-like data integration based on bidirectional mappings. Sources and mappings are associated with a domain model to support a semantically meaningful interoperability. BioFuice extends the generic iFuice integration platform which utilizes specific operators for data fusion and workflow-like script programs. BioFuice supports explorative data analysis and query and search capabilities. We outline the integration approach by an illustrating scenario, the architecture of BioFuice and its query interface.

1 Introduction

Many biological and medical applications require access to a variety of molecular-biological objects, such as genes, proteins, their interrelationships and functions, and their correlations with phenotypical effects. These objects are maintained in a high number of diverse web-accessible data sources [Ga05] as well as in local (private) data sources, e.g. specific analysis results such as a particular list of genes or medical data on patients participating in clinical trials. Typically, such data is highly diverse so that their integration is laborious and error-prone and difficult to perform by domain experts.

Traditional data integration approaches like data warehousing and mediators are often applicable but also time-consuming to deploy and may lack sufficient support for features such as explorative data analysis. These integration approaches typically require a unified global schema to obtain a consistent view over data from different sources. However, creating such a schema for more than a few data sources is almost impossible due to the high diversity, complexity and fast evolution of sources. Each new source to consider may require adapting the global schema as well as applications built upon this schema.

A promising alternative to the traditional data warehousing and mediator solutions using a global schema are so-called peer-to-peer approaches for data integration [Ha03]. They are based on bilateral mappings between autonomous data sources, called data peers, instead of mappings between data sources and a global schema. Adding a new data source can thus be achieved by mapping it to only one existing peer instead of adapting the global schema and mapping the source to it. In bioinformatics, a peer-to-peer approach seems especially appropriate since bilateral mappings can often be derived from existing cross-references between objects of different

U. Leser, F. Naumann, and B. Eckman (Eds.): DILS 2006, LNBI 4075, pp. 124 – 135, 2006.

sources. Such cross-references refer to so-called accessions, i.e. unique object identifiers, and are omnipresent in public data sources. The cross-references are typically maintained by domain experts and thus of high quality. However, they are currently used mostly for manual web navigation which is unsuitable for evaluating large sets of objects, e.g. for gene expression analysis. Moreover, the semantics of the cross-references is typically not made explicit making it difficult for the user to find and correctly use all relevant sources and mappings for a given application task.

iFuice (information Fusion utilizing instance correspondences and peer mappings) [Ra05] is a recently proposed approach for peer-to-peer data integration. It utilizes mappings, e.g. sets of cross-references, to combine or fuse information from different sources. Sources and mappings are related to a domain model to support semantically meaningful information fusion. The iFuice architecture incorporates a mapping mediator offering both interactive and script-driven, workflow-like access to the sources and their mappings. The script programmer can use powerful generic operators to execute and manipulate mappings and their results. iFuice is a generic data integration approach which is not targeted for a specific application domain. An initial use case of iFuice was to combine bibliographic data for a citation analysis of database publications [Ra05, RT05].

In this paper we describe how iFuice and its extension BioFuice can be used for data integration in bioinformatics applications. Key characteristics of BioFuice include:

- *Peer-to-peer integration:* By following the iFuice paradigm BioFuice aims at utilizing instance-level cross-references which already exist, e.g. as web links, or can be generated by bioinformatics tools, such as BLAST. New sources can be dynamically integrated as needed by mapping the new source to (at least) one already integrated source.
- *Semantic integration:* To address semantic integration, BioFuice utilizes a high-level domain model containing domain-specific object types and mapping types. The domain model is used to categorize specific sources and mappings so that they can be selected and accessed according to current application requirements.
- *Comprehensive query capabilities:* BioFuice utilizes the high-level operators and scripting facility of iFuice to perform data access, mapping execution and data fusion. This infrastructure makes it possible to react to new application needs and to support complex data integration and analysis workflows. BioFuice substantially extends the generic iFuice facilities by providing a graphical query interface for explorative analysis and automatically generating script programs from interactively specified queries. Both predefined queries as well as keyword searches are supported.
- *Local data sources:* BioFuice integrates both public and local (private) data sources. In particular, query and script results or copies of entire sources may be stored within a local database for later reuse. BioFuice can also be operated in an offline mode (e.g. on a notebook) by only evaluating local data sources.

