Universität Bielefeld

Technische Fakultät

Arbeitsgruppe Bioinformatik / Medizinische Informatik

Semi-automated Reconstruction of Biological Networks Based on a Life Science Data Warehouse

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur (**Dr.-Ing.**)

vorgelegt der Technischen Fakultät
der Universität Bielefeld

von:	Dipl.-Inform. Benjamin Kormeier
geb. am:	12. August 1977 in Bielefeld

Benjamin Kormeier:
Semi-automated Reconstruction of Biological Networks
Based on a Life Science Data Warehouse

Der Technischen Fakultät der Universität Bielefeld
am 09. März 2010 vorgelegt,
am 27. Mai 2010 verteidigt und genehmigt.

Gutachter:
Prof. Dr. Hofestädt, Universität Bielefeld
Prof. Dr. Conrad, Heinrich-Heine-Universität Düsseldorf

Prüfungsausschuß:
Prof. Dr. Cimiano, Universität Bielefeld
Prof. Dr. Conrad, Heinrich-Heine-Universität Düsseldorf
Prof. Dr. Hofestädt, Universität Bielefeld
Dr. Kopp, Universität Bielefeld

148 Seiten
55 Abbildungen
16 Tabellen

Gedruckt auf alterungsbeständigem Papier (ISO 9706)

Abstract

The progress in the area of biological research in recent years leads to a multiplicity of different databases and information systems. Typically, those data is available via the world wide web for further investigation. Usually, biological and life science data that describe different aspects of a biological system are distributed and spread over the whole world. Moreover, molecular biology deals with complex problems and an enormous amount of versatile data will be produced by high-throughput techniques. Hence, the total number of databases, as well the data itself, is continuously increasing, whereas rises the distribution and heterogeneity of the data. For a comprehensive and efficient use of life science data it is necessary to integrate the distributed and heterogeneous data and provide them for further analysis to the researcher. Beside data integration the user has to be supported by applicable tools for navigation within the integrated data sets that supports an efficient and precise processing of the data.

The importance of database integration has been recognized for many years. Therefore, this work describes an evolving data warehouse infrastructure for constructing life science data warehouses that integrate multiple heterogeneous biological databases within a single physical data physical database management system to facilitate queries that span multiple databases. In addition, the accurate representation of the integrated research data in a user-friendly format is high demanded among scientists. Information must be visualized in a clear and understandable way. Otherwise important information can get lost. Therefore, a specific data warehouse approach related to cardiovascular disease and a general data warehouse approach to browse and explore life science data are presented. The systems enable intuitive search of integrated life science data, simple navigation to related information as well as visualization of biological domains and their relationships. In addition, this thesis presents a software framework for visualizing and modeling biological networks. The user is able to create a user-specific pathway without any restrictions. Moreover, the editor is connected to the data warehouse approach, so that the user can take advantage of a wide range of biomedical data sources. Additionally, an easy-to-use web-based application for modeling of biological networks as Petri nets is motivated. The system supports semi-automatic generation of hybrid Petri net models. Then, it is possible to use generated networks in external simulation environments for qualitative and quantitative simulation. The work was funded in context of the *Cardioworkbench* EU project. Thus, the applications are already in use within the Cardioworkbench project as well as in ongoing in-house projects.

As result, this work presents a powerful and flexible data warehouse infrastructure that can be used for building project specific information systems and data warehouses. Furthermore, the system is the basis for the network modeling application pathway reconstruction tool. Finally, this work shows the semi-automated reconstruction approach of biological network based on life science data integration supported by the developed tools within this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

AC	Accession number
ASCII	American Standard Code for Information Interchange
AFL	Academic Free License
API	Application Programming Interface
ASP.NET	Active Server Pages .NET language
BioPax	Biological Pathways Exchange Format
Bio-SPICE	Biological Simulation Program for Intra- and Inter Cellular Evolution
BRENDA	Braunschweiger Enzymdatenbank
CAS	Chemical Abstracts Service
CAD/CVD	Cardiovascular Disease
CI	Cell Illustrator
CRUD	Create, Retrieve, Update, Delete
CSML	Cell System Markup Language
DBMS	Database Management System
DCM	Dilatative Cardiomyopathy
DNA	Deoxyribonucleic acid
DWH	Data Warehouse
EBI	European Bioinformatics Institute
ER	Entity Relationship
ETL	Extraction-Transform-Load
EU	European Union
FDBS	Federated Database System
FPN	Functional Petri Net
FTP	File Transfer Protocol

GML	Graph Modelling Language
GNU	GNU's Not Unix (reverse acronym)
GO	Gene Ontology
GraphML	Graph Markup Language
GRN	Gene regulatory network
GUI	Graphical User Interface
HDN	Hybrid Dynamic Net
HFPN	Hybrid Functional Petri Net
HGP	Human Genome Project
HPN	Hybrid Petri Net
HPRD	Human Protein Reference Database
HQL	Hibernate Query Language
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IUBMB	International Union of Biochemistry and Molecular Biology
IUPAC	International Union of Pure and Applied Chemistry
JAR	Java Archive
JDBC	Java Database Connectivity
JFC	Java Foundation Classes
JPA	Java Persistence Architecture
JRE	Java Runtime Environment
JSP	Java Server Pages
JUNG	Java Universal Network/Graph framework
KEGG	Kyoto Encyclopedia of Genes and Genomes
MINT	Molecular Interaction database
MPL	Mozilla Public License
mRNA	messenger Ribonucleic acid
NAR	Nucleic Acids Research
ODS	Operational Data Store
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
OMIM	Online Inheritance in Man
ORM	Object-Relational Mapping
PDB	Protein Data Bank
PHP	Hypertext Preprocessor (reverse acronym)
PIR	Protein Information Resource
PKU	Phenylketonuria
PNG	Portable Network Graphic
PNML	Petri Net Markup Language
PNTD	Petri Net Type Definition
PPI	Protein-Protein Interaction
PSI-MI	Proteomics Standard Initiative Molecular Interaction XML Format
P/T net	Place/Transition net (Petri net)

RDBMS	Relational Database Management System
RNA	Ribonucleic acid
SBML	System Biology Markup Language
SIB	Swiss Institute of Bioinformatics
SCOP	Structural Classification of Proteins
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SRS	Sequence Retrieval System
SVG	Scalable Vector Graphics
UML	Unified Markup Language
URL	Uniform Resource Locator
WWW	Worl Wide Web
XML	Extensible Markup Language

1 Introduction

The first chapter gives an overview of this thesis. In section 1.1 the case of data integration of biological and life science data will be delineated. Furthermore, the motivation for modeling, visualization of life science data, as well as the simulation of biological networks will be illustrated. The main objectives of this work will be introduced in section 1.2 and the structure of this thesis will be presented in section 1.3.

1.1 Motivation

The volume of biological knowledge is increasing significantly. Determining the biological function of genes and understanding the interaction of metabolism has become a major challenge in the post-genomic era. Several software tools allow scientists to integrate complex metabolic networks that take place in the living cell. Various biological databases have been created to represent, analyze and gain deeper insights into the complex processes and interactions inside organisms. The latest Nucleic Acids Research (NAR) database issue describes about 1,000 different molecular biology databases [GC09].

The importance of data integration in bioinformatics has been recognized for many years. Molecular biology research without computer-based analysis and data management is unthinkable. Hence, it is essential for scientists to access and analyze information from multiple heterogeneous data sources to meet their objectives. Molecular biological data has a high heterogeneity that is caused by experimental data extracted from a series of experiments. Molecular biology deals with complex problems, hence enormous and versatile data will be produced. The total number of databases, as well the data itself, is continuously increasing, alongside increases in the complexity in distribution and heterogeneity of the data. This data heterogeneity causes big problems in molecular biological data integration. Therefore, this thesis presents a bioinformatics data warehouse that integrates biological data from different public sources into a local database management system. This approach can be used as a general software infrastructure for bioinformatics research and development.

Due the huge amount of biological data provided by integrated information systems and data warehouses, an accurate visualization and analysis of those data becomes more and more important in bioinformatics. Hence, software applications which provide visualization, analysis tools and information systems are in huge demand among scientist. Many groups have contributed to the task of developing software solutions which try to meet those requirements. Usually, such software solutions focus on a particular problem. But none of them are that powerful they would be able to address all problems. After a detailed research it was clear that none of the existing software solutions can provide a clear and complete answer to these questions. Additionally, no existing software solution is able to handle and analyze acquired data sets from biological experiments. Therefore, a software framework is necessary that gives the researcher the opportunity to characterize biological questions with all its relevant details.

1.2 Aims

The integration of life science and biological data from heterogeneous, autonomous and distributed data sources is an important task in bioinformatics. The challenge is to integrate huge data sets regarding the large heterogeneity of the databases on the semantic and technical level. Therefore, the aims for the *Cardioworkbench* is to implement data warehouse software kit that integrates biological information from multiple public life science data sources into a local database management system. Several widely used life science information systems should be integrated, such as: UniProt, KEGG, OMIM, GO, Enzyme, BRENDA. The system should enable intuitive search of integrated life science data, simple navigation to related information as well as visualization of biological domains and their relationships. Furthermore, tools for modeling and visualizing of biochemical pathways should be developed. Moreover, the biological network models should be exported to a simulation environment such as Cell Illustrator by using standardized export formats.

So, the main objective of this work is to establish a general and flexible data warehouse infrastructure for biological and life science data that is independent from the underlying relational database management system. Configuration of the infrastructure and its tools should be possible via a graphical user interface (GUI) and standardized formats such as Extensible Markup Language (XML). Relational database systems are universalized and well established, because of their flexibility and robustness. The relational model makes it possible to protect and guarantee the integrity of data for relational databases. There exist several relational database management systems, so that a data warehouse infrastructure has to be independent from the underlying database management system. Therefore, a persistence layer is necessary to achieve this independence from the database management system. Most integrated database systems and data warehouse approaches available are not up-to-date or must be manually updated. Therefore, the monitoring of data sources is needed to keep the system up-to-date. Moreover the integration process should be observed and in case of failures to start a simple recovery process. This error recovery algorithm guarantees a consistent state of a data warehouse. Finally, the data warehouse approach should provide an easy-to-use graphical user interface for administration and configuration. The system architecture of system should follow a general data warehouse design.

Moreover it is necessary to provide a software framework that gives the researcher the opportunity to characterize biological questions with all its relevant details based on available biological data. Many biological processes are involved in a cell. Typically, those processes and their elements can be modeled as network. The accurate representation of research data is one of the main tasks in computer science. Information has to be visualized in a clear and understandable way to meet the aims of research activities. Otherwise important information can get lost. One of the most important tasks is on the one hand the data acquisition and on the other hand developing a network editor with graphical user interface. So, it is possible to explore integrated life science data for relevant information that might be helpful for biological and medical research. Furthermore, the user should be able to create, edit and investigate biological networks supported by the life science data. The researcher should be able to handle any kind of biological network with all its biological and medical elements. Moreover, a user should also have the possibility to share and export the biological networks using different widely used file formats.

1.3 Structure

The work presented in this thesis is outlined in the following. The *Basics* in the relevant disciplines are introduced and reviewed in chapter 2. This includes on the one hand, biological basics in general with special regard to database integration. On the other hand the current state of molecular biological databases, their heterogeneity and the integration of molecular biological databases is described.

In chapter 3 relevant integration approaches in field of data warehouses as well as modeling and simulation software approaches will be introduced. The focus will be on several widely used data warehouse approaches. Additionally, all data integration approaches will be compared based on multiple criteria. Moreover, the most relevant data sources for database integration will be shortly introduced. These are databases in the field of metabolic pathways, signaling pathways, nucleotide sequences, proteins, enzymes and diseases. Finally, this chapter deals with applications and tools that were realized in order to create, edit and analyze biological networks. It gives an insight into the related network modeling software solutions and approaches. A selection of widely used modeling and simulation environments will be given.

The principles and requirements of database integration system will be presented in the fourth chapter *Life Science Data Integration*. General requirements for the integration of biological databases are presented regarding to their feasibility of implementation in the *BioDWH* data warehouse infrastructure. Particularly the functions of the software toolkit will be illustrated more precisely. Furthermore, the realization, the structure as well as the function of the data warehouse infrastructure will be illustrated.

Chapter 5 gives an insight into the concept of building web-based data warehouse systems based on the requirements that are delineated in the first section of the chapter. Two novel web-based DWH information systems, *CardioVINEdb* and *DAWIS-M.D.*, will be

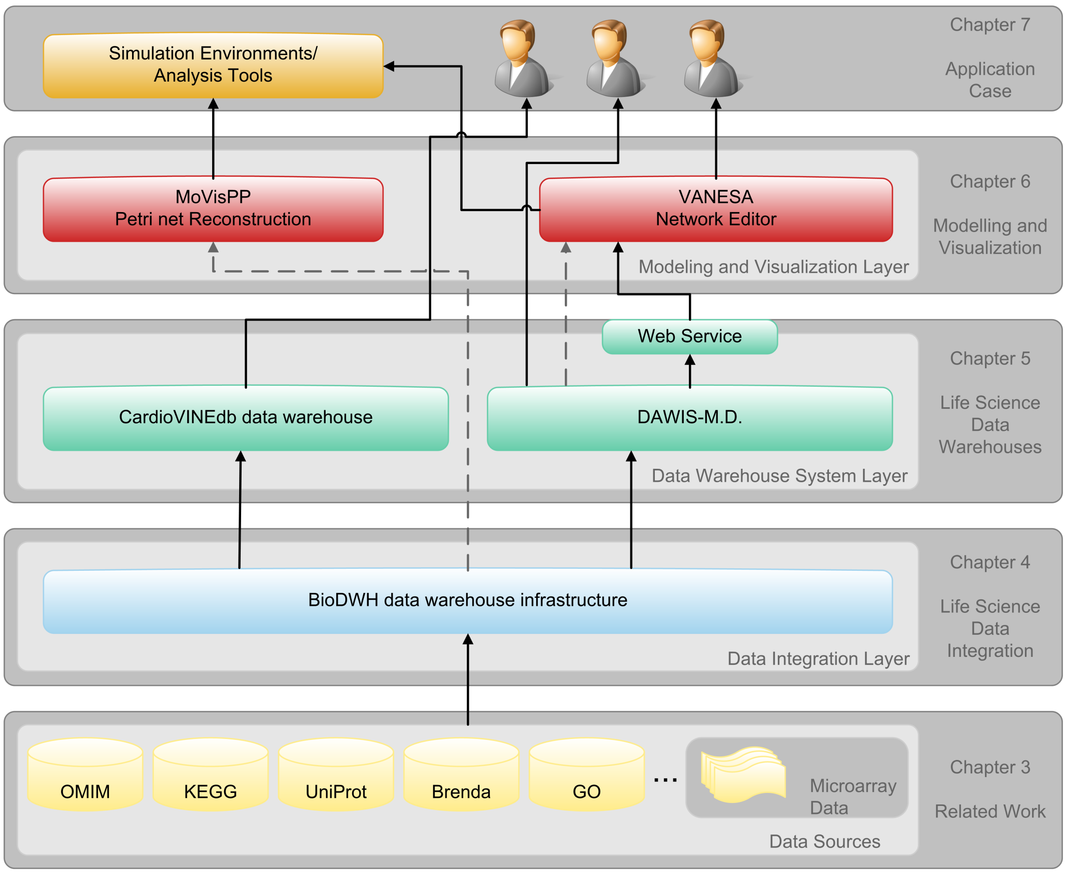

Simulation Environments/ Analysis Tools
Chapter 7
Application Case
MoVisPP Petri net Reconstruction
VANESA Network Editor
Chapter 6
Modelling and Visualization
Modeling and Visualization Layer
Web Service
CardioVINEdb data warehouse
DAWIS-M.D.
Chapter 5
Life Science Data Warehouses
Data Warehouse System Layer
BioDWH data warehouse infrastructure
Chapter 4
Life Science Data Integration
Data Integration Layer
OMIM
KEGG
UniProt
Brenda
GO
...
Microarray Data
Chapter 3
Related Work
Data Sources

Figure 1.1: Overview of the applications for the reconstruction of biological networks based on life science data integration and their relation ordered by chapter.

introduced. Details of the integration process, the integrated database schema, the system architecture and its components as well as the structure and function of each system will be presented.

An insight into the concept of modeling and visualization of biological networks based on life science data integration is described in chapter 6. Therefore two modeling and visualization software approaches *MoVisPP* and *VANESA* will be introduced. For both systems, the concept of modeling, the concept of visualization, the integrated databases, the system architecture and its components will be described. Furthermore, the structure and function of each application will be illustrated based on different figures.

The *Application Case* chapter presents two applications for semi-automated reconstruction of biological networks. First, the *Cardioworkbench* project will be introduced. In detail the project focused on atherosclerosis. The aim of Cardioworkbench consortium that consists of 11 partners from 7 different countries is to improve the target selection/-validation process and optimize drug design for cardiovascular diseases. In this context, the projects *BioDWH, CardioVINEdb* and *MoVisPP* were developed. The first application describes the semi-automated construction of biological networks based on Petri Nets. For this application, the two genes coming from the Cardioworkbench project were investigated using the CardioVINEdb information system and MoVisPP network visualization web application. The second application presents the relation between the cytoplasmic proteome profile and the metabolic pathways of three samples coming from patients of the Cardioworkbench project affected by dilatative cardiomyopathy. The proteome analysis has been done by using microarray experiments. To characterize the specific functional pathways, that are dysregulated in each pathological sample, the DAWIS-M.D and VANESA system were used.

The last Chapter 8 summarizes the work that has been done and discusses ongoing and further development possibilities. Additionally, in the discussion, the presented software tools are compared to existing database integration and visualization systems. It concludes the results that have been carried out, the possibilities and the perspectives for the future.

An overview of the different software approaches related to their chapter in this thesis could be found in figure 1.1.

2 Basics

In this chapter, a brief introduction to relevant basics of biology and computer science is given for a better understanding of the present work. This work is addressed to computer scientists, biologists and those who are interested in bioinformatics. Therefore, an understanding of the properties of the different subjects in academics in interdisciplinary sciences is important. An overview of the most relevant basics will be presented in the following.

In the first section, the basic terms in the field of molecular biology will be explained. They are important for an understanding and reflection of different life science data sources. Following, definitions and techniques of computer science that are necessary for this work will be presented. In addition, problems in data integration and the data warehouse approaches will be outlined.

2.1 Biological Basics

Since Human Genome Project, gene therapies and cloning the deoxyribonucleic acid (DNA) becomes more and more important in scientific research, and recently the public have a great interest. In face of vast progress in the field of biochemistry intra and extra cellular processes are highly complex. High throughput sequence investigation tools, array technologies for gene/protein analysis generate more and more data, but far away from being able to represent these complex processes in a computer. A first step in reaching this goal is to understand the metabolism, which is based on gene-controlled biochemical reactions. Accordingly, modeling and simulation of metabolic networks is important. Therefore, basic elements such as enzymes and biological pathways will be discussed in the next sections.

2.1.1 DNA as Carrier of Genetic Information

The genome of a species is organized in chromosomes, those again are composed of nucleic acids. Indeed, deoxyribonucleic acid (DNA) and ribonucleic acid (RNA) are two kinds of nucleic acids that can be found in living organisms. DNA has two purine bases adenine and guanine, while the pyrimidines bases are thymine and cytosine; in RNA thymine is replaced by uracil (table 2.1).

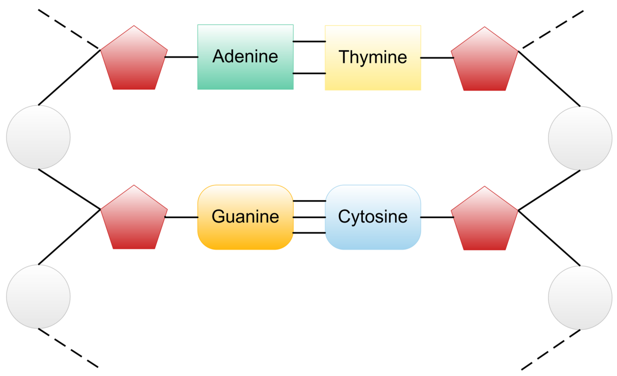
Adenine	A
Guanine	G
Thymine	T
Guanine	G
Uracil	U

Table 2.1: Nucleotide bases with their abbreviation.

Chemically, a DNA consists of two long polymers of nucleotides with backbones made of sugars and phosphate groups joined by ester bonds (see figure 2.1). Each type of base on one strand forms a bond with just one type of base on the other strand, so-called complementary base pairing. The arrangement of two nucleotides binding together across the double helix is called a base pair. Complementarity means all information in the double-stranded sequence of a DNA helix is duplicated on each strand, which is important in DNA replication. The two DNA strands run in opposite directions to each other, so-called anti-parallel. Hence, the DNA can be pulled apart like a zipper. Usually, the double helix is a right-handed spiral. As the DNA strands wind around each other, they leave gaps between each set of phosphate backbones, revealing the sides of the bases inside. There are two kinds of these grooves twisting around the surface of the double helix. A first model of DNA structure was developed by Watson and Crick in 1953 [WC53].

Finally, the DNA is a set of blueprints that contains the instructions needed to construct other components of cells, such as RNAs, proteins, and other molecules. Modifications of genetic information or the genetic code are called *mutations*. Mutations evolve from environmental conditions or from instability of nucleotide bases of the DNA. Those modifications are rare, but they can cause in some cases diseases. Mutated genes could cause deficiencies protein synthesis of inoperable proteins. Different kinds of mutation are given here:

- ⋆ *Genome mutation* is a drastic change of the whole genome, for instance in the number of chromosomes that leads to trisomy.
- ⋆ *Chromosome mutation* are changes in the form and structure of the chromosome including:
  - *Translocation:* Shift of a part of the chromosome from its origin to another location of the same chromosome or another chromosome.
  - *Deletion:* Loss of chromosome parts.
  - *Insertion:* Assembly of a DNA fragment into the chromosome.
  - *Inversion:* Twist of a chromosome segment.



(a) Abstract DNA model.

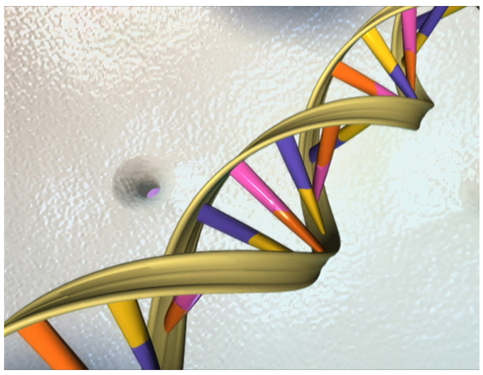

(b) DNA double helix.

Figure 2.1: DNA is a double helix structure made of nucleotides. Each nucleotide consists of a deoxyribose sugar (red pentagon), a phosphate group (grey circle), and one of four bases. Sugars and phosphates join to make a double backbone and bases are joined by hydrogen bonds. Adenine joins with thymine and guanine with cytosine. The major groove is 22 Å wide and the other, the minor groove, is 12 Å wide. (DNA double helix taken from http://www.genome.gov/pressDisplay.cfm?photoID=96)

2.1.2 Proteins and Enzymes

Proteins are basic molecules and involved in most of biochemical reactions in a cell. A protein is made of amino acids arranged in a linear chain and joined together by peptide bonds. There are 20 possible types of amino acids involved in a protein. The sequence of amino acids is defined by the sequence of a gene, which is encoded in the genetic code of the DNA. An ordinary polypeptide chain is usually between 100 and 800 amino acids. Sequences with less than 20 amino acids are called *peptides*. The activity of a protein is highly connected to the structure of the protein. Prediction of the three dimensional structure based on the sequence is still an open question. In addition, the folding of a protein is important for interaction between proteins and other molecules. However, in many biological processes proteins play a major role. A human organism has about 50.000 different proteins, they act as catalysts in chemical reaction or take over or conduct other important functions from transport of specific molecules up to immune defense. Proteins are macromolecules that have the ability to react specifically with various molecules. Those proteins have so-called functional groups where substances (substrates) they catalyze are able to dock. This mechanism follows the principle of coaptation or *lock and key* model [Fis94].

Enzymes are a subset of proteins and act in most biochemical reactions as catalysts. Those biomolecules increase the rates of chemical reactions by between 10^8 to 10^{20}. Catalysts, i.e. enzymes, are not consumed by the reactions they catalyze. In addition, enzymes differ from most other catalysts by being much more specific. About 4.000 enzymes are known to catalyze biochemical reactions [Bai00]. The activity of an enzyme can be affected by other molecules such as inhibitors, activators or cofactors. Inhibitors are molecules that decrease enzyme activity, while activators increase activity. A lot of drugs and poisons are enzyme inhibitors. Temperature, the chemical environment and the concentration of substrates can also affect the enzyme activity.

The name of an enzyme is often derived from its substrate or the chemical reaction it catalyzes, usually ending with *-ase*. Well known examples are DNA polymerase, acetylcholinesterase or lactase. Isozymes are enzymes with the same function having the same basic name. They differ in amino acid sequence but catalyze the same chemical reaction. However, the International Union of Biochemistry and Molecular Biology (IUBMB) has developed a nomenclature called *EC numbers* for enzymes [Bar97]. Each enzyme is described by a four digit number starting with "EC". In following the top-level classification of EC nomenclature is described.

