

**POLITECNICO DI MILANO**

V Facoltà di Ingegneria  
Laurea in Ingegneria Informatica  
Dipartimento di Elettronica e Informazione



*Integration of biomolecular interaction data in a  
genomic and proteomic data warehouse*

Advisor: Prof. Marco Masseroli, Ph.D

Co-advisor: Eng. Giorgio Ghisalberti

Final Thesis by: Arif Canakoglu  
ID Number.: 709607

Academic year 2009-2010



**POLITECNICO DI MILANO**

V Facoltà di Ingegneria  
Laurea in Ingegneria Informatica  
Dipartimento di Elettronica e Informazione



*Integration of biomolecular interaction data in a  
genomic and proteomic data warehouse*

Thesis by:

\_\_\_\_\_

(Arif Canakoglu)

Advisor:

\_\_\_\_\_

(Prof. Marco Masseroli)

Academic year 2009-2010



# Acknowledgements

First of all, I wish to thank to Prof. Masseroli for his guidance and support from the initial to the final level of my thesis and also thank to Giorgio Ghisalberti for his support and help resolve my problems.

Thanks to all of my friends who are support me difficulties not only for the university life but also difficulties of living abroad, Mercan, Tugce, Ugur, Guzide, Burak, Anil my housemates Riccardo, Giulia and Giorgia, and all that I forgot to write here.

Last but not least thanks to my beloved family for their endless love and precious support to me.

Thanks also to everyone I forgot...

# Table of Contents

1	Abstract.....	1
2	Introduction .....	5
2.1	Genomic and proteomic research.....	7
2.2	Controlled vocabularies, ontologies and functional annotations.....	8
2.3	Biomolecular databanks .....	10
2.3.1	Data file formats.....	13
2.4	Difficulties in comprehensively using available biomolecular data .....	15
2.5	Genomic and Proteomic Data Warehouse (GPDW) .....	15
2.5.1	Data and metadata schema.....	16
2.5.2	Software framework for automatic creation and updating of GPDW .....	17
3	Thesis goals .....	24
4	Extension of GPDW software framework for biomolecular interaction data management.....	25
4.1	Data import procedures .....	25
4.2	Data integration procedures.....	26
4.3	Metadata computation and storing .....	26
5	Considered biomolecular interaction databanks.....	30
5.1	IntAct: an open source molecular interaction databank.....	33
5.2	MINT: the Molecular INTeraction databank .....	33
6	Design of GPDW sections for biomolecular interaction data .....	34
6.1	Conceptual schema.....	35
6.2	Logical schema.....	38
7	Implementation of automatic procedures for biomolecular interaction data import.....	41
7.1	Tabular file format.....	41
7.2	XML file format .....	44
8	Validation and testing.....	47
8.1	Quantification of imported data and running times.....	47

9 Conclusions ..... 51

10 Future developments ..... 52

11 Bibliography..... 53

## List of figures

Figure 1 Moore's Law .....	10
Figure 2 Internet domain survey host count.....	11
Figure 3 Databank growth.....	12
Figure 4 Biomolecular databanks and their correlations .....	13
Figure 5 Data file formats .....	13
Figure 6 Tabular file types .....	14
Figure 7 Structure of GPDW framework.....	17
Figure 8 Sequence diagram of the main tasks of the implemented .....	18
Figure 9 Sequence diagram of the main tasks of the implemented .....	21
Figure 10 Import level relation between same feature.....	25
Figure 11 Integrate level relation between same feature .....	26
Figure 12 Metadata schema of feature unfolding tables .....	27
Figure 13 Metadata schema of source to feature unfolding tables .....	28
Figure 14 Metadata schema if source2feature_association_display_url.....	30
Figure 15 Example of PSI MI.....	32
Figure 16 Complex expansion algorithms .....	33
Figure 17 Conceptual schema of interactions between protein and protein. ....	36
Figure 18 Conceptual schema of interactions between dna sequence and protein. ....	37
Figure 19 Logical schema of interactions between protein and protein. ....	39
Figure 20 Conceptual schema of interactions between dna sequence and protein. ....	40
Figure 21 Tabular loader workflow .....	44
Figure 22 Interaction.....	45
Figure 23 XML loader workflow.....	46



## List of tables

Table 1 Logical schema legend colors of tables .....	38
Table 2 Loader execution time.....	47
Table 3 Number of entries in interaction tables.....	48
Table 4 Number of entries in similarity tables.....	48
Table 5 Number of entries in IntAct interaction related tables.....	50
Table 6 Number of entries in MINT interaction related tables.....	50

# 1 Abstract

The growing available genomic information provides new opportunities for novel research approaches and original biomedical applications that can provide effective data management and analysis support. In fact, integration and comprehensive evaluation of available controlled data can highlight information patterns leading to unveil new biomedical knowledge.

The goal of bioinformatics is to organize databases, analyze the knowledge acquired in the genome and proteome and finally store, retrieve and monitor effectively the information available today. There are many public Web databases that allow online consultation and provide the possibility to download such information freely.

However these data, which are available and important to biologists, doctors and researchers, are very heterogeneous and distributed. Therefore it is needed a tool that overcomes cross-search problem on various data-sources and returns the information which is not possible to get from individual data sources.

For this purpose, in the University of Politecnico di Milano, a project, Genomic and Proteomic Data Warehouse (GPDW), is creating a data warehouse that integrates information from many sources of genomic and proteomic data on the basis of a conceptual framework that relates molecular entities and biomedical characteristics (features).

The primary goal of this Thesis is to develop an extension for biomolecular interaction data on the GPDW framework and the second goal is to implement the integration of such type of data, from two considered databanks, to the GPDW project, by using the framework extension implemented as first goal of the Thesis.

After this abstract, chapter 2, Introduction, of this Thesis is about the conceptual meaning of bioinformatics, with the origins and the historical development of the discipline and the tasks and goals of the discipline. Then information about genomic and proteomic fields is briefly explained. Next the chapter talks about controlled vocabularies, ontologies and functional annotations and their usage in bioinformatics. In this first part, the main concepts of bioinformatics are given. Thereafter, it is introduced another important part of the bioinformatics, regarding biomolecular databanks and data, and the current difficulties of

using such biomolecular data. This chapter continues giving information about the GPDW project and its current status.

In chapter 3, the goals of the Thesis are discussed.

In chapter 4, the extension of the GPDW project for integrating biomolecular interaction data and the necessary development to be achieved for this extension are discussed. This is explained in the chapter subsections as data importing procedures, for importing data from external databanks, and data integration procedures, which is about integration of the data into the data warehouse and metadata computation and storing, which is as about storing the metadata of the data warehouse into the metadata schema of the database.

In chapter 5, the databanks considered in this Thesis are presented by explaining some general, historical and statistical information; information about the provided types of data and files are also given.

In chapter 6, the design of the integration of the considered databank data is described. This mainly generated entity relationship diagrams and logical diagrams of the considered databank data.

Chapter 7 describes the software architecture and methodologies implemented for the automatic import of data from the databank provided data files, the contents of those files, and the design choices and strategies adopted to achieve a correct and consistent import.

Chapter 8 shows some quantitative results related to the imported data and the time taken to import them.

In chapter 9, the conclusions, which confirm the legitimacy of the design choices and activities undertaken to achieve the objectives, are discussed.

Chapter 10 includes references to books, scientific articles and web sites referred in the elaboration of this Thesis.

## Sommario

Le crescenti informazioni genomiche disponibili offrono nuove opportunità per nuovi approcci di ricerca e originali applicazioni biomediche, in grado di fornire un'efficace gestione dei dati e il supporto all'analisi. Infatti, l'integrazione e la valutazione globale dei dati controllati disponibili può evidenziare modelli di informazione che permettono di svelare nuove conoscenze biomediche.

L'obiettivo della bioinformatica è quello di organizzare banche dati, analizzare le conoscenze acquisite del genoma e del proteoma e, infine, archiviare, recuperare e controllare in modo efficace le informazioni attualmente disponibili. Esistono molte banche dati Web pubbliche che consentono la consultazione on-line e forniscono la possibilità di scaricare liberamente tali informazioni.

Tuttavia questi dati, che sono disponibili ed importanti per biologi, medici e ricercatori, sono molto eterogenei e distribuiti. Pertanto è necessario disporre di uno strumento che permetta di superare i problemi di ricerca tra le varie fonti di dati ed in grado di restituire le informazioni che non è possibile ottenere dalla consultazione di singole fonti dati. A tal fine, presso il Politecnico di Milano, è stato creato un progetto denominato Genomic and Proteomic Data Warehouse (GPDW): si tratta di un data warehouse che integra le informazioni provenienti da molte fonti dati genomiche e proteomiche, sulla base di un quadro concettuale che si riferisce ad entità molecolari e caratteristiche biomediche (funzionalità). L'obiettivo primario di questa tesi è quello di sviluppare un'estensione per l'interazione biomolecolare di dati nel framework GPDW, mentre il secondo obiettivo è quello di implementare l'integrazione di questo tipo di dati, provenienti da due banche dati considerate utilizzando l'estensione implementata come primo obiettivo della tesi.

Dopo questo riassunto, nel capitolo 2, Introduzione riguarda il significato concettuale della bioinformatica, le origini e lo sviluppo storico della disciplina, nonché i compiti e gli obiettivi. In seguito vengono brevemente fornite informazioni sulla genomica e proteomica. Il capitolo successivo tratta i vocabolari controllati, le ontologie e le annotazioni funzionali ed il loro utilizzo nella bioinformatica. In questa prima parte, vengono spiegati i concetti principali della bioinformatica. Successivamente, è introdotto un altro elemento importante della bioinformatica, riguardante le banche dati biomolecolari e le attuali difficoltà di utilizzare tali

informazioni. Questo capitolo continua fornendo informazioni sul progetto del GPDW ed il suo stato attuale.

Nel capitolo 3 sono discussi gli obiettivi della Tesi.

Nel capitolo 4 sono trattati l'estensione del progetto GPDW per l'integrazione di dati di interazione biomolecolare e lo sviluppo raggiunto per questa estensione. Questo aspetto è spiegato nelle sottosezioni del capitolo, tra cui le procedure di importazione di dati da banche dati esterne, le procedure di integrazione dei dati che riguardano l'integrazione dei dati nel data warehouse, e la memorizzazione e creazione dei metadati, che riguardano la conservazione dei metadati del data warehouse nello schema dei metadati del database.

Nel capitolo 5 sono presentate le banche dati considerate in questa tesi, spiegando alcune informazioni generali, storiche e statistiche, come anche informazioni sui tipi di dato e i tipi di file forniti.

Nel capitolo 6 è descritto il progetto di integrazione delle banca dati considerate. Sono stati realizzati diagrammi entità-relazione e diagrammi logici delle banca dati prese in considerazione.

Il capitolo 7 descrive l'architettura software e le metodologie attuate per l'importazione automatica dei dati dai file forniti dalle banche dati, il contenuto di tali file, e le scelte progettuali e le strategie adottate per raggiungere una corretta e consistente importazione.

Il capitolo 8 mostra alcuni risultati quantitativi relativi ai dati importati e il tempo necessario per importarli.

Nel capitolo 9 vengono discusse le conclusioni, che confermano la legittimità delle scelte progettuali e le attività intraprese per raggiungere gli obiettivi.

Il capitolo 10 include riferimenti a libri, articoli scientifici e siti web a cui si fa riferimento nell'elaboratodi questa tesi.

## 2 Introduction

Bioinformatics (1) is the application of statistics and computer science to the field of molecular biology.

Bioinformatics and computational biology involve the use of techniques including applied mathematics, informatics, statistics, computer science, artificial intelligence, chemistry, and biochemistry to solve biological problems usually on the molecular level.

In general, the aims of bioinformatics are three-fold. First, at its simplest bioinformatics organizes data in a way that allows researchers to access existing information and to submit new entries as they are produced. While data-curation is an essential task, the information stored in these databases is essentially useless until analyzed. Thus the purpose of bioinformatics extends much further. The second aim is to develop tools and resources that aid in the analysis of data. Development of such resources dictates expertise in computational theory, as well as a thorough understanding of biology. The third aim is to use these tools to analyze the data and interpret the results in a biologically meaningful manner. Traditionally, biological studies examined individual systems in detail, and frequently compared to them with a few that are related. In bioinformatics, we can now conduct global analyses of all the available data with the aim of uncovering common principles that apply across many systems and highlight novel features.

Over the past few decades, major advances in the field of molecular biology, coupled with advances in genomic technologies, have led to an explosive growth in the biological information generated by the scientific community. This deluge of genomic information has, in turn, led to an absolute requirement for computerized databases to store, organize, and index the data and for specialized tools to view and analyze the data.

A biological database is a large, organized body of persistent data, usually associated with computerized software designed to update, query, and retrieve components of the data stored within the system. A simple database might be a single file containing many records, each of which includes the same set of information. For example, a record associated with a nucleotide sequence database typically contains information such as contact name; the input sequence with a description of the type of molecule; the scientific name of the source organism from which it was isolated; and, often, literature citations associated with the sequence.

Thus bioinformatics is the field of science in which biology, computer science, and information technology merge to form a single discipline. The ultimate goal of the field is to enable the discovery of new biological insights as well as to create a global perspective from which unifying principles in biology can be discerned. At the beginning of the "genomic revolution", a bioinformatics concern was the creation and maintenance of a database to store biological information, such as nucleotide and amino acid sequences. Development of this type of database involved not only design issues but the development of complex interfaces whereby researchers could both access existing data as well as submit new or revised data.

In order to study how normal cellular activities are altered in different disease states, the biological data must be combined to form a comprehensive picture of these activities. Therefore, the field of bioinformatics has evolved such that the most pressing task now involves the analysis and interpretation of various types of data, including nucleotide and amino acid sequences, protein domains, and protein structures. The actual process of analyzing and interpreting data is referred to as computational biology. Important sub-disciplines within bioinformatics and computational biology include:

- the development and implementation of tools that enable efficient access to, and use and management of, various types of information.
- the development of new algorithms (mathematical formulas) and statistics with which to assess relationships among members of large data sets, such as methods to locate a gene within a sequence, predict protein structure and/or function, and cluster protein sequences into families of related sequences.

The first challenge facing the bioinformatics community today is the intelligent and efficient storage of this mass of data. It is then their responsibility to provide easy and reliable access to this data. The data itself is meaningless before analysis and the sheer volume present makes it impossible for even a trained biologist to begin to interpret it manually. Therefore, incisive computer tools must be developed to allow the extraction of meaningful biological information.

There are three central biological processes around which bioinformatics tools must be developed:

- DNA sequence determines protein sequence
- Protein sequence determines protein structure
- Protein structure determines protein function

The integration of information learned about these key biological processes should allow us to achieve the long term goal of the complete understanding of the biology of organisms.

## 2.1 Genomic and proteomic research

Genomics is a discipline in genetics concerning the study of the genomes of organisms. In particular, it focuses on the structure, content, function and evolution of genomes. It is based on bioinformatics in order to process and display enormous amount of data.