The rest of the paper is organized as follows. In the next section we introduce the basic idea of the BioFuice approach by using an illustrating scenario. We also outline selected high-level operators and their usage. In Section 3, we introduce the BioFuice

system architecture. Section 4 describes the interactive query and search capabilities for explorative analysis. We discuss related work in Section 5 before we conclude in Section 6.

2 Illustrating Scenario

To illustrate our data integration approach we consider an analysis task on human expressed sequence tag (EST) sequences. Typically, such ESTs are short DNA sequences of a specific organism that are generated by sequence machines. We assume that the analysis application needs to classify EST sequences into three classes which are represented by the following queries:

Query 1 (Q1): Return all unaligned EST sequences of a given set, i.e. EST sequences for which no corresponding DNA sequence can be found.

Query 2 (Q2): Return all EST sequences of a given set that are associated with protein-coding DNA sequences.

Query 3 (Q3): Return all EST sequences of a given set that are associated with non-coding DNA sequences.

Such a classification is a typical data integration problem since the given set of EST sequences has to be combined with further molecular-biological data on genes and proteins to decide which sequences fall into which class.

Figure 1a shows four (physical) sources and associated mappings we want to use for this scenario within a so-called s*ource mapping model (SMM)*. There are three public data sources (Ensembl [Bi04], NetAffx [Li03], SwissProt [Bo03]) and one local data source, MyEstSet, specifying the EST sequences of interest. A *physical data source (PDS)*, e.g. Ensembl, may offer objects of different types. We call the object types of one PDS the *logical data sources (LDS)*. The notation <ObjectType>@<PhysicalDataSource> is used to denote a specific LDS, e.g. Gene@Ensembl or Protein@SwissProt. Each LDS has an identifying attribute (accession) plus additional attributes.

Object types of any source are represented in the abstract domain model (Figure 1b). Each mapping between source instances has a mapping type which is also represented in the domain model. For example, mappings of type GeneCodedProteins relate gene instances to their associated proteins. The domain model is used to semantically categorize data sources and mappings and at a much higher conceptual level than a global schema. We do not include attributes for object types to accommodate a large variety of data sources and to make it easy to construct the domain model. In many cases, we expect a small set of object and mapping types to be sufficient.

New sources can be flexibly included by adopting relevant metadata, i.e. the physical source name and its associated object type (building the new LDS) as well as definitions for querying and searching within the source. In addition, at least one mapping should be defined to connect the new LDS to an existing LDS so that the new LDS can be used together with others. The domain model is automatically adapted to the changes made within the source mapping model.

Mappings can often be represented by sets of cross-references between objects/instances of different LDS. For instance, the mapping between Gene@Ensembl and Protein@SwissProt can be derived from the existing SwissProt references in the Ensembl gene instances. Alternatively, mappings can be derived on demand by executing queries or a program (script). In our example, the private source MyEstSet contains ESTs that are only described by a sequence. Hence, neither the PDS MyEstSet nor the public PDS Ensembl provide correspondences between the instances they offer. In this case, a BLAST[1]-like tool can be used to determine for a set of EST sequences the most similar DNA sequences within Ensembl. This creates a mapping between EstSequence@MyEstSet and SequenceRegion@Ensembl thereby integrating the local source into the peer-to-peer network represented by the SMM. Special mapping types are so-called *same-mappings* interrelating semantically equivalent instances of the same object type. In Figure 1a there is one same-mapping on genes between Ensembl and NetAffx.