- ⋆ EC 1: Oxidoreductases
- ⋆ EC 2: Transferases
- ⋆ EC 3: Hydrolases
- ⋆ EC 4: Lyases
- ⋆ EC 5: Isomerases
- ⋆ EC 6: Ligases

For instance, the number of lactase is 3.2.1.108, which indicates that this enzyme is a hydrolase (EC 3.*.*.*), that acts on "glycosylases" (EC 3.2.*.*), hydrolysing *O*- and *S*-glycosyl compounds (EC 3.2.1.*).

2.1.3 Gene Expression

The blueprint for protein synthesis is coded in the DNA double helix as described in section 2.1.1. *Gene expression* is the whole process from reading genetic information to the completed protein. Localization of DNA in a cell depends on the organism. In prokaryotes the DNA is located in the cytoplasm (cytosol), while in eukaryotes the DNA is located in the cytoblast (nucleus).

Gene regulation gives the cell control over structure and function. The process of gene expression starts with transcription. First, an enzyme (RNA polymerase II) binds to the DNA and the double strand will be opened. Several proteins help the RNA polymerase II to perform its task. These proteins are called *transcription factors*. They build up, step by step, the pre-initial complex for the start of transcription. However, the RNA polymerase produces a complementary nucleotide RNA strand, called *messenger RNA* (mRNA). Translation is the production of proteins by decoding mRNA produced in transcription. The cytoplasm is where the ribosomes are located and the translation occurs.

Ribosomes are made of a small and large subunit which surrounds the mRNA. Now, mRNA is decoded to produce a specific polypeptide according to the rules specified by the genetic code. Thereupon, the three dimensional folding of the protein based on the amino acid sequence occurs.

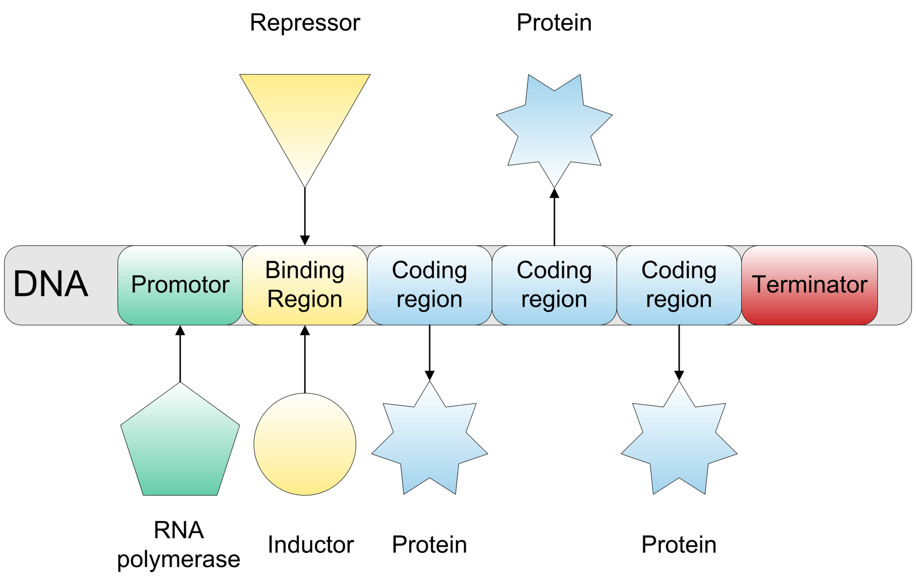


Figure 2.2: Simplified principle of gene regulation in prokaryotes.

A gene is a segment of DNA that contains a coding sequence, that determines when the gene is expressed or active. The human genome has about 30.000 genes. Initiation of transcription starts with the binding of RNA polymerase at the promoter region of the gene. The efficiency and specificity with which a promoter is recognized depends on the participating DNA motifs, which are recognized by RNA polymerase. Next region is the RNA coding sequence which is transcribed into mRNA during transcription. Finally, transcription stops when RNA polymerase reaches a sequence called *terminator*. Overall, there are two opposed regulation mechanism.

- ⋆ Negative gene control (repressor)
- ⋆ Positive gene control (inductor)

A negative gene control is a systematic cutoff of a gene. This is necessary if sufficient amount of a substance is available in the cell. In this case a repressor is required. On the other hand, if a specific substance is not available in the environment of a cell and is required, then an activation of gene expression is necessary. This mechanism is called positive gene control. Usually, an inductor is required to start transcription.

Figure 2.2 shows a generalized principle of gene expression control in prokaryotes. A small molecule (inductor) assembles to a specific binding region near the promoter. The efficiency and specificity with which a promoter is recognized depends on the participating DNA motifs, which are recognized by upstream factors and those inducible factors. Now, RNA polymerase is able to start transcribing. One well known model to explain

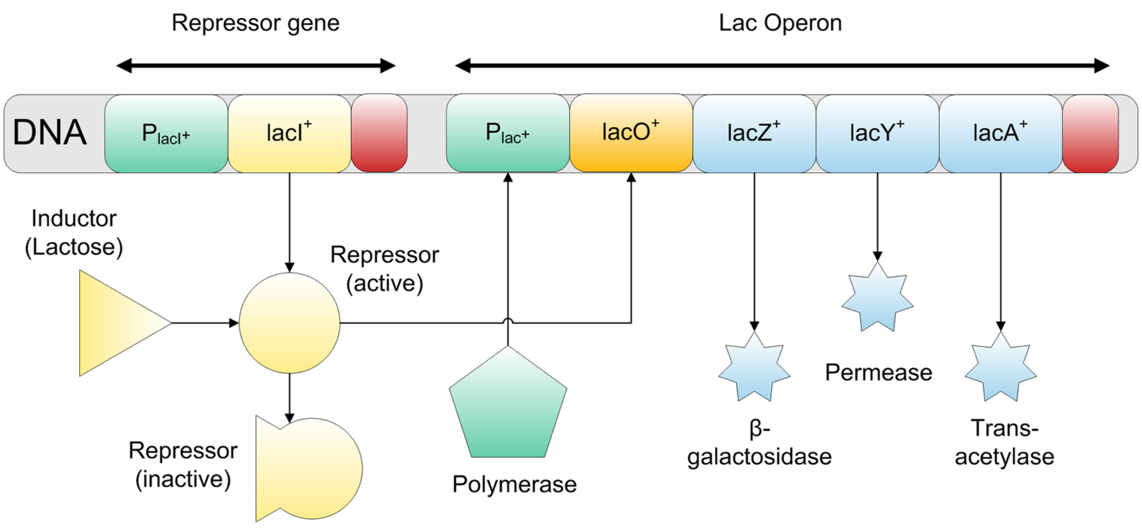


Figure 2.3: Regulation of lac operon in *E. coli*.

gene regulation in prokaryotes is the operon model by Jacob and Monod [JM61]. Figure 2.3 represents the genes lacZ, lacY and lacA in an operon model, they are clustered in a transcription unit. The regulatory response to lactose is controlled by an intracellular regulatory protein called lactose repressor to inhibit production of β-galactosidase in the absence of lactose. A structure gene called lacI for the repressor lies nearby the lac operon and is always expressed. If lactose is missing the repressor binds to a short DNA sequence downstream of the promoter near the beginning of lacZ called the lac operator. However, in presence of lactose the repressor binds to the DNA and the proteins β-galactosidase (lacZ), β-galactoside permease (lacY) and β-galactoside transacetylase (lacA) are expressed. On the other hand, in absence of lactose, the repressor binds to the operon and transcription is interrupted. Reduction of lactose releases the repressor protein and transcription is to be continued.

In comparison to prokaryotes, gene regulation in eukaryotes is much more complex. The eukaryotic RNA polymerase requires an initiation complex of transcription factors to bind on the DNA. Furthermore, specific short consensus elements located upstream of the promoter region which increases (enhancer) or decreases (silencer) the transcription rate.

2.1.4 Biological Pathways

In previous sections cellular processes that are based on molecular processes which are catalyzed by enzymatic reaction were discussed. Therefore, the gene regulation process is fundamental in molecular processes. The functioning of these molecular processes and metabolic systems has been studied in the last decades. High-throughput sequence tools, array technologies for gene and protein analysis, as well as the electronic laboratory infrastructure for the investigation of molecular data do support the understanding of metabolic systems in the cell. However, the role of metabolic networks and their role in context in a living cell are still vague, due to lack of information and their extreme complexity. Biological networks on molecular level can be divided into four classifications [JS08].

- ⋆ Gene regulatory networks
- ⋆ Metabolic networks
- ⋆ Signal transduction networks
- ⋆ Protein interaction networks

Metabolic networks deal with flow mass and energy, while gene regulation processes are involved to transform genetic information, i.e. gene, into the encoded protein. Signaling networks recognize intra/extra cellular stimuli and transduce their recognition into a response. Usually, this response is a change in cellular activity. Protein interaction networks are typically generated out of different large-scale approaches by genetic, biochemical and biophysical techniques [vMKS+02]. In general, biological pathways are biological processes that occur in metabolic systems of a cell.

2.1.4.1 Metabolic Pathways

Metabolic pathways are one of the most investigated and studied field in bioinformatics and biology. However, living cells require energy for many processes such as storing molecules, replication and repair of DNA, movement and other processes. Metabolism is a set of chemical reactions that are catalyzed by enzymes. Those processes are highly organized [KHK+05]. A metabolic network consists of biochemical reactions that transform by modification one type of molecule into another type. Now, a metabolic pathway is a particular part of a metabolic network, or the other way around, a collection of metabolic pathways is called metabolic network. Traditionally, metabolic pathways are given by (wall) chart, such as the Boehringer Mannheim, now Roche chart [Mic99]. But more and more charts are digitally available as dynamic maps via the internet like in KEGG (see figure 2.4) or Reactome.

Each metabolic pathway is set of chemical reactions catalyzed by enzymes. Enzymes have an *active site* where a specific molecule, called substrate, can bind. Furthermore, the enzyme transforms the substrate into a specific product. An enzymatic reaction can look like this:

$$E + S \rightleftharpoons ES \rightleftharpoons EP \rightleftharpoons E + P$$

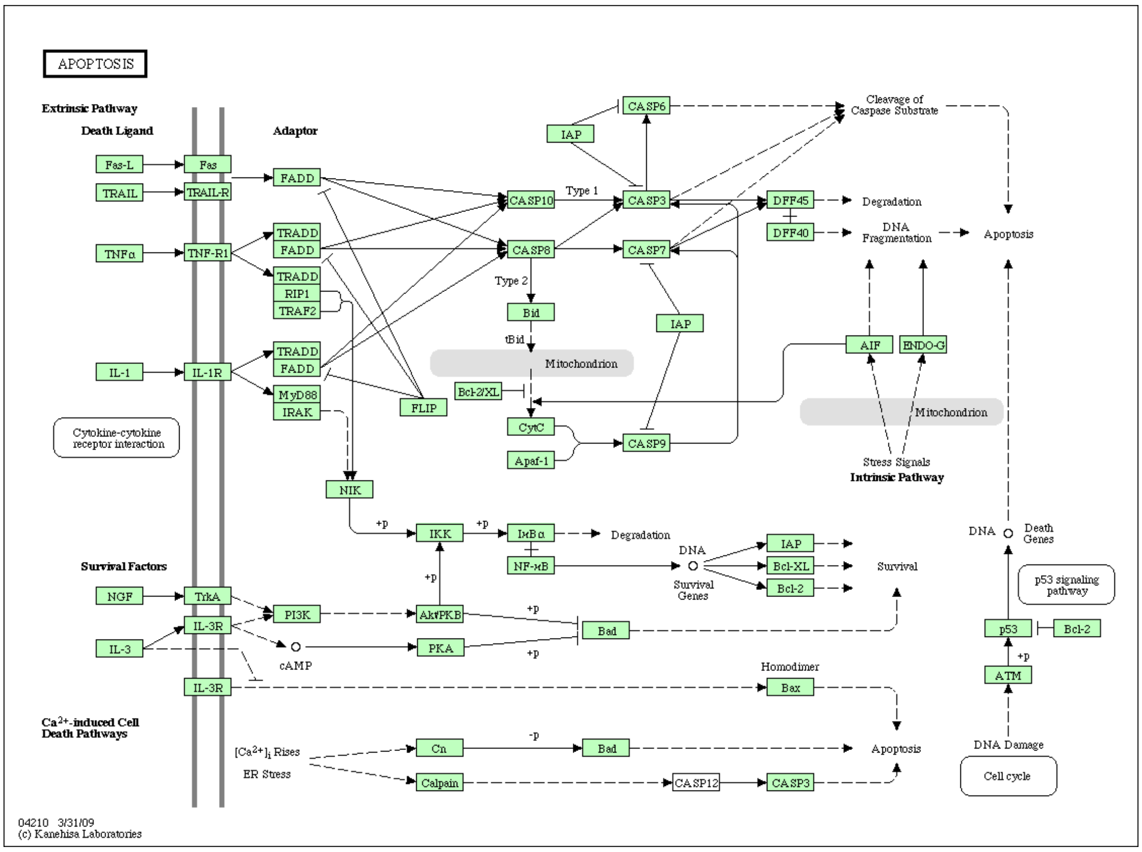


Figure 2.4: Graphical representation of the human apoptosis pathway. (Taken from http://www.genome.jp/kegg/pathway/hsa/hsa04210.html)

First, enzyme E and substrate S forms an enzyme-substrate complex ES, afterwards an enzyme-product complex EP. Finally, the enzyme-substrate complex is split by a chemical step into enzyme and product P, while the product will be released. Enzymes are able to catalyze several million reactions per second. Most of these processes are bi-directional. The flux of biosynthetic processes is controlled by enzyme affinity, enzyme concentration and reaction rate. Kinetic affects are important regarding to concentration rates. The kinetic behavior of an enzyme seems to be enzyme specific. Typically, most biochemical reactions follow *Michaelis-Menten kinetic*. In 1923 L. Michaelis and M. Menten developed a kinetic model for enzymes. The enzyme velocity V_0 is a function of substrate concentration:

$$V_0 = \frac{V_{max}[S]}{K_m+[S]}$$

where $[S]$ substrate concentration, K_m Michaelis constant, V_{max} the maximum rate of reaction.

Typically, in computer science a metabolic pathway is illustrated as a graph (V, E), where V is a finite vertex set and E is a finite set of edges. Each vertex represents a metabolite and each edge displays a biochemical reaction that is catalyzed by an enzyme. Undirected graphs are graphs where connections between vertices are without a direction. In directed graphs edges are also called arcs, an edge between two vertices u and v is represented by the ordered vertex pair (u, v). Usually, the direction of an edge is illustrated by an arrowhead at the end of the edge [JS08]. In living cells the metabolite flow is largely uni-directional, therefore an irreversible directed graph is usually used to model metabolic pathways.

2.1.4.2 Gene Regulatory Networks

In the last decades *gene regulatory networks* (GRNs) have been studied extensively by scientists. Gene regulatory networks are one of the important biological processes in the organizational level in the cell where signals from the cell state and the outside environment are integrated in terms of activation and inhibition of genes [CH08]. They are the on/off switches and controller of a cell operating at the gene level. GRNs dynamically influence the level of expression for each gene in the genome. That means whether or how strong a gene will be transcribed into RNA. A simple GRN consists of one or more input signaling pathways, regulatory proteins that manipulate the input signals, some target genes, and of course the RNA and proteins produced by the target genes (see figure 2.5). In many cases GRNs include dynamic feedback loops that provide further network regulation and output.

In general, GRNs act similar to biochemical computer models [Gar69] to specify the identity and level of expression of the target genes. Essential for this computation are DNA recognition sequences with which the efficiency and specificity transcription factors are able to bind. These cis-regulatory elements[1] are often binding sites which one or more trans-acting factors. Every gene has its own cis-element that regulates its expression. Transcription factors can specifically repress (down-regulate) or induce (up-regulate) synthesis of the corresponding RNA. Finally, those regulations cause changes in the structure of the cell, the metabolic capacity in the cell, or behavior of the cell mediated by new expression levels of up-regulated proteins and elimination of down-regulated proteins.

Figure 2.6 illustrates an example of a gene regulation network. A $lambda$-phage is able to injects its DNA into its *E. coli* and then the phage usually enters the lytic or lysogenic pathway where it replicates its DNA. However, the replication rate is regulated by the amount of CII in the host cell. When neither CI nor Cro bind to the operator then cI and cro will be expressed. The concentration of Cro protein is regulated at some level by the feedback control of the Cro protein itself. If the concentration of CII protein is high, the CII protein binds to the promoter and anti-sense RNA of the gene cro is produced. Thereby concentration of Cro protein decrease and concentration of CI protein is regulated at some level by the feedback mechanism of the CI protein itself.

[1] *cis*[latin] means “on the same side as”

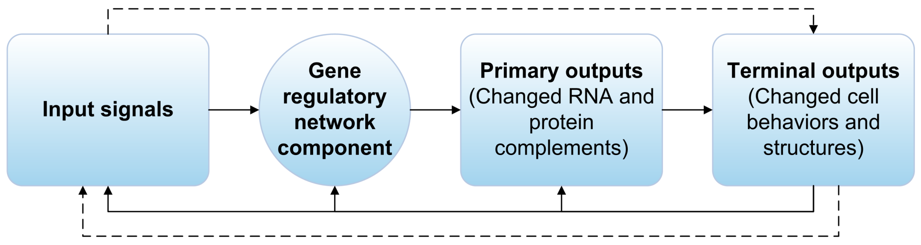

Input signals
Gene regulatory network component
Primary outputs (Changed RNA and protein complements)
Terminal outputs (Changed cell behaviors and structures)

Figure 2.5: Control process of a minimal Gene regulatory network (according to http://genomics.energy.gov). Beyond GRN boundaries are signaling responses and feedbacks which do not involve regulation of gene expression but instead act directly on proteins and protein machine assemblies (dashed arrows).

2.1.4.3 Signal Transduction Pathways

Cells have to communicate with each other. Intracellular signal transduction, in general, is a biochemical process where chemical signals from outside the cell passed through the cytoplasma to cellular systems. Signal transduction pathways are known for their non-linear, highly connected nature compared to metabolic pathways (described in section 2.1.4.1) or gene regulatory networks as described in section 2.1.4.2. Chemical signals from the cell environment are received by receptors that are located in the cell membrane. Those receptors can induce, by stimulation, the activation of a network of at least more than hundreds proteins. These networks are associated with highly cellular functions, often activate common sets of proteins [JI98, FMKL99, JLI00].

Cells recognize stimuli and transduce their recognition into response. This response is typically a change in cellular activity. A response is usually divided into three steps.

1. A stimulus activates a receptor. Typically, a chemical ligand binds to the receptor.
2. An active receptor transduces the stimulus into a chemical signal inside the cell. This implicates a change in the amount of small messenger molecules or a change in the activity of the messenger molecule. One type of signal (stimulus) is converted into another type of signal (messenger). This step is called *transduction*.
3. Those messenger acts upon several effector systems.

Now, the effector systems modify the state of the cell. Furthermore, the response at transduction and effector stages are influenced by signals from parallel signaling pathways or feed loops. Signals could be light, gases, steroid hormones or growth factors. Receptors are always proteins and about 20 families of receptors are known. Within each family multiple isoforms provide a wide range of receptors with specificity for particular stimuli (according to [PE02]). An example of a signal transduction pathway is given in figure 2.7. However, a full understanding of the mechanism of intracellular signal transduction is still one of the major challenges in cell biology [HKHR08].

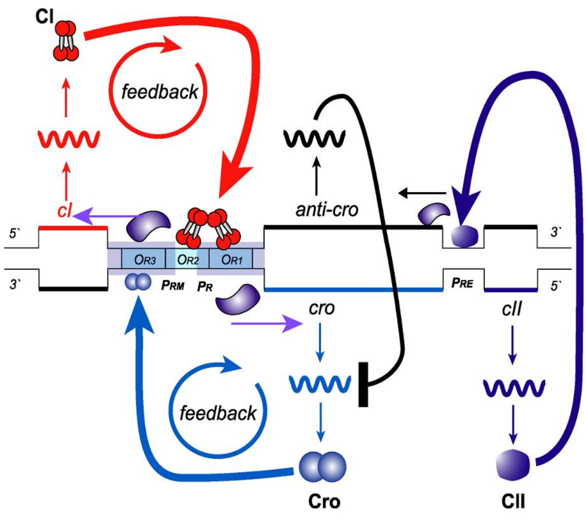


Figure 2.6: Transcription of the genes *cro*, *cII* in *E. coli*. The condition of *E. coli* gives an effect to the concentration of CII protein. If the concentration of CII protein is low, the transcription continues and keeps the concentration of Cro protein at some level by the feedback control of the Cro protein itself. If the concentration of CII protein is high, transcription of cI gene is followed and concentration of CI protein keeps at some level by the feedback control of the CI protein itself. (Taken from http://genomicobject.net/member3/index.html)

2.2 Molecular Database Integration

Since the Human Genome Project, the integration of fast growing data is one of the major challenges in bioinformatics. Today, high-throughput methods generate, in short time, data about the whole genome of an organism or a species. A manual analysis of this data or investigation of related data sources is no longer possible. Biologists have to be supported in their research by tools and applications which can analyze, compare or accumulate experimental data with external data sources or data from other experiments of other researchers. This challenge leads us to the problem of database integration which will be discussed in this section.

Typically, data of genomes, genes, proteins, enzymes, chemical compounds, diseases etc is stored in databases with worldwide availability. A good overview of important databases is provided by the annual special issue of *Nucleic Acids Research* [GC09]. The number of molecular databases is continuously increasing in the last decade (see figure 2.9). Several databases are connected with each other via specific links or foreign key relations. An example of relationships between different databases is given in Figure 2.8 for the DBGET database system [FGM+98]. Classification of databases is not a minor problem, because the biological data of a database could be very complex. A good classification is given by Galperin and Cochrane [GC09]. They divide biological database in the following 14 major classes.

1. Nucleotide Sequence Databases
2. RNA Sequence, Structure and Functions
3. Protein Sequence Databases
4. Structure Databases
5. Genomics Databases (non-human)
6. Metabolic Enzymes and Pathways; Signaling Pathways
7. Human and other Vertebrate Genomes
8. Human Genes and Diseases
9. Microarray Data and other Gene Expression Databases
10. Proteomics Resources
11. Other Molecular Biology Databases
12. Organelle Databases
13. Plant Databases
14. Immunological Databases

Some classes are divided into subclasses, it shows how complex and comprehensive the data structure could be. A detailed classification schema could be found in [GC09]. To perform a certain quality molecular databases have to support following standards [LR03].

⋆ *Transparency.* Users of the system do not need knowledge about data organization nor database queries of integrated data sources.

⋆ *Integrity.* The whole data of each and every data resource is accessible without restrictions via the system.

⋆ *Semantic correctness and non redundancy.* The database schema is semantically correct and each element is unique. That means semantically equal elements from different data sources have a unique global schema element. All data from the different data sources is accurately inserted into the system.

⋆ *Actuality.* Availability of up-to-date data.

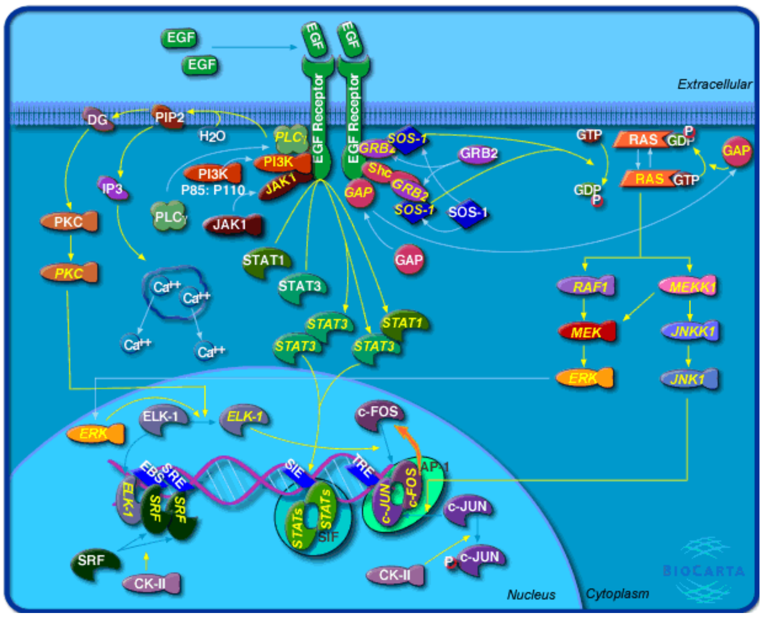


Figure 2.7: The epidermal growth factor (EGF) pathway.
(Taken from http://www.biocarta.com/pathfiles/h_egfPathway.asp)

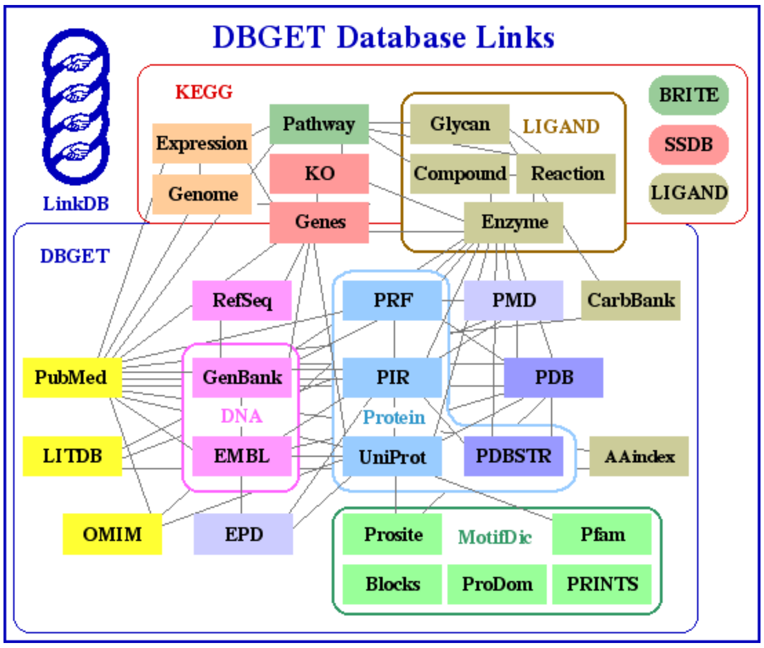


Figure 2.8: DBGET database links diagram.
(Taken from http://www.genome.jp/dbget/dbget.links.html)

⋆ *Performance.* For each application, the performance of database queries is very important. Particularly, for high-throughput experiments a high performance is necessary. Usually, virtual integration methods are not able to provide this performance, particularly in regard to distributed data sources. Normally, integration projects use hybrid or materialized approaches to ensure high performance.