The next step in relation to genomics has led to the identification of a new discipline called proteomics. The term proteome defines the entire protein complement in a given cell, tissue or organism. In its wider sense, proteomics research also assesses protein activities, modifications and localization, and interactions of proteins in complexes. It is very much a technology-driven enterprise, and this collection of reviews reflects the progress made and future developments needed to identify proteins and protein complexes in biological samples comprehensively and quantitatively with both high sensitivity and fidelity. The term "proteomics" was first coined in 1997(2) to make an analogy with genomics, the study of the genes. The word "proteome" is a blend of "protein" and "genome", and was coined by Marc Wilkins in 1994 while working on the concept as a PhD student.

The area of genomic and proteomic research concerns activities based on molecular research of transcripts and proteins expressed in a cellular compartment. This area is very multidisciplinary and requires the integration of biochemical, bioanalytic, bioinformatics and biomolecular knowledge. Unlike the genome, the proteome is a far more complex and dynamic system, which undergoes radical changes both in ontogeny and in various physiological and pathological states. The proteome of each cellular type of an organism is its unique character. Only a small part of potentially active genome is transcribed and translated in a certain type of cells, with the number RNA copies which does not strictly correlate to the number of proteins synthesized in these cells. Other factors determining the proteome variability include mRNA editing, alternative splicing, co- and posttranslational modification, and processing. It should be noted that the "DNA => RNA => protein => phenotype" concept could be applied with confidence only in the case of monogenic diseases. However, even in such a case phenotypes are substantially modulated by many factors. Most of the proteins display functional activity, and qualitative (synthesis of new proteins, posttranslational modifications) and quantitative (differences in expression level, encoded expression)



proteome analysis could provide valuable information on the dynamics of genome expression in varying conditions.

## 2.2 Controlled vocabularies, ontologies and functional annotations

Controlled vocabularies (3) provide a way to organize knowledge for subsequent retrieval in an orderly, readable and preferably unique. They can be indexed for using simply and quickly by computer. A controlled vocabulary consists of a set of selected terms and they are approved by the designer, in contrast to natural language vocabularies, where there is no restriction on the vocabulary. Controlled vocabularies solve the problems of homographs, synonyms and polysemes by a bijection between concepts and authorized terms. In short, controlled vocabularies reduce ambiguity inherent in normal human languages where the same concept can be given different names and ensure consistency.

Each term in this controlled vocabulary is uniquely identified by an alphanumeric code that represents a particular concept or feature.

In the context of genomic annotation, there are a lots of controlled vocabularies that can be divided into two main categories:

- flat controlled vocabulary or terminology: the entities in the vocabulary are not structured or linked through relationships;
- structured controlled vocabularies or ontologies: the words in the vocabulary are structured hierarchically, from most generic (root) to the most specific (leaf) through relationships characterized by a particular semantic meaning.

In continuation of the current thesis will tend to limit the terms such as gene or protein, when the topics will be valid for both objects, so you will prefer the expression biomolecular entities to include both terms in a single subject.

A controlled vocabulary provides a methodology for organizing knowledge in an orderly, readable and preferably so unique that it can be used simply and quickly by a computer. A controlled vocabulary consists of a set of terms selected and approved by the designer of the same,

this is in contrast with natural language as in the latter there are no restrictions on the vocabulary used and is therefore made possible the presence of homographies, synonyms and polysemy. These factors, which in fact would increase the ambiguity of information, are absent, or nearly so, in controlled vocabularies.

Controlled vocabularies provide a way to organize knowledge for subsequent retrieval in an orderly, readable and preferably unique. They can be indexed for using simply and quickly by computer. A controlled vocabulary consists of a set of selected terms and they are approved by the designer, in contrast to natural language vocabularies, where there is no restriction on the vocabulary. Controlled vocabularies solve the problems of homographs, synonyms and polysemes by a bijection between concepts and authorized terms. In short, controlled vocabularies reduce ambiguity inherent in normal human languages where the same concept can be given different names and ensure consistency.

Each term in this controlled vocabulary is uniquely identified by an alphanumeric code that represents a particular concept or feature.

In the context of genomic annotation, there are lots of controlled vocabularies that can be divided into two main categories:

- flat controlled vocabulary or terminology: the words in the vocabulary are not structured or linked through relationships;
- structured controlled vocabularies or ontologies: the words in the vocabulary are structured hierarchically, from most generic (root) to the most specific (leaf) through relationships characterized by a particular semantic meaning.

Ontology (4) is a formal representation of knowledge as a set of concepts within a domain, and the relationships between those concepts. It is used to reason about the entities within that domain, and may be used to describe the domain.

Some biomedical ontologies are:

- ChEBI (Chemical Entities of Biological Interest): dictionary and ontological classification of molecular entities focused on 'small' chemical compounds
- Gene Ontology (GO): controlled vocabularies for molecular functions, biological processes and cellular components of gene products
- Systems Biology Ontology (SBO): controlled vocabularies and ontologies for systems biology, especially in the context of computational modeling

- NCBI taxonomy: controlled vocabulary and classification of organisms
- MI ontology: (5)and it is an OBO ontology which is defined below:

Open Biomedical Ontologies (abbreviated OBO; formerly Open Biological Ontologies) is an effort to create controlled vocabularies for shared use across different biological and medical domains. As of 2006, OBO forms part of the resources of the U.S. National Center for Biomedical Ontology, where it will form a central element of the NCBO's BioPortal.

## 2.3 Biomolecular databanks

Since the building of the first computers, and especially in the last fifteen years the power of computers has experienced an exponential growth, in agreement with Moore's Law which states: "The performance of the processors and the number of transistors on it are doubling almost every 2 years.", as seen from the graph in Figure 1.

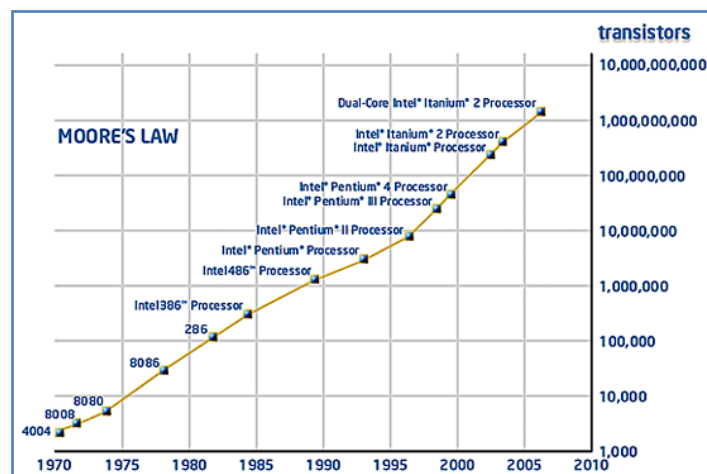


Figure 1 Moore's Law

In parallel to this development and consequently also Internet has evolved in a manner equally sudden (Figure 2). The great growth of the Internet has contributed to the development of bioinformatics.

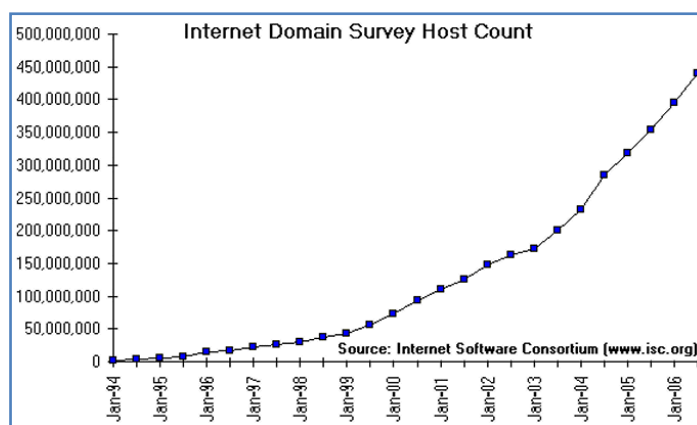


Figure 2 Internet domain survey host count

The biological databases contain wide spectrum data fields of molecular biology.

It is essential that these databases are easily accessible and that it provided an intuitive system of questions to enable researchers to obtain specific information about a particular biological argument.

They were created specialized databases for particular subjects such as:

- Databases of scientific literature;
- Databases of taxonomy;
- Databases of nucleotides;
- genomic databases;
- Databases of protein;
- Banks microarray data.

As shown in Figure 3, the databases have been a large increase in recent years:

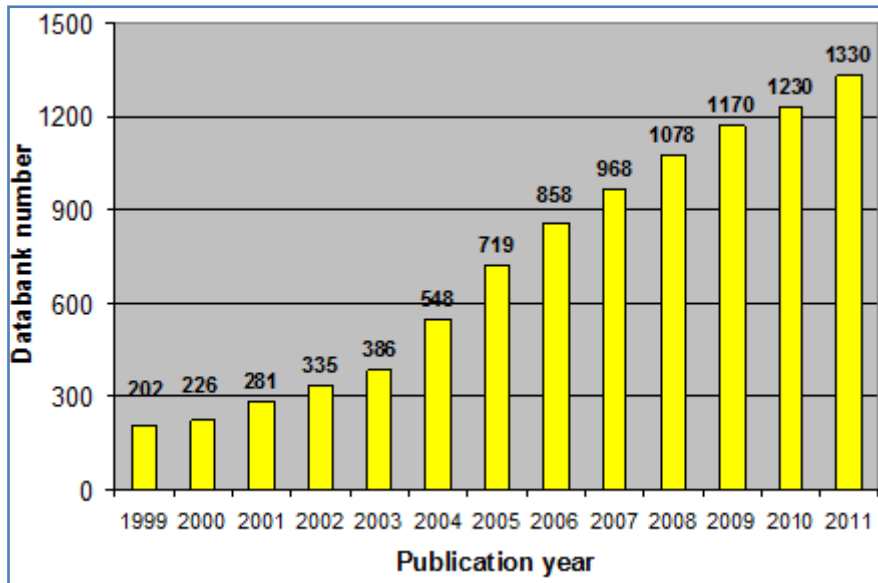


Figure 3 Databank growth

Different databases provide different access methods:

- Access via the Web interfaces (HTML or XML): The information is unstructured data and they are heterogeneous interfaces, and query results are for single sequence and so it requires much time for answers.
- Access via Web Services: This service is available for a few databases with a limited number of items and it is for people who have some computer skills.
- Access via FTP servers: it is the necessary re-implementation of the database locally.
- Direct access: this is rarely used for safety issues and it shows the lack of a common vocabulary.

Figure 4 shows the largest databases on the Web.

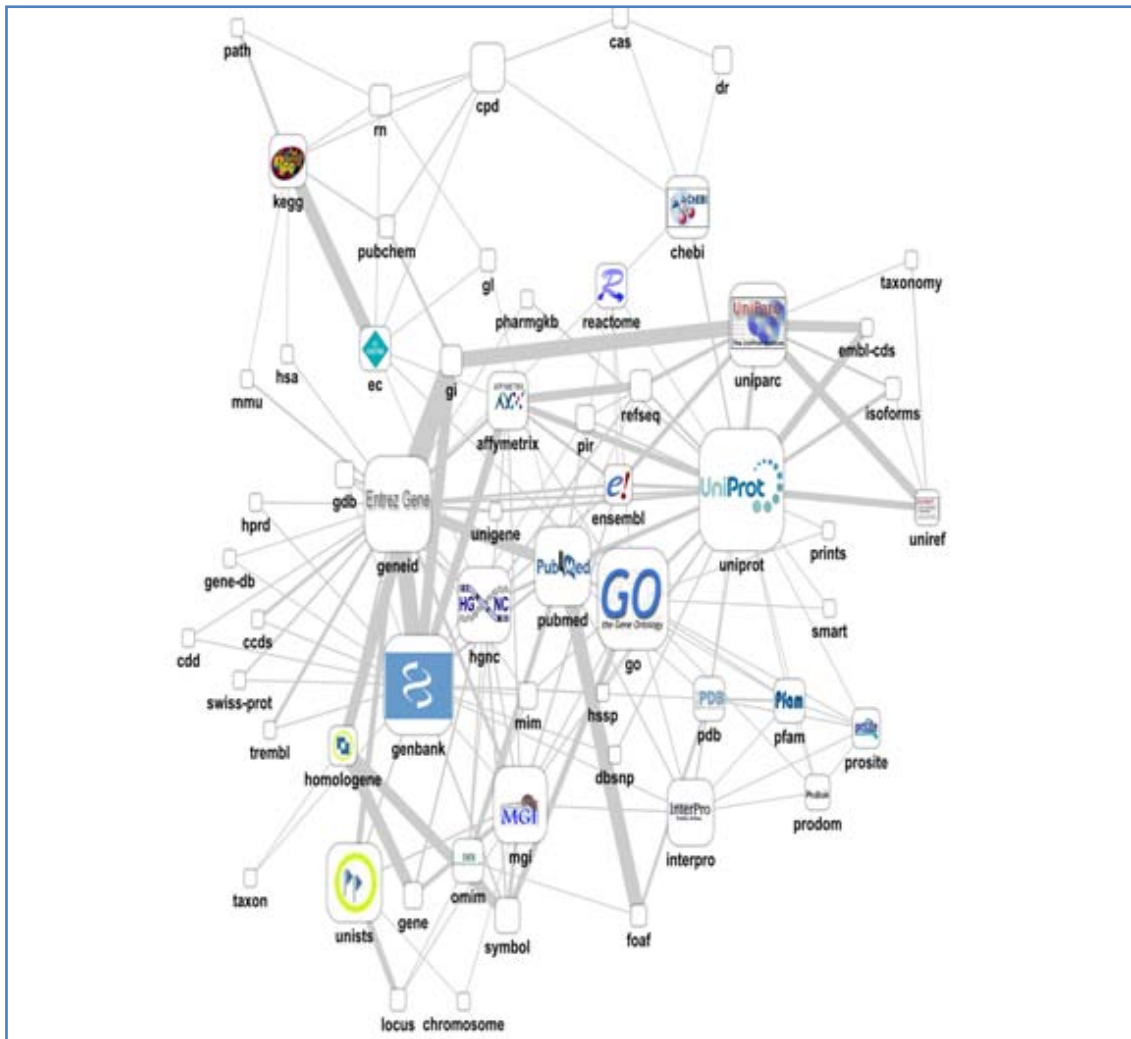


Figure 4 Biomolecular databanks and their correlations

### 2.3.1 Data file formats

The databases can provide the same data in different formats. There is no common standard for the file type and format which represent information. Genomic data are usually provided with different types of files, as shown Figure 5.

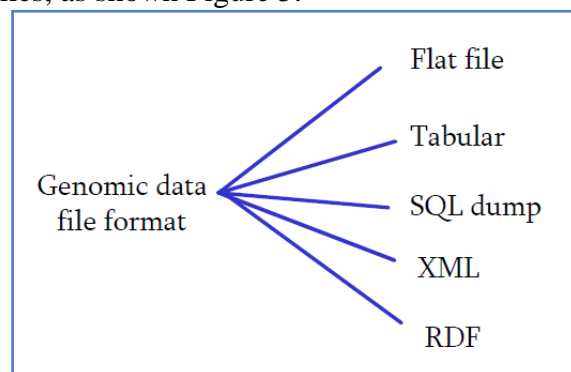


Figure 5 Data file formats

The flat file type is defined as a structured text file containing values and relations of these values. In the field of genomic data, a flat file is a text file in which each row has a different semantics and the semantics are defined by the label which is inserted at the beginning of the line. It is possible that in the same line there can be two or more labels where the successive labels specify the semantics of the previous ones, and if the beginning of a line is not indicated by any particular label, the line inherits the semantics of the previous line.