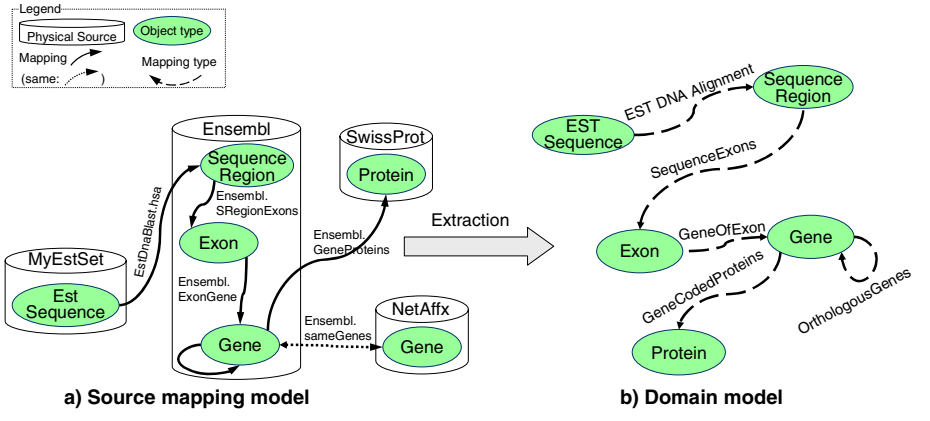

Fig. 1. Data integration scenario

To process data and mappings, iFuice and thus BioFuice offer a set of high-level operators which can be combined within script programs. Table 1 shows a selection of these operators that are relevant for the examples in this paper; the full definitions are given in [Ra05]. The operators typically operate on a set of input objects, e.g. an entire LDS, and generate a set of output objects which can be used as the input of further operators. In Table 1, OI denotes a set of object instances from one object type; objects are identified by their ids which are assumed to also identify the LDS the objects belong to.

To solve the EST classification problem posed in the beginning of this section we can use the following simple script determining three sets of EST sequences:

```
$alignedEstMR:=map(MyEstSet,{EstDnaBlast.hsa});
$unalignedEstOI:=diff(MyEstSet,domain($alinedEstMR));
$codingEstMR:=compose($alignedEstMR,
                  map(range($alignedEstMR),{Ensembl.SRegionExons}));
$proteinCodingEstOI:=(domain($codingEstMR));
$nonCodingEstOI:= diff (domain($alignedEstMR), $proteinCodingEstOI);
```

[1] BLAST stands for Basic Local Alignment Search Tool [Al90].

Table 1. Selected iFuice script operators (OI = set of object instances)

Operator	Description
OI:=queryInstances(LDS, query condition)	executes a query on the specified LDS and returns object instances (OI) meeting the query condition
OI:=searchInstances(LDS, {keywords})	executes a keyword search on the specified LDS and returns object instances containing at least one of the specified keywords
OI:=getInstances(OI)	returns complete instances for objects identified by their id values
OI:=traverse(OI,{mapping names})	traverse specified mappings on input instances; multiple mappings are automatically composed
OI:=traverseSame(OI,PDS)	traverse same mappings to target PDS
MR:=map(OI,{mapping names})	returns a mapping result MR (mapping table) that associates each input object to the corresponding output objects by executing specified mappings
OI:=domain(MR)	returns the domain (input objects) of a MR
OI:=range(MR)	returns the range (output objects) of a MR
MR:=compose(MR,MR)	composes two given mapping results
OI:=diff(OI,OI)	returns the difference set of object instances between the first and second input set
AO:=aggregateSame(OI,PDS)	fuses objects of two different PDS interrelated by a same mapping

In the first step, we associate each EST sequence of the local source MyEstSet to the associated DNA sequence regions in Ensembl by executing the *map* operator on mapping EstDnaBlast.hsa. This mapping was created by performing a BLAST search during the integration of MyEstSet. The map result is stored in variable *alignedEstMR* indicating the EST sequences which could be mapped. The set of unaligned ESTs is computed by taking the difference between all given ESTs in MyEstSet and the aligned ESTs in (the domain of) *alignedEstMR* as shown in Step 2 (answer to query Q1). To further distinguish between protein-coding and non-coding aligned EST sequences, we consider that genes typically consist of multiple intron and exon sequences. Usually, intron sequences are spliced out before the protein coding (translation) process starts. Therefore, sequence regions within introns are typically non-coding sequences. Conversely, exon sequences are highly involved in the protein coding process. In Step 3 we apply the mapping Ensembl.SRegionExons to determine the exons associated with the aligned DNA sequence regions. The domain of this composed mapping thus corresponds to the protein-coding aligned EST sequences (Step 4; answer to Q2). The set of non-coding and aligned EST sequences can be derived as the difference set between all aligned ESTs and all protein-coding and aligned EST sequences (Step 5; answer to Q3).