On the basis of those properties it is possible to identify problems in the field of data integration. The following problems in integration of molecular databases can occur.

- ⋆ Large distributed data
- ⋆ Problem of autonomy
- ⋆ High redundancy
- ⋆ High technical/structural/semantic heterogeneity

In the next section problems of database integration will be discussed in more detail. Additional information about molecular database integration could be found in [LR03, LN07].

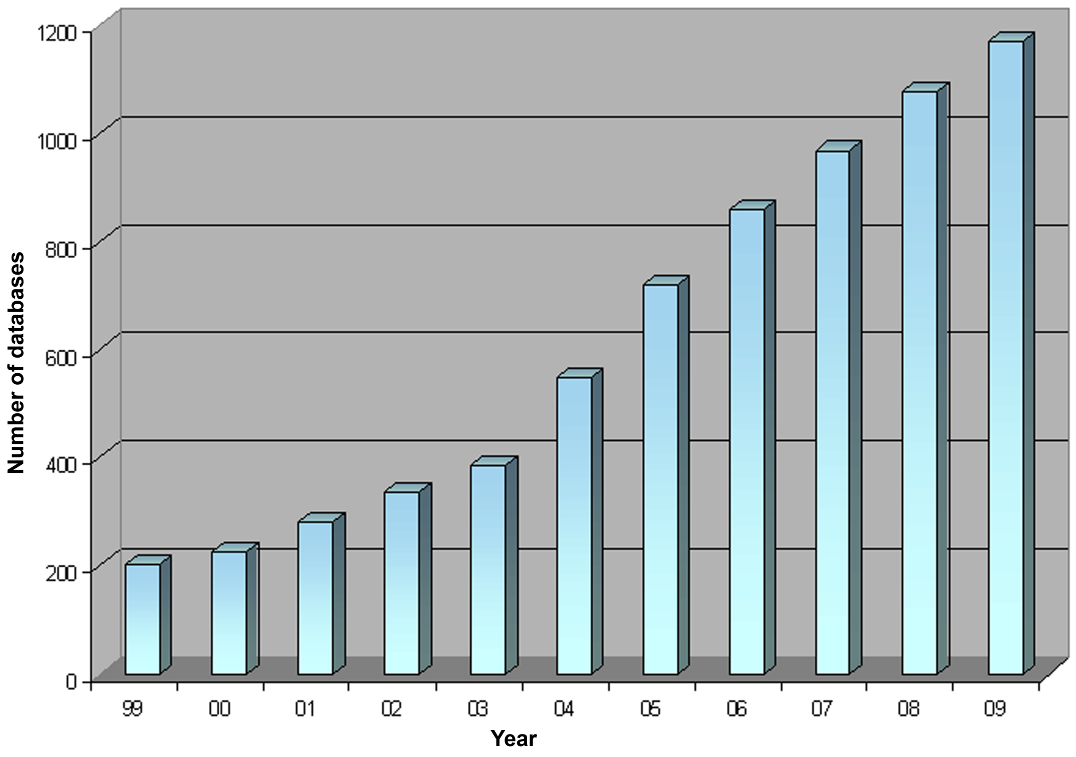


Figure 2.9: Increasing number of NAR listed databases in the last years [GC09].

2.2.1 Problems in Database Integration

Molecular biological data has a high *semantic heterogeneity* that is caused by (experimental) data extracted from a series of experiments. Molecular biology deals with complex problems, hence enormous and versatile data will be produced. The total number of databases, as well the data itself, is continuously increasing, as is rises the distribution and heterogeneity of the data. Particularly, data heterogeneity causes big problems in molecular biological data integration. *Technical heterogeneity* is caused by a high number of different formats and interfaces of the different data sources. Furthermore, the data is usually not available in a standard format which causes *structural heterogeneity*. Moreover, there is a level of semantic heterogeneity, because of missing standards and consensus for basic biological terms. Beside the problems of molecular biological databases there are some more problems in data integration. In the next sections basic problems of data integration in the field of *distribution*, *autonomy* and *heterogeneity* will be presented, and Leser and Naumann those fields are called "orthogonal dimensions of data integration" [LN07] will be discussed. Thereby, in each dimension can independently occur problems.

2.2.1.1 Distribution

Usually, data sources of an integrated system are distributed. That means, each and every source is located on separate systems and different locations. It will be distinguished between *physical* and *logical distribution*. Physical distribution means the data is physically and as well geographically organized on different distributed systems. That leads to following problems in data integration.

- ⋆ Localization of data
- ⋆ Data is represented in multiple schemata
- ⋆ Optimization of distributed queries

Problems of physical distribution can be solved by data warehousing. The topic of data warehouses will be discussed in section 2.2.2.4.

Logical distribution means that homogeneous data of the system is located at different logical places. Hence, the system is redundant and several problems can occur. The localization of this data is difficult and ambiguous. Now, a user has problems to track the origin of the data. A solution could be to provide metadata, for instance a global schema. Additionally, duplicates and conflicts can occur with logical distribution. These problems have to be identified and fixed by the system to guarantee consistent data.

2.2.1.2 Autonomy

The distribution of several data sources leads automatically to the problem of autonomy. Regarding to data integration, autonomy means independence of the data source that refers to access, configuration, development and administration. Generally, autonomy can divided into following four types [LN07].

- ⋆ Design autonomy
- ⋆ Interface autonomy
- ⋆ Access autonomy
- ⋆ Legal autonomy

Whether a data source has the freedom to decide how its data can be provided and represented, it is called *design autonomy*. This autonomy is related to the data model, schema and transaction management. A data resource has *interface autonomy*, whether it has the freedom to define the way of access. For instance, to define a protocol for the query language of the system. Interface autonomy is strongly related to design autonomy, because the storage of data typically determines the data access. The data source is access autonomous, whether the system is able to decide who can access which data. Legal autonomy is achieved, if the integration of a resource is prohibited. Additional kinds of autonomy can be found in [Con97].

2.2.1.3 Heterogeneity

The major problem of data integration is heterogeneity that is caused by autonomy. Moreover, distribution can also cause heterogeneity, but not in general. Two information systems that have identical methods, nor provide identical models and structures for data access, are called heterogeneous. Different kinds of heterogeneity are defined according to [LN07] as follows.

- ⋆ Technical heterogeneity
- ⋆ Syntactic heterogeneity
- ⋆ Data model heterogeneity
- ⋆ Structural heterogeneity
- ⋆ Schematic heterogeneity
- ⋆ Semantic heterogeneity

Technical heterogeneity is the implementation of different access methods to the data of data source. This kind of problem is solved, if the integrated system is able to query the data source and the request returns a correct result set. Different representations of the same issue is called *syntactic heterogeneity*. An example is different character encodings in a data set. This problem can easily be solved by converting the data into a unique format. Data sets of a data source could be managed by different data models. One data model is, for instance, object-oriented and the other is relational. In this case there exists a so-called *data model heterogeneity*. On the other hand, if both data models are equivalent, then a data model heterogeneity is nonexistent. Data model heterogeneity can be resolved by converting a semantically weaker model into a semantically stronger model. Often design autonomy causes structural, schematic and semantic heterogeneity in data integration. Structural differences in the representation of data are called *structural heterogeneity*. If each semantic concept is structurally equally modeled, then the structural heterogeneity is solved. A special case of structural heterogeneity is *schematic heterogeneity*, where different concepts of a data model describe the same issue or data. Semantic heterogeneity characterizes the differences in sense, interpretation, types of terms and concepts. In particular synonyms and homonyms play a major role in those conflicts. Consequently, these problems can be solved if schema elements have the same meaning and an identical name.

It is possible to force specific properties to be homogenous by restricting autonomy of a data source. This can be achieved by standards in exchange formats, interfaces and protocols.

2.2.2 Approaches of Database Integration

The development of an integrated database system is a complex task. Particularly, if a large number of heterogeneous databases have to be integrated. Hence, an elaborate blueprint of the architecture of the system is essential. However, another non-trivial problem is the availability of databases that should be integrated. The number of molecular databases is still increasing (see section 2.2), but on the other hand, many databases are not publicly available. It is understandable that in many cases owners of databases, which have been developed within publicly funded research project, do not make their data completely available. The assembling of a molecular database is in many cases the result of a large number of experiments or of manual extraction of literature research. In fact, most databases can be searched or queried via a web page, but this does not mean that the data is completely available. In comparison to the total number of databases, only small amounts of data can be accessed via web. Copyright protection of databases outside of Europe is non-existent [MHO01]. Providers use limited accessibility to protect their information, for example, they slow down the connection speed so that a user can interact with the web page.

Generally, there exists two architectures for integration. They are divided into *materialized integration* and *virtual integration*. Materialized integration means the whole data set of source is stored persistent via a global component, typically in a database management system (DBMS). Updates and extensions are transferred via periodic update strategies to the global system. Then the integration system normalizes the data. Furthermore, duplicates and failures will be removed. Finally, the central database will be updated to provide an up-to-date data set. Advantages of materialized integration are the high velocity, because there is no communication between different data sources, and no restriction of queries, which could be the case in virtual integration systems.

In comparison to materialized integration, virtual integration does not store data in a global way. Rather the data is located on different local systems and queried by a global schema if required. A complex normalization and transformation, in comparison to materialized integration, does not happen. Queries are managed by a global schema, while the underlying data is "virtual" available. Thereby, complex queries have to be generated to get, transform and aggregate adequate data from different data sources. Moreover, some data sources provide only restricted interfaces, whereby queries of the global schema can not be answered or executed.

Different approaches of database integration have been frequently discussed and reviewed in the last years. Three approaches will be discussed in this thesis in more detail.

- ⋆ *Hypertext navigation systems.* HTML frontends linked to molecular biological databases.
- ⋆ *Federated database systems and mediator-based systems.* Virtual integration does not store any data in a global schema. Federated systems integrate multiple autonomous database systems into a virtual single federated database. Usually, each database is interconnected via a computer network. The databases may be geographically decentralized.
- ⋆ *Multi-database systems.* In comparison to federated database systems multi-database systems do not have a global schema, rather these systems interactively generate queries for several databases at the same time.
- ⋆ *Data warehouses.* Materialized integration stores data persistent in a global data repository, which typically has a DMBS as backend.

The aims of these approaches are all the same, providing techniques to handle several kinds of heterogeneous data as discussed in section 2.2.1.3 and to build an unique retrieval system for researchers to support their activities.

2.2.2.1 Hypertext Navigation Systems

Today, most databases are connected to world wide web and can be accessed with a common browser. Typically, many of these databases provide links to other databases. Usually, accession numbers (AC), unique identifiers or other database identifier are used for linking database entries. In fact, many databases use different identifiers or terms for the same entries, so that interlinking databases is a major task. Furthermore, pair-wise or binary mappings between database entries have to be generated to provide links between different databases. Thus, databases only provide cross links to most relevant databases.

In addition, many other database attributes can be used for linking databases to each other. Examples are EC numbers (see section 2.1.2), CAS (Chemical Abstracts Service) registry numbers, GO (Gene Ontology) terms or other controlled vocabularies. Often databases are not linked to each other, even if they use the same controlled vocabulary. However, it is not possible to link all the databases to each other. Providers are in fact not aware of all other relevant databases. Nevertheless, interlinked web sites are common way of database "integration".

2.2.2.2 Multi-Database Systems

Federated and central database systems contain respectively one database management system that manages the whole dataset, while *multi-database systems* are usually a network of database systems [Con97]. Therefore, the management of the whole dataset is not controlled by the overall system, rather the data is controlled by independent partitions. Thereby, the user has access via a common query language to the different data sources. Conflicts of integration, for instance, multiple redundant data, structural differences between data sources or semantic heterogeneities are covered by the provided query language.

On one hand, if data sources maintain a certain level of autonomy, they are called federated database systems. On the other hand, while a central system takes control of data sets the system is not federated anymore. It is diverse by which level of autonomy the border between federated database systems and multi-database systems is arranged. Figure 2.10 shows a proposed architecture according to [LMR90] that includes following schemata.

- ⋆ *Physical schema* characterizes the physical or internal structure of the different partitions of the databases.
- ⋆ *Internal logical schema* illustrates conception schema of each component. The schema is independent from implementation.
- ⋆ *Conceptual schema* provides users views or special views of the internal logical schema.
- ⋆ *External schema* defines a virtual database as a named set of relations. Usually these schemas define distributed databases.
- ⋆ *Dependency schema* describes dependencies between the data of the different databases.

The schemata are divided into the internal layer, the conceptual layer and the external layer. The user defines independently a view for the required data. A query spanning multiple databases is specified by the multi-database query language. In a central unit the query is fractionalized and sent to the different databases. Finally, the result sets are sent to the processing unit and returned as a merged result to the user.

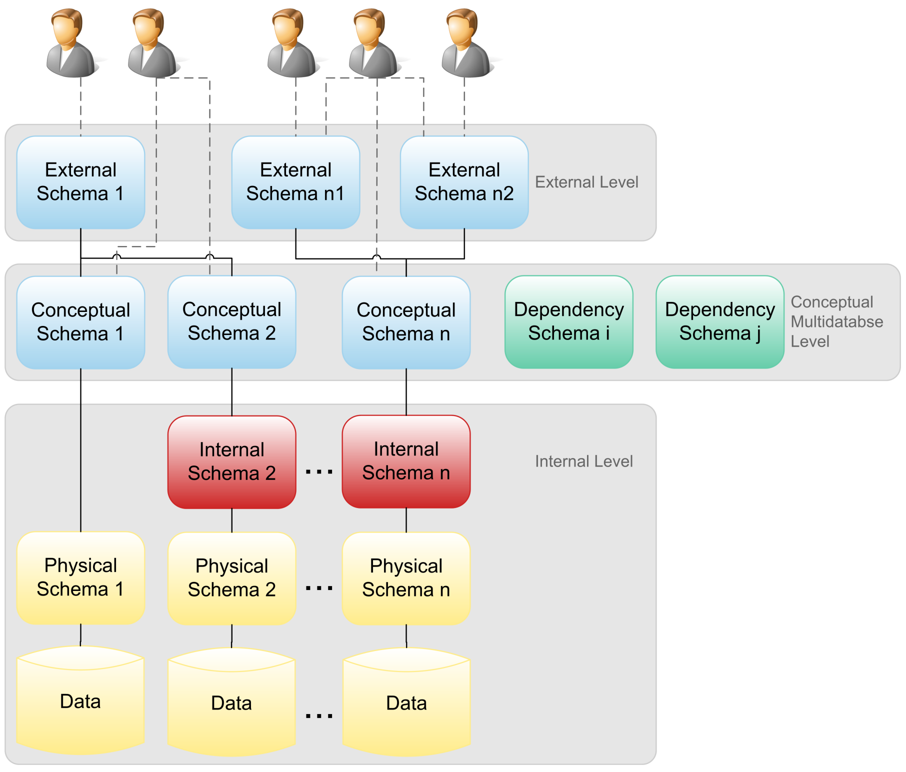


Figure 2.10: Reference architecture of a multi-database system according to [LMR90].

2.2.2.3 Federated Database Systems

The term *federated database system* (FDBS) was first introduced by Heimbigner and McLeod in 1985 [HM85] and later defined in more detail by Sheth and Larson [SL90]. A federated database system consists of multiple cooperating component systems that are autonomous and a federated database management system that controls the component systems. Federated architectures differ based on levels of integration with the component database systems and the extent of services offered by the federation. A general architecture of a federated database system is illustrated in figure 2.11.

Depending on the level of coupling federated database systems can be categorized as loosely or tightly coupled systems. In a loosely coupled FDBS each user is the administrator of his own federated schema. Each user is responsible for understanding the semantics of objects in the export schemata and as well for elimination of heterogeneities from the DBMS. Furthermore, users can store their schema under their own accounts. A schema can be deleted at any time by the user [SL90].

In tightly coupled FDBS export schemata are created by negotiation between component database administrator and federation database administrator. Usually, the component database administrator has control over the export schemata, while the federation database administrator has the authority to read the database to help determine what data is available and where it is located. The federation database creates and controls the federated schemata [SL90].

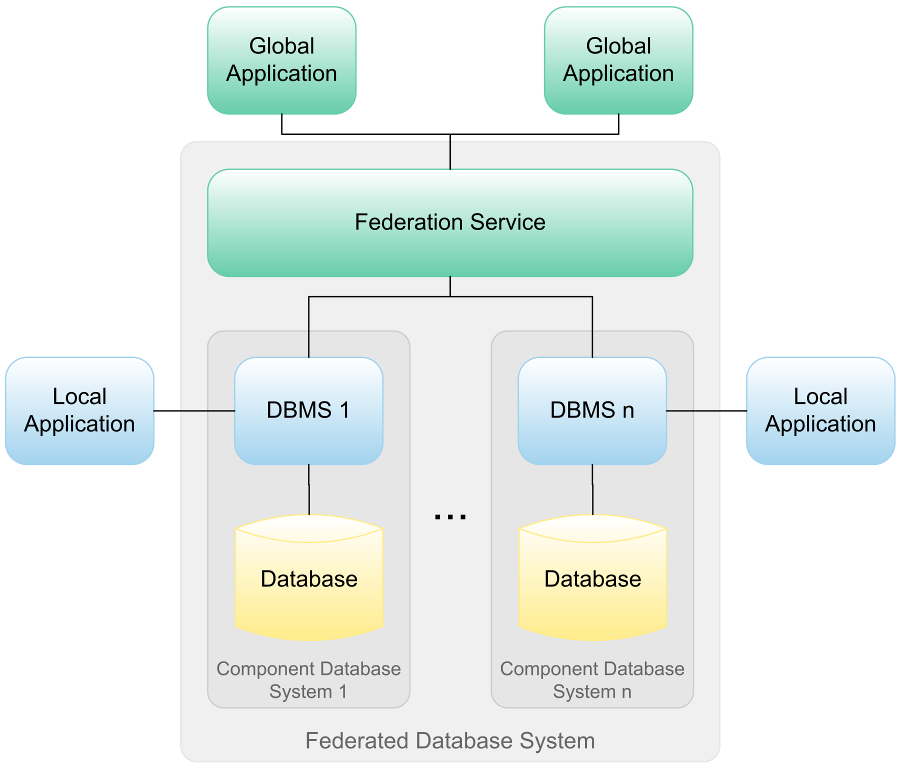


Figure 2.11: General architecture of federated database systems according to [Con97].

2.2.2.4 Data Warehouses

This section will give an introduction into *data warehouse systems* that are essential for this work. *Data warehouses* (DWH) are one of the widely used architectures of materialized integration. Usually, data warehouses are used in the field of information management. In particular data analysis, data mining and long-term storage of business intelligence in companies are the major advantages of data warehouse systems. In bioinformatics DWHs are usually used for data integration. Virtual integration has some disadvantages, thus DWHs are often preferred. The disadvantages are listed in the following.

- ⋆ No write access.
- ⋆ Poor speed of request handling.

- ⋆ Problems in availability of data sources.
- ⋆ Complexity of queries.

This work will present a materialized data integration approach realized as data warehouse. In this field a lot of work has already been done. Some projects such as ONDEX, BioWarehouse and Columba will be discussed in chapter 3 in more detail.

Currently, there is no consistent definition of the DWH term. Since a while different consortia such as the OLAP Council are trying to standardize the DWH term, a first definition was given by Inmon in 1996 [Inm96].

> A data warehouse is a subject oriented, integrated, non-volatile, and time variant collection of data support of management's decision.

Therefore, a data warehouse has following four properties according to Immon.

1. *Subject orientation.* The purpose of the database is not only to administrate the data, such as personal data, rather a data warehouse is designed to help users analyze data. The ability to define a data warehouse by subject matter makes the data warehouse subject oriented.
2. *Integration* is closely related to subject orientation. A data warehouse needs to have the data from disparate sources put into a consistent format. This means they have to solve heterogeneity and other integration problems as discussed in section 2.2.1.
3. *Non-volatile* means that the data should not change once entered into the warehouse. This is logical because the purpose of a warehouse is to analyze what has occurred.
4. *Time variance.* A data warehouses is time variant in the sense that it maintains both historical and current data. Usually, a data warehouse stores many months or years of data. This is necessary to support historical analysis and reporting.

On one hand this definition is not meaningful enough to use the definition in practice or theory. On the other hand this definition is so restrictive that many application areas and approaches are not covered. Thus, a new definition is necessary that was defined by Bauer and Günzel. "A data warehouse is a physical database that provides an integrated view of arbitrary data for analysis." [BG04]. A DWH can not be assigned to classical OLTP (Online Transaction Processing) systems, which are optimized for fast and reliable transaction handling. Compared to data warehouse systems, most OLTP interactions will involve a relatively small number of rows, but a larger group of tables. DWHs are assigned to OLAP (Online Analytical Processing) systems, which are able to quickly answer multi-dimensional analytical queries. OLAP is part of the category business intelligence, which also includes relational reporting and data mining. Typically, in DWH, new data will be added, already stored data should not be manipulated or overwritten.

Figure 2.12 illustrates a data warehouse reference architecture according to [BG04]. The central component of a DWH is the *Data Warehouse Manager*, that initialize, controls and monitors each process of the system from data extraction to data analysis. As described in figure 2.12 the Data Warehouse Manager controls all components of the DWH. The components are responsible for internal and external data sources as well as for querying and representation of the data.

- ⋆ *Monitors*: detect and report changes within different data sources, relevant for the DWH, to the data warehouse manager.
- ⋆ *Extractors*: select and transport data from the sources into the Staging Area.
- ⋆ *Transformators*: standardize, integrate, consolidate, aggregate and complete extracted data of the staging area.
- ⋆ *Loading components*: after finishing ETL process, loading transformed data from Staging Area into the *Operational Data Store* (ODS) and then into the DWH.
- ⋆ *Analysis components*: analysis and presentation of data of the DWH.

A data warehouse process is divided into four phases. The first phase obtains the data from different resources. That means the data will be extracted and transformed. This phase is called *ETL process (Extraction-Transform-Load-Process)*. Afterwards, the data will be saved persistently in the DWH. In the third phase the separated data will be divided into several data marts, if necessary. The last phase analyzes the data of the DWH or data marts and provides data to external application.

2.3 Summary

In this chapter, the basics of two different disciplines, biology and computer science, were presented. This was necessary to understand basic connections in biology that is necessary for molecular biology data integration. Also basic methods in computer science for data storage were discussed.

First basic nodes on DNA as carrier of genetic information were introduced. In addition, notes to protein synthesis and the role of special proteins, as well enzymes, in metabolism and on general biochemical reactions were presented, followed by biological networks that are a combination of metabolic pathways, regulatory networks, signal pathways and cell communication processes were illustrated. In the second part of computer science, problems in data integration were delineated. Finally, we outlined the data warehouse approach that is essential for this work.