In tabular text format, the data are organized into rows and columns separated by one or more separators. In this case, the semantic value of a given position depends on its row and column. Figure 6 shows the different types of existing tabular file.

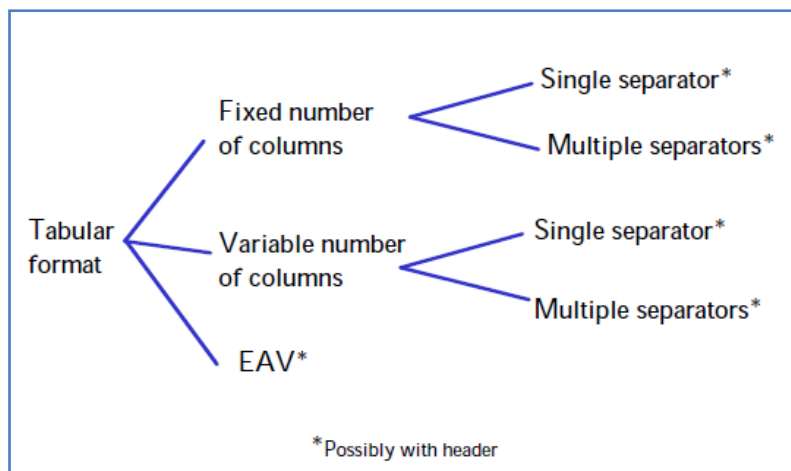


Figure 6 Tabular file types

The headings in this type of file help to understand the contents of the file.

Some databanks provide their data as SQL dump file: DBMS specific and different DBMS or earlier versions of the same DBMS cannot import them properly.

XML or eXtensible Markup Language, a meta-language is created and managed by the World Wide Web Consortium (W3C) is used to create new language to describe structured documents and acts as a means of exporting data from heterogeneous sources. Because of these characteristics many databases provide their data in XML format. The structure of the instance of an XML document can be described by the Document Type Definition (DTD) or XML-Schema.

Some banks provide genomic data in standard XML but some in RDF.

The Resource Description Framework (RDF) is the basis proposed by W3C for the encoding, exchange and reuse of structured metadata provides interoperability between applications that exchange information over Web

## **2.4 Difficulties in comprehensively using available biomolecular data**

First, there are geographically distributed databases which can include duplicate data. In recent year it is working on the integration of these data into one central database. For this purpose GPDW and other data integration systems can manage and maintain the content. There is, first, the problem of managing the large amount of heterogeneous data to integrate and to offer the users as a system as homogeneous as possible.

Secondly there is the issue concerning the controlled vocabulary, despite claiming to be as orthogonal to each other, there are relationships between a controlled vocabulary set and another, if these vocabularies are managed by different organizations.

Finally, there are problems concerning the goodness of annotations in databases, the difficulties faced by the curators to validate new records and the difficulty in managing and maintaining such records.

This thesis tries to give an answer, efficiently and effectively, these issues in the development of data warehouse developed in the laboratories of Politecnico di Milano.

## **2.5 Genomic and Proteomic Data Warehouse (GPDW)**

To effectively take advantage of the numerous genomic and proteomic information sparsely available in many heterogeneous and distributed biomolecular databases accessible via the Internet, we previously developed the Genome Function INtegrated Discoverer (GFINDER) project (<http://www.bioinformatics.polimi.it/GFINDER/>) (6,7). GFINDER is a publicly available Web system used by several thousand worldwide scientists (we counted about 110,000 accesses from more than 6,000 distinct IP addresses in the last 5 years). GFINDER supports comprehensive statistical enrichment analysis and data mining of functional and phenotypic annotations of large-scale lists of user-classified genes, such as those identified by high-throughput biomolecular experiments. It automatically retrieves annotations of several functional and phenotypic categories from different sources, identifies the categories enriched



in each class of a user-classified gene list and calculates statistical significance values for each category. Moreover, GFINDER enables the functional classification of genes according to mined functional categories and the statistical analysis of the classifications obtained, aiding better interpretation of high-throughput experiment results.

GFINDER is based on a multi-organism genomic and proteomic data warehouse (GPDW). In the GPDW several controlled terminologies and ontologies, which describe gene and gene product related biomolecular processes, functions and phenotypes, are imported and stored together with their associations (annotations) with genes and proteins of several organisms. In the GPDW all such data from several different databases are integrated by interconnecting the imported annotations to the genes and proteins they refer to by means of their provided IDs and cross-references.

### 2.5.1 Data and metadata schema

The GPDW has four main schemas:

- *metadata*: where are stored information about the metadata, as for example name of the data sources and their features ,etc.
- *public*: where are stored all the different types of data of imported and integrated data
- *flag*: where are stored information about encoded fields (i.e. fields that act as labels for other fields (data fields) in a data record)
- *log*: where are stored data that are used by developers for the testing

The **flags schema** describes all the information that regards encoded flag fields. Indeed in the public schema tables or log schema tables, which are correlated to public schema, we have some data that have been encoded, in the flag schema we want to store the mapping between the encoded value and the name of the flag.

The **log schema** describes all the data has errors in the public schema tables. This schema contains the duplicated or erroneous data from data sources.

In **metadata schema**, they are described the data contained in the public database: name of main tables, name of encoded fields, name of all source and features, which feature are provided by each source, which feature/source relations or annotations are provided by each source, which source are imported and which are synthesized, which source/feature is

ontological, which sources/features have history or similarity data and in which tables they are stored, and many other information

In **public schema**, there are all the tables that contain all the imported and integrated data in the GPDW.

In this schema there are different types of tables, grouped by two macro-areas: imported tables and integrated tables. Integrated tables are generated after all the import procedure is finished.

## 2.5.2 Software framework for automatic creation and updating of GPDW

GPDW framework handles the import and integration process, starting from the creation of a database that contains the metadata necessary for the operation of the application.

The method for integration of heterogeneous data collected from the web is divided in two macro-phases:

- import data from different sources
- integration of imported data.

It is performing these operations automatically by the software architecture whose main components are identified in Figure 7.

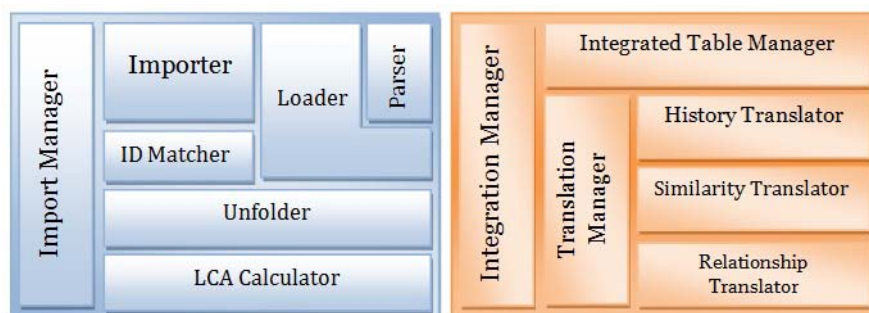


Figure 7 Structure of GPDW framework

### 2.5.2.1 Data import procedures

The data import operations require execution of the following standard processes and in particular requiring the registration of the data source definition file *data\_source.xml*.

There are four main components of import procedures:

- ImportManager;
- Importer;
- Parser;
- Loader.

The main tasks performed by the implemented automatic procedure in the data import step of the integration factory are illustrated in the sequence diagram in Figure 8.

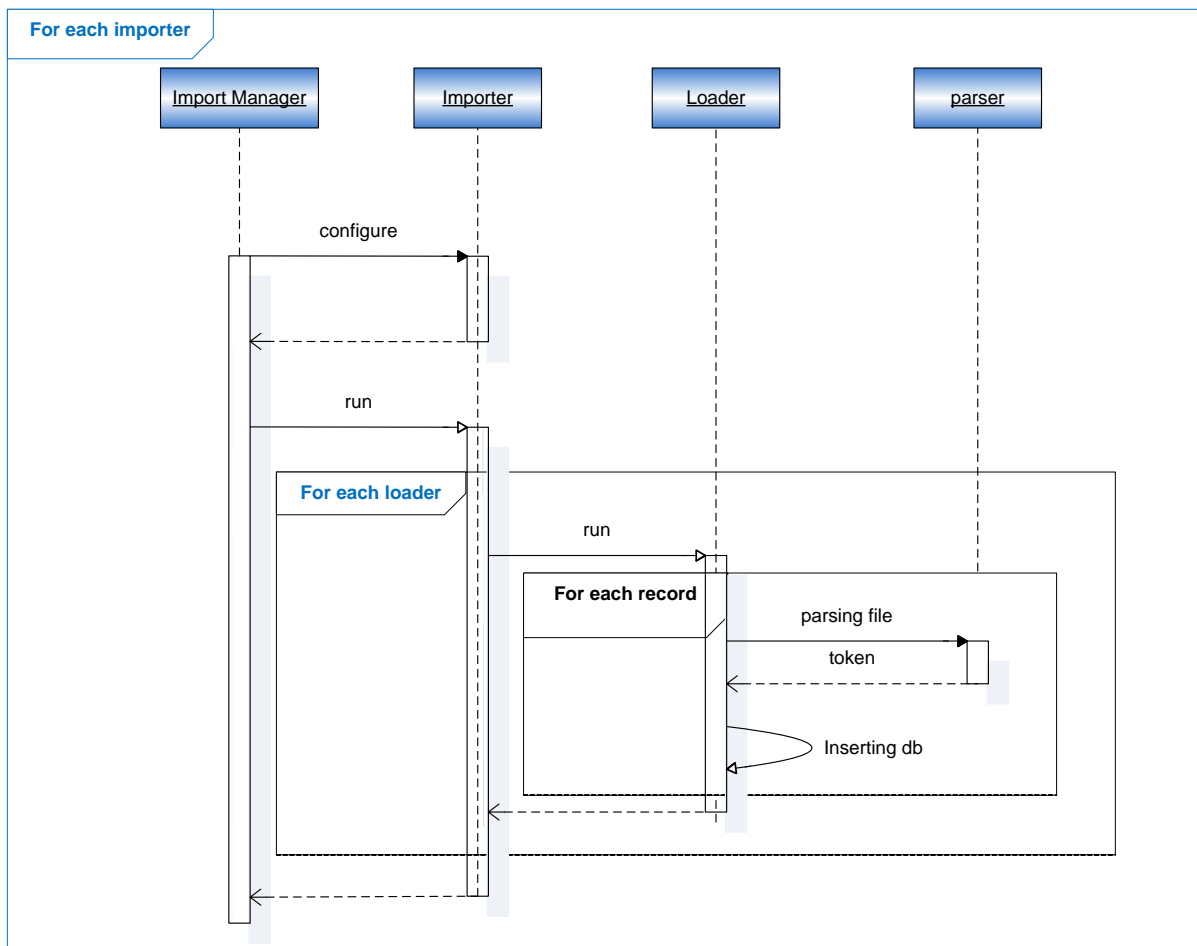


Figure 8 Sequence diagram of the main tasks of the implemented

The procedure is guided by an Import manager that configures an Importer for each considered data source and starts the procedure. Since each importer is designed to be self contained, data from many different sources can be imported in parallel to speed up the process. Each source specific importer coordinates a set of Loaders (a loader for each data file, or group of homogeneous data files, taken from the source) and a set of Parsers (a parser for each data file format). Our framework provides a number of parsers: some standard parsers for the most common data formats and ad hoc designed ones for some unusual formats. Each parser extracts the data from its associated input files and produces data tokens usable by the loader. Loader and parser use a producer–consumer pattern. Each loader is responsible for associating a semantic meaning to the tokens produced by the associated parser(s) and inserting them into the data warehouse.

On the imported data that describe ontology, the Import manager also performs a standard unfolding processing of the ontology DAG. Unfolding is done in order to speed up subsequent computational analysis on such data and their annotations. It is performed by inserting into the data warehouse an explicit association between each term (node) of the ontology and all its ancestors. Since the DAG may contain more than one path between a node and its ancestors, all possible paths are also recorded. After the unfolding has been completed, each node is tagged with its position in the DAG and its levels (for each possible path, the distance between the node and the root of the DAG). In addition, the Import manager computes also the Lowest Common Ancestor (LCA) [31] for each pair of nodes in the ontological DAG; this is a fundamental component for evaluating the similarity between two terms of an ontology according to the various metrics that are commonly used to analyze the similarity between genes or proteins based on their ontological annotations.

Each actor of the data import process is independent from the others: the import manager manages importers via a standard interface based on java reflection; the parser is aware of the data file format, but agnostic of the semantic of the file; the loader receives data in a standard format and inserts them in the proper data warehouse table fields. The importing is built to be very flexible and to allow adding new sources as easily as possible. To reach this objective, the process is guided by a XML configuration file that contains the list of registered data sources to be imported, the list of all the high level features (biomolecular entities or biomedical features) described by the data to be imported, and their bindings. In this file, metadata represent the characteristics of each data source in terms of features (what biomolecular entities or biomedical features are described by the source data), structure

(whether the feature data are ontological, i.e. describe a hierarchical graph), and relationships with other data sources. Each biomolecular entity and biomedical feature is then defined in term of its components (according to the data to be imported) by using two orthogonal sets of information metadata: its structure, which is later mapped to one or more tables in the data warehouse, and its bindings, which are used to populate those tables.

The importing framework assigns to each imported “data record” an OID, which is unique across the data warehouse. It is used as the primary identification of the data entries, since there is no guarantee that the IDs provided by the different sources do not conflict with each other. In order to ensure correctness of imported data, a set of rules has been defined to check and identify the IDs (see Section 5.1). Such rules are used by the ID Matcher, an additional component of our framework that acts as mediator between the loaders and the data warehouse. The main role of this mediator is to identify the type of each ID in order to insert the information in the appropriate data warehouse tables. During this process, each inserted tuple is also coupled with provenance meta-data to track its source.

Since the input data may contain errors and the structure of the input data files is subject to modifications in their subsequent versions, which are loaded by the automatic updates of the data warehouse, checking of input data is strictly enforced. Verification is done in three steps: during parsing, during data loading and at the end of the process. Both parsers and loaders are designed and coded to be robust w.r.t. changes in the structure of the input. Parsers are generally able to complete their task even when small modifications of the input format are detected (e.g., a new column in a tabular file, or a new attribute in a XML file) and they are able to detect (and point out) the modifications, even if they are unable to automatically deal with them. Loaders are decoupled from the actual format of the input, so they mainly deal with the correctness of the data extracted by the parsers. In doing so, they also call the previously described ID Matcher, which is also responsible for signaling unknown ID types in the input data. At the end of the import process, index, unique, primary and foreign key integrity constraints are defined and enforced upon the data warehouse tables. This allows detecting possible data duplications and inconsistencies, and improving subsequent access time to the imported data.

### ***2.5.2.2 Data integration procedures***

The main tasks automatically performed in the data integration step of our factory are described in the sequence diagram in Figure 9. They can be grouped into an aggregation and

an integration phase. In the former, data from the different sources imported in the previous data import step are gathered and normalized into a single representation in the instance-aggregation tier of our global data model. In the latter, data are organized into informative clusters in the concept-integration tier of the global model.