The example illustrates the power of the set-oriented operators for interconnecting data from different sources. The operators make it also easy to react to new analysis needs. For instance, we can associate the found protein-coding EST sequences not only to SwissProt proteins but also to genes of Ensembl and NetAffx, e.g. for microarray-based gene expression analysis. This is achieved by the following script extension.

```
$codingEstProteinMR:=compose($codingEstMR, map(range($codingEstMR),
                        {Ensembl.ExonGene,Ensembl.GeneProteins}));
$codingEstGeneOI:=traverse(range($codingEstMR),{Ensembl.ExonGene});
$fusedGeneAO:=aggregateSame($codingEstGeneOI,NetAffx);
```

The first statement returns a mapping result associating each protein-coding EST to the corresponding protein in SwissProt, the second statement determines all associated genes available in the Ensembl data source. This set of genes can further be fused with information on genes of the NetAffx source by traversing the corresponding same mapping. The fused gene information contains the attributes of both LDS, Gene@Ensembl and Gene@NetAffx.

3 BioFuice Architecture

Figure 2 gives an overview of the BioFuice system architecture. It consists of two main components, the iFuice core and the BioFuice query component. BioFuice utilizes the generic iFuice core to execute script programs and fuse data from several sources. The BioFuice query component provides interactive query functionality, supports local storage of analysis results and meets specific bioinformatics requirements, e.g. to export genomic sequences in specific formats for later use in other analysis tools. BioFuice query and iFuice core may run on the same machine or different machines. The BioFuice query component can also be run in stand-alone mode independent of iFuice, e.g. offline on a laptop. In this case analysis and query processing are restricted to local data sources. BioFuice is already in use for different applications, in particular for gene expression analysis, protein interaction analysis and analysis of non-coding RNAs.