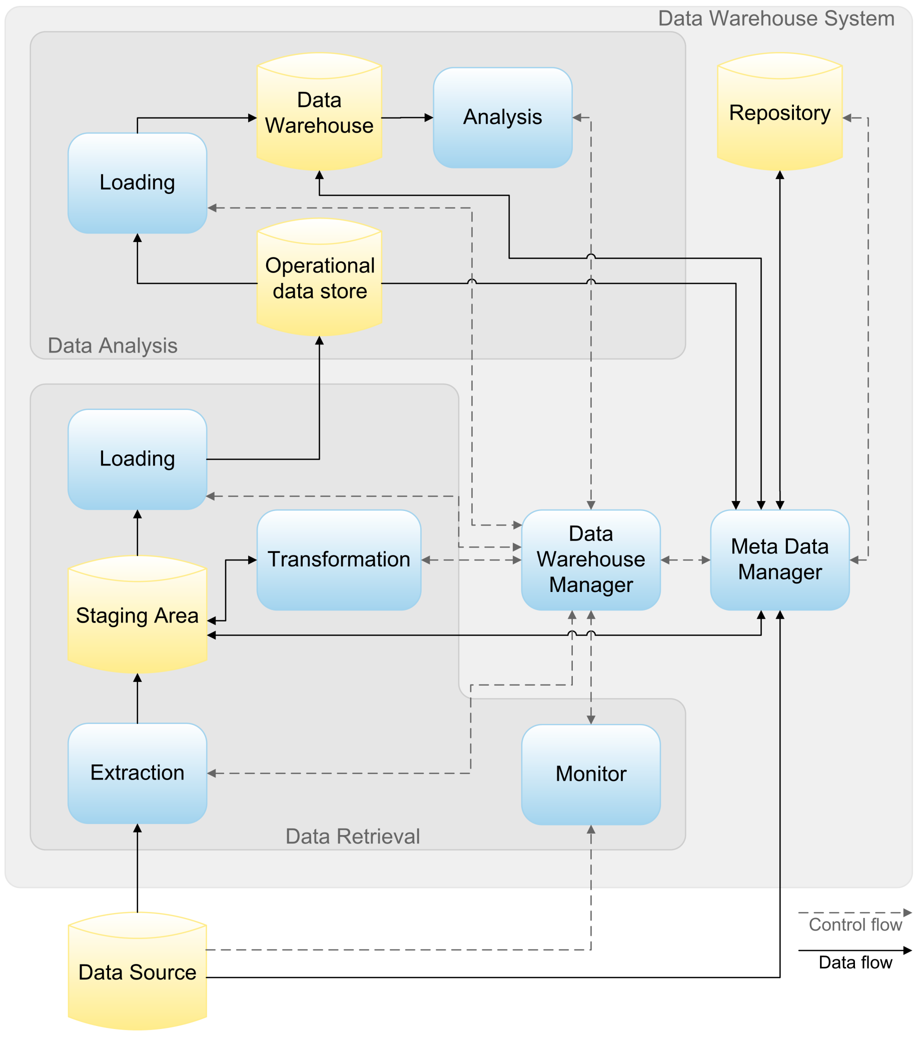


Figure 2.12: Reference architecture of a data warehouse system according to [BG04].

3 Related Work

Following to the chapters Introduction and Basics of biology and computer science in this chapter relevant data integration approaches and data sources will be discussed. Usually, the data is distributed in multiple data sources. Those sources differ in the biological context, internal representation, used underlying systems, access possibility and complexity.

In the first section of this chapter relevant integration approaches in field of data warehouses will be introduced. The focus lies on the data warehouse approaches Atlas, Columba, BioWarehouse, ONDEX and CoryneRegNet, because the systems are equipped with important requirements in biological data integration that are relevant for this work. Moreover they are well known examples in the literature of data integration approaches in bioinformatics. Additionally, in table 3.2 all data integration approaches will be compared. Generally, in bioinformatics integration approaches are divided into following classes [LN07].

- *Indexing systems:* e.g. SRS (Sequence Retrieval System) [EUA96], Entrez and BioRS
- *Multi-databases:* e.g. OPM (Object Protocol Model) [CM95] and DiscoveryLink [HSK$^+$01]
- *Ontology-based integration:* e.g. TAMBIS (Transparent Access to Multiple Bioinformatics Information Sources) [SBB$^+$00] and ONDEX [KBT$^+$06]
- *Data warehouse:* e.g. Atlas [SHX$^+$05], BioWarehouse [LPW$^+$06], Columba [TRM$^+$05], CoryneRegNet [Bau07], ONDEX [KBT$^+$06] and SYSTOMONAS [CML$^+$07]

To integrate medical and molecular biological data, the first step is to structure and evaluate the amount of available data sources. A comparison and choice of data sources is only possible on the basis of an adequate set of criteria. Therefore, in this chapter we only focus on the most important data sources that are relevant for this work. Based on relevant biological database and information systems data integration is an essential step in constructing biological networks. Therefore, in section 3.3 most relevant software solutions for visualization and simulation of biochemical networks, such as Cytoscape, VisANT and VANTED, will be introduced. Finally, Petri nets for modeling and simulation of biological networks will be motivated.

3.1 Related Data Integration Approaches

In this section the data warehouse infrastructure, Atlas and BioWarehouse will be introduced. Both systems provide a software infrastructure that can be installed and configured locally. They give insight into biological data integration that are relevant for the data warehouse infrastructure in chapter 4. Afterwards the Columba data warehouse with its web-application (section 3.1.3) and the ontology-based data warehouse approach CoryneRegNet (section 3.1.5) will be introduced. Both systems give an understanding in building web-based data warehouses that is important for chapter 5.

3.1.1 Atlas

The Atlas system was developed at UBC Bioinformatics Centre (University of Bristish Columbia) in Canada and is available at present in Version 0.2.1. Atlas is freely available and is protected under terms of GNU General Public License. An Unix operation system is required, because the software is not platform independent.

The goal of Atlas is to provide data as well as a software infrastructure for bioinformatics research and development. The biological data warehouse locally stores and integrates the following kind of data.

- ⋆ biological sequences
- ⋆ molecular interactions
- ⋆ homology information
- ⋆ functional annotations of genes
- ⋆ biological ontologies

The system is made up of the *data sources*, an *ontology system*, the *relational data models*, different *APIs* (Application Programming Interfaces) and *applications*. Figure 3.1 illustrates the system architecture of Atlas. The data sources of Atlas are categorized into four classes: *sequence*, *molecular interactions*, *gene related resources* and *ontology*. According to Figure 3.1 some data sources are available, such as OMIM, UniProt or GO, those are also relevant for this work. A full list of the Atlas data sources could be found in [SHX$^+$05].

Each mentioned category has its own database schema in the Atlas relational database model. In [SHX$^+$05] is a complete entity relationship schema of the data warehouse presented. Depending on the level of coupling, this approach can be categorized as a tightly coupled system. As relational backend Atlas uses the open source software DBMS MySQL.

The different APIs of Atlas are developed in three programming languages C++, Java and Perl. But not every API is available in the respective programming language. Furthermore, the APIs are divided into two classes *loader* and *retrieval*. The loader APIs consist of parser to populate the relational schemata and store the data in the Atlas databases. The other class of APIs are the retrieval APIs. These APIs allow to retrieve the data stored in the data warehouse. Additionally, they are required for developing custom retrieval applications. The Atlas system provides a lot of Unix command line tools, such as *ac2seq* that is able to find a sequence in FASTA format[1] on the basis of accession numbers. Moreover, the user is able to send direct SQL queries via MySQL client to the data warehouse.

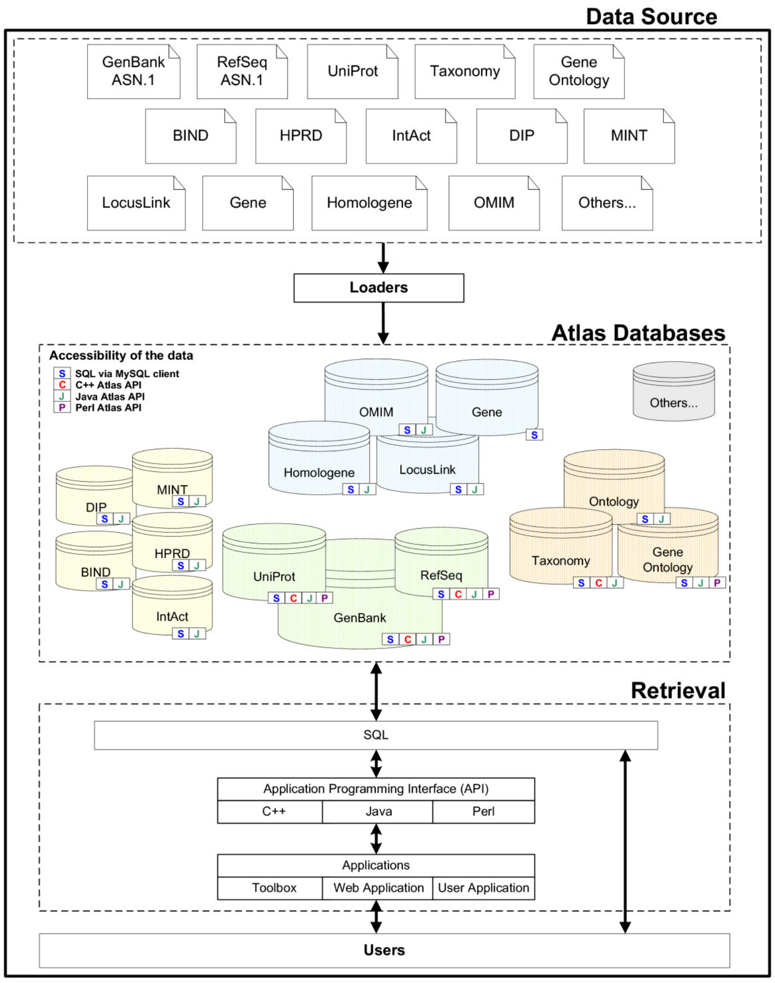


Figure 3.1: System architecture of the Atlas data warehouse [SHX+05].

The atlas system is designed to run as a service on a local computer system or server. According to [SHX+05] it is also possible to access Atlas via the web interface, whereas this web application is currently not available. The following advantages and disadvantages of the Atlas system have been identified.

[1]http://www.ncbi.nlm.nih.gov/blast/fasta.shtml

- ⋆ **Advantages**
  - a lot of tools
  - integrative database schema
  - short response time, because of local installation
  - complete access to the database
- ⋆ **Disadvantages**
  - extensive maintenance of the system
  - high system requirement
  - not platform independent
  - actuality of the data depends on user or administrator
  - tools only available for command line
  - missing web interface
  - only MySQL is supported as database management system

3.1.2 BioWarehouse

The BioWarehouse system was developed by the Bioinformatics Research Group (SRI International), Computer Science Laboratory (SRI International) and Stanford Medical Informatics (Stanford University). It is also part of the Bio-SPICE (Biological Simulation Program for Intra- and Inter-Cellular Evaluation) project, an open source framework and software toolset for Systems Biology. BioWarehouse is an open source toolkit that integrates a multiplicity of biological databases. The software is protected under terms of the MPL (Mozilla Public License) and is currently available in version 4.5. A Linux system is required, because the software is not platform independent. BioWarehouse facilitates to create user-defined and user-specific data warehouse instances. Several data sources for the toolkit are available such as ENZYME, KEGG, GO and UniProt that are also part of this work. More available data sources for this toolkit could be found in [LPW$^+$06]. Similar to Atlas system different relational database schema exist according to the different data types, therefore this approach can be characterized as loosely coupled system. BioWarehouse supports MySQL and Oracle database management systems in comparison to Atlas that only supports MySQL. Figure 3.2 illustrates the BioWarehouse database schema whereas the entities symbolize the particular data type. Integration of the different data sources is realized by a specific loader. Each loader is adapted for a particular data source. Due to the fact of heterogeneities between different data sources, the data is transformed into a consistent format and afterwards transferred into the database schema. Loaders have been implemented in programming languages C and Java. A special feature of the loaders is the error tolerance during integration, in case of an error the integration process will be finished and the incorrect data sets will be marked. Furthermore, the BioWarehouse implementation provides a set of Java utility classes that are useful for developers who want to construct their own loaders or applications.

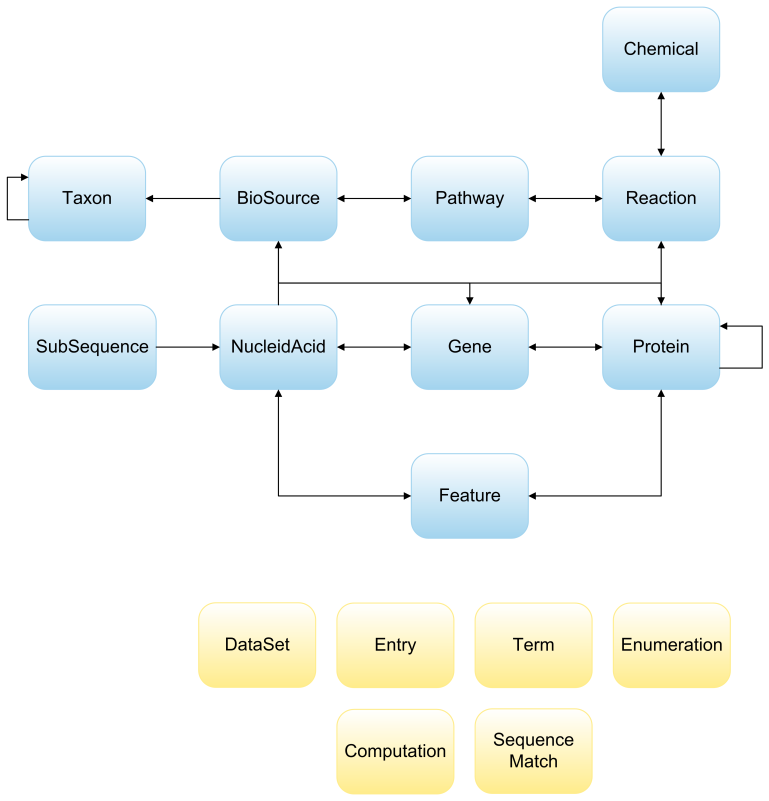


Figure 3.2: The main datatypes in the BioWarehouse schema according to [LPW+06], and the relationships between them. An arc represents a connection between two datatypes. For instance, the datatype *Gene* contains a column that references datatype *Protein*.

The BioWarehouse system can be used in two different ways. A public available version of BioWarehouse called PublicHouse that is available via the internet. This application is based on BioWarehouse version 3.7 and only a couple of public available data sources are accessible. In addition, a user has to register to access PublicHouse. The access to the system occurs via a MySQL client. Complex queries are not recommendable, because the performance of the server is limited. The second alternative to use the system is to install the software locally on the computer and select the required data sources. In the locally installed version the user is able to manage the data sources and have not only read access. Following advantages and disadvantages of the system could be figured out.

- ⋆ **Advantages**
  - integrative database schema
  - short response time because of local installation

  - full access to the database
  - MySQL and Oracle are supported

⋆ **Disadvantages**

  - extensive maintenance of the system
  - high system requirements
  - it is the administrators responsibility to keep the data up-to-date
  - missing web interface
  - SQL knowledge is required

3.1.3 Columba

The Columba data warehouse system was developed by the following institutes.

⋆ Department of Computer Science, Humboldt-Universität zu Berlin

⋆ Department of Biochemistry, Charité Universitätmedizin Berlin

⋆ Zuse Institute Berlin

⋆ University of Applied Sciences Berlin

Columba integrates data from 12 heterogeneous biological data sources in the field of protein structures and protein annotations. An important aspect in Columba is the protein structure from the database PDB which plays a major role in the system. This data is extracted and enriched with additional information such as protein sequence, protein function, involvement in biological networks and membership of protein families. Usually, several data sources are provided in different exchange formats. For this reason parsers were developed for those formats when not available from other projects. Table 3.1 presents data sources, exchange formats as well as parsers of the system. Beside parser of the BioSQL and BioPython project new parsers have been developed. Therefore the programming languages Python and Perl were used. Columba uses the PostgreSQL database management system to manage the data. The idea behind the schema of the Columba database is comparable with a star schema. On the basis of this schema it is clear that the core of Columba is the protein structures of PDB. Additional information related to the protein structures are included into specific sub-schemata that comes from different sources that are arranged around the main table. Figure 3.3 illustrates the database schema of Columba. Each data source is modeled as a different dimension and has its own sub-schema within the overall schema of Columba. This approach has the following advantages.

⋆ simple maintenance of the system

- ⋆ intuitive query model
- ⋆ high recognition value of the information

Data source	**Exchange format**	**Parser**
PDB	Flatfile	BioPython
SCOP	Flatfile	BioPython
CATH	Flatfile	Columba
DSSP	unknown	Columba
ENZYME	Flatfile	BioPython
Boehringer	HTML	Columba
KEGG	HTML	Columba
Swiss-Prot	Flatfile	BioSQL
GO	Flatfile	BioSQL
GOA	Database dump	no parser required
Taxonomy	Flatfile	BioSQL
PISCES	Database dump	Columba

Table 3.1: Data sources, exchange formats and parser of the Columba data warehouse system [TRM+05].

Columba is accessible through a web interface that allows full text search as well as attribute specific searches. Full text search is implemented with the extension Tsearch2 in PostgreSQL. The search engine supports Boolean operators to link keywords with each other.

Details of the result are provided in HTML or XML format. The PDB entries represent the basic structure that is extended by the different data sources. All detailed results are uniquely identified by a PDB-ID.

The Columba data warehouse is freely available and not usable for commercial purpose. Data sources will be updated immediately if a new version of the source is available according to [TRM+05]. Columba integrates different data sources about proteins that simplifies research in this area. On the other hand Columba does not provide comprehensive knowledge about molecular biology which is a crucial disadvantage. Redundancies could not be excluded, because every data source is an independent dimension in the model. Depending on the level of coupling this approach can be categorized as a loosely coupled system.

3.1.4 ONDEX

ONDEX was developed at the Division of Biomathematics and Bioinformatics Rothamsted Research and Faculty of Technology at Bielefeld University. The ONDEX framework

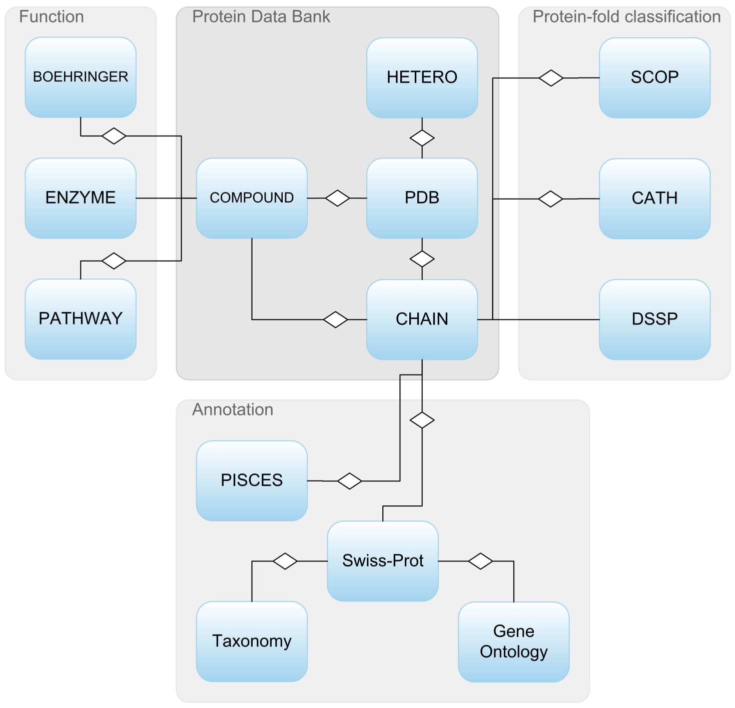


Figure 3.3: Entity Relationship diagram of Columba according to [TRM+05].

combines: semantic database integration, sequence analysis, text mining and graph analysis [KBT+06]. Thus, the main idea behind semantic database is to represent biological databases using graphs in which the nodes and edges have different attributes. Nodes represent any concept that can occur in a database, whether it is a gene, an enzyme, a transcription factor, a pathway or any other biological element. The edges of the graph represent the relation between concepts. Each concept is assigned to a concept class as well as each relation belongs to one relation type. Concepts and relations can both have properties and optional characteristics. Figure 3.4 illustrated the entity relationship (ER) diagram representing the data structure used for ONDEX.

Furthermore, ONDEX has a text mining component that extracts digitally available biological knowledge that is not stored in databases. The graph analysis and visualization in ONDEX works on an internal graph object that is in general independent from any graph library. Therefore, the internal graph object provides several interfaces and adaptors in order to provide the user with a wide range of possibilities to layout, process or filter the graph. The graphical user interface of the ONDEX system is implemented in Java.

In summary, the modular and generic architecture of the ONDEX framework is flexible enough to combine a number of graph libraries and filter. Several data sources, such as AraCyc, KEGG, TRANSAFC, TRANSPATH, DRASTIC, etc are integrated and linked by ONDEX. Additionally, graph analysis functions and visualization methods can operate on a large scale graph and networks. Moreover, ONDEX provide microarray analysis and

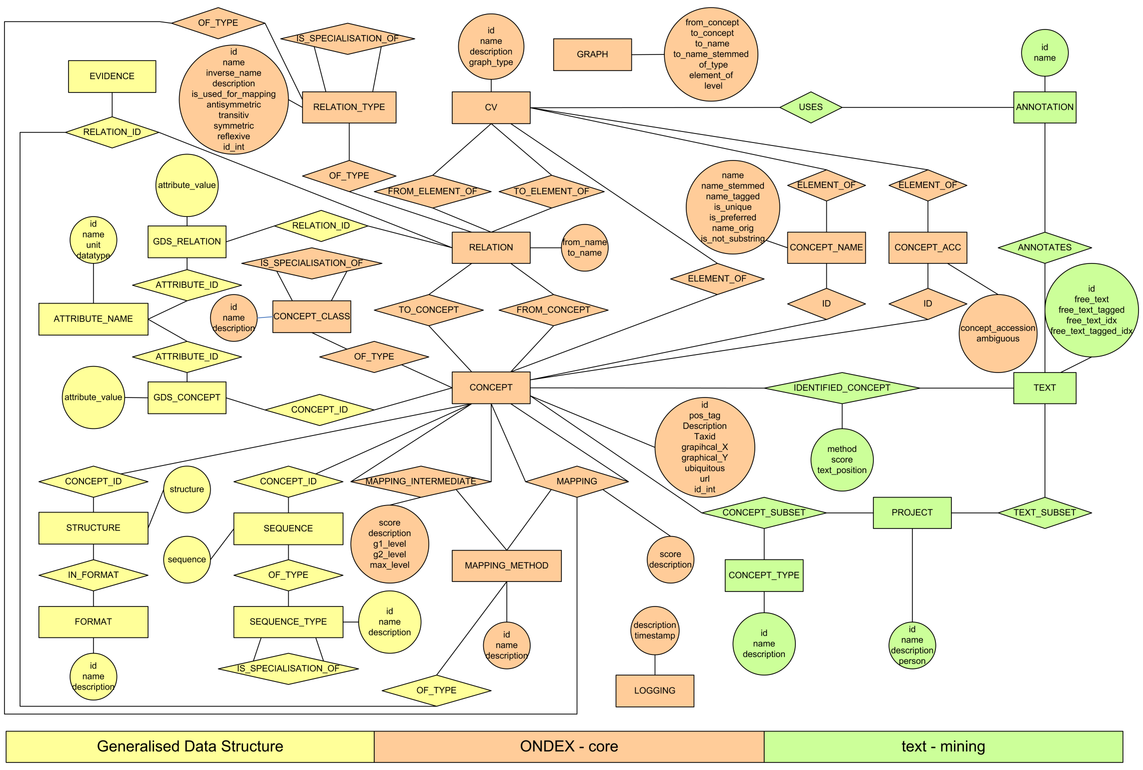


Figure 3.4: Ontology based Entity Relationship diagram of ONDEX.

statistical support. Currently, it is supported by several institution of the Ondex SABR project. On the other hand ONDEX is difficult to install on a local system and it is resource consuming. Numerous functions and analysis tools make the user interface quite complex.

3.1.5 CoryneRegNet

The CoryneRegNet system was developed at the Center for Biotechnology (CeBiTec), Bielefeld University. CoryneRegNet is an acronym for **Coryne**bacteric Transcription Factors and **Reg**ulatory **Net**works. This software is an ontology-based data warehouse approach that provides data about transcription factors and gene regulatory networks. At present the system provides data about 6 corynebacteria, 2 mycobacteria and model oragnism *Escherichia coli* whereas the tool focus on corynebacterium. The software is protected under terms of AFL (Academic Free License) and is available in version 4 [Bau07]. Figure 3.5 illustrates the system architecture of CoryneRegNet. Parser transform different data sources into consistent object-orientated ontology-based data structure. Those data structures are closely related to the ONDEX system, described in section 3.1.4. Afterwards, the data structures are transformed into a relational database model. Figure 3.6 shows the entity relationship diagram of CoryneRegNet that consist of *Generalised Data Structure* and *Ontological Data Structure*. Furthermore, the ontological data structure stores all essential data such as genes and proteins. Consequently, the generalized data structure manages common system-relevant data.