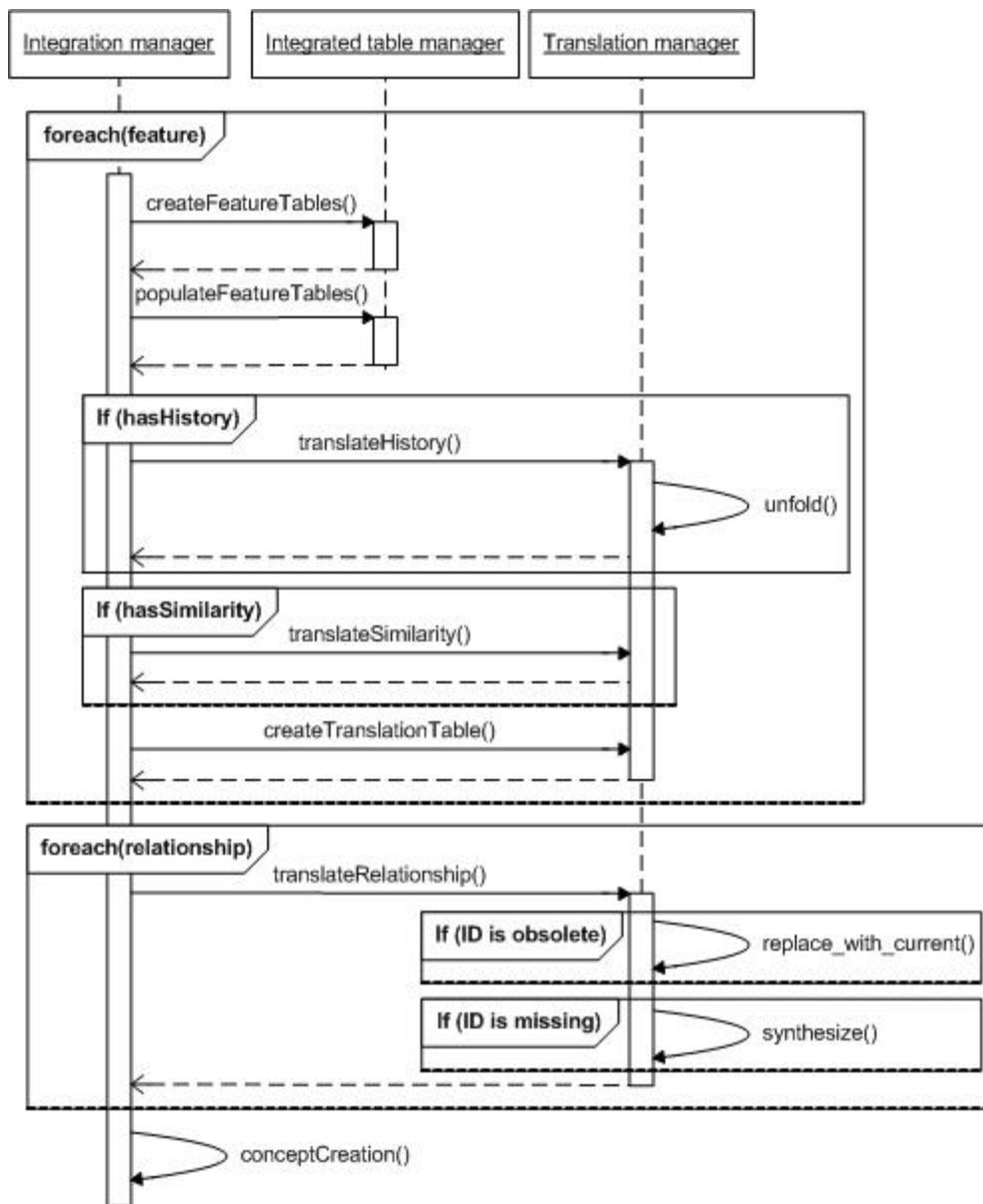


Figure 9 Sequence diagram of the main tasks of the implemented

During the initial aggregation phase, integrated tables of the features described by the imported data are created and populated. Then, similarity and historical ID data are created by translating the IDs provided by the data sources to our internal OIDs. Both similarity and historical ID data are extremely valuable for subsequent data integration tasks. The former ones express similarity between different entries of the same feature (e.g., homology between genes or proteins, or alias of feature IDs). The latter ones, which are sometimes provided by

the data sources, describe obsolete feature IDs and the IDs to which they have been propagated. Unfolding of their translated OIDs is performed, so as to associate repeatedly superseded and discontinued IDs to the latest IDs before their OID translation. These integrated similarity entries are also marked as inferred through historical data in order to keep full track of their generation process. Special translation tables for biomolecular entity and biomedical feature IDs are also created by using translated similarity data and unfolded historical ID data. These tables serve as main entry points to explore the data warehouse, since they allow the conversion from a number of diverse and possible obsolete user-provided identifiers to a set of current OIDs, usable to navigate the data warehouse.

Finally, relationships (annotations) between pairs of feature entries are created by performing OID translation of the imported relationship data expressed through the “native” IDs. In doing so, relationship data are coupled with the related feature entries. Due on the imported data sources and on their mutual synchronization, relationship data may refer to feature entries, or even features, that have not been imported in the data warehouse. In this case, missing integrated feature entries are synthesized and marked as such (i.e., inferred through synthesis from relationship data). However, if a missing entry has an obsolete ID and through unfolded translated historical data it is possible to extract a more current ID for it, the relationship is first transferred to the latest ID and is marked as inferred through historical data. This relationship translation policy preserves, between the integrated data, all the relationships expressed by the imported relationship data and allows their subsequent use for biomedical knowledge discovery (e.g., by transitive closure inference, involving also the synthesized entries).

During the final integration phase, by doing a similarity analysis, it is tested whether single feature instances from different sources represent the same feature concept. In this case, they are associated with a new created single concept OID. Different biomolecular entity concept instances are also further integrated under a single gene concept when they represent biomolecular entities related to the same gene. New entries can then be inferred from the integrated data (e.g., annotations can be inferred from other annotations by transitive closure inference). The Inferred field is used in all integrated tables to keep track of the method of inference used to derive an entry. Furthermore, a summary quality value for each concept instance is calculated based on the source specific instances contributing to the concept instance and their Inferred attribute value.

At the end of the integration process, on all integrated tables the defined index, unique, primary and foreign key integrity constraints are enforced in order to detect (and resolve) possible data duplications and inconsistencies, thus improving the time of access to the integrated data, as well as their overall quality.

### ***2.5.2.3 Metadata procedures***

In **metadata schema**, they are described the data contained in the public database: name of main tables, name of encoded fields, name of all source and features, which feature are provided by each source, which feature/source relations or annotations are provided by each source, which source are imported and which are synthesized, which source/feature is ontological, which sources/features have history or similarity data and in which tables they are stored, and many other information

As it is written in the chapter 2.5.1 Data and metadata schema, metadata schema store information about the metadata, as for example name of the data sources and their provided features

Metadata procedures are the function and class which are generate the the tables explained in the previous section. There was some problem in unfolding of the metadata and they are not optimized.



### 3 Thesis goals

The integration of biomolecular data is an important aspect of the bioinformatics research. Especially in biomolecular researches, it is possible to answer questions of interest by analyzing the various types of data and knowledge available in order to obtain evidences in support of the searched answer. The number of completely sequenced genomes is increasing at a rapid speed, as the amount of available proteomics data is. All these different types of data present a huge challenge for data integration, because they are usually stored in different databases without overall data standards or universal links. To a certain amount, data can be linked ('integrated') through manual, literature-based curation, but, with the quantity of available data continuing to increase exponentially, automatic methods for data integration need to be developed. The project GPDW (Genomic and Proteomic Data Warehouse) is working on this problem. The GPDW project aims not only to create a local data warehouse that integrates records from different databases, but also to keep the integrated information frequently updated.

This Thesis has, as primary purpose, the extension of the procedures in the GPDW framework for the integration of biomolecular interaction data; the second Thesis objective is to use these extensions to import and integrate the information provided by some databases that are not yet considered in the project GPDW.

Especially the development of the second objective requires:

- Conceptual and logical modeling of interaction data from the considered databases;
- Implementation of interaction data importing procedures, through the use of automated parser of standard data and specifications according to the specific data file type, to extract data from considered files and import them into the data warehouse;
- Configuration of automatic procedures for the importing and integration of interaction data into the data warehouse and reporting anomalies in the data.

## 4 Extension of GPDW software framework for biomolecular interaction data management

Before this thesis is developed, GPDW project is implemented for only the relations of the different features. The project is updated to cope with also the relations between same feature types.

### 4.1 Data import procedures

In this part first of all it was added new type of table template to feature\_definitions.xml. Previously all the features has different types of the integration so the source and destination id can be called directly with the [feature name]\_id, however when the name of the feature are same this become a problem because they have the same name in the database which is not possible. So it becomes [feature\_name]1\_id and [feature\_name]2\_id as seen in the Figure 10

```
- <template name="feature_relationship_same_imported">
- <table name="featureA2featureB_imported" xmlns="http://polimi.it/gfinder2/table_definition">
  <attribute name="annotation_oid" type="BIGINT" nullable="false" />
  <attribute name="featureA1_id" type="VARCHAR(128)" nullable="false" />
  <attribute name="origin_data_source" type="DATA_SOURCE_ID_TYPE" nullable="true" />
  <attribute name="featureB2_id" type="VARCHAR(128)" nullable="false" />
  <attribute name="destination_data_source" type="DATA_SOURCE_ID_TYPE" nullable="true" />
  <attribute name="reference" type="DATA_SOURCE_ID_TYPE" nullable="false" />
  <attribute name="reference_file" type="bit(256)" nullable="true" />
  <attribute name="association_type" type="INTEGER" nullable="true" />
- <primary_key>
  <attribute name="annotation_oid" />
</primary_key>
- <index name="featureA2featureB_imported_idx1">
  <attribute name="featureA1_id" />
  <attribute name="origin_data_source" />
</index>
- <index name="featureA2featureB_imported_idx2">
  <attribute name="featureB2_id" />
  <attribute name="destination_data_source" />
</index>
- <unique name="featureA2featureB_uidx">
  <attribute name="featureA1_id" />
  <attribute name="origin_data_source" />
  <attribute name="featureB2_id" />
  <attribute name="destination_data_source" />
  <attribute name="reference" />
</unique>
</table>
</template>
```

Figure 10 Import level relation between same feature

The necessary changes are done which is using this template automatically and they are updated if they are not checking if the which is created by using this template and all the hardcoded part of the import phase which are working on the relation tables. They were

rewrite to check if the features are same then add 1 or 2 at the end of the feature name in ordered.

## 4.2 Data integration procedures

In this part first of all it was added new type of table template to feature\_definitions.xml. previously all the features has different types of the integration so the source and destination id can be called directly with the [feature name]\_id, however when the name of the feature are same this become a problem because they have the same name in the database which is not possible. So it becomes [feature\_name]1\_id and [feature\_name]2\_id as seen in the Figure 11

```
- <template name="feature_relationship_same">
- <table name="featureA2featureB" xmlns="http://polimi.it/gfinder2/table_definition">
  <attribute name="annotation_oid" type="BIGINT" nullable="false" integrated="true" />
  <attribute name="featureA1_oid" type="BIGINT" nullable="false" integrated="true" />
  <attribute name="featureB2_oid" type="BIGINT" nullable="false" integrated="true" />
  <attribute name="reference" type="DATA_SOURCE_ID_TYPE" nullable="false" integrated="true" />
  <attribute name="association_type" type="INTEGER" nullable="true" integrated="true" encoded="true" />
  <attribute name="inferred" type="INTEGER" nullable="true" integrated="true" encoded="true" />
- <primary_key>
  <attribute name="annotation_oid" />
</primary_key>
- <foreign_key name="featureA1_fk" references="featureA">
  <attribute name="featureA1_oid" references="featureA_oid" />
</foreign_key>
- <foreign_key name="featureB2_fk" references="featureB">
  <attribute name="featureB2_oid" references="featureB_oid" />
</foreign_key>
- <index name="featureA2featureB_idx1">
  <attribute name="featureA1_oid" integrated="true" />
</index>
- <index name="featureA2featureB_idx2">
  <attribute name="featureB2_oid" integrated="true" />
</index>
- <index name="featureA2featureB_idx3">
  <attribute name="inferred" integrated="true" />
</index>
- <unique name="featureA2featureB_uidx">
  <attribute name="featureA1_oid" integrated="true" />
  <attribute name="featureB2_oid" integrated="true" />
  <attribute name="reference" integrated="true" />
</unique>
</table>
</template>
```

Figure 11 Integrate level relation between same feature

## 4.3 Metadata computation and storing

As mentioned before metadata schema contains metadata information of the public schema. Some processes are not working optimized and correctly. The most important change is on the unfolding of feature association and source to feature association. This unfolding procedure is designed to create metadata in Figure 12 and Figure 13:

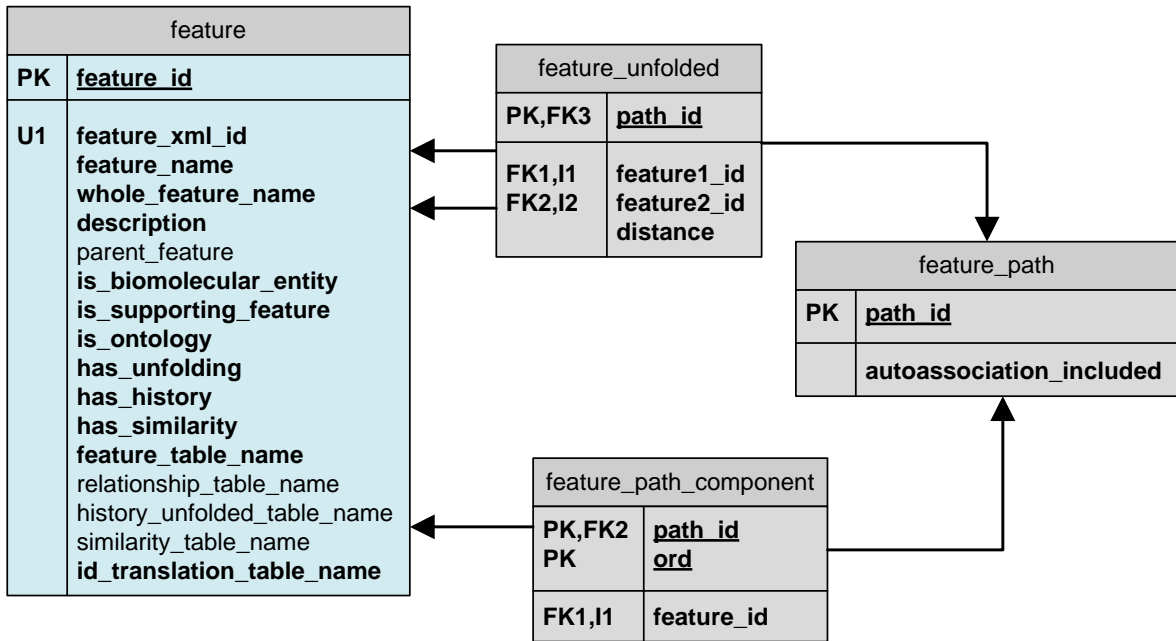


Figure 12 Metadata schema of feature unfolding tables

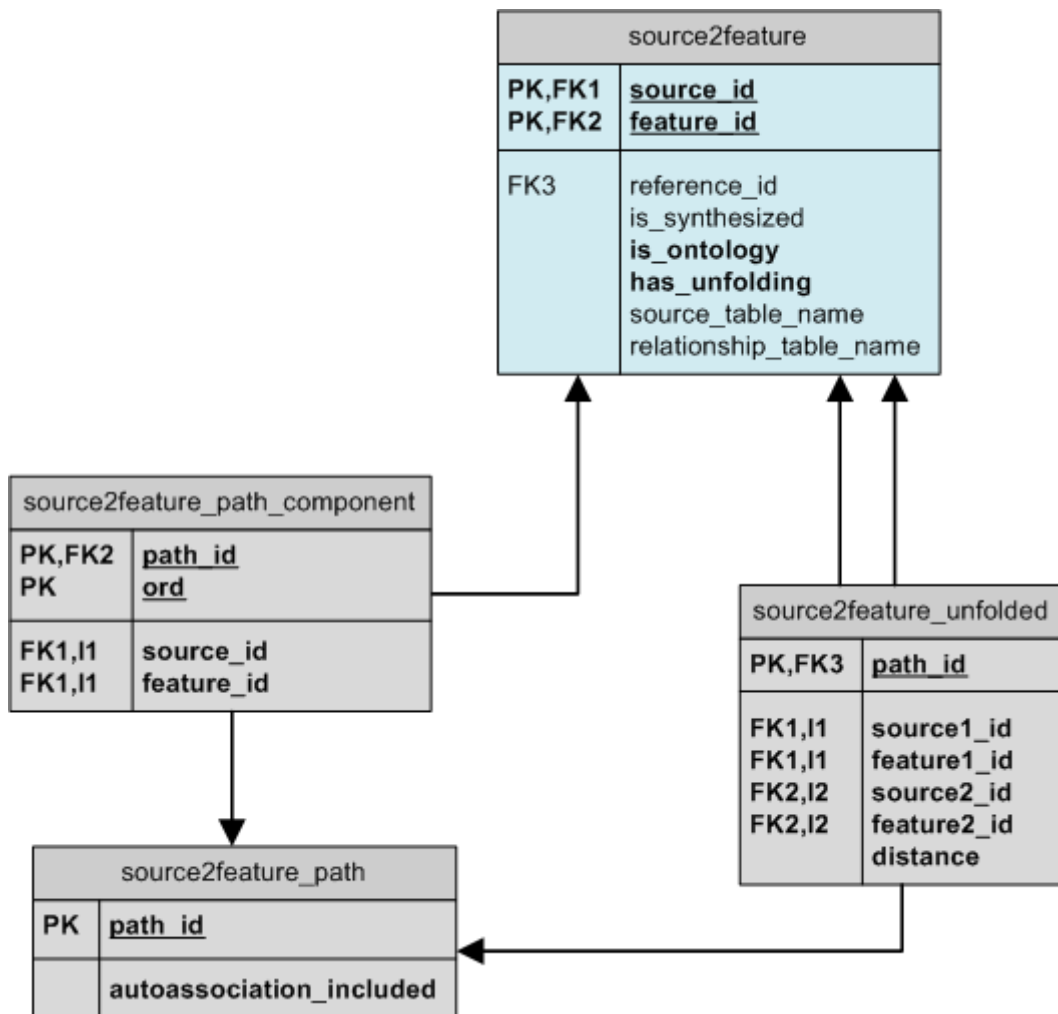


Figure 13 Metadata schema of source to feature unfolding tables

For generating these tables the algorithm below is used:

```

unfold()
  for each "node" defined in database(from source2feature table or feature table)
    "children":find all connected nodes of the "node"
    if there is no children create a new path with a new path_id
    and also insert "node" to path_componant table with this path id order 0
    continue with the next "node"
  elseif
    create a new path with a new "pathId" contains starts with this node(as order = 0)
    and also insert "node" to path_componant table with this path id order 0

    push "node" to "stack"
    for each "children" as "child"
      (stack now contains only "node")
      call unfoldRecursive function:with pathId,node as ancestorNode,child as
currentNode,1 as ord, stack as "stack"
    endfor
    pop "stack"
  endif
endfor
endunfold
unfoldRecursive(pathId,ancestorNode,currentNode,ord,stack)
  insert "currentNode" to path_componant table with "pathId", "ord" as order //each recursive call ord increase by 1

  insert "ancestorNode","currentNode" to unfolded table with "ord" as distance(because order is also shows the
distance to ancestor node)
  "children":find all connected nodes of the "currentNode"(in both case from node1->node2 or node2->node1)
  for each "children" as "child"
    //checking a loop before adding current node to stack in order to allow only same node to same node
path once and consecutively
    if "stack" is not contains "child"
      stack.push(currentNode);
      /* Clone the path */
      create a "newPathId" and add path table
      add elements to path_componant table with "newPathId" from ancestorNode to currentNode
with their order.

      call unfoldRecursive function:with "newPathId", "ancestorNode" as ancestorNode,child as
currentNode,"ord"+1 as ord, stack as "stack" (currentnode added to stack)
      stack.pop();
    endif
  endfor
endunfoldRecursive

```

This algorithm is allows loop only if the cycle is consecutively in a node and only once.

For the feature case, there are not many associations so the unfolding is not generating so many path. However for the source to feature the current number of associations and the connections between association lead to nearly 200 millions of tuples in the unfolded table. This much of data(huge tables) will make the data warehouse slow. So the algorithm updated to traverse the association graph till a given number of depth. The system is traversing till this fixed depth.

This thesis also added a new table to metadata schema which is source2feature\_association\_display\_url table. This is necessary for the sources which is not importing any feature but the relations. So this display\_url table for the association is showing the url of the database for this association. The data is taken from the data\_sources.xml in the path data\_sources/data\_source\_definition/feature/display\_urls. The metadata schema of this table is in Figure 14. So for each couples of source feature pair and reference id the url is inserted into database.

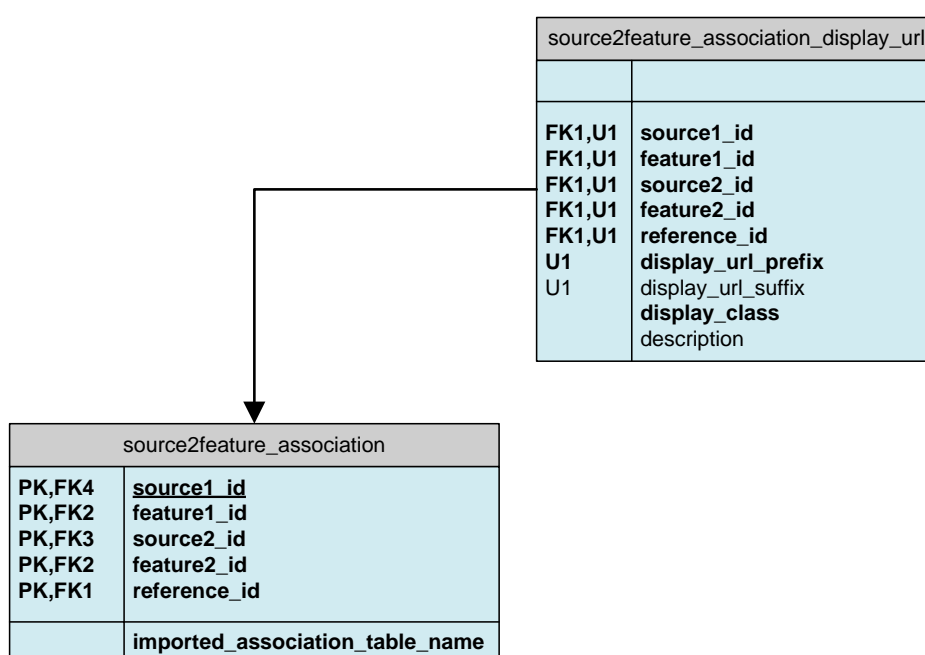


Figure 14 Metadata schema if source2feature\_association\_display\_url

## 5 Considered biomolecular interaction databanks

The considered databanks have information about molecular interactions (MIs) by extracting experimental details from work published in peer-reviewed journals. In another words, the data available in the databases originates entirely from published literature and is manually annotated by expert biologists to a high level of detail, including experimental methods, conditions and interacting domains.

The IntAct and MINT projects support PSI-MI 2.5 standard. The PSI MI format is a data exchange format for molecular interactions. This thesis is focused on mainly MITAB2.6 and PSI-MI XML v2.5. MITAB is data interchange format, a common tab delimited format XML standard is described below:

The Proteomics Standards Initiative Molecular Interaction XML format (PSI MI)(5) was developed by the Proteomics Standards Initiative, one initiative of the Human Proteome Organisation (HUPO). The aim of the initiative is to develop standards for data representation in proteomics to facilitate data comparison, exchange and verification. One of those is the PSI MI standard for protein–protein interaction. The format is intended for exchange of data on protein interactions.

All data in PSI MI are structured around an entry. An entry describes one or more interactions that are grouped together for some reason. Note that a PSI MI model is not intended to be a pathway, an entry can be any set of interactions. In the entry the two tags source and the availabilitylist are used for describing the source of the data, usually an organization, and where the data can be accessed, typically a database. The experimentlist describes experiments and links to publications where the interactions are verified.

The pathway itself is described via the interactorlist, which is a list of proteins participating in the interaction, and the interactionlist, a list of the actual interactions. For each interactor information about, for instance, substructure can be defined. For each interaction it is possible to set the type of interaction and also a database reference to more information about the interaction. The type of the interaction, e.g. aggregation, is chosen from an externally defined controlled vocabulary, that can be chosen by the user. The participating proteins are described by their names or references to the interactorlist. It is also possible to set a confidence level for detecting this protein in the experiment, the role of the protein and whether the protein was tagged or overexpressed in the experiment. In addition each interaction has a description of availability and experiments which normally are references to the lists above.

Finally, the attributelist gives the user the possibility to add further information that does not fit into the entries above. Extra attributes can be used in all the parts described above. An abbreviated example pathway represented in PSI MI is shown in Figure 15.



```

<entry>
<interactorList>
  <proteinInteractor id="Succinate">
    <names>
      <shortLabel>Succinate</shortLabel>
      <fullName>Succinate</fullName>
    </names>
  </proteinInteractor>
  ...
</interactorList>
<interactionList>
  <interaction>
    <names>
      <shortLabel> Succinate dehydrogenas catalysis </shortLabel>
      <fullName>Interaction between ....</fullName>
    </names>
    <participantList>
      <proteinParticipant>
        <proteinInteractorRef ref="Succinate"/> <role>neutral</role>
      </proteinParticipant>
      <proteinParticipant>
        <proteinInteractorRef ref="Fumarate"/><role>neutral</role>
      </proteinParticipant>
      <proteinParticipant>
        <proteinInteractorRef ref="Succdeh"/><role>neutral</role>
      </proteinParticipant>
    </participantList>
  </interaction>
</interactionList>

```

Figure 15 Example of PSI MI.

There can be two types of interactions which are binary and complex. The binary interaction is an interaction between only two interactors; however complex interactions are between with multiple interactors. In the data warehouse the complex interactions are imported as multiple binary interactions. There are two main algorithms for this:

- Spoke expansion: Links the bait molecule to all prey molecules. If N is the count of molecule in the complex, it generated N-1 binary interactions.
- Matrix expansion: Links all molecule to all other molecule present in the complex. If N is the count of molecule in the complex, it generated  $(N*(N-1))/2$  binary interactions.

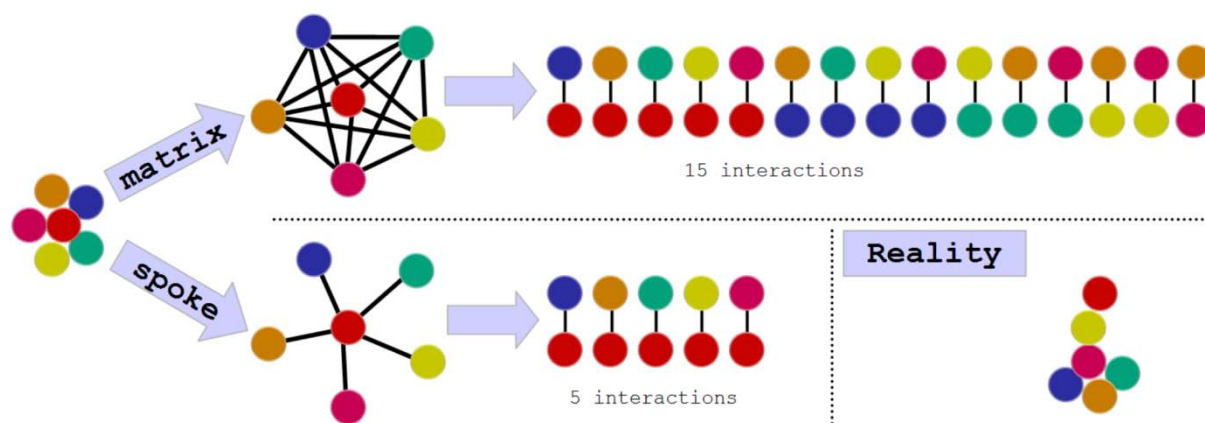


Figure 16 Complex expansion algorithms

## 5.1 IntAct: an open source molecular interaction databank

IntAct(8) is an open-source, open data molecular interaction database and toolkit. Data is abstracted from the literature or from direct data depositions by expert curators following a deep annotation model providing a high level of detail. IntAct contains over 200.000 curated binary interaction evidences. In response to the growing data volume and user requests, IntAct now provides a two-tiered view of the interaction data. The search interface allows the user to iteratively develop complex queries, exploiting the detailed annotation with hierarchical controlled vocabularies. Results are provided at any stage in a simplified, tabular view.

It is possible to download PSI MI XML 2.5 format and MITAB2.6(definition can be found: <ftp://ftp.ebi.ac.uk/pub/databases/intact/current/psimitab/README>) version with additional columns.

It has one single tabular file that contains all the relations with name *intact.txt* and has multiple XML files with divided publication ids. The procedure runs all these XML files and import them to database.

## 5.2 MINT: the Molecular INTERaction databank

The Molecular INTERaction database (9) (MINT) aims at storing, in a structured format, information about molecular interactions (MIs) by extracting experimental details from work

published in peer-reviewed journals. At present the MINT team focuses the curation work on physical interactions between proteins. Genetic or computationally inferred interactions are not included in the database. Over the past four years MINT has undergone extensive revision. The new version of MINT is based on a completely remodeled database structure, which offers more efficient data exploration and analysis, and is characterized by entries with a richer annotation. Over the past few years the number of curated physical interactions has soared to over 95 000. The whole dataset can be freely accessed online in both interactive and batch modes through web-based interfaces and an FTP server. MINT now includes, as an integrated addition, HomoMINT, a database of interactions between human proteins inferred from experiments with ortholog proteins in model organisms (<http://mint.bio.uniroma2.it/mint/>).

MINT databank contains interactions only between protein and protein.

It has a single tabular file for binary interactions with name *[creation date]-mint-full-binary.txt* and another tabular file that contains complex interactions with name *[creation date]-mint-full-complex.txt* and has multiple XML files not to create huge files with name *full[id number].psi25.xml*.