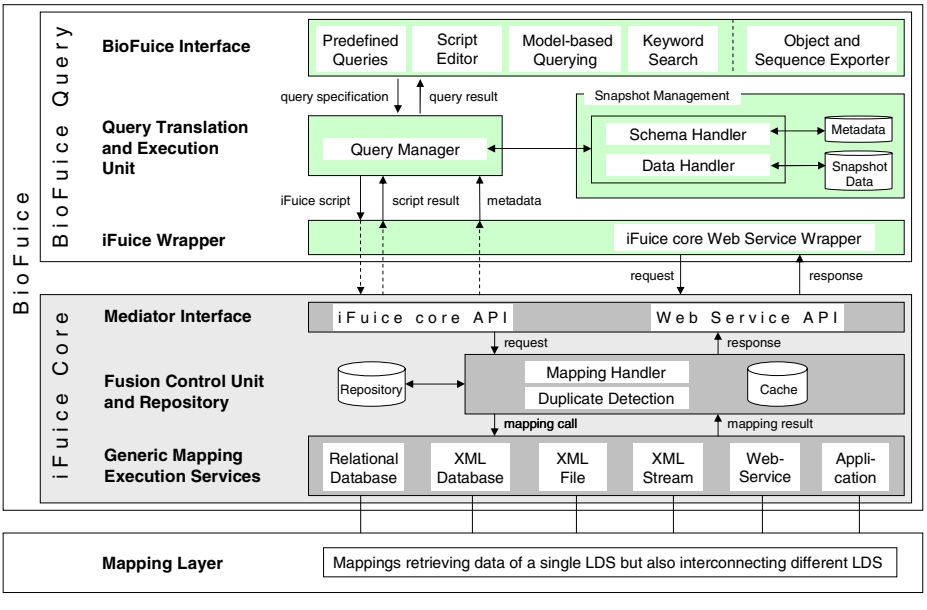

Fig. 2. BioFuice system architecture

The iFuice core consists of mapping execution services, a fusion control unit, a repository and the mediator interface. The mediator interface supports access to the iFuice functionality by a basic application interface but also by specific web service methods (so that iFuice core may run on a separate server machine). Typically, the interface is used to start a script combining multiple data and mapping operations. The mapping handler executes the script, temporarily caches the results and provides the results to the application via the mediator interface. The iFuice repository contains all metadata of the source mapping model and the domain model. In particular, it stores all LDS descriptions and mapping definitions. Each mapping definition is associated with mapping execution services that implement the specified mapping, e.g. a web service, Java application or SQL query. The spectrum of available mapping execution services supports mappings for sources of different formats, such as relational databases, XML-based sources but also application tools. The implementation of mappings by bioinformatics tools allows complex analysis and integration workflows, e.g. to perform search (blast) and data cleaning tasks.

Like the iFuice core the BioFuice query component is modularly structured. Subcomponents include a user interface, a query translation and execution unit and a local snapshot management unit. The user interface not only supports query specification but also visualization and export of query results. Query capabilities include predefined structured queries and unstructured keyword search, and are further described in the next section. The query manager translates interactively specified queries into an internal query format and automatically generates an iFuice script whenever the user decides to utilize the original (non-local) sources. Alternatively, queries may be restricted to local data, in particular materialized analysis results and copies (snapshots) of public sources. In this case, the query manager maps user queries to the corresponding statements on local sources, e.g. SQL statements for snapshots stored within a relational database.

4 Interactive Query Processing

BioFuice provides different query and search capabilities for explorative analysis and repeated execution of predefined analysis workflows. In addition to the iFuice scripting facility BioFuice supports canned queries, model-based querying, and a keyword search. Canned queries are parameterized predefined queries. Query parameters are provided by the user at runtime to specify specific query conditions.

Since predefined queries are sometimes too static on the one hand and the scripting capability could be too complex for end users on the other hand, BioFuice provides the model-based querying and keyword search. Model-based querying directly utilizes the source mapping and the domain models. Figure 3a shows the GUI for this query capability. Both models are illustrated as graphs on the GUI's left hand side (top: domain model, bottom: source mapping model). The nodes represent object types (logical sources) and edges stand for mapping types (mappings) within the domain model (source mapping model).

Users can use the GUI to select relevant sources and specify query conditions (keywords) for specific LDS on the right hand side of the interface. Furthermore, the query targets are specified, i.e. LDS for which object instances are finally retrieved.

Objects of target LDS sharing the same object type can be aggregated, i.e. attribute sets of corresponding instances from these LDS are merged. BioFuice automatically determines available mapping paths from the source mapping model connecting the selected LDS and query targets; the paths are visualized on the right hand side for user selection. Before the query is executed the user can also specify whether the original sources or the local snapshots should be used to answer the specified query.