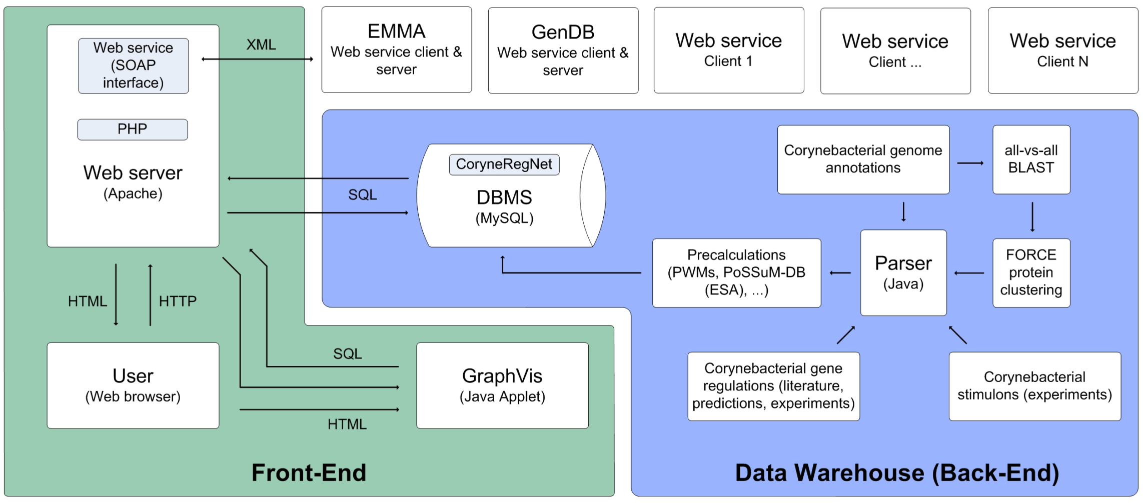

Web service (SOAP interface)
PHP
Web server (Apache)
XML
EMMA Web service client & server
GenDB Web service client & server
Web service Client 1
Web service Client ...
Web service Client N
SQL
CoryneRegNet
DBMS (MySQL)
Corynebacterial genome annotations
all-vs-all BLAST
Precalculations (PWMs, PoSSuM-DB (ESA), ...)
Parser (Java)
FORCE protein clustering
HTML
HTTP
User (Web browser)
SQL
GraphVis (Java Applet)
HTML
Corynebacterial gene regulations (literature, predictions, experiments)
Corynebacterial stimulons (experiments)
Front-End
Data Warehouse (Back-End)

Figure 3.5: System architecture of CoryneRegNet [Bau07].

In addition the software is accessible via a web-service that is illustrated in figure 3.5. Based on this web-service it is possible to access additional and up to date data dynamically. A web-service can also have disadvantages in terms of performance and security, performance delays can occur because of high traffic. Figure 3.5 shows the communication between the applications EMMA and GenDB that also communicate with the web-service, so that the system has to take care of the security.

CoryneRegNet is realized as web application and thereby platform independent. The software is developed using PHP and Java programming language. A MySQL database management system is used as relational backend. The web interface provides several search and analysis possibilities. In addition, transcription factors and regulatory networks will be visualized. The regulatory networks are displayed in a Java applet. Furthermore, it is possible to adjust the depth of the visualized graphs.

One of the main advantages of CoryneRegNet is the user-friendly and intuitive web application. Moreover, other application can access CoryneRegNet via web service and provide additional information. However, a Java installation is required to visualize networks in a Java applet. Furthermore, longer loading times by the Java applets as well by the web service can occur. Similar to the Columba, CoryneRegNet has not general molecular knowledge, because it is limited to bacteria.

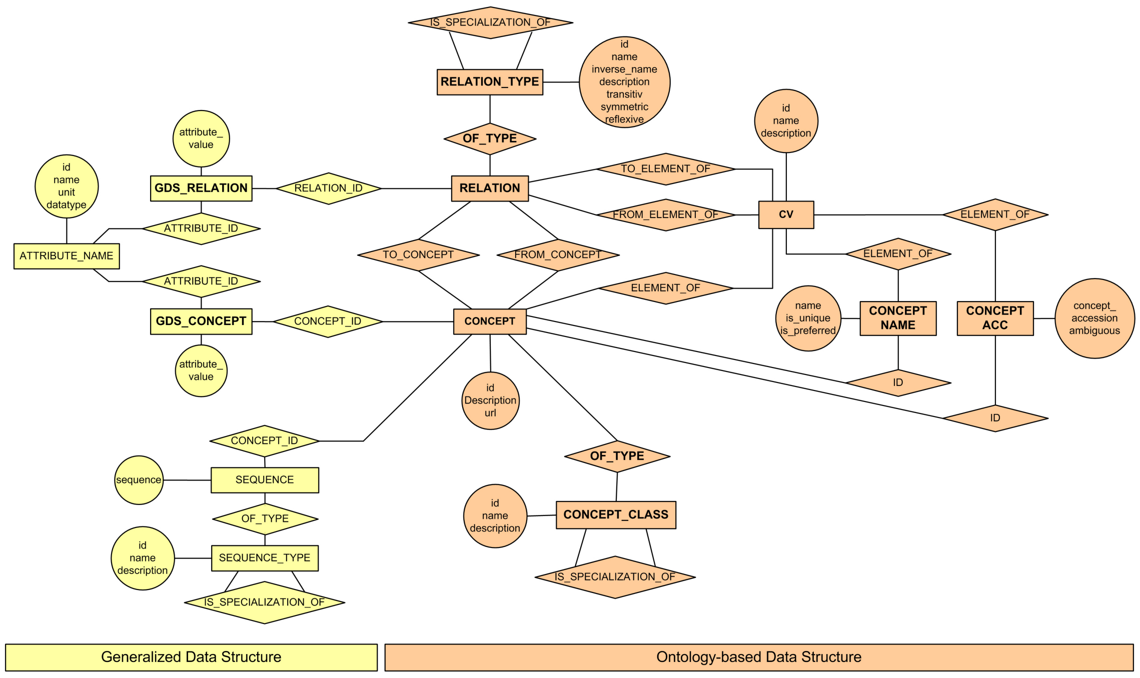


Figure 3.6: Ontology-based entity relationship diagram of CoryneRegNet [BBC+06].

3.1.6 Summary

In this section the data integration approaches Atlas, BioWarehouse, Columba, ONDEX and CoryneRegNet were introduced. All projects use the data warehouse technique for data integration, whereas CoryneRegNet and ONDEX provide a web service. Atlas, BioWarehouse and ONDEX provide a software infrastructure for data integration, rather Columba, CoryneRegNet. They provide a web interface and therefore they are directly useable. In table 3.2 a comparison of the data warehouse systems on the bases of different criteria are given. Although advantages and disadvantages of the particular software solution are marked.

An important criterion is platform independence and user-friendly system. Typically, web applications are preferred, because they are very flexible and interactive. The term *Web 2.0* characterizes flexible and interactive web applications. A selection of the DBMS is not possible with Atlas, Columba, ONDEX and CoryneRegNet, because they provide a specific DBMS. In comparison, BioWarehouse provides a choice between MySQL and Oracle database management system. Columba, CoryneRegNet only provide a restricted access to the database, but provide suitable query forms. On the other hand, Atlas and BioWarehouse support a complete database access. Finally, an overview of the comparison of the different approaches is given in table 3.2.

Property	**Atlas**	**BioWarehouse**	**Columba**	**ONDEX**	**CoryneRegNet**
Integration	tightly coupled	tightly coupled	loose coupled	loose coupled	tightly coupled
DBMS	MySQL	MySQL, Oracle	PostgreSQL	MySQL	PostgreSQL
Programming language	**Java, C++, Perl**	Java, C	Python, Perl	Java	PHP, Java
Architecture	Application	Application	**Web interface**	Application	**Web interface**
Platform independence	No (Unix systems)	No (Linux systems)	**Yes**	**Yes**	**Yes**
Updates	Manually	Manually	Old	**High**	Unknown
Maintance & developement	Unknown	Periodical	Project ended	Periodical	Periodical
License	**GNU**	MPL	Freely available on request	**GNU**	AFL
Open source	**Yes**	**Yes**	Unknown	**Yes**	**Yes**

Table 3.2: Comparison of different data warehouse approaches. Advantages are marked in bold letters.

3.2 Related Data Sources

In this section the most relevant data sources will be introduced. Databases in the field of metabolic pathways, signaling pathways, nucleotide sequences, proteins, enzymes and diseases will be presented.

3.2.1 KEGG

The Kyoto Encyclopedia of Genes and Genomes [KAG+08] also called KEGG is a freely available database that is part of the NAR category *Genomics Databases (non-human)* (see section 2.2). Originally, KEGG was launched in 1995 within the *Human Genome Project*, but it now covers a wider range of organisms. Responsible for development and actuality are the Kaneshia Laboratories in the Bioinfomatics Center of Kyoto University and Human Genome Center of the University of Tokyo. Data of following categories are organized in an encyclopedic system.

- ⋆ Organism specific metabolic knowledge
- ⋆ Genomic, chemical and systematic functional information

One of the major advantages of KEGG is the integration and linkage of genomic and chemical information. In addition to this, KEGG has a powerful connection between individual categories (systems information, genomic information, chemical information) as well as additional links to other databases such as UniProt (Universal Protein), OMIM (Online Mendelian Inheritance in Man) and GO (Gene Ontology). Table 3.3 illustrates the design and the integrated databases of KEGG. The complex structure of KEGG requires an unique identification of the different databases. Usually, the KEGG identifiers are composed of a prefix and a number. Table 3.4 shows the identity keys for the specific databases. Typically, the identity keys of the KEGG GENES database are *locus_tags* that are provided by the INSDC (International Nucleotid-Sequence Database Collaboration). UniProt belongs to the most substantial protein sequence databases and uses different kinds of identifiers. For this reason, KEGG implements an automatic conversion of these identifiers. This work also uses UniProt, so conversion will be explained on the example of UniProt. Now, it is possible to identify for the UniProt identifier *Q2JKT8* for KEGG GENES identifiers. KEGG GENES provides two entries *cyb:CYB_0854* and *cyb:CYB_1736*. The identifiers are composed of the organism *cyb* and the *locus_tag*. Figure 3.7 shows the KEGG PATHWAY entry of the Urea Cycle.

3.2.2 UniProt

The UniProt database [ABW+04, BBA+09] belongs to the category of *Protein Sequence Databases* and is one of the biggest database in this field. UniProt contains data about pro-

Category	**Database**	**Content**
System information	KEGG PATHWAY	Pathway maps
	KEGG BRITE	Functional hierarchies
	KEGG MODULE	Pathway modules
	KEGG DISEASE	Diseases
Genomic information	KEGG ORTHOLOGY	KEGG orthology (KO) groups
	KEGG GENOME	organisms
	KEGG GENES	genes in high quality genomes
	KEGG DGENES	Genes in draft genomes
	KEGG EGENES	Genes as EST contigs
	KEGG VGENOME	Viral genomes
	KEGG OGENES	Genes in organelle genomes
	KEGG SSDB	Sequence similarities and best hit relations
Chemical information	KEGG COMPOUND	Metabolites and other chemical compounds
	KEGG DRUGS	Drugs
	KEGG GLYCAN	Glycans
	KEGG ENZYME	Enzymes
	KEGG REACTION	Enzymatic reactions
	KEGG RPAIR	Reactant pairs and chemical transformations

Table 3.3: KEGG databases according to [KAG+08].

teins as well their function and structure. The Universal Protein Knowledgebase (UniProt) consortium has been united from EBI (European Bioinformatics Institute), SIB (Swiss Institute of Bioinformatics), PIR (Protein Information Resource) group at the University Center and National Biomedical Research Foundation. Since 2003 the UniProt database is online available and is updated in a fourteen day cycle. Following databases are integrated in UniProt.

- SWISS-Prot
- TrEMBL
- PIR-PSD

The UniProt database consists of three databases mentioned above and is organized in the following three layers.

- *UniProtKB:* The UniProt Knowledgebase provides data about accurate, consistent protein sequences and functional annotations. Additional, links to other databases are included.
- *UniRef:* The UniProt Reference Cluster provides non-redundant data collections based UniProtKB. It contains three kinds of sequences from UniProt: UniRef100, UniRef90 and UniRef50.

Prefix	Description	Database	Example
map	Reference metabolic pathway map	PATHWAY	map00010
org	Organism specific pathway or BRITE hierarchy	PATHWAY	hsa00010
br	Reference BRITE hierarchy	BRITE	br08301
ko	Reference metabolic pathway / BRITE hierarchy	PATHWAY/BRITE	ko00010
K	Otholog group bases on PATHWAY and BRITE	ORTHOLOGY	K00001
T	Organism	GENOME	T01001
C	Chemical compound	COMPOUND	C00001
D	Drug	DRUG	D00001
G	Glycan	GLYCAN	G00001
R	Reaction	REACTION	R00001
RP	Reaction pair	RPAIR	RP00001
EC	Enzyme nomenclature	ENZYME	EC1.1.1.1
H	Disease	DISEASE	H00001
M	Pathway modules	MODULE	M00001

Table 3.4: KEGG identity keys according to http://www.genome.jp/kegg/kegg3.html.

⋆ *UniParc:* The UniProt Archive provides stable, comprehensive and non-redundant sequence collection. The collection stores protein sequences from public available databases such as Ensembl, RefSeq (Reference Sequence) and FlyBase.

Figure 3.8 shows the structure and connection of UniProt databases. UniMES (UniProt Metagenomic and Environmental Sequences) is a database specialized on metagenomic and ecological data. Currently, UniMES contains data from the Global Ocean Sampling Expidition (GOS).

3.2.3 BRENDA

The BRENDA database [SCE+04, CSG+09] could be categorized in *Metabolic Enzymes and Pathways; Signaling Pathways* as described in section 2.2. In 1987 BRENDA (**Br**aunschweiger **En**zym**da**tenbank) was developed by Prof. Dietmar Schomburg at the GBF (Gesellschaft für Biotechnologische Forschung) Braunschweig (today Helmholtz-Zentrum für Infektionsfosrchung). Between 1996 and 2007 BRENDA was enhanced at the University of Cologne to one of the most used molecular information systems worldwide. Since mid 2007 the database is hosted at TU Braunschweig and will be developed further.

For academic users BRENDA is freely available via the Internet. A version is provided by Biobase can be purchased for commercial use. The BRENDA database contains information about following topics.

⋆ Classification and nomenclature

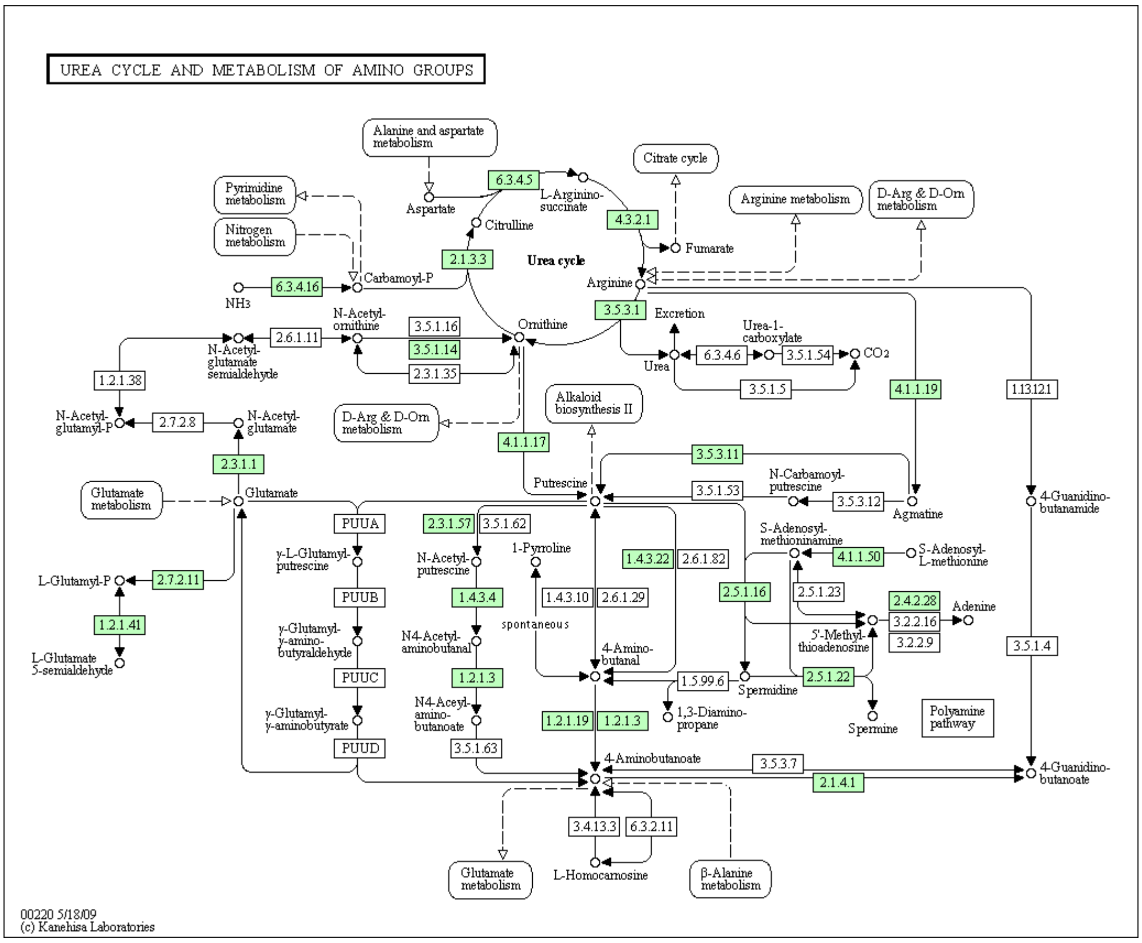


Figure 3.7: Graphical representation of the human urea cycle pathway. (Taken from http://www.genome.jp/kegg/pathway/hsa/hsa00220.html

⋆ Reaction and specificity

⋆ Functional parameters

⋆ Organism-related information

⋆ Enzyme structure

⋆ Isolation and preparation

⋆ Literature references

⋆ Application and engineering

⋆ Enzyme-disease relationships

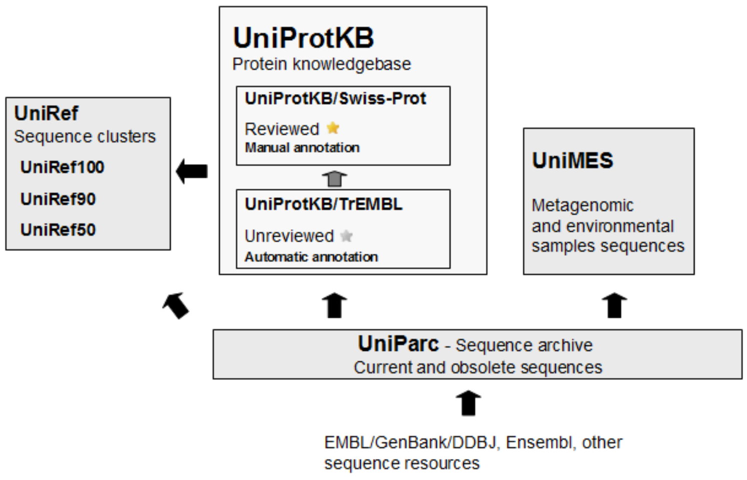


Figure 3.8: Overview and structure of the UniProt database.
(Taken from http://www.uniprot.org/help/about)

All information is extracted from scientific articles, proof-checked and finally stored in the database. The manual database curation is supported by text-mining techniques. Currently, BRENDA has information about 85,000 different scientific articles. Table 3.5 illustrates the current statistics (according to [CSG$^+$09]) of the different areas in BRENDA.

Category	Amount
Names and synonyms	102540
Isolation and preperation	68126
Stability	38196
Reaction and specificity	475510
Enzyme structure	716397
Functional and kinetic parameters	253795
Organism-related information	106218
References	113626
Mutant enzymes	31421
IC50	8474

Table 3.5: Statistic of BRENDA [CSG$^+$09].

3.2.4 TRANSFAC®/TRANSPATH®

In addition to BRENDA the company Biobase sells a multiple of additional products for medical and pharmaceutical research. Biobase provides biological databases, knowledge tools and analysis software for the life science research.

The *TRANSFAC® Suite* is a collection of databases which deal with information about gene expression. Following databases are included in TRANSFAC®[Win04, MKMF$^+$06, Win08].

- ⋆ *TRANSCompel®:* Focuses on composite elements of transcription factors.
- ⋆ *TRANSPro:* Provides systematic promoter analysis of co-regulated genes. Known transcription factor binding sites are indicated.
- ⋆ *PathoDB:* Links gene mutations to their corresponding diseases.
- ⋆ *S/MART DB:* Analyzes eukaryotic regulatory chromatin domains.

The suite is based on the TRANSFAC® database which could be categorized in *Nucleotide Sequence Databases*. TRANSFAC® contains information about transcription factors, their binding sites, nucleotide distribution matrices and regulated genes. Among these commercial versions, a public available database of TRANSFAC® is available. However, this alternative is only accessible for academic use or nonprofit organizations. Furthermore, the content of the database is not that comprehensive and updated than the commercial version of the database. Products like TRANSCompel® are also not available in the public version of the TRANSFAC® database. Table 3.6 illustrates the differences between the public and commercial version. In the meantime the TRANSFAC® database has more than 50 tables whereas the listed main tables, shown in the list below, have nearly links to all other tables. Each of these main tables have specific information related to a particular transcription factor.

- ⋆ *Factor:* Information about transcription factors.
- ⋆ *Site:* Artificial binding sites, which are sites from random oligonucleotide selection assays and IUPAC consensus sequences.
- ⋆ *Gene:* Information about regulated genes.
- ⋆ *Cell:* Cell lines and other kinds of factor sources. Those sources can be used for detection of binding sites or binding activity.
- ⋆ *Matrix:* Stores nucleotide distribution matrices, which are derived from a collection of binding sites for a particular transcription factor.
- ⋆ *Class:* Classification of factor entries into a classification hierarchy.

Stored data can be processed by specialized programs such as *PatSearch* [GLL$^+$03], *MatInspector* [CFG$^+$05] and *FastM* [KFQW99]. For instance, with these programs it is possible to predict or find transcription factor binding sites in DNA sequences. In addition to this, the TRANSFAC® Suite provides two programs *Match*™ and *Patch*™ for transcription factor binding site searches.

Table	**Commercial: TRANSFAC® 2008.3**	**Public available: TRANSFAC® 7.0**
Factor	11683	6133
Site	30227	7915
Gene	33159	7915
CHIP-chip Fragments	141595	Not available
Matrix	856	398
Reference	18959	Not available

Table 3.6: Comparison of the commercial TRANSFAC® 2008.3 and public available TRANSFAC® 7.0 version by the number of database entries.

The TRANSPATH® database [KPV+06] is also a product of Biobase and extends the knowledge of TRANSFAC®. Similar to TRANSFAC®, TRANSPATH® is also a member of the category *Nucleotide Sequence Databases*. The database contains data about signaling molecules, metabolic enzymes, second messengers, endogenous metabolites, miRNAs and the reactions that are involved in signal transduction pathways of a cell. Furthermore, TRANSPATH® provides upstream regulators and downstream targets of each molecule in the regulatory network. Today, the understanding of signaling pathways is more and more important for the development of new drugs. Therefore, *PathwayBuilder™* and *ArrayAnalyzer™* have been included in TRANSPATH® [CKK+04]. Hence, the PathwayBuilder™ provides several types of network visualizations and the ArrayAnalyzer™ is used for gene expression array integration. Moreover, it is possible to identify key molecules within signaling pathways.

Table	**Commercial: TRANSPATH® 2008.3**	**Public available: TRANSFAC® 7.0**
Molecule	97202	23384
Reaction	144759	32752
Gene	29391	10962
Maps	71	5
Reference	34706	9531

Table 3.7: Comparison of the commercial TRANSPATH® 2008.3 and public available TRANSPATH® 6.0 version by the number of database entries.

Among these commercial versions a public available database of TRANSPATH® is available. However, this alternative is only accessible for academic use or nonprofit organizations. Furthermore, the content of the database is not that comprehensive and is not often updated, unlike the commercial version which is updated on a regular basis. Products like PathwayBuilder™ and ArrayAnalyzer™ are also not available in the public version of the TRANSPATH® database. Table 3.7 illustrates the differences between the public available and commercial version. In the meantime the TRANSPATH® database has more than 30 tables whereas the listed main tables, shown in the list below, have nearly links to all other tables.

- ⋆ *Molecule:* Proteins and other components that transduce extracellular signals to target genes.
- ⋆ *Reaction:* Information on reactions between signaling molecules that constitute regulatory pathways and networks.
- ⋆ *Gene:* Information on target genes and gene expression as starting points for regulatory pathways or feedback loops.
- ⋆ *Pathway:* .Canonical pathways for specific signaling molecules (mostly ligands and receptors).
- ⋆ *Reference:* References with links to PubMed for each entry.

3.2.5 OMIM

The OMIM database [HSA$^+$05] could be categorized in *Human Genes and Diseases*. **O**nline **M**endelian **I**nheritance in **M**an (OMIM™) is a database that contains knowledge about human genes and genetic disorders. Moreover, the information should support human genetics research of clinical genetics. OMIM was initiated in 1966 by Dr. Victor A. McKusick as a catalog of mendelian traits and disorders, entitled Mendelian Inheritance in Man (MIM). In 1985 OMIM was launched by a collaboration between the National Library of Medicine and the William H. Welch Medical Library at Johns Hopkins. Since 1987 a public version of OMIM is available on the internet and since 1995 the database was developed for the World Wide Web by NCBI (National Center for Biotechnology Information). Furthermore, OMIM was enhanced with links to sequence databases and medical literature. Today, OMIM is part of the Entrez information system of the NCBI.

Indentifier	**Description**
1 - - - - - (100000-)	Autosomal loci or phenotypes (entries created before May 15, 1994)
2 - - - - - (200000-)	
3 - - - - - (300000-)	X-linked loci or phenotypes
4 - - - - - (400000-)	Y-linked loci or phenotypes
5 - - - - - (500000-)	Mitochondrial loci or phenotypes
6 - - - - - (600000-)	Autosomal loci or phenotypes (entries created after May 15, 1994)

Table 3.8: Numbering system that is used in the OMIM database.