## 6 Design of GPDW sections for biomolecular interaction data

The design of the database is organized in three steps:

- *Conceptual design*: This phase is to create the conceptual or ER (entity relationship) diagram that represents the informal specifications of the database in terms of a complete formal description but independent of the criteria used in the performance management system based data. In these representations, it is not taken into account neither the manner in which this information will be encoded nor efficiency of programs that will use this information;
- *Logical design*: This phase is to translate the conceptual schema, which is defined in the previous phase, the model of the data representation adopted by the management system of the database used. In this phase, the logical schema, a logical data model refers to a as in the preceding stage, is generated that refers to a as in the preceding

stage and the representation is independent of physical details. In this case, the design choices are based on optimization criteria necessary operations on data.

- *Physical Design*: In this stage, the logical schema is completed with the specification of physical data storage, file organization and indexes. This phase refers to a physical data model and depends on the specific data management system chosen.

The first two phases provide for the creation of the schemes through the use of design software, which in this case Microsoft Visio ® is used, and the physical design is performed by an automated framework, thanks to the information source and features defined in the XML configuration file.

The schemas are designed for tabular format (MITAB) and XML format (PSI MI XML)The interactions can be between:

- protein and protein
- transcript and transcript
- dna sequence and dna sequence
- small molecule and small molecule
- transcript and protein
- dna sequence and protein
- protein and small molecule
- transcript and small molecule
- dna sequence and small molecule

## 6.1 Conceptual schema

Figure 17 and Figure 18 show the conceptual diagram of the database of interaction between protein and protein and interaction between dna sequence and protein, respectively. The other interactions are similar to these two.

Attributes shown in red are not imported into database.

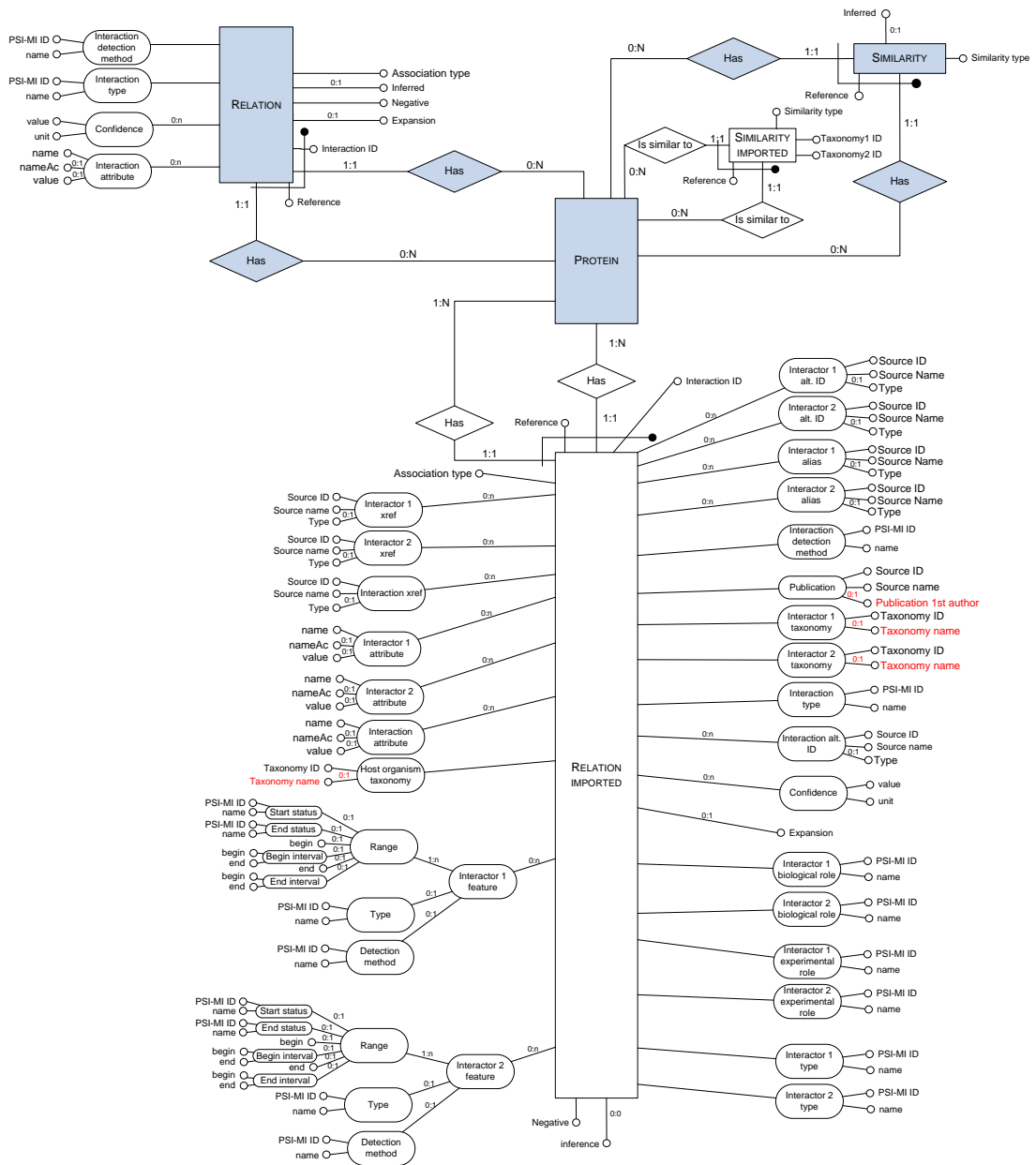


Figure 17 Conceptual schema of interactions between protein and protein.

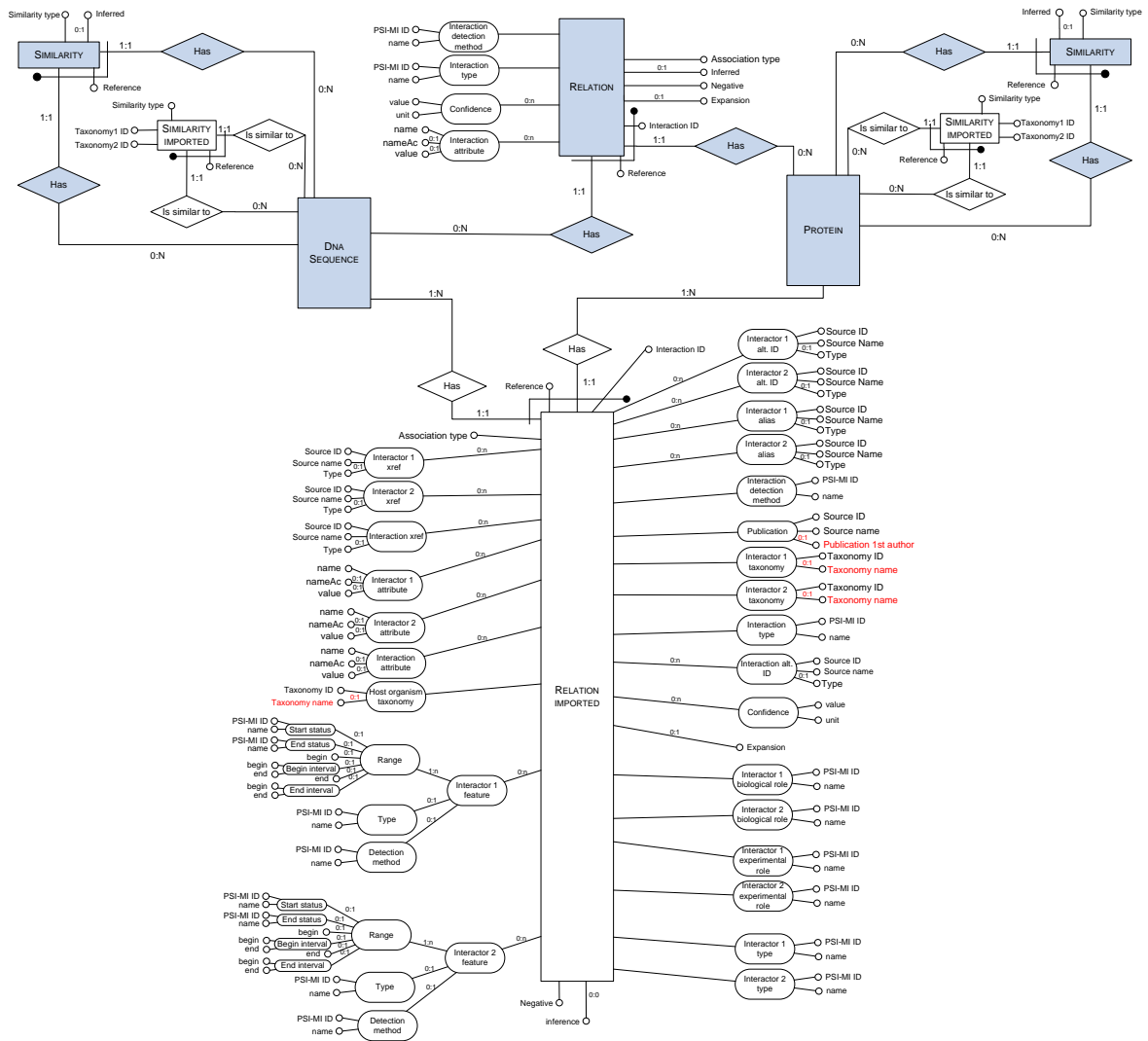


Figure 18 Conceptual schema of interactions between dna sequence and protein.

The interaction between below are like in Figure 17:

- protein and protein
- transcript and transcript
- dna sequence and dna sequence
- small molecule and small molecule

The interaction between below are like in Figure 18:

- transcript and protein
- dna sequence and protein
- protein and small molecule

## 6.2 Logical schema

The logical schema is showing the type of the tables and their attributes.

The type of the tables is based on their background color as shown in Table 1.







Color of table	Type of table
	ID translation table
	Relation or annotation integrated table
	Similarity integrated table
	Integrated table
	Relation or annotation imported table
	Similarity imported table

Table 1 Logical schema legend colors of tables

It is necessary to identify some more aspects:

Bold fields show required attributes.

The field with **PK** is table's primary key.

The field with **FK** is foreign key to another table. The links show also the foreign key between two tables.

The field with **UX** shows the unique key. U1 unique key can be defined as pseudo primary key and U2 of the related tables is necessary for to integrate the table.

The field with **IX** shows the index of the table

The fields with \* and + are encoded fields. The values of fields with \* are stored in the flag schema of the database. The values of fields with + are stored in the metadata schema of the database.

The Figure 19 and Figure 20 show the logic diagram of the molecular interaction databases.

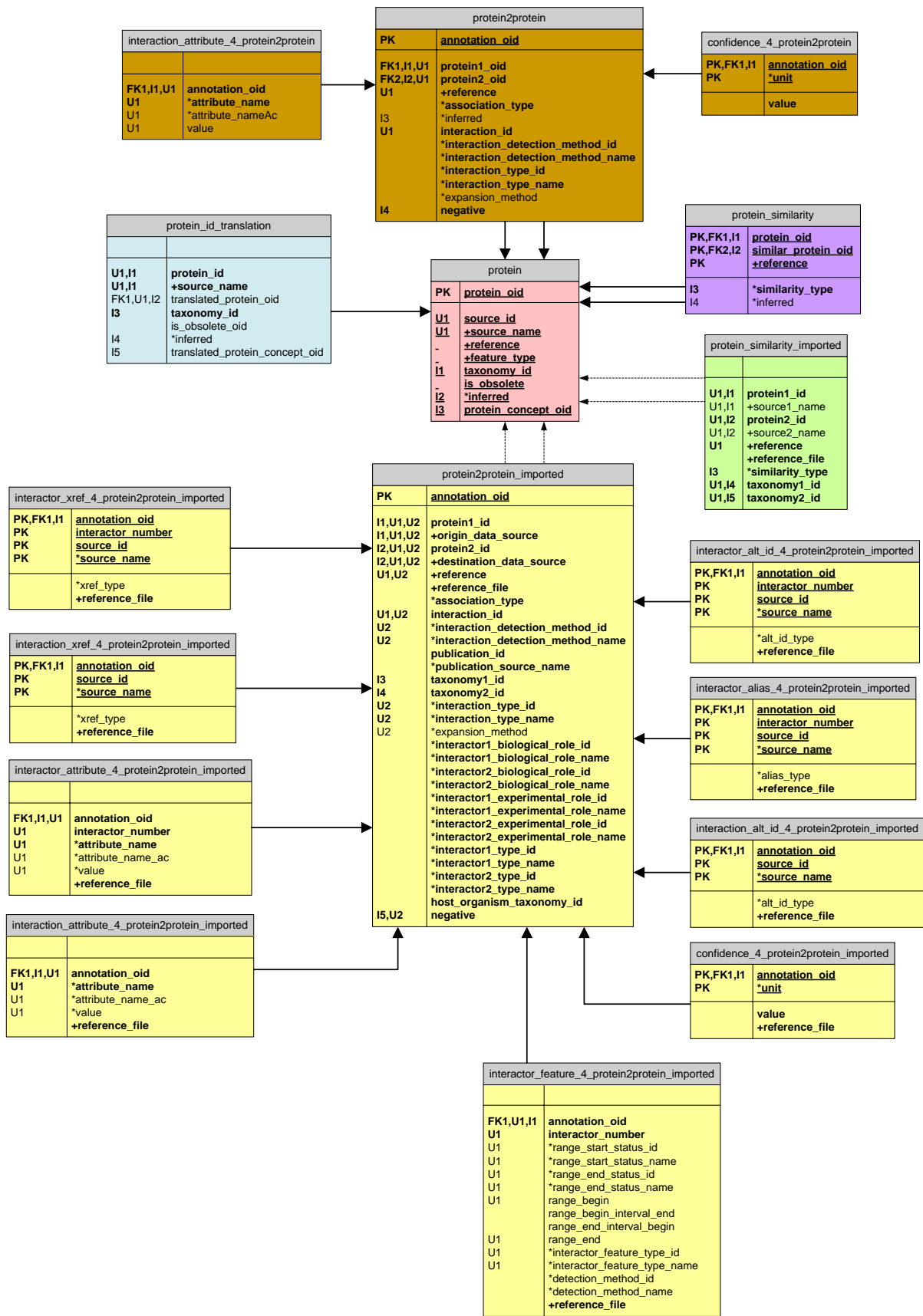


Figure 19 Logical schema of interactions between protein and protein.