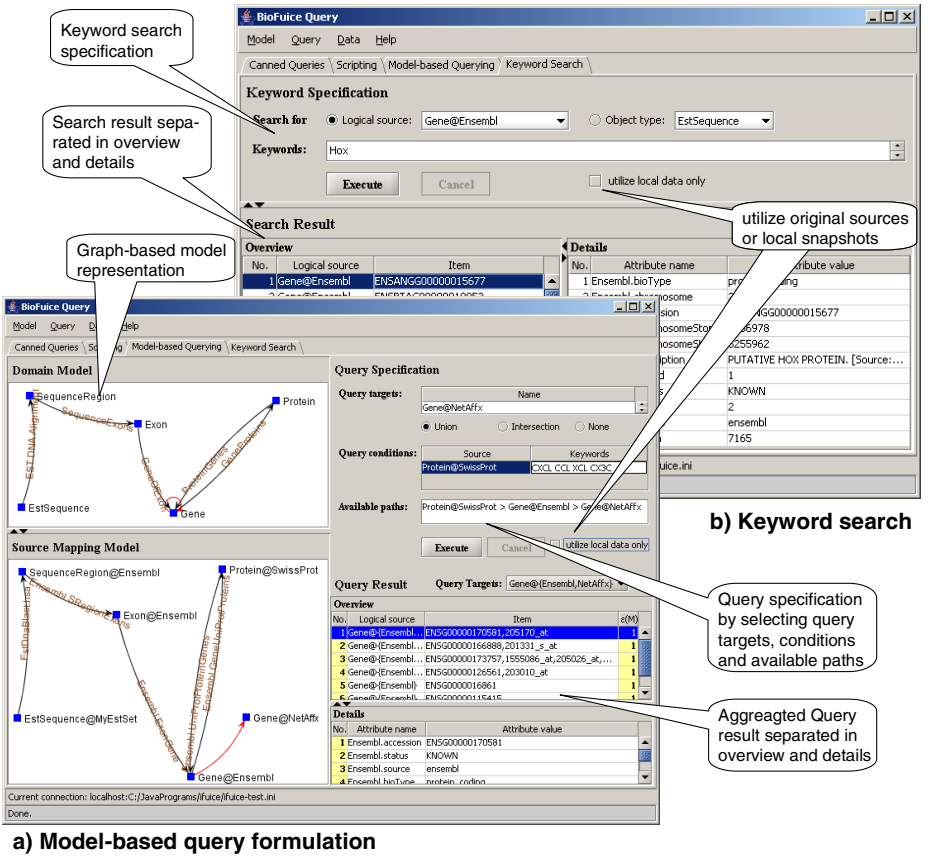

Fig. 3. Selected BioFuice query capabilities

Figure 3a illustrates such a query specification for a simple explorative analysis task. The goal is to find all genes of the source NetAffx corresponding to Chemokine proteins, i.e. special proteins that are responsible for cell-cell interactions. Based on the SMM of Figure 1, the plan is to use SwissProt to determine the relevant proteins and to traverse to the associated genes in Ensembl and then to the corresponding genes in NetAffx. BioFuice translates the interactively specified query into the following iFuice script:

```
$Proteins:=searchInstances(Protein@SwissProt,"CXCL CCL XCL CX3C");
$Genes:=traverse($Proteins,{Ensembl.ProtGenes});
$FusedGenes:=aggregateSame($Genes,NetAffx);
```

In Step 1, we utilize the LDS Protein@SwissProt to search for *Chemokine* proteins. [Ta05] provides a list of such proteins that are systematically classified in four groups *CXC*, *CC*, *XC*, and *CX3C*[2]. These group names can be used to search for relevant proteins since they are part of the protein name. In Step 2, the found proteins are associated with Ensembl genes which are fused with NetAffx genes in Step 3 as result.

In contrast to the model-based query capability, the keyword search looks either for objects of one selected LDS or of all LDS with the same chosen object type by considering the specified keywords. Figure 3b shows an example for keyword search within the GUI where the user is interested in finding relevant *Hox* genes of the Ensembl data source. This query specification is then taken by the query manager and either translated into an iFuice script program or a snapshot query in dependence of the user choice. In the first case, the query manager generates the following iFuice script which returns all object instances of the LDS containing the specified keyword.

```
$HoxGenes:=searchInstances(Gene@Ensembl,"Hox");
```