OMIM is freely accessible and is frequently edited and updated by the John Hopkins University. Currently, OMIM has about 18344 gene and phenotype entries. All entries have an unique identifier, which consists of a number and a prefix. Table 3.8 and table 3.9 illustrates the components of the identity key. Hence, the number describes the kind of inheritance. Additionally, an identification of different mutations is possible by the four digit number. For instance, an OMIM identifier has the following number *#261600*. The *#261600* number is conform to the PKU (Phenylketonuria) disease. A mutation of a gene that codes for the enzyme PAH is in charge for the PKU disease. By now more than 400 mutations of genes are known that are related to PKU. However, the OMIM number **612349.001* is one mutation of this gene.

Prefix	**Description**
*	A gene of known sequence
#	Descriptive entry, usually of a phenotype, and does not represent a unique locus. The reason for the use of the #-sign is given in the first paragraph of the entry. Discussion of any gene(s) related to the phenotype resides in another entry(ies) as described in the first paragraph
+	Entry contains the description of a gene of known sequence and a phenotype.
%	Entry describes a confirmed mendelian phenotype or phenotypic locus for which the underlying molecular basis is not known.
^	Entry no longer exists because it was removed from the database or moved to another entry as indicated.

Table 3.9: Symbols preceding a MIM number and their meaning.

3.2.6 GO

The term *ontology* has been used in computer science for approximately 15 years. Ontology is the philosophical study of the nature of being, existence or reality. Traditionally, ontology deals with questions concerning what entities exist or can be said to exist. Ontology has developed to encompass the research into how such entities can be grouped within a hierarchy, and subdivided according to similarities and differences.

Beside the Human Genome Project (HGP) many other genome projects are arisen. Therefore, it was necessary to develop a catalog that annotates proteins of different organism in the same manner. Based on this vocabulary it is possible to avoid different annotations, e. g. *translation* and *protein synthesis*, for the same protein.

The Gene Ontology project [HCI+04] was founded in 1998 as collaboration of the databases FlyBase, Saccharomyces Genome Database (SDG) and Mouse Genome Database (MGD). In 2004 the Gene Ontology Consortium was founded as consequence and since then many databases has been included. GO (Gene Ontology) could be categorize in *Genomics Databases (non-human)* as described in section 2.2. The vocabularies are structured in a classification that supports *is-a*, *part-of*, *regulates, positively_regulates* and *negatively_regulates* relation-ships. In general GO consists of three controlled vocabularies *molecular function, biological process* and *cellular component.* For instance a gene product can be part of all three vocabularies, so it is clear the catalog is not hierarchical organized, but rather a network that consists of many nodes. Those nodes could have more than one root. Such a structure makes sense, because gene products are not controlled by a hierarchy, rather they act together as a network.

3.2.7 Summary

In this section a selection of most relevant data sources related to this work were introduced. Starting from KEGG which provides information about organism specific metabolic knowledge and genomic, chemical and systematic information. An advantage of KEGG is the integration and linkage of genomic and chemical information. As well as general and organism specific metabolic pathway maps. UniProt contains data about proteins and their function and structure. The Universal Protein Knowledgebase is online available and is updated in a fourteen days cycle. It consists of the three protein databases SWISS-Prot, TrEMBL and PIR-PSD. The BRENDA database contains information about a wide range of enzymes. All information is extracted from scientific articles, proofed and finally stored in the database. The manual database curation is supported by text mining techniques. TRANSFAC® contains information about transcription factors, their binding sites, nucleotide distribution matrices and regulated genes. In addition, TRANSPATH® database contains data about signaling molecules, metabolic enzymes, second messengers, endogenous metabolites, mRNAs and the reactions that are involved in signal transduction pathways of a cell. Furthermore, the database provides upstream regulators and downstream targets of each molecule in the regulatory network. OMIM contains knowledge about human genes and genetic disorders. Moreover, the information should support human genetics research of clinical genetics. Finally, GO consists of three controlled vocabularies: molecular function, biological process and cellular component to specify gene annotations. An overview of the mentioned databases is shown in table 3.10.

In order to take advantage of the potential of various valuable biology databases, it must be considered that bioinformatics is an inherently integrative discipline, requiring access to data from a wide range of sources. Consequently, the integration of several data sources is necessary to provide information with a new level of quality.

3.3 Modeling and Simulation of Biological Networks

The vast complexity of biological systems requires modeling to design and interpret biological experiments. For the implementation of the virtual cell, the fundamental question is how to model and simulate complex metabolic networks. Since last decades research activities took place on biological phenomena with the motivation to understand biolog-

Database	NAR Category	Access	Interface	Exchange format	Identifier	Updates	Costs
KEGG	Genomics Database (non-huma)	Flatfile, WWW, SOAP	WWW, SRS, DBGET	HTML, XML, ASCII	Unique identifier	Periodic updates	Free
UniProt	Protein Sequence database	Flatfiles, WWW	WWW, FASTA, DBGET, BLAST	HTML, XML, ACSII, FASTA	Unique entry name	14 days periodic update	Free
BRENDA	Metabolic Enzymes and Pathways; Singaling Pathways	Flatfiles, WWW, SOAP	WWW, SRS	HTML, ASCII	Unique EC number	Periodic updates	Free, License
TRANSFAC®	Nucleotide Sequence Databases	Flatfiles, WWW,	WWW, SRS, DBGET	HTML, ACSII	Unique identifier	Daily update, quarterly updates	(Free), Commercial
TRANSPATH®	Nucleotide Sequence Databases	Flatfiles, WWW	WWW, SRS	HTML, ASCII	Unique identifier	Daily update, quarterly updates	(Free) Commercial
GO	Genomics Databases (non-human)	WWW, MySQL	WWW, DBGET, BLAST, AmiGO	HTML, XML, ASCII	Unique GO number	Periodic updates	Free

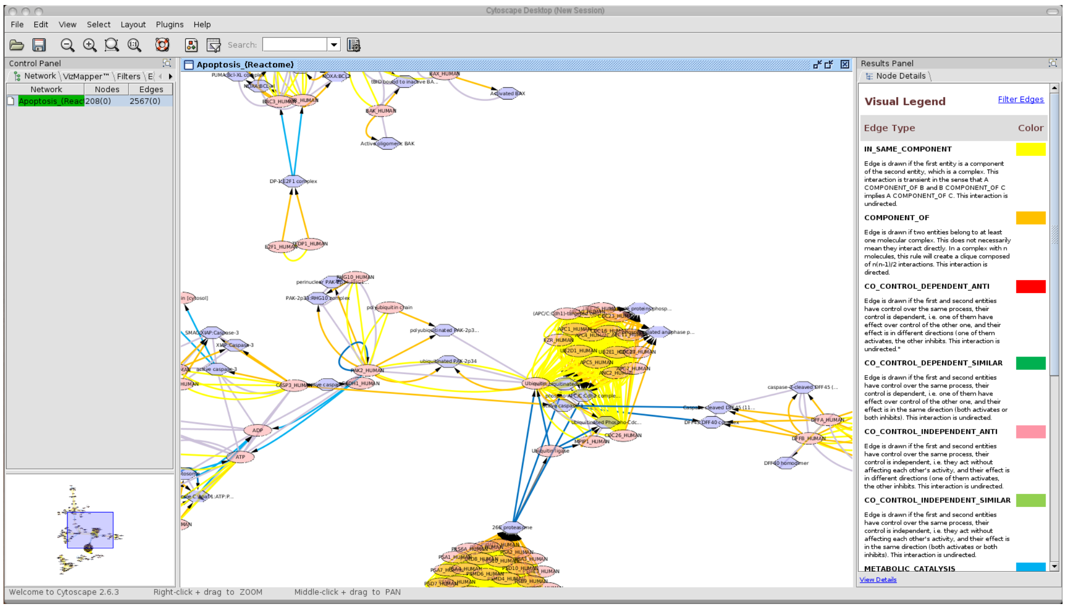
Table 3.10: Overview of similarities and differences of the databases related to this work.

ical processes. And of course the field of bioinformatics has made a contribution to this research as well. This section deals with some applications and tools that were realized in order to create, edit and analyze biological networks. It gives an insight into the related network modeling software solutions and approaches. Accordingly, only a selection of widely used modeling and simulation environments will be given. Cytoscape is one of the most widely used bioinformatics software platforms for visualizing molecular interaction networks and biological with a lot of useful plug-in tools. VisANT and VANTED are visualization and analysis approaches that are also used in several projects as described in literature.

3.3.1 Cytoscape

Cytoscape is an open source software environment to visualize and integrate biomolecular interaction networks [SMO+03]. The software provides the integration of networks with high-throughput expression data and other molecular states into a unified conceptual framework. Moreover, Cytoscape is applicable to almost all systems of molecular components and interactions. The user is able to access several protein-protein, protein-DNA and genetic interaction databases. In addition to this, the user has the opportunity to layout and query the network for information of interest. Furthermore, the application is able to visually integrate the network with expression profiles, phenotypes and other molecular information (see figure 3.9). It is also possible to link the created networks to databases that contain functional annotations. The core of the framework is extensible through a plug-in architecture. It allows development of additional computational analyses and features. By this plug-in structure it is possible to extend the software and its possibilities by new developments and features without changing the core of the software framework. Plug-ins can be developed by using the Cytoscape open API based on Java. At the moment about eight institutions are involved in the development of the software. Hence, many plug-ins are available for Cytoscape. The main goal of the software is to cover all fields of activities which can be concerned to biological network modeling and analysis.

One of the main advantages of software is the extensible plug-in architecture. Using Cytoscape it is possible to integrate a plenty of important biological databases, including BRENDA and KEGG, into the workflow. But integration of further databases seems to be quite complex. In addition, the integrated databases need to be queried by specific criteria, but Cytoscape only provides a database search by names and element ids. Research activities need a more powerful search that enables a powerful navigation through the databases in order to find specific information as much as relations between biological elements and systems.


Cytoscape Desktop (New Session)
File Edit View Select Layout Plugins Help
Control Panel
Network VizMapper Filters
Apoptosis_(Reactome)
Results Panel
Node Details
Welcome to Cytoscape 2.6.3
Right-click + drag to ZOOM
Middle-click + drag to PAN

Network	Nodes	Edges
Apoptosis_(Reac	208(0)	2567(0)

Visual Legend Filter Edges

Edge Type	Color
IN_SAME_COMPONENT Edge is drawn if the first entity is a component of the second entity, which is a complex. This interaction is transient in the sense that A COMPONENT_OF B and B COMPONENT_OF C implies A COMPONENT_OF C. This interaction is undirected.	🟨
COMPONENT_OF Edge is drawn if two entities belong to at least one molecular complex. This does not necessarily mean they interact directly. In a complex with n molecules, this rule will create a clique composed of n(n-1)/2 interactions. This interaction is directed.	🟧
CO_CONTROL_DEPENDENT_ANTI Edge is drawn if the first and second entities have control over the same process, their control is dependent, i.e. one of them have effect over control of the other one, and their effect is in different directions (one of them activates, the other inhibits. This interaction is undirected."	🟥
CO_CONTROL_DEPENDENT_SIMILAR Edge is drawn if the first and second entities have control over the same process, their control is dependent, i.e. one of them have effect over control of the other one, and their effect is in the same direction (both activates or both inhibits). This interaction is undirected.	🟩
CO_CONTROL_INDEPENDENT_ANTI Edge is drawn if the first and second entities have control over the same process, their control is independent, i.e. they act without affecting each other's activity, and their effect is in different directions (one of them activates, the other inhibits. This interaction is undirected.	🟥
CO_CONTROL_INDEPENDENT_SIMILAR Edge is drawn if the first and second entities have control over the same process, their control is independent, i.e. they act without affecting each other's activity, and their effect is in the same direction (both activates or both inhibits). This interaction is undirected.	🟩
METABOLIC_CATALYSIS	🟦

View Details

Figure 3.9: Visualization of the apoptosis pathway with Cytoscape.

3.3.2 VisANT

VisANT is a web-based software framework for visualizing and analyzing biological networks. This framework provides the user with a visual graphical interface to combine and annotate network data [HMW$^+$05]. Furthermore, VisANT supports functional and annotation data from KEGG and GO. The software includes statistical and analytical tools that are useful for extracting topological properties of user-defined networks. Additionally, the user can extract network models out of different data sources such as KEGG, GO or Saccharomyces genome database. Moreover, the user can customize, modify, save and share network views with other users. It is also possible to import basic network data representations from own data sources and exchange formats such as PSI-MI (Proteomics Standards Initiative Molecular Interaction XML Format) and BioPAX (Biological Pathways Exchange Format). The possibility to analyse network data for topological statistics and features is of broad prospect. VisANT supports the development of visualization and analysis functions through an open API based on Java for plug-ins.

The idea of integrating information from a variety of sources can give some interesting insights into a biological system. VisANT provides a wide range of visualization properties and analysis function. An example expression profile is shown in figure 3.10. On the other hand, one of the main disadvantages is that the user is not able to create his own network model. Rather than the software only provides the possibility to import already existing networks. Although, the user is able to customize and modify loaded networks. The user is limited to very simple relations and biological elements like enzymes, proteins and compounds. Biological aspects and relations cannot be captured in its relevant details by

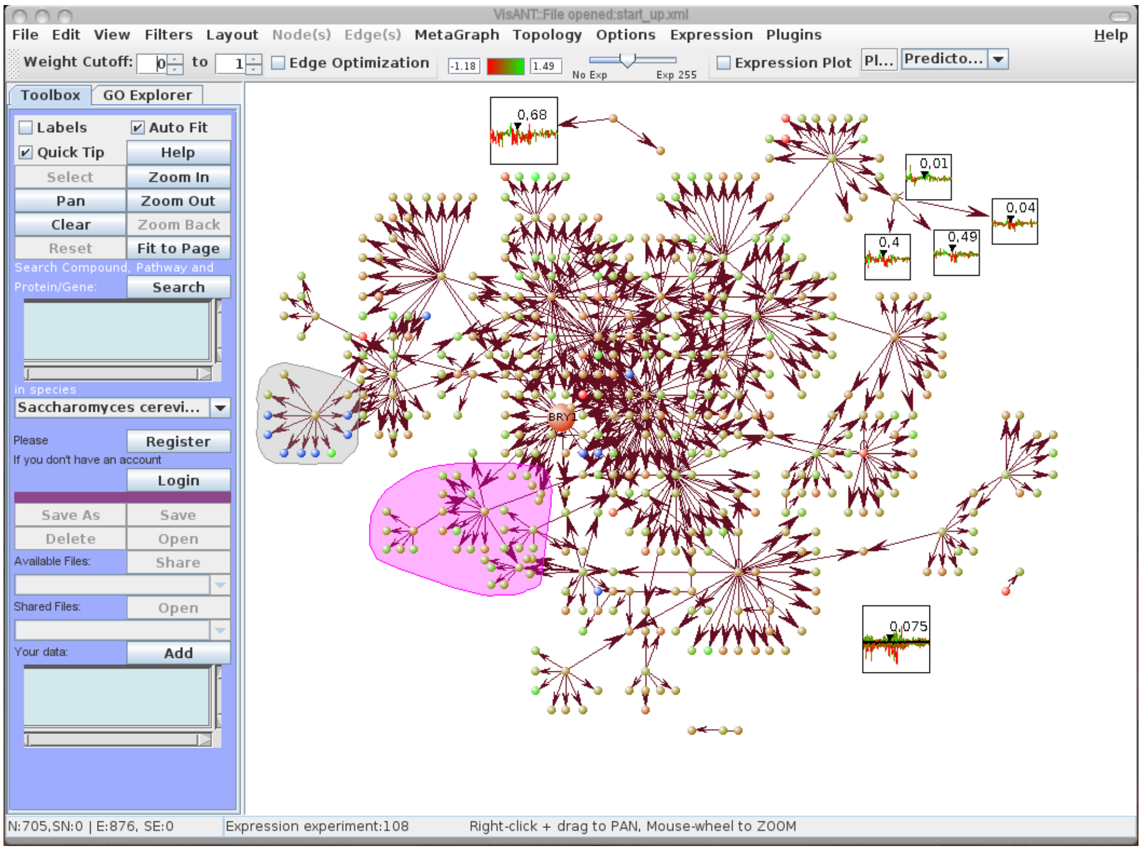


Figure 3.10: Visualization of an expression experiment using VisANT.

this framework.

3.3.3 VANTED

VANTED is a visualization and analysis tool for biological networks that are related to experimental data [JKS06]. The software supports import of large-scale biochemical data via a Microsoft Excel-based form. The data can be mapped on the network that is drawn by the application. Furthermore, it is possible to download networks directly from the KEGG database or import networks via standard exchange formats such as GML (Graph Modelling Language) or SBML (Systems Biology Markup Language). Different types of data such as transcript, enzyme and metabolite data can be visualized in context of the underlying networks. Moreover several statistical tests, correlation analysis and self-organizing maps for the clustering of time series data sets are implemented.

VANTED is a platform-independent tool based on Java that allows the visual analysis of large-scale biochemical data sets in the context of relevant networks. One of the main advantages of VANTED is the creation of correlation networks from biochemical data.

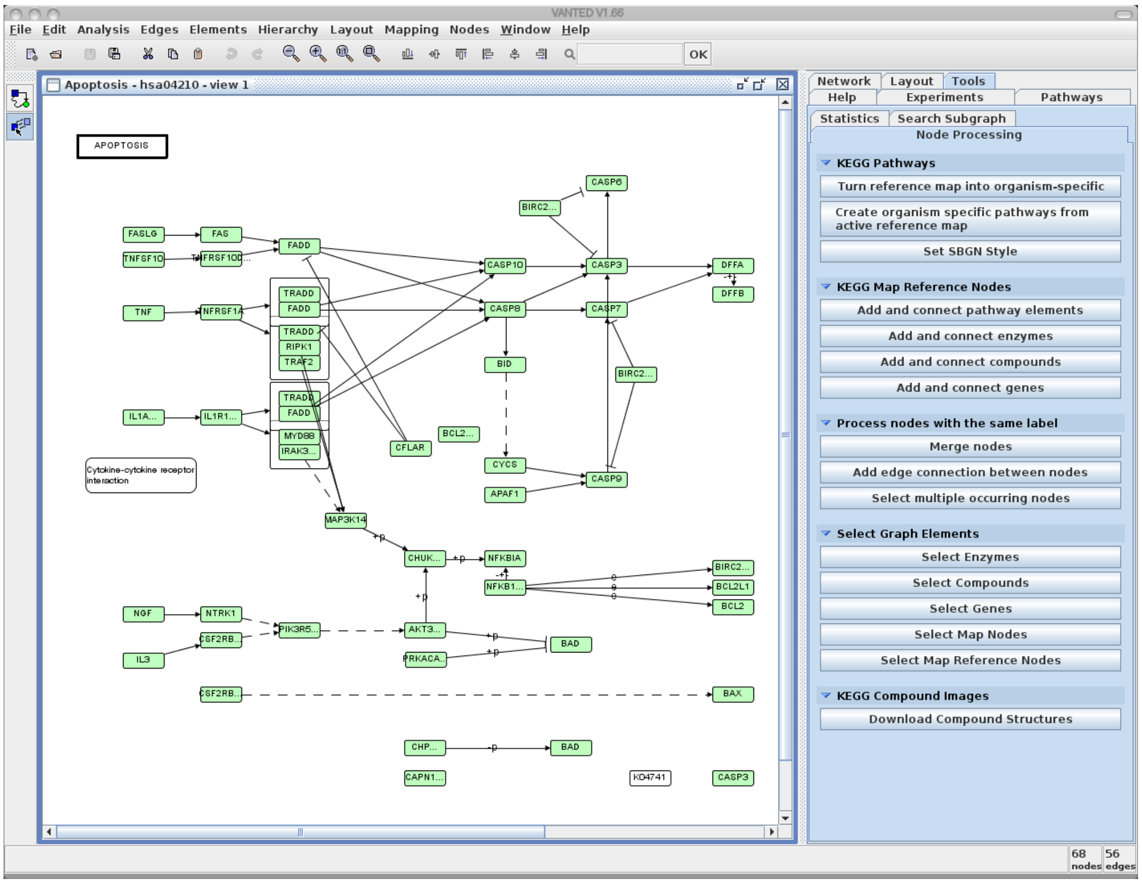

VANTED V1.66
File Edit Analysis Edges Elements Hierarchy Layout Mapping Nodes Window Help
Apoptosis - hsa04210 - view 1
Network Layout Tools
Help Experiments Pathways
Statistics Search Subgraph
Node Processing
KEGG Pathways
Turn reference map into organism-specific
Create organism specific pathways from active reference map
Set SBGN Style
KEGG Map Reference Nodes
Add and connect pathway elements
Add and connect enzymes
Add and connect compounds
Add and connect genes
Process nodes with the same label
Merge nodes
Add edge connection between nodes
Select multiple occurring nodes
Select Graph Elements
Select Enzymes
Select Compounds
Select Genes
Select Map Nodes
Select Map Reference Nodes
KEGG Compound Images
Download Compound Structures
68 nodes
56 edges

Figure 3.11: Visualization of the apoptosis pathway with VANTED.

Hence, the tool provides researcher the possibility to generate and visualize networks in the context of biochemical pathways as shown in figure 3.11. The limitation of the software is that is not practicable to visualize genome-wide data. Integration of further pathways and biological databases might be useful for research and better insight into biological networks.

3.3.4 Summary

In this section a selection of widely used modeling and simulation environments were presented. All of the presented software solutions have their own way to visualize and model biological networks. They provide a wide range of visualization properties and analysis function, as well as plug-in interfaces to support the development analysis functions. However, the main disadvantages of all these applications are a limited connection to multiple databases or a data warehouse. Integrated databases and data warehouses are an essential step in constructing biological networks. Based on data integration we will present systems for semi-automated reconstruction of biological networks in this work.

3.4 Petri Net Modeling and Simulation

Petri nets are a well known formalism to model dynamic aspects of systems. They have found many applications over the last decades and are increasingly used for greatly varying purposes. Regarding the published papers, it is clear that hybrid functional Petri nets are an adequate method to model complex biological networks. Prof. Dr. Carl Adam Petri presented the formalism of Petri net. The Petri nets were developed for the simulation and description of concurrency during the 60s [Pet62, Pet66]. Petri nets are natural, simple, and powerful methods for describing and analyzing the flow of information and control in systems. Their properties, concepts, and techniques are simple and could be approached and utilized in a large number of ways. In particular, Petri nets are systems that may exhibit asynchronous and concurrent activities. Their major use is the modeling of systems of events in which it is possible for some events to occur concurrently. On the concurrence, precedence, or frequency of these occurrences are constraints defined [Pet77].

3.4.1 Petri Nets

A Petri net (also known as a place/transition net or P/T net) is one of several mathematical representations of discrete distributed systems. As a modeling language, it graphically depicts the structure of a distributed system as a *directed bipartite graph* with annotations. A Petri net consists of *places, transitions and directed arcs*. The nodes are connected with directed arcs, where an arc is never a connection between two nodes of the same type. Places may be marked by an integer number of tokens. A transition has concession (a transition fires) if all conditions of concession are satisfied. Concession of a transition means that the number of tokens at every place before the transition is decreased by one and the number of tokens at every place behind it is increased by one. "A place *before* a transition" means the existence of an arc leading from that place to the transition. Correspondingly "*after* a transition" means the existence of and arc leading from the transition to that place. The concession condition for a transition is satisfied if the amount of tokens of every place before the transition is greater or equal than one. In graphical representation places are displayed as circles and transitions as rectangles. A formal definition of an ordinary Petri net according to [PW03] is given in the following.

Definition 3.1 *A Petri net graph, PN is a tuple $(P, T, \mathbb{F}, \mathbb{B})$ of*

- ⋆ *a finite, ordered set $P = p_1, \ldots, p_{|P|}$ of places,*
- ⋆ *a finite, ordered set $T = t_1, \ldots, t_{|T|}$ of transitions,*
- ⋆ *with $P \cap T = \emptyset$,*
- ⋆ *a $|P| \times |T|$-matrix $\mathbb{F}$ of $\mathbb{B}$ with*
- ⋆ *a $|P| \times |T|$-matrix $\mathbb{B}$ of $\mathbb{N}$*

In the graphical notation each place is represented as *circle* and each transition as *rectangle* and from each and every place p_i has exact $\mathbb{F}_{i,j}$ arcs lead to transistion t_j. Thus, from transition t_j has exact $\mathbb{B}_{i,j}$ arrows lead to place p_i.

$\mathbb{B}$ is also called *backward matrix* and consequently $\mathbb{F}$ is called *forward matrix*. From the view of the places: Arrows starting from a place, means forward directed, so they are written in the forward matrix. On the other hand looking back from a place, means looking along the arc to a place, the places are written in the backward matrix.