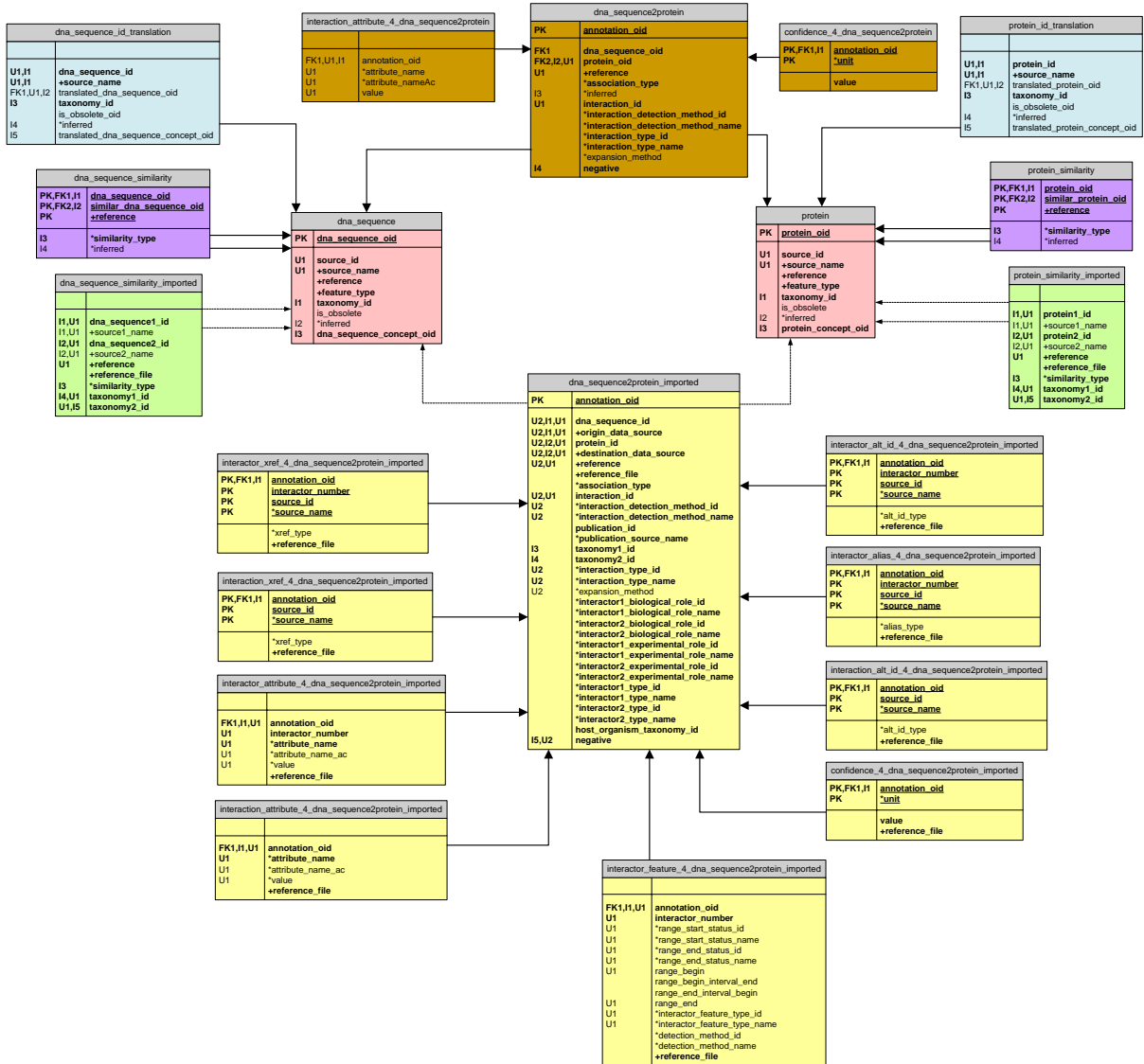


Figure 20 Conceptual schema of interactions between dna sequence and protein.

The interaction between below are like in Figure 19:

- protein and protein
- transcript and transcript
- dna sequence and dna sequence
- small molecule and small molecule

The interaction between below are like in Figure 20:

- transcript and protein
- dna sequence and protein
- protein and small molecule

## 7 Implementation of automatic procedures for biomolecular interaction data import

In this section, the processes are performed, the design choices made for the import of data and explaining the technical solutions used are explained.

### 7.1 Tabular file format

The tabular files can be downloaded ftp server of the databanks and these are one file for IntAct and Two files for MINT. In MINT case binary interactions and complex interactions are in separate files.

There are 40 columns from the MINT database and 31 columns for the IntAct case. First 15 columns are same for both databanks. The other columns are defined in the databanks' read me files.

First 15 columns are described as below:

1. **Unique identifier for interactor A**, represented as `databaseName:ac`, where `databaseName` is the name of the corresponding database as defined in the PSI-MI controlled vocabulary, and `ac` is the unique primary identifier of the molecule in the database. Identifiers from multiple databases can be separated by "|". It is recommended that proteins be identified by stable identifiers such as their UniProtKB or RefSeq accession number.
2. **Unique identifier for interactor B**.
3. **Alternative identifier for interactor A**, for example the official gene symbol as defined by a recognised nomenclature committee. Representation as `databaseName:identifier`. Multiple identifiers separated by "|".
4. **Alternative identifier for interactor B**.
5. **Aliases for A**, separated by "|". Representation as `databaseName:identifier`. Multiple identifiers separated by "|".
6. **Aliases for B**.
7. **Interaction detection methods**, taken from the corresponding PSI-MI controlled Vocabulary, and represented as `databaseName:identifier(methodName)`, separated by "|".

8. **First author** surname(s) of the publication(s) in which this interaction has been shown, optionally followed by additional indicators, e.g. "Doe-2005-a". Separated by "|".
9. **Identifier of the publication** in which this interaction has been shown. Database name taken from the PSI-MI controlled vocabulary, represented as databaseName:identifier. Multiple identifiers separated by "|".
10. **NCBI Taxonomy identifier for interactor A.** Database name for NCBI taxid taken from the PSI-MI controlled vocabulary, represented as databaseName:identifier. Multiple identifiers separated by "|". Note: In this column, the databaseName:identifier(speciesName) notation is only there for consistency. Currently no taxonomy identifiers other than NCBI taxid are anticipated, apart from the use of -1 to indicate "in vitro" and -2 to indicate "chemical synthesis".
11. **NCBI Taxonomy identifier for interactor B.**
12. **Interaction types**, taken from the corresponding PSI-MI controlled vocabulary, and represented as dataBaseName:identifier(interactionType), separated by "|".
13. **Source databases** and identifiers, taken from the corresponding PSI-MI controlled vocabulary, and represented as databaseName:identifier(sourceName). Multiple source databases can be separated by "|".
14. **Interaction identifier(s)** in the corresponding source database, represented by databaseName:identifier
15. **Confidence score.** Denoted as scoreType:value. There are many different types of confidence score, but so far no controlled vocabulary. Thus the only current recommendation is to use score types consistently within one source. Multiple scores separated by "|".

The loader java files are different for the MINT and IntAct and they are used the same procedure for inserting into database.

Each file consist of three sub filed divided by first field which is external database name and a colon (: ) and the id of the field and in the parenthesis the name for the id.

Each column is parsed and then after checking the correctness of the data they are imported into database. The automatic parser class is implemented and makes the fields ready to use them. Parse the field of the line is done by this class.

After parsing the data it is checking the correctness of the data especially the id is recognized by the system or not. If so it inserts the data into database.

As explained in the chapter 2.5.2, each databank has an importer that imports each file with loader of appropriate source. Tabular loader of each databank(MintMitabLoader or IntActMitabLoader) is an extension of a generic loader(MitabLoader.java). MitabLoader has an object which is MIUtils which has database insert methods and data checking methods. Those methods are used also by the XML loaders.

The expansion of the complex interaction in tabular format is given by the data source. Currently both MINT and IntAct are using spoke model

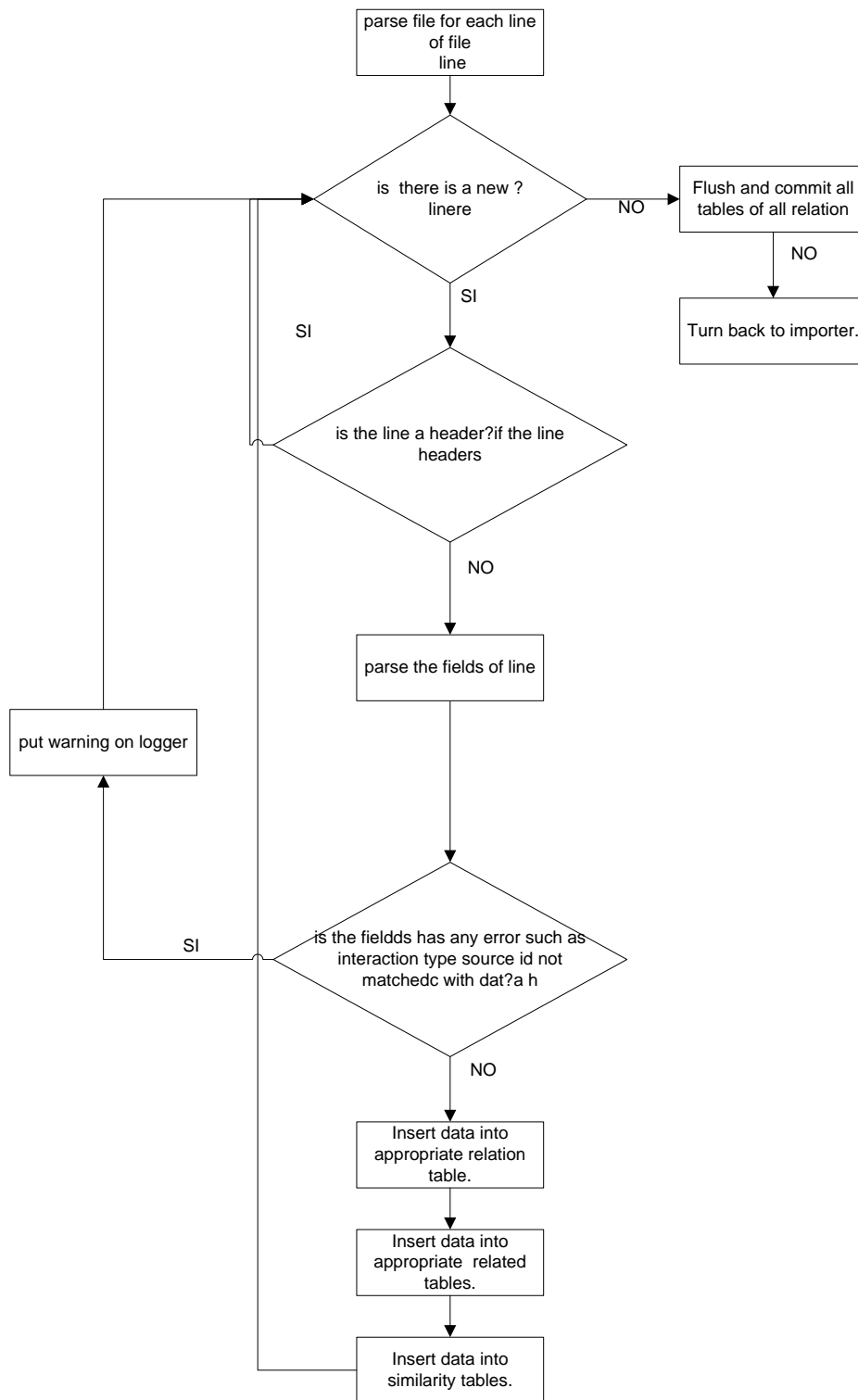


Figure 21 Tabular loader workflow

## 7.2 XML file format

XML files were generated with the schema in the link:

<http://psidev.sourceforge.net/mi/rel25/src/MIF25.xsd>.

The parser of XML was implemented by PSI-MI Developer on Google codes. This parser parses the interaction automatically and then it is easy to get the interactions. It parses all the data in the xml file and returns the easily manageable java class.

After automatic parse the necessary columns of the database is found with traversing parsed XML. The XML parser is also using the same procedure with the tabular parser for checking the data quality and inserting into database. The importing additional tables are also done with this parser. The common procedures are explained in the previous sub-chapter.

The XML file has structure as below for each interaction:

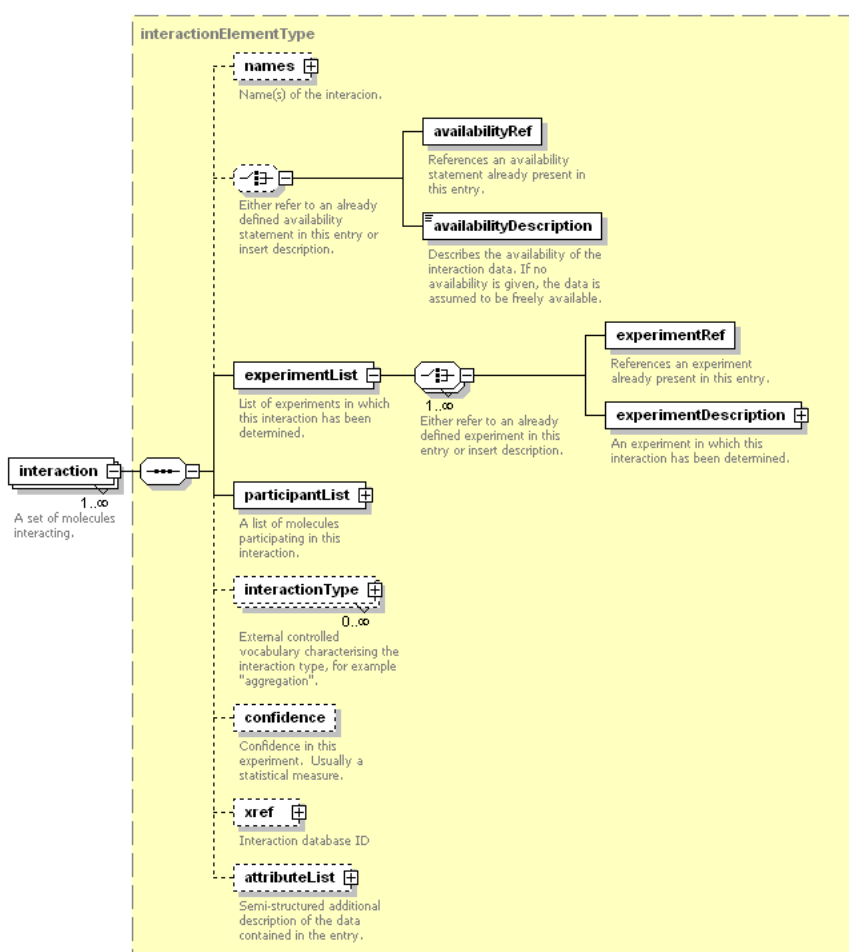


Figure 22 Interaction

So it is easy to parse XML calling for all file for each interaction and adding them to database.

The expansion of the complex interaction in XML file format is given by the data source. If the all the interactor of interaction are neutral then matrix algorithm is used and if the interaction is prey- bait model then spoke algorithm is used.