5 Related Work

Previous data integration approaches in bioinformatics [HK04, LC03, Iv05] have already made heavy use of cross-references between data sources and navigational access. In the simplest case, users have to manually navigate between sources and objects by following hypertext links, e.g. supported by a portal. Widely used systems like Entrez [Sc96] and SRS [Et03] support automatic navigational access in combination with web-based search and retrieval but are limited to sources at NCBI (Entrez) or local copies (SRS). BioFuice similarly utilizes cross-references to interrelate objects of different sources, but offers more query flexibility through its use of high-level operators. In particular, BioFuice utilizes operators fusing objects from different sources which is currently not possible with Entrez and SRS. Furthermore, BioFuice provides a semantic domain model for both sources and mappings helping to identify relevant sources and focusing the analysis on semantically meaningful mappings. Our previous integration approaches of [DR04] (GenMapper) and [KDKR05] also utilize existing cross-references but do not consider the semantics of objects and mappings. Moreover, they use a central database to store all cross-references. Like SRS, BIS [La03], and others, BioFuice is not limited to pre-existing cross-references but can also compute mappings, e.g. by queries or bioinformatics tools such as BLAST.

Many data warehouse and mediator systems utilize a global schema aiming at a consistent view over different data sources. As discussed in the introduction, such a schema is hard to create and maintain due to the large number of relevant sources and their high degree of heterogeneity. Some approaches such as ALADIN [LN05], HumMer [Bi05], AutoMed Toolkit [Ma05], and ANNODA [PC05] try to address this problem by using automatic schema matching. Other approaches simply take the union of the local schemas as the global schema. This exposes the heterogeneity and complexity to the user but still suffers from the need to change the global schema (and thus dependent mappings and applications) whenever one of the sources changes or a new source is added. In contrast to these approaches, BioFuice avoids the

[2] The additional character 'L' within the script identifies ligand proteins.

construction of a global schema but uses bidirectional peer mappings between sources. The domain model used is at a much higher abstraction level than a global schema and does not include details like the exact set of source attributes.

The BioFuice domain model can be seen as a domain-specific ontology. Ontologies provide a common understanding of a domain and thus are of interest for data integration. An overview of ontology-based approaches for data integration is presented in [Wa01]. While BioFuice currently utilizes user-defined object types as concepts and mapping types as their semantic relationships, other approaches reuse pre-defined ontologies [Me00] and focus on efficient query processing and rewriting by applying description logic [St03], different kinds of rules [NF05] and a quality model [He05]. Moreover, associating each relevant source attribute to an ontology concept need much more fine-grained ontologies than we use in BioFuice. Creating and maintaining fine-grained ontologies has similar problems than a global schema.

Peer-to-peer data management systems typically avoid the construction of a global schema. Database-oriented systems like Piazza [Ha03], PeerDB [Ng03] and Orchestra [Iv05] allow queries to be formulated on one peer and to be propagated through the system. Conversely, BioFuice can execute queries as well as mappings containing instance correspondences between sources, and can aggregate data from different sources. In contrast to Orchestra, BioFuice does not need to copy remote sources to local copies but uses the source schemas and their instances as provided.

6 Conclusions

We presented the BioFuice approach to integrate data from decentralized private and public data sources and ontologies. BioFuice follows a peer-to-peer-like data integration based on bidirectional mappings. Sources and mappings are associated with a domain model to support a semantically meaningful interoperability. BioFuice extends the generic iFuice integration platform which utilizes specific operators for data fusion and workflow-like script programs. BioFuice supports explorative data analysis and interactive query and search capabilities. BioFuice is operational and being used in different applications, such as for gene expression analysis, protein interactions analysis, and detection and analysis of non-coding RNAs.

Acknowledgements

The authors thank Andreas Thor, Nick Golovin and David Aumüller for useful discussions and their collaboration in developing the iFuice core component. We also thank the unknown reviewers for their constructive hints to improve the paper. The work is supported by the German Research Foundation, grant BIZ 1/3-1.

References

[Al90] Altschul, S. F. et al.: Basic Local Alignment Search Tool. Journal of Molecular Biology 215(3):403-10, 1990.

[Bi04] Birney, E. et al.: An Overview of Ensembl. Genome Research 14: 925-928, 2004.

[Bi05] Bilke, A. et al: Automatic Data Fusion with HumMer. Proc. 31st VLDB Conf., Demo description, 2005.

[Bo03] Boeckmann, B. et al.: The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Research 31: 365-370, 2003.