Definition 3.2 *Let $(P, T, \mathbb{F}, \mathbb{B})$ be a Petri net. Thus $F : P \times T \cup T \times P \to \mathbb{N}$ is the arc function, defined as $\forall x, y \in P \cup T$:*

$$F(x,y) := \begin{cases} \mathbb{F}_{i,j}, \text{ if } x = p_i \text{ and } y = t_j \\ \mathbb{B}_{i,j}, \text{ if } x = t_j \text{ and } y = p_i \end{cases}$$

For $x \in P \cup T$ we call $\bullet x := \{y \in P \cup T | F(y,x) \geq 1\}$ pre-set of x,
$x\bullet := \{y \in P \cup T | F(x,y) \geq 1\}$ post-set of x.

Analogue for $X \subseteq P \cup T : \bullet X := \bigcup_{x\in X} \bullet x$, $X\bullet := \bigcup_{x \in X} x\bullet$.

Typically, Petri nets are also defined as (P, T, F) nets. This illustration is in principle equivalent to definition 3.1. If a Petri net does not contain multiple arcs, then F can be considered as a subset of $P \times T \cup T \times P$. In *generalized Petri nets* are explicitly multiple arcs allowed.

Definition 3.3 *Let $N = (P, T, \mathbb{F}, \mathbb{B})$ respectively $N = (P, T, F)$ be a Petri net. The state space of those Petri nets is $\mathbb{N}^P$. A mapping $s : P \to \mathbb{N}$ is called* ***state*** *or* ***marking****. If $s(p_i) = k$ we can say that place p_i has k token in state s. We can also write states as column vector.*

Graphically token are marked as dots in a place.

The dynamic behavior of a Petri net is expressed by changing markings. A marking changes when a transition fires, and a transition may fire when it is enabled.

Definition 3.4 *Let $N = (P, T, \mathbb{F}, \mathbb{B})$, $s \in \mathbb{N}^P$ a state of N an $t \in T$ a transition of N. We call t s-enabled (transition may fire in state s) if*

$s \geq \mathbb{F}(t)$
(i.e. $\forall p \in P : s(p) \geq \mathbb{F}_{p,t} = F(p,t)$)

Continuing, t fires from state s to state s', if

- $\star$ $s \geq \mathbb{F}(t)$
- $\star$ $s' = s - \mathbb{F}(t) + \mathbb{B}(t)$

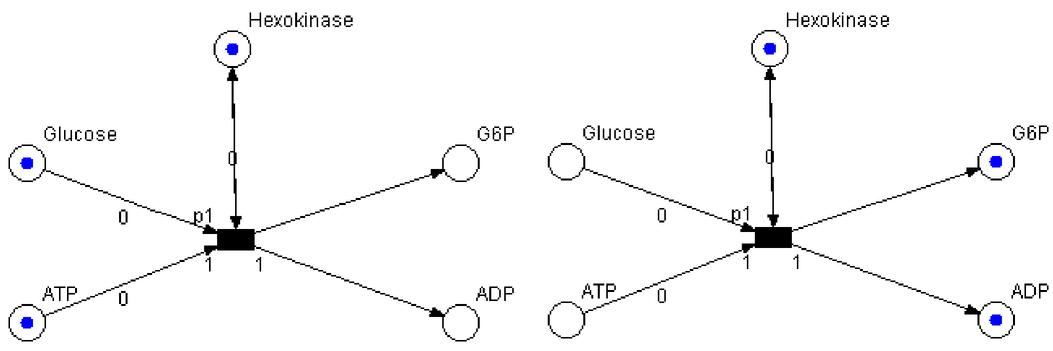
State changes are carried out by firing enabled transitions. In an ordinary Petri net a transition is enabled when all its input places have at least one token. When an enabled transition t is fired, tokens are consumed by each input place of t and tokens are placed at each output place of t. Finally, it gives a new state.

3.4.2 Petri Net Models for Biological Networks

During the 90s, when the field of bioinformatics was getting stronger, modeling and simulation of metabolic pathways became a fundamental research topic. The first Petri net application for metabolic simulation was published by Reddy [RML93] and Hofestädt [Hof94]. They showed that formalism of Petri nets is useful to model simple metabolic networks. The first papers already illustrated that in addition to the possibility of modeling parallel processes. One important advantage of this formalism is that the Petri net simulation can start with a discrete model. Moreover, these discrete models can be extended to a quantitative model step-by-step and at any point in time [HT98].

Consequently, Petri nets are useful for the representation of biochemical reactions and metabolic pathways. Figure 3.12 shows a model of an abstract enzyme-controlled reaction, which is the basic element of metabolic pathways. First of all the paper of Reddy [RML93] addressed the modeling of metabolic pathways as Petri nets. The disadvantage of this application is that kinetic effects cannot be included in the model. Based on this paper the work of Hofestädt [Hof94] showed that a Petri net can also model gene-controlled metabolic networks and cell communication processes. The idea of Hofestädt and Thelen [HT98] was to include the kinetic effects. Therefore, they introduce the *functional Petri net* (FPN), which allows the kinetic simulation of metabolic networks by placing specific functions to the arcs. Kinetic effects of biological networks can be simulated can be simulated by using FPNs. Thus, any qualitative Petri net model of a biological network can be extended by using the FPN. Of course quantitative experimental data (for instance quantitative proteomic data or kinetic data) can be included. In general, extensions of more complexity lead to high-level Petri-Nets. These high-level nets include following type of nets.

- ⋆ *Hierarchical Petri nets*: Allow a place or a transition being a place-holder for a predefined net.
- ⋆ *Coloured Petri nets*: Allow different kinds of tokens in one net. The tokens can have a complex internal structure. Accordingly, transitions can have more complex firing rules.
- ⋆ *Stochastic Petri nets*: Places and transitions may be assigned with a probability distribution.
- ⋆ *Timed Petri nets*: Assign a delay time to each transition.



(a) Start configuration of the enzymatic reaction shows an abstract glucose molecule, an abstract enzyme hexokinase and energy (ATP).

(b) Final condition of the Petri net shows the result of the biochemical reaction.

Figure 3.12: The enzyme-catalyze process of glucose into glucose-6-phosphate.)

⋆ *Continuous Petri nets*: They do not have tokens, but introduce the concept of continuous flow. Using this concept, it is no longer a natural number that is assigned to a place, but a real number. Transitions can be equipped with a complex mathematical function implying actual assignment of places. Hence, continuous Petri nets are expedient for modeling of metabolic pathways with regard to the concentration flow.

⋆ *Hybrid Petri nets*: A mixture of ordinary (discrete) and continuous Petri nets. Places and transitions can be either discrete or continuous so that they deal with natural numbers (tokens) or a continuous flow. This type of high-level Petri net extends the continuous net with the useful ability to model logical coherences.

Other publications published by Matsuno [MDNM00] showed that higher Petri nets, in particular hybrid Petri nets, present the formalism which is useful to model and simulate complex biological networks. The motivation to define the *hybrid functional Petri net* (HFPN) was to allow the quantitative modeling of biological networks, which is still one of the major tasks of Systems Biology today. Having more and more quantitative molecular data the realistic simulation of biological networks required more extensions. The usage of real numbers is one important aspect. The HFPN is an extension of the *Hybrid Petri net* (HPN) [AD98] and hybrid dynamic net (HDN) [Dra98]. Additionally, HFPN has the feature of the functional Petri nets and allows assigning a function of values to any arc.

Based on these ideas several simulation tools were created and implemented, such as the Cell Illustrator.

3.4.3 Cell Illustrator

The Cell Illustrator[1] is a promising and widely used tool for biological network modeling and simulation. A first version of the Cell Illustrator (CI) is available since mid 2003 and has been evolved from *Genome Object Viewer* [NDMM05]. Since this version CI has been developed to a standard software for modeling and simulation of pathways.

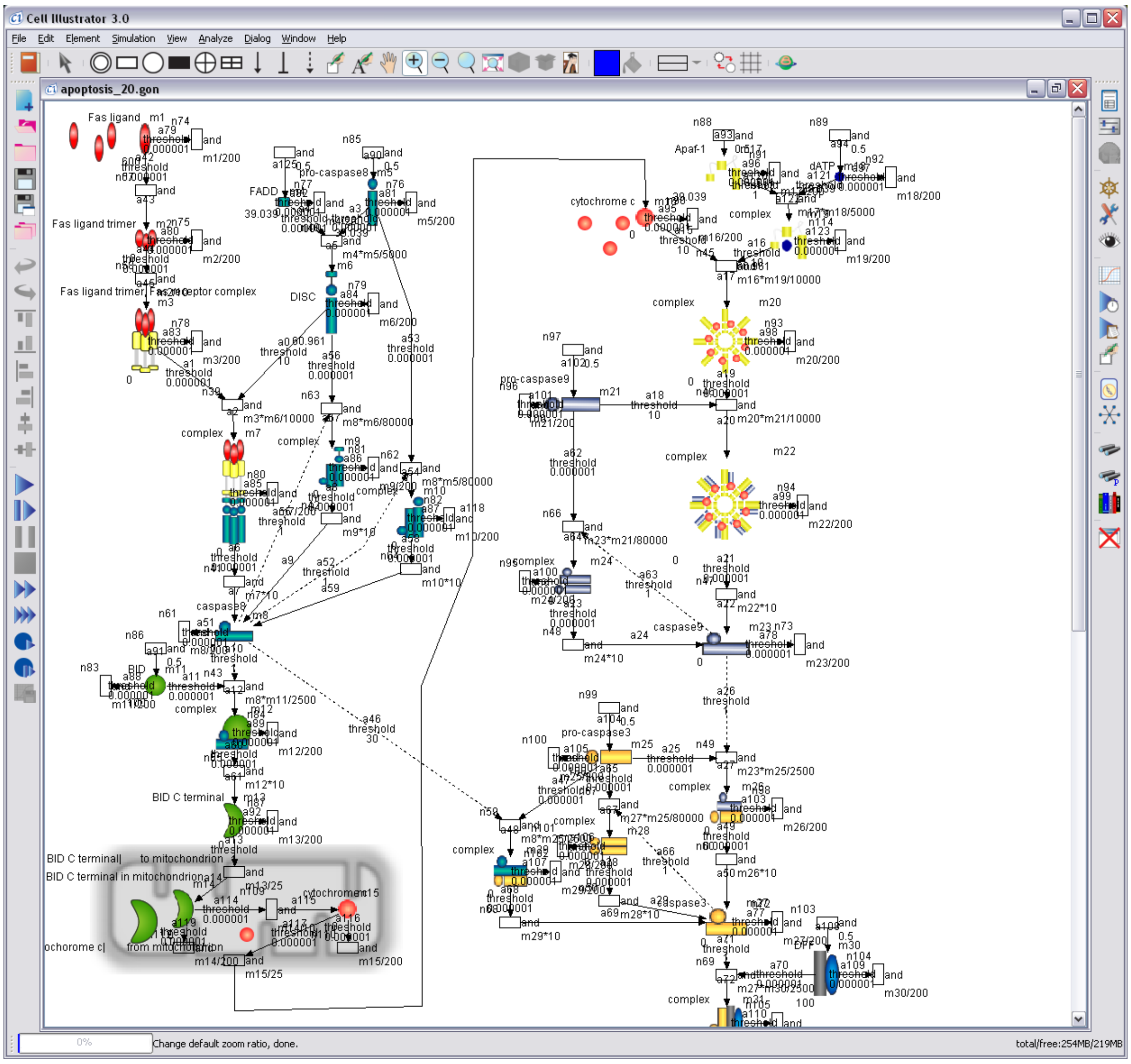


Figure 3.13: Visualization of the apoptosis pathway with the Cell Illustrator.

The CI employs the HFPN (described in section 3.4.2) as a basic architecture. CSML (Cell System Markup Language) as description language was developed for the CI. The goal of CSML 3.0 is to create a usable XML format for visualization, modeling and simulation of biological pathways that are based on HFPNs. Moreover, CSML is able to store biological

[1]http://www.cellillustrator.com/

images, result sets of simulation, as well as biological annotations and therefore it is more powerful than CellML or SBML. Both formats, SBML and CellML are widely used and many models are already available, so it is possible to import via conversion tools these formats into the CI [NDMM05].

However, the CI enables scientist to draw, edit, model and simulate complex biological systems and networks. Regarding to the HFPNs it is possible to create pathways of discrete, continuous and generic entities and transitions, as well as typical, inhibitory and "test" arcs. A "test arc" can be used like typical arcs. In contrast a "test arc" does not consume any tokens of the place at the source of the arc by firing. An inhibitory arc can be used to represent the function of "repress" in gene regulation. A graphical representation of all elements in a HFPN is shown in figure 3.14. For each and every element in the pathway the CI provides miscellaneous properties. Starting from labeling, initial concentration, shape and color, up to biological functions, database identifier and images every setting is possible. Figure 3.13 shows an example network of CI tool. Biological networks modeled with the Cell Illustrator can be visualized and animated using Cell Animator™. Additionally, the CI since version 3.0 includes a library of SVG graphical elements, enhancements in the Petri net simulation engine, improved charts, external references, and gene network view. In summary, Cell Illustrator provides biologists, biochemists, and scientists with comprehensive visual representations of biochemical processes which form large and complex networks.

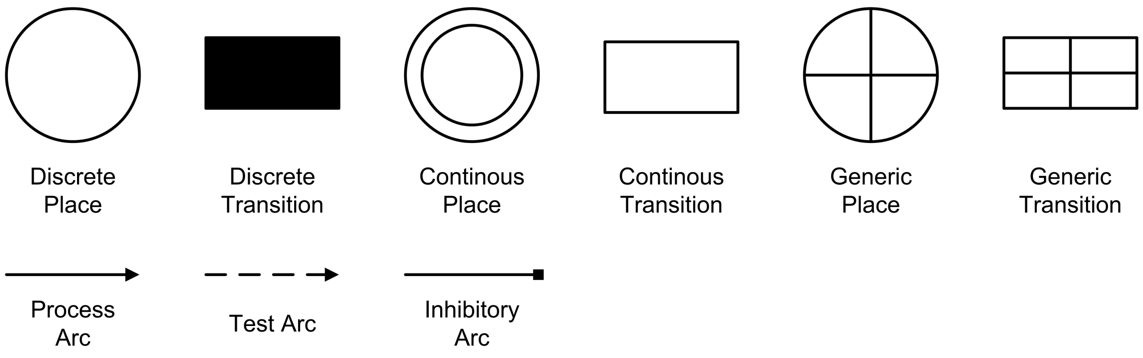


Figure 3.14: Graphical representation of HFPN components.

3.5 Summary

One of the major challenges in bioinformatics is to integrate, combine diverse and multiple data and to bring them into a homogenous, consistent state. So that scientists can analyze information from different data sources to meet their objectives. Consequently, in section 3.1, were presented existing systems Atlas, BioWarehouse, Columba, ONDEX and CoryneRegNet with a comparison illustrated in table 3.2.

Information about biological networks and cellular processes can be found in different databases around the world. Therefore, a selection of the most relevant databases, such as KEGG, UniProt, BRENDA, TRANSFAC/TRANSPATH, OMIM and GO are given in section 3.2. All these databases are expert systems for different biological topics or elements. The vision of a virtual cell combines bioinformatics and systems biology. But scientists are still far away from implementing a simple virtual cell *in silico*. The understanding of the metabolism, which is based on gene-controlled biochemical reactions, is a first step in reaching this goal. Therefore, modeling and simulation of metabolic networks becomes more and more important. In section 3.3 the modeling and simulation tools Cytoscape, ViSANT and VANTED were introduced. It is important to perform computer simulations on cellular processes, because these processes are the backbone of a virtual cell. Although cellular systems are information processing systems that are highly complex. Petri nets, in this case hybrid functional Petri nets, are the adequate method to model complex biological networks. Thus, the formalism of Petri nets is described in section 3.4.1.

4 Life Science Data Integration

In the previous chapter several principles and approaches for database integration and network visualization were introduced. A couple of the principles of the introduced integration systems in chapter 3 are well suited to be used within the database integration system that will be presented in this chapter. Furthermore, the requirements for the implementation of a general data warehouse infrastructure will be discussed. General requirements for the integration of biological databases are presented regarding to their feasibility. Particularly the functions of the software toolkit *BioDWH* [TKKH08] will be illustrated more precisely in section 4.2. On the basis of figure 4.1 the different functions of the different components will be illustrated.

An enumeration of requirements for building a data warehouse integration infrastructure using relational databases is given in section 4.1. Furthermore, section 4.2 illustrates the realization of the infrastructure including parser plug-in mechanism, object-relational mapping and graphical user interface. Finally section 4.3 presents the structure as well as the function of the BioDWH data warehouse infrastructure.

4.1 Requirements for Life Science Data Integration

The importance of database integration problems in bioinformatics have been recognized for many years. This section describes an evolving open source toolkit for constructing data warehouses that integrate multiple heterogeneous biological databases within a single physical database management system to facilitate queries that span multiple databases.

For that purpose in this section a software infrastructure for building life science data warehouses using different common relational database management systems is introduced.

Requirement 4.1 *The approach should be implemented as platform independent system that fulfils the purpose of database integration.*

The main goal of this work is to present a novel bioinformatics data warehouse software toolkit that integrates biological information from different public available data sources into a local database management system. To get hold of a wide range of users the system should be platform independent.

Requirement 4.2 *A graphical user interface (GUI) should enable a user to easily construct a user-defined data warehouse.*

The system should provide a graphical user interface, so that a user must not have experience in general scripting, database management systems or programming languages. Thus, the system is addressed to computer scientists, biologists as well as users from outside of computer science. Additionally, the GUI should be easy-to-use and configure automatically the integration process of the data sources.

Requirement 4.3 *The data integration infrastructure should be independent from the underlying relational database system.*

Relational database systems are flexible and robust. The theoretical basis of the relational model makes it possible to protect and guarantee the integrity of the data. There are many different relational database systems (RDBMS) such as MySQL, Oracle, PostgreSQL that are used in laboratories and companies all over the world. Therefore, data integration infrastructure should be independent from the underlying DBMS. Accordingly, a persistence layer that will be described in section 4.2.2 is necessary to achieve independence from the RDBMS.

Requirement 4.4 *Configuration of the infrastructure and its components via Extensible Markup Language (XML).*

XML is human readable, well formatted, easy to access and standardized. Therefore, it is easy for users to create, modify and edit configurations manually, if necessary.

Requirement 4.5 *A monitor utility is needed to keep the system up-to-date.*

Most database integration systems and data warehouse approaches do not contain up-to-date data or will not be updated anymore. Therefore, a monitoring utility is necessary to keep the system up-to-date. The software should provide a monitoring component that observes changes in the data sources of the data warehouse.

Requirement 4.6 *A logging mechanism with a recovery function is needed to guarantee a consistent state of the database.*

A logging component should observe the integration process and start a simple recovery process. The error recovery and logging algorithm should also guarantee a consistent state of the data warehouse.

Requirement 4.7 *The data integration infrastructure should provide a wide range of ready-to-use parser.*

The integration software toolkit should provide a number of ready-to-use parsers that extract data from public available data sources and to store the content into a data warehouse. A parser interface should provide a plug-in mechanism. Hence, it should be easy and possible for user to add or remove parser.

4.2 Data Warehouse Infrastructure Application

In this section the architecture of data warehouse infrastructure software to integrate molecular biological and life science data will be outlined. First, an overview of the system will be given and afterwards a detailed description of the different components will be presented.

Point of origin for the concept of the data warehouse system are the requirements introduced in the previous section and the general data warehouse schema (figure 2.12) presented in section 2.2.2.4. Therefore, this section presents a novel concept for a bioinformatics data warehouse software infrastructure that integrates biological information from multiple public life science data sources into a local database management system. It stands out from other approaches, as described in chapter 3.1, by providing up-to-date integrated knowledge, platform and database independence as well as high usability and customization. This open source software toolkit can be used as a general infrastructure for integrative bioinformatics research and development. The advantages of the approach are realized by using a Java-based system architecture and object-relational mapping (ORM) technology.

Figure 4.1 shows the system architecture of the software. Basically, the system consists of the *Data Retrieval* module, the *Data Warehouse Manager* and a *Graphical User Interface* (GUI). The user is able to control the infrastructure via GUI and XML configuration files. Now, the core of the system is the Data Retrieval component that is composed of *Loading-*, *Transformation-*, *Extraction Component* i.e. the *Parser*, *Monitor component* and the *Staging Area*. Beyond this component and beside the *User*, the external *Applications/Analysis Tools* and existing *Data Sources* are essential components for the system. But the most important component is *Data Warehouse Manager* that is appropriate for initiation, control and observation of individual processes starting from extraction of the data sources to the integration of the data into the data warehouse. Accordingly, the Data Warehouse Manager is the interface to all components and the relational database system.

In the following a deeper insight of the different data warehouse infrastructure components will be presented regarding to their requirements.

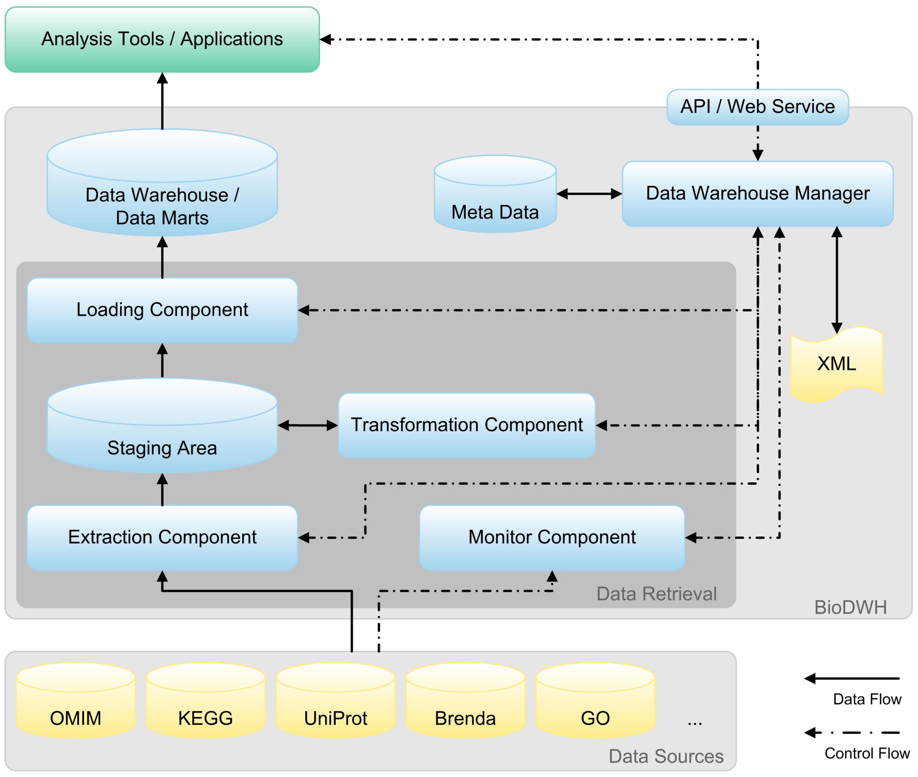


Figure 4.1: Schematic illustration of the BioDWH system architecture following the general data warehouse design.

4.2.1 Parser Plug-in Mechnism

The parser library provides a large number of ready-to-use-parsers for biological and life science databases which are available, such as: UniProt, KEGG, OMIM, GO, Enzyme, BRENDA, HPRD, MINT, SCOP, OMIM, Reactome, iProClass, TRANSFAC and TRANSPATH. Each and every parser extends the *BioDWHParser* interface that makes parser usable for the application. The interface implements methods that contain general information about the parser and an abstract method for the parser itself. The *BioDWHParser* interface is illustrated in table 4.1. Typically, the parser is appropriate for the data extraction of the sources, the transformation of the information and the insertion into the database. Usually, the database sources are available in ASCII flat files or XML files. To achieve independence from the RDBMS a persistence layer is necessary between the parser and DBMS.

For that purpose, a well-engineered object-relational mapping framework called *Hibernate*[1] was used as a persistence layer. Hibernate performs well and is independent from manufacturers like MySQL, PostgreSQL or Oracle. Furthermore, the Hibernate framework fits perfectly into the infrastructure of the BioDWH data warehouse appli-

[1]https://www.hibernate.org/

Function	**Description**
start()	Starts the parser using database connection provided by hibernate session or java persistence manager (JPA).
abort()	Aborts the parser.
getParserName()	Returns the name of the parser.
getVersion()	Returns the version number of the parser.
GetParserID();	Returns the identifier of the parser. Parser identifier is **obligatory**.
getCreationDate()	Returns creation date of the parser.
getDefaultDownloadURL()	Returns default download location of the flat files.
getFileNames();	Returns a list of filenames that are required for the parser.
getParserDescription()	Returns a short description of the parser and useful information about the parser.
getParserAuthor()	Returns the name of the author.
getEntityPackage()	Returns the package name of the entities that belong to this parser.
getProgress()	Returns the progress of the parser in percent.

Table 4.1: Description of the *BioDWHParser* interface.

cation. Finally, the *BioDWHParser* interface and the object-relational mapping classes as well as Java Persistence Architecture (JPA) classes constitute the plug-in architecture of BioDWH. Now then, a parser consists of one or more Java classes that do the extraction, transformation and loading and a set of annotated Java classes for the object relational mapping that are organized in Java packages. All these classes are compressed JAR file (Java Archive). The archive file has to be copied into the parser repository, i.e. parser directory of BioDWH software, and the parser is applicable for the toolkit. Hence, it is easy to plug-in a new parser into the infrastructure.