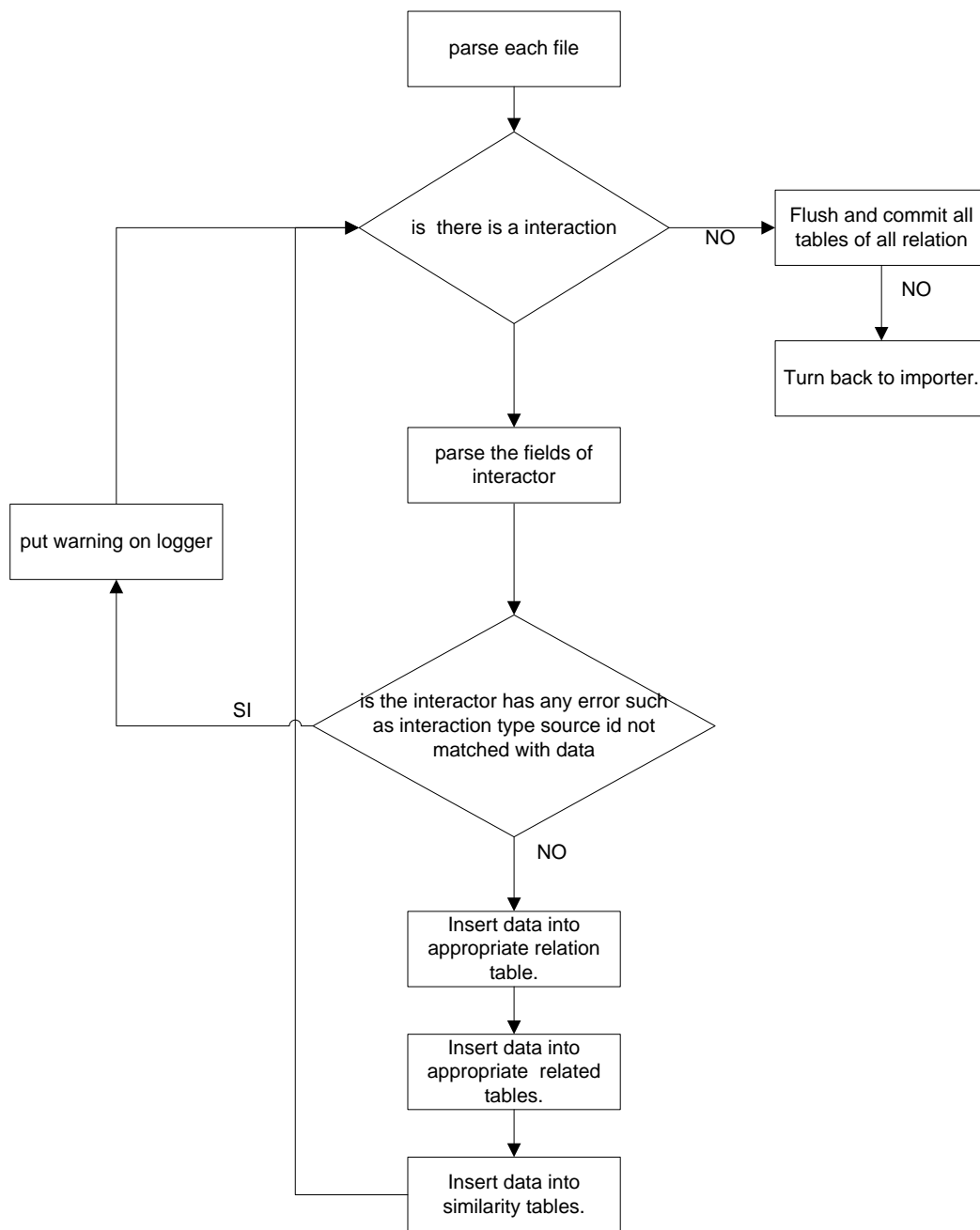


Figure 23 XML loader workflow

## 8 Validation and testing

### 8.1 Quantification of imported data and running times

Following tables contains running times of import phase and number of imported entries.

Table 2: show the execution time of the importer and number of relation entries and number of all entries in relation tables and their related tables.

Table 4: show the number of entries in similarity tables.

Table 5 and Table 6 show number of entries of interaction related tables from IntAct and MINT databanks, respectively.

Databank	Name of file	Number of tables populated	Number of relation entries	Number of all entries	Execution time(sec)
MINT	2010-12-15-mint-full-binary.mitab26.txt	11	116880	1632032	248
	2010-12-15-mint-full-complexes.mitab26.txt	11	8207	138689	23
IntAct	intact.txt	98	271522	13608653	1621

Table 2 Loader execution time

Databank	Name of file	Interaction name	Number of entries
MINT	2010-12-15-mint-full-binary.mitab26.txt	protein to protein	116880
	2010-12-15-mint-full-complexes.mitab26.txt	protein to protein	8207
IntAct	intact.txt	protein to protein	268051
IntAct	intact.txt	protein to small_molecule	1936
IntAct	intact.txt	protein to transcript	559
IntAct	intact.txt	protein to dna_sequence	913
IntAct	intact.txt	dna_sequence to dna_sequence	10



IntAct	intact.txt	dna_sequence to small molecule	20
IntAct	intact.txt	transcript to transcript	11
IntAct	intact.txt	transcript to small molecule	1
IntAct	intact.txt	small molecule to small molecule	21

Table 3 Number of entries in interaction tables

Databank	Name of file	Feature name	Number of entries
MINT	2010-12-15-mint-full-binary.mitab26.txt	protein	496
	2010-12-15-mint-full-complexes.mitab26.txt	protein	276
IntAct	intact.txt	protein	697620
IntAct	intact.txt	small_molecule	1166
IntAct	intact.txt	transcript	0
IntAct	intact.txt	dna_sequence	0

Table 4 Number of entries in similarity tables

Table name	Number of entries
confidence_4_dna_sequence2dna_sequence_imported	0
confidence_4_dna_sequence2protein_imported	0
confidence_4_dna_sequence2small_molec_imported	0
confidence_4_protein2protein_imported	0
confidence_4_protein2small_molec_imported	0
confidence_4_small_molec2small_molec_imported	0
confidence_4_transcript2protein_imported	0
confidence_4_transcript2small_molec_imported	0
confidence_4_transcript2transcript_imported	0

interaction_alt_id_4_dna_sequence2dna_sequence_imported	12
interaction_alt_id_4_dna_sequence2protein_imported	1366
interaction_alt_id_4_dna_sequence2small_molec_imported	28
interaction_alt_id_4_protein2protein_imported	354150
interaction_alt_id_4_protein2small_molec_imported	2192
interaction_alt_id_4_small_molec2small_molec_imported	29
interaction_alt_id_4_transcript2protein_imported	631
interaction_alt_id_4_transcript2small_molec_imported	1
interaction_alt_id_4_transcript2transcript_imported	15
interaction_attribute_4_dna_sequence2dna_sequence_imported	2
interaction_attribute_4_dna_sequence2protein_imported	413
interaction_attribute_4_dna_sequence2small_molec_imported	10
interaction_attribute_4_protein2protein_imported	27213
interaction_attribute_4_protein2small_molec_imported	221
interaction_attribute_4_small_molec2small_molec_imported	3
interaction_attribute_4_transcript2protein_imported	55
interaction_attribute_4_transcript2small_molec_imported	0
interaction_attribute_4_transcript2transcript_imported	0
interaction_xref_4_dna_sequence2dna_sequence_imported	0
interaction_xref_4_dna_sequence2protein_imported	0
interaction_xref_4_dna_sequence2small_molec_imported	0
interaction_xref_4_protein2protein_imported	0
interaction_xref_4_protein2small_molec_imported	0
interaction_xref_4_small_molec2small_molec_imported	0
interaction_xref_4_transcript2protein_imported	0
interaction_xref_4_transcript2small_molec_imported	0
interaction_xref_4_transcript2transcript_imported	0
interactor_alias_4_dna_sequence2dna_sequence_imported	0
interactor_alias_4_dna_sequence2protein_imported	987
interactor_alias_4_dna_sequence2small_molec_imported	0
interactor_alias_4_protein2protein_imported	473423
interactor_alias_4_protein2small_molec_imported	1826
interactor_alias_4_small_molec2small_molec_imported	
interactor_alias_4_transcript2protein_imported	469
interactor_alias_4_transcript2small_molec_imported	0
interactor_alias_4_transcript2transcript_imported	0
interactor_alt_id_4_dna_sequence2dna_sequence_imported	40
interactor_alt_id_4_dna_sequence2protein_imported	7332
interactor_alt_id_4_dna_sequence2small_molec_imported	82
interactor_alt_id_4_protein2protein_imported	2598669
interactor_alt_id_4_protein2small_molec_imported	14104
interactor_alt_id_4_small_molec2small_molec_imported	84
interactor_alt_id_4_transcript2protein_imported	3778
interactor_alt_id_4_transcript2small_molec_imported	4

interactor_alt_id_4_transcript2transcript_imported	62
interactor_attribute_4_dna_sequence2dna_sequence_imported	1
interactor_attribute_4_dna_sequence2protein_imported	139
interactor_attribute_4_dna_sequence2small_molec_imported	21
interactor_attribute_4_protein2protein_imported	20565
interactor_attribute_4_protein2small_molec_imported	2492
interactor_attribute_4_small_molec2small_molec_imported	38
interactor_attribute_4_transcript2protein_imported	152
interactor_attribute_4_transcript2small_molec_imported	1
interactor_attribute_4_transcript2transcript_imported	11
interactor_feature_4_dna_sequence2dna_sequence_imported	0
interactor_feature_4_dna_sequence2protein_imported	0
interactor_feature_4_dna_sequence2small_molec_imported	0
interactor_feature_4_protein2protein_imported	0
interactor_feature_4_protein2small_molec_imported	0
interactor_feature_4_small_molec2small_molec_imported	0
interactor_feature_4_transcript2protein_imported	0
interactor_feature_4_transcript2small_molec_imported	0
interactor_feature_4_transcript2transcript_imported	0
interactor_xref_4_dna_sequence2dna_sequence_imported	0
interactor_xref_4_dna_sequence2protein_imported	34103
interactor_xref_4_dna_sequence2small_molec_imported	25
interactor_xref_4_protein2protein_imported	9723538
interactor_xref_4_protein2small_molec_imported	59228
interactor_xref_4_small_molec2small_molec_imported	44
interactor_xref_4_transcript2protein_imported	9555
interactor_xref_4_transcript2small_molec_imported	1
interactor_xref_4_transcript2transcript_imported	16

Table 5 Number of entries in IntAct interaction related tables

Table name	Number of entries (binary interaction file)	Number of entries (binary interaction file)
confidence_4_protein2protein_imported	149948	7144
interaction_alt_id_4_protein2protein_imported	19184	4807
interaction_attribute_4_protein2protein_imported	424229	46653
interaction_xref_4_protein2protein_imported	790	821
interactor_alias_4_protein2protein_imported	866025	70009
interactor_alt_id_4_protein2protein_imported	0	0
interactor_attribute_4_protein2protein_imported	0	0
interactor_feature_4_protein2protein_imported	34348	1020
interactor_xref_4_protein2protein_imported	20628	28

Table 6 Number of entries in MINT interaction related tables

## 9 Conclusions

The main result of this Thesis is the implementation of software automated procedures for the importing and integration of genomic and proteomic interaction data provided by bioinformatics databanks. These procedures are integrated into the GPDW project that continues to develop into the final version providing an integrated and consistent data warehouse, which will be made available to researchers with the addition of the important interaction data considered in this Thesis.

This Thesis work was divided into three main stages:

1. analysis of the existing architecture of the GPDW framework in order to identify possible design errors and optimization of components;
2. definition and implementation of procedures for automatic importing and integration of biomolecular interaction data.
3. application of the implemented procedures to import, in the GPDW data warehouse, data supplied by IntAct and MINT databanks; this stage developed in the following different steps:
  - a. analysis of data from each source file, description of data and conceptual and logical design of relational model for each considered databank;
  - b. development of Parser, Importer and Loader software classes for specific interaction data, by extending the generic classes available, for extracting data from the considered sources and their automated importing in the data warehouse; the importing procedures have been designed by taking into account the goals and requirements defined, considering the need to develop capabilities that can be inherited and reused;
  - c. integration with the existing software architecture and changing the structure of the XML configuration file for the import of new data;
  - d. use of developed software for the creation of new parts of the GPDW data warehouse and import into them the considered interaction data;
  - e. testing and verification of the developed software and its use; this step involved the quantification of the imported data, their errors and inconsistencies detected by the automatic verification procedures implemented, and evaluation of the times taken for the importation and control of such data.

## 10 Future developments

In order to meet the goal of creating GPDW, it is needed to complete the implementation of the automatic integration activities, and to increase the number of databanks whose data are imported into the GPDW; this will generate a huge amount of structured data representing consolidated information and knowledge available worldwide, which will constitute the knowledge base that will enable the Virtual BioInformatics Lab users to efficiently analyze, and enrich biomedical experimental results, in order to translate new experimental findings into a better understanding of the underlying biological mechanisms involved.

The method for importing the data considered in this Thesis, which is currently applied only to data from the MINT and IntAct databanks, will be applied also to other databanks that provide Molecular Integration data files.

In addition, a softer module dedicated to download data files provided by the various databanks considered in the GPDW project will be implemented in order to support the maintenance of constantly updated information in the GPDW.

In the near future, the public access to GPDW data warehouse will be possible. The idea is to provide various Web services that will help the answer of classic bioinformatics problems in order to support scientists and researchers in the analysis of experimental data.

# 11 Bibliography

- 1 Masseroli M. Biomolecular databanks. *Bioinformatica e biologia computazionale per la medicina molecolare*. [Online].; 2009. Available from: [http://www.bioinformatics.polimi.it/masseroli/bbcm/dispense/9\\_BiomolecularDataBanks.pdf](http://www.bioinformatics.polimi.it/masseroli/bbcm/dispense/9_BiomolecularDataBanks.pdf).
- 2 Wilkins MR, Pasquali C, Appel RD, Ou K, Golaz O, Sanchez JC, Yan JX, Gooley AA, Hughes G, Humphery-Smith I, Williams KL, Hochstrasser DF. From proteins to proteomes: large scale protein identification by two-dimensional electrophoresis and amino acid analysis. *Bio/technology* (Nature Publishing Company). 1996 Jan; 14(1): p. 61-5.
- 3 Karl Fast, Fred Leise and Mike Steckel. What is a controlled vocabulary? [Online].; 2011. Available from: [http://www.boxesandarrows.com/view/what\\_is\\_a\\_controlled\\_vocabulary](http://www.boxesandarrows.com/view/what_is_a_controlled_vocabulary).
- 4 Masseroli M, Pinciroli F. Using Gene Ontology and genomic controlled vocabularies to analyze high-throughput gene lists: three tool comparison. *Computers in biology and medicine*. 2006 Jul-Aug; 36(7-8): p. 731-47.
- 5 Hermjakob H, Montecchi-Palazzi L, Bader G, Wojcik J, Salwinski L, Ceol A, Moore S, Orchard S, Sarkans U, et al. The HUPO PSI's molecular interaction format--a community standard for the representation of protein interaction data. *Nature biotechnology*. 2004 Feb; 22(2): p. 177-83.
- 6 Masseroli M., Galati O., Pinciroli F. GFINDER: genetic disease and phenotype location statistical analysis and mining of dynamically annotated gene lists. *Nucleic Acids Research*. 2005 Jul 1; 33(Web Server issue): p. W717-W723.
- 7 Masseroli M., Galati O., Pinciroli F. GFINDER: Genome Function INtegrated Discoverer through dynamic annotation, statistical analysis, and mining. *Nucleic acids research*. 2004 Jul 1; 32(Web Server issue): p. W293-300.
- 8 Aranda B, Achuthan P, Alam-Faruque Y, Armean I, Bridge A, Derow C, Feuermann M et al. The IntAct molecular interaction database in 2010. *Nucleic acids research*. 2010 Jan; 38(Database issue).
- 9 Chatr-aryamontri A, Ceol A, Palazzi LM, Nardelli G, Schneider MV, Castagnoli L,

. Cesareni G. MINT: the Molecular INTERaction database. Nucleic acids research. 2007 Jan; 35(Database issue): p. D572-4.