[Et03] Etzold, T. et al.: SRS: An Integration Platform for Databanks and Analysis Tools in Bioinformatics. In [LC03]: 109-145.

[DR04] Do, H.-H.; Rahm, E.: Flexible Integration of Molecular-biological Annotation Data: The GenMapper Approach. Proc. EDBT Conf., 2004.

[Ga05] Galperin, M. Y.: The Molecular Biology Database Collection: 2005 Update", Nucleic Acids Research, 33, D5-D24, 2005.

[Ha03] Halevy, A. et al.: Piazza: data management infrastructure for semantic web applications. Proc. WWW, 2003.

[He05] Heese, R. et al: Self-extending Peer Data Management. Proc. Database Systems in Business, Technology and Web (BTW), 2005.

[HK04] Hernandez, T.; Kambhampati, S.: Integration of Biological Sources: Current Systems and Challenges Ahead. SIGMOD Record 33(3), 2004.

[Iv05] Ives, Z. et al.: Orchestra: Rapid, Collaborative Sharing of Dynamic Data. Proc. of Conf. on Innovative Data Systems Research (CIDR), 2005.

[KDKR05] Kirsten, T.; Do, H.-H.; Körner, C.; Rahm, E.: Hybrid Integration of molecular-biological Annotation Data. Proc. 2nd Int. Workshop on Data Integration in the Life Sciences (DILS), 2005.

[La03] Lacroix, Z. et al.: The Biological Integration System. Proc. 5th ACM Int. Workshop on Web Information and Data Management, 2003.

[LC03] Lacroix, Z.; Critchlow T. (Eds.): Bioinformatics: Managing Scientific Data. Morgan Kaufmann, 2003.

[Li03] Liu, G. et al.: NetAffx: Affymetrix probesets and annotations. Nucleic Acids Research, 31(1): 82-86, 2003.

[LN05] Leser, U.; Naumann, F.: (Almost) Hands-Off Information Integration for the Life Sciences. Proc. 2nd Conf. on Innovative Data Systems Research (CIDR), 2005.

[Ma05] Maibaum, M. et al.: Cluster based Integration of heterogeneous biological Databases using the AutoMed Toolkit. Proc. 2nd Int. Workshop on Data Integration in the Life Sciences (DILS), 2005.

[Me00] Mena, E. et al.: Observer: An Approach fro Query processing in Global Information Systems based on Interoperation across pre-existing Ontologies. Distributed and Parallel Databases 8(2): 223-271, 2000.

[NF05] Necib, C. B.; Freytag, J.-C.: Query Processing Using Ontologies. Proc. 17th Conf. on Advanced Information Systems Engineering (CAISE), 2005.

[Ng03] Ng, W. S. et al.: PeerDB A P2P-based System for Distributed Data Sharing. Proc. 19th Int. Conf. on Data Engineering, 2003.

[PC05] Prompramote, S.; Chen, Y.P.: Annonda: Tool for integrating molecular-biological Annotation Data. Proc. 21st Int. Conf. on Data Engineering (ICDE), 2005.

[Ra05] Rahm, E. et al.: iFuice - Information Fusion utilizing Instance Correspondences and Peer Mappings. Proc. 8th Int. Workshop on the Web & Databases (WebDB), 2005.

[RT05] Rahm, E.; Thor, A.: Citation analysis of database publications. SIGMOD Record 34(4), 2005.

[Sc96] Schuler, G. D. et al.: Entrez: Molecular biology database and retrieval system. Journal of Methods in Enzymology 266:141-62, 1996.

[St03] Stevens, R. et al.: Complex Query Formulation over diverse Information Sources in TAMBIS. In [LC03], 190-224, 2003.

[Ta05] Tanaka, Toshiyuki et al.: Chemokines in tumor progression and metastasis. Cancer Science 96(6): 317-322, 2005.

[Wa01] Wache, H. et al.: Ontology-based Integration of Information - A Survey of existing Approaches. Proc. Workshop on Ontologies and Information Sharing (IJCAI), 2001.