4.2.2 Object-relational Mapping

Object-relational mapping (ORM) is a powerful, transparent and automated persistence method for Java application to represent objects in a relational database system. In particular a mapping between objects and meta data of the database is described. In principle, ORM works with reversible transformation of data from one representation into another. An ORM solution consists of four parts:

1. Application programming interface (API) that executes simple CRUD (create, retrieve, update, delete) operations using objects of persistent classes.
2. Programming language or API to formulate queries that depend on Java entity classes or properties of classes.
3. A facility for mapping metadata.

4. Techniques of an ORM implementation to handle interactions of dirty checking, lazy association fetching and other optimization functions of transactional objects.

In addition, four different options exist to implement an ORM [BK07]:

1. *Straight relational* means the whole application, including the user interface, is designed based on the relational model and SQL-based operations. It is possible to turn SQL in any direction, but there are major problems with portability and maintainability of direct SQL, especially in the long run. However, this approach could be a good solution for huge applications. In most cases, these applications use stored procedures and shift tasks from the business layer to the database.

2. Entities are represented as classes in *the light object mapping* approach. They are manually mapped to relational tables. Manual coded SQL is hidden from the application logic. This approach is widely used and works well for applications with a small number of entities.

3. *Medium object mapping* applications are based on an object model. SQL statements will be compiled at runtime using framework code or will be generated by a code generator tool. Objects are linked via persistence layer and queries, which can be specified by an object-oriented language. Objects are cached by the persistence layer. The medium object mapping is usually used by mid-size applications that deal with complex transactions. Compatibility between different database brands is supported.

4. *Full object mapping* supports elaborate object modeling for composition, inheritance, polymorphism and persistence by reach ability. Transparent persistence is implemented by a persistence layer. Persistent classes have to implement a particular interface and can not inherent special classes. Lazy, eager and pre-fetching as well as caching strategies are implemented transparent in the application layer. Several open source and commercial Java ORM frameworks reach this level of quality. This level meets the definition of ORM using with BioDWH.

4.2.3 Monitoring

A monitor is appropriate for detection of data manipulation or changes within a data source. To keep the staging area and of course the data warehouse up-to-date, updates of the data sources have to be incrementally propagated. The BioDWH system provides the following monitoring strategies.

- ⋆ *Timestamp-based*: Each and every dataset or source file has a timestamp. On the basis of the timestamp the system is able to decide which source has been changed since the last extraction.

- *Filesize-based*: The monitor is able to determine the file size of all source files. On the basis of the filesize the system is able to decide which source has been changed since the last extraction.

The monitor component is able to download the source file via Hypertext Transfer Protocol (HTTP) or File Transfer Protocol (FTP). Furthermore, the component compares the downloaded source file against the extracted or current source files on the basis of the above mentioned strategies. If a downloaded source is newer than the extracted source, the monitor informs the Data Warehouse Manager. Then the Data Warehouse Manager starts a new ETL process.

4.2.4 Data Warehouse Management

As central component of data warehouse system, the *Data Warehouse Manager* is responsible for initiation, control and observation of individual processes starting from extraction of the data sources to the integration of the data into the data warehouse. Regarding to figure 4.1 the Data Warehouse Manager controls all components of the DWH. The components of the DWH are responsible for internal and external data sources as well as for querying and representation of the data.

- *Monitor Component*: Detect and report changes within different data sources, relevant for the DWH, to the Data Warehouse Manager.
- *Extraction Component*: Select and transport data from the sources into the Staging Area. The extraction of data is done by the different parser.
- *Transformation Component*: Standardize, integrate, consolidate, aggregate and complete extracted data of the Staging Area. The transformation is also done by the parser using the object relational mapping.
- *Loading Component*: After finishing ETL process, loading transformed data from Staging Area into the data warehouse or data mart. A persistent loading into the staging area is supported by the Hibernate framework. Every parser uses functions from the *BioDWHParser* interface to store the data into the underlying RDBS.

A major task of the Data Warehouse Manager in this context is to initiate the data retrieval process. The start of the data retrieval process can be activated in two different ways.

- In *regular time intervals*, for instance every week or once a month. At this time the extraction of the data from the sources and the loading into the staging area is activated by the Data Warehouse Manager.
- The process starts on *explicit* demand of a user or an DWH administrator.

After the Data Warehouse Manager has activated the loading process, the manager takes care to amend and ensures the correct execution of the different tasks and steps of the data retrieval process such as transformation, integration etc.. The data retrieval process is executed sequentially, that means the Data Warehouse Manager waits until a step is finished and then a new step will be started. It is also possible to run an integration process in parallel. That means, for instance, two different parser can extract data at the same time and load them into the staging area. Errors, which can occur in the data retrieval process, are documented by a logging mechanism. The logging mechanism logs errors into different log files or into meta data schema of the data warehouse. Moreover, the Data Warehouse Manager is able to start a simple recovery process to keep the data warehouse in a consistent state.

The recovery process is realized in a simple way. At the beginning of an integration process all tables in the staging area will be renamed for recovery. New tables will be created for the integration process. If the integration process fails, current tables will be deleted and the renamed tables will be recovered. On the other hand, if integration is successful, backup tables will be deleted.

In summary, to control the data retrieval process the Data Warehouse Manager uses information from that are stored in the repository (i.e. logfiles) or *Meta Data* database. At the same time the Data Warehouse Manager is the interface to all components and the repository of the DWH. Furthermore, the manager controls the access of every component and its parameters and values, those are required at runtime.

4.2.5 Implementation

The BioDWH system is realized as a Java-based open source application that is supported on different platforms with an installed *Java Runtime Environment* (JRE). Today, Java is wide-spread and usually installed on most of the computers. Additionally, Java is available on most platforms such as Windows, Linux, Mac and Solaris. Thus, Java applications have a high degree of platform independence. Moreover, Java applications offer flexible software solutions that can be provided to a large audience. In this way the software solutions can become widely used.

In chapter 4.1 the requirements of the data integration infrastructure for biological and life science data were presented. BioDWH provides an easy-to-use Java application for parsing and loading the source data into a data warehouse. The graphical user interface of BioDWH is basically realized with *Swing*, whereas Swing is a widget toolkit for Java. It is part of Sun Microsystems' Java Foundation Classes (JFC). JFC is an API for providing a graphical user interface for Java programs. Moreover *SwingX*[2] was used to have more interactive and easy-to-use GUI. SwingX contains a collection of new, powerful and useful Swing components. All components of the data warehouse are implemented in Java.

[2]http://swinglabs.org/

4.3 Structure and Function of the Application

In this section the functionality of the different components of the DWH infrastructure are presented. Thereby the function as well as the structure of BioDWH will be introduced in detail on the basis of figures.

A user can easily start the BioDWH Java application by using Linux/Windows startup script. After starting the main window will appear. Now it is possible to start the new data warehouse project, to load an existing project or import existing projects.

4.3.1 Project Configuration

Creating a new data warehouse is possible by using the *Project Wizard* of the graphical user interface. The Project Wizard consists of four configuration steps to create a DWH project. Figure 4.2 illustrates the data warehouse configuration as UML activity diagram. First, the DWH project can be described by *Name, Project description* and *Email.* Thus, the project can be assigned to an integration project as well as to a user or administrator.

In the next step the database settings have to be configured. First, the database system with the specific dialect can be specified via the top-down menu. Table 4.2 shows an overview of the supported database systems that are equal to the databases supported by the Hibernate framework. Furthermore the sever settings with a *Hostname* and *Port* and the *Database* can be configured. An experienced user could use the *Connection URL* that is similar to a JDBC connection URL. At least *Username* and *Password* has to be complete. Afterwards the database connection settings will be tested as illustrated in figure 4.2. For that purpose a *HibernateConnector* interface class is implemented to establish a database connection via Hibernate persistence layer.

Another important step is the parser configuration. A list of implemented parser is available, that can be easily plugged-in as described in section 4.2.1, via the drop-down menu. Information about the version, author and of course a description of the parser is given. The directory of the flat files for a parser can be chosen by file browser or by typing in the directory directly into the text field. Moreover, it is possible to backup the database tables for recovery related to this parser by using *Rename old tables* checkbox. The recovery process is described in section 4.2.4. Furthermore, it is possible to edit and remove parser from the current configuration.

Finally, for each parser it is possible to configure a monitor that watches for changes of the data sources if necessary and if available via the internet. A monitor for a specific parser is able to update the data sources in pre-defined cycles such as *Daily, Weekly* or *Monthly*. Additionally, a day in a month or week and a time can be specified via a calendar for starting the update process. Flat files can be downloaded, extracted and backed-up automatically by enabling designated checkboxes. For downloading of flat files an URL can be specified by the user. The monitor component supports the FTP and HTTP protocol. Moreover a

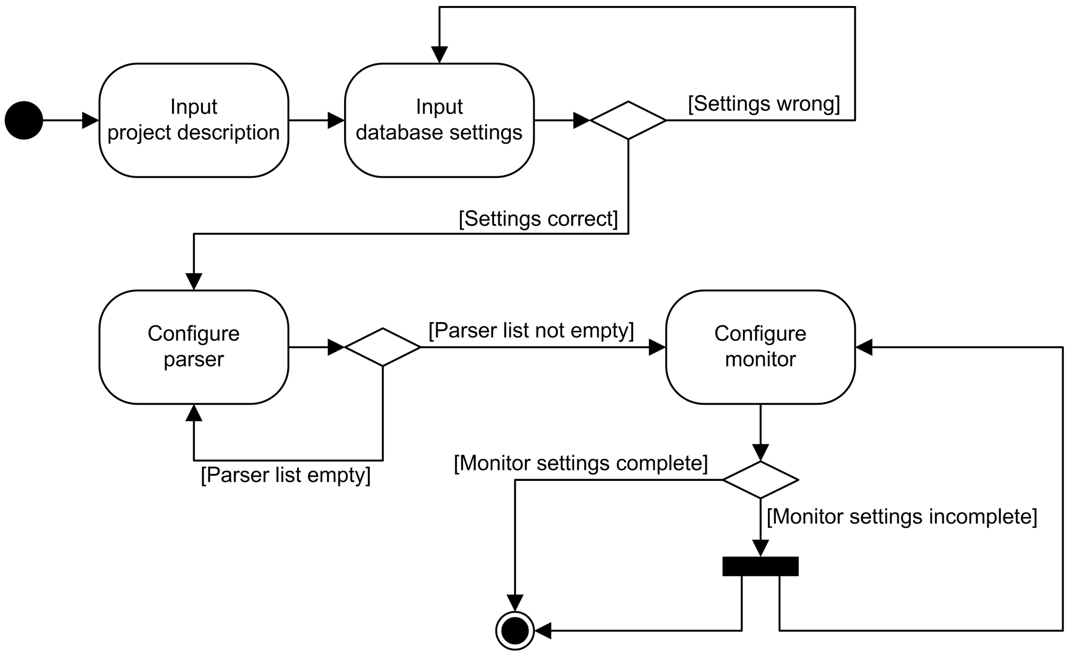
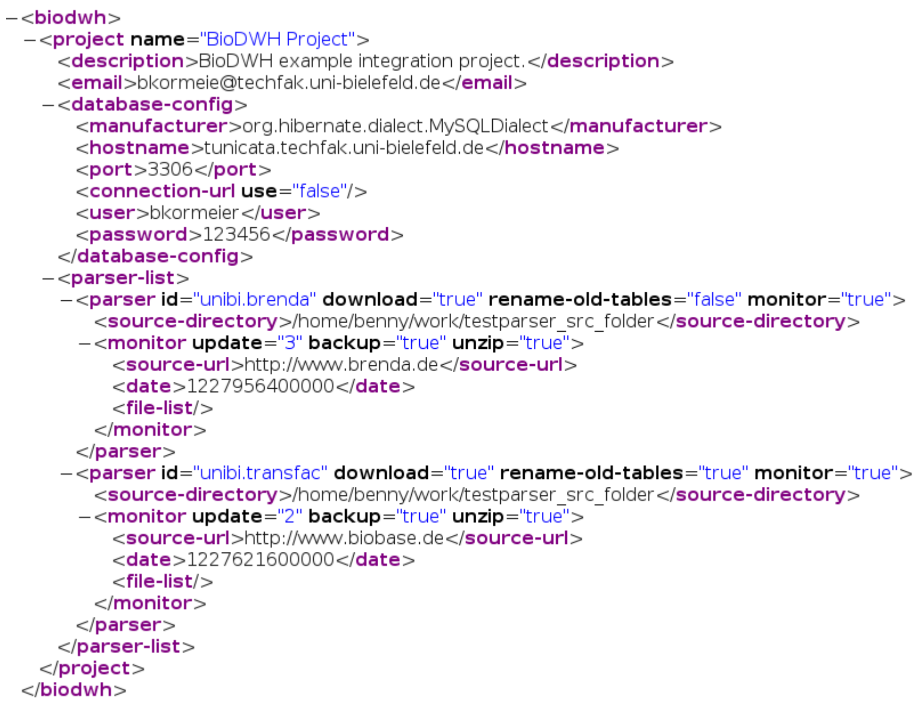


Figure 4.2: UML activity diagram of the BioDWH configuration wizard.

proxy can be configured via dialog easily, if necessary. The different update strategies for monitoring are described in section 4.2.3. At least a list of files or a complete directory can be defined for download.

For experienced users and users that are familiar with the BioDWH software, it is also possible to configure a project or DWH configuration via XML. An example XML configuration is shown in figure 4.3 and will be explained in the following. A project has a *name* attribute which is the public name of the project. Furthermore, a project contains a general *description* of the project, an *email* address of the owner or administrator, the *database-config*uration and a *parser-list*. In particular, a database configuration includes a *manufacturer* i.e. the SQL dialect of the database management system. In addition this configuration includes *hostname* or IP of the server where the database is located and as well on which *port* the DBMS is listening. Optional a *connection-url* can be used to establish the connection to the database server. Therefore, *user*name and *password* are obligatory. The different parser for the integration are described in the *parser-list*. Each parser has a unique *id* that is defined by the parser as described in table 4.1. Furthermore, the *download* of the source files and data recovery (*rename-old-tables*), as described in the paragraph above, can be enabled or disabled. The *monitor* configuration includes the *update* cycle, the *date* as long number, the URL of the source files and the *files* to download. If the *file-list* is empty, the whole directory will be downloaded from the file server of the given URL. Otherwise only the listed files will be downloads. Moreover, it is also possible to manage several projects in one configuration file.

In summary, an easy-to-use Project Wizard is implemented that supports the user or ad-



```
– <biodwh>
  – <project name="BioDWH Project">
      <description>BioDWH example integration project.</description>
      <email>bkormeie@techfak.uni-bielefeld.de</email>
    – <database-config>
        <manufacturer>org.hibernate.dialect.MySQLDialect</manufacturer>
        <hostname>tunicata.techfak.uni-bielefeld.de</hostname>
        <port>3306</port>
        <connection-url use="false"/>
        <user>bkormeier</user>
        <password>123456</password>
      </database-config>
    – <parser-list>
      – <parser id="unibi.brenda" download="true" rename-old-tables="false" monitor="true">
          <source-directory>/home/benny/work/testparser_src_folder</source-directory>
        – <monitor update="3" backup="true" unzip="true">
            <source-url>http://www.brenda.de</source-url>
            <date>1227956400000</date>
            <file-list/>
          </monitor>
        </parser>
      – <parser id="unibi.transfac" download="true" rename-old-tables="true" monitor="true">
          <source-directory>/home/benny/work/testparser_src_folder</source-directory>
        – <monitor update="2" backup="true" unzip="true">
            <source-url>http://www.biobase.de</source-url>
            <date>1227621600000</date>
            <file-list/>
          </monitor>
        </parser>
      </parser-list>
    </project>
  </biodwh>
```

Figure 4.3: BioDWH XML example configuration file.

ministrator to configure a DWH integration process in four steps. No additional knowledge in database systems or computer science is necessary. The whole configuration starting from database connection settings, via parser configuration to monitor configuration is supported by the graphical user interface. Furthermore, it is possible for an experienced user to create and edit projects via XML configuration files.

4.3.2 Project Management

The BioDWH main window contains all necessary information to manage several integration projects. It is possible to *Open* multiple DWH configurations via the *File* menu. In addition it is also possible to import single or multiple existing project configurations into the actual BioDWH system. On the other hand it is possible to *Save* the current configuration of BioDWH or export single projects. Figure 4.4 shows the main window of the BioDWH software infrastructure with a project tree on the left. Every node in the tree represents a project that is managed by the software. Each project contains one or more parser that are appropriate for an integration process of a database. The leaves of the different parser symbolize one step during the integration process. Additionally, the status of the integration process can be followed by the symbols of the leaves.

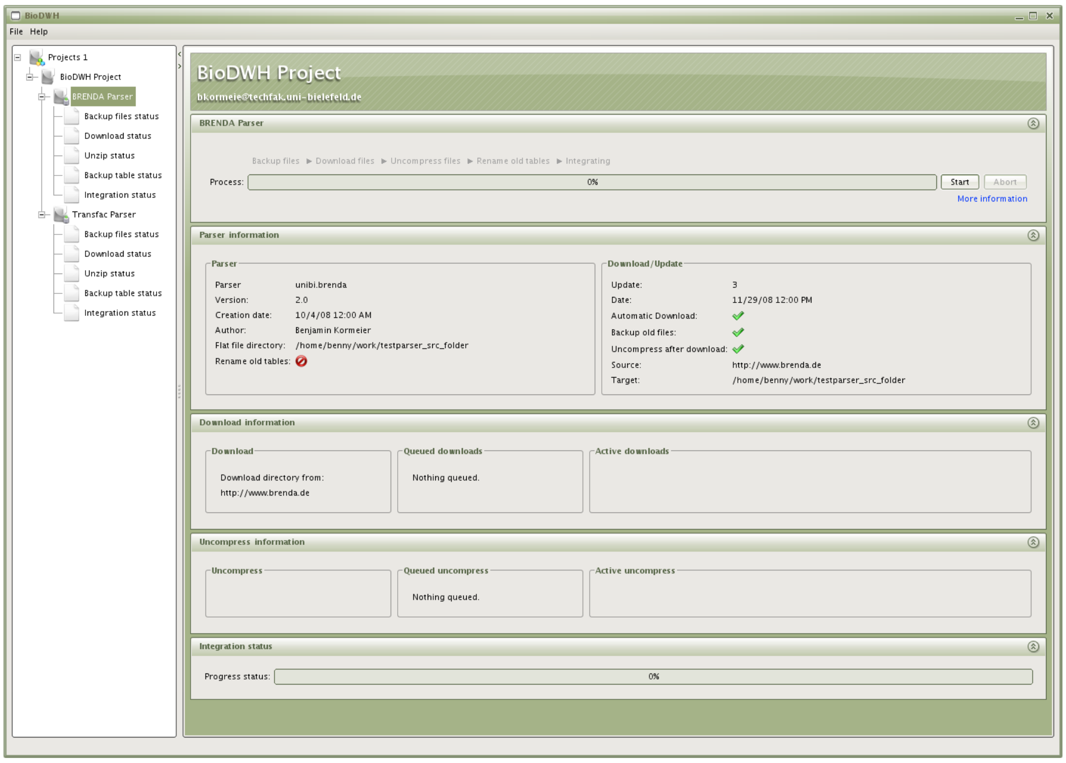
An overview of the important information and a detailed status of the integration process are given on the right panel of the main window. Starting from the root note *Projects*

Manufacturer	**Dialect**	**URL**
IBM	DB2, DB2 AS 400, DB2 OS390	http://www.ibm.com/db2/
Apache	Derby	http://db.apache.org/derby/
PostgreSQL Global Development Group	PostgreSQL	http://www.postgresql.org/
SUN Microsystems MySQL	InnoDB, MyISAM	http://www.mysql.de/
Oracle	Oracle 9i, Oracle 10g	http://www.oracle.com/
Sybase	Sybase Anywhere	http://www.sybase.de/
Microsoft	Microsoft SQL	http://www.microsoft.com/germany/sql/
SAP	SAP	http://www.sap.com
IBM	Informix	http://www.ibm.com/software/data/informix/
hsql Development Group	HSQL	http://hsqldb.org/
Ingres VectorWise project	Ingres	http://community.ingres.com/
Progress Software	Progress	http://web.progress.com/
Diehl and Associates	Mckoi	http://www.mckoi.com/
Embarcadero Technologies	Interbase	http://www.embarcadero.com/
IBM	Pointbase	http://www.ibm.com/software/data/integration/dm/
FrontBase	Frontbase	http://www.frontbase.com/
Firebird Project	Firebird	http://www.firebirdsql.org/

Table 4.2: Supported databases of BioDWH infrastructure.

a list of the different projects will appear that are managed by the DWH infrastructure. Detailed information about the database connection and a list of involved parsers will be displayed. The project node shows the general description of the project as well as the overall status of the project that is illustrated by a progress bar. More detailed information for each parser is displayed by selecting a particular parser node. Each integration step has its own task pane as illustrated in figure 4.4. First, the general status of the parser is given, followed by the *Parser information* including the *Download/Update* settings that have been described in section 4.3.1. Furthermore, *Download Information* of queued and active downloads are presented in a task pane, if this step is enabled. The task pane for file uncompress is similar. Finally, the status of the data integration process, which means extraction, transformation and loading, is illustrated by a progress bar.

The BioDWH infrastructure is running with multiple threads which means it is possible to run several download processes, uncompress processes or integration processes in parallel. Integration processes are volatile tasks, hence it is strongly dependent on the hardware of the server or computer. Therefore, at the moment the BioDWH software is restricted to run two integration processes in parallel. An overview of the running processes could be found in the *Process queue* window that is accessible via the *Properties* menu. The same functionality is implemented for the file downloads that are listed in the *Download queue*. From the point of implementation it is possible to run more parallel processes. This feature will be available via configuration settings in the near future.



Process: 0%

Parser

Parser unibi.brenda
Version: 2.0
Creation date: 10/4/08 12:00 AM
Author: Benjamin Kormeier
Flat file directory: /home/benny/work/testparser_src_folder
Rename old tables: 🚫

Download/Update

Update: 3
Date: 11/29/08 12:00 PM
Automatic Download: ✓
Backup old files: ✓
Uncompress after download: ✓
Source: http://www.brenda.de
Target: /home/benny/work/testparser_src_folder

Download

Download directory from:
http://www.brenda.de

Queued downloads

Nothing queued.

Active downloads

Uncompress

Queued uncompress

Nothing queued.

Active uncompress

Progress status: 0%

Figure 4.4: Screenshot of the BioDWH graphical user interface.

4.4 Summary

In this chapter the concept for a general data warehouse infrastructure for life science data integration was described. Starting in section 4.1 requirements to a data warehouse infrastructure were presented. Based on the requirements the concept and system architecture of the data warehouse infrastructure application was introduced in section 4.2. Section 4.2.5 presents the implementation of the BioDWH data warehouse infrastructure approach regarding the general DWH concept in the previous section. In detail the plug-in mechanism for parser, object-relational mapping including persistence, monitoring strategies and the general management of the system were presented. In the next section 4.3 the structure and the function of DWH infrastructure approach were illustrated on the basis of the graphical user interface.

In summary, a flexible DWH infrastructure for bioinformatics, which is independent from the underlying RDBMS, was introduced. The independence between the application and the RDBMS is enabled by an ORM approach. The system is developed with Java, so that the software is platform independent. Furthermore, the data warehouse approach provides an easy-to-use graphical user interface for administration and configuration. Additionally, configuration of the infrastructure and its tools is also possible via XML, because it is human readable, well-formatted, easy to access and standardized. A logging mechanism watches the integration process and starts a simple recovery process to guarantee a consistent state of the data warehouse.

5 Life Science Data Warehouses

This chapter gives an insight into the concept of building web-based data warehouse systems on the basis of the requirements introduced in section 5.1. This section gives an enumeration of important requirements for building a life science data warehouse using relational database management systems. In section 5.2 two web-based DWH systems, called *CardioVINEdb* [KHTH09] and *DAWIS-M.D.*, will be introduced. In detail the integration process, the integrated database schema, the system architecture and its components will be illustrated. Based on screenshots of the web-application in section 5.2.1.3 and section 5.2.2.3 the structure and function of each system will be presented. Finally, a summary of the chapter is given in section 5.3.

5.1 Requirements for Building Life Science Data Warehouses

The importance of building biological and life science data warehouses has already been discussed in chapter 3. Furthermore, advantages and disadvantages of DWH technique has been discussed in chapter 2. Basically in bioinformatics the DWH architecture for integration of molecular data is used.

Based on the experience in data integration and on the experience of existing data warehouses (as described section 3.1) an enumeration of requirements for building life science data warehouses is given. In addition, requirements of the *Cardioworkbench*[1] project, that will be introduced in chapter 7, are included to achieve the requests of the scientists.

Requirement 5.1 *Molecular and biological data will be integrated to construct a general data warehouse. This warehouse approach will not be focused on a specific problem, but will be general in concept.*

The data warehouse system should cover a wide spectrum of molecular and biological data. In comparison systems like Columba, CoryneRegNet or SYSTOMONAS, that were introduced in section 3.1, do not contain general biological knowledge but rather specific knowledge in a special research area.

[1]http://www.cardioworkbench.eu/

Requirement 5.2 *The integration of multiple databases in different research fields will use the BioDWH data integration infrastructure.*

BioDWH provides a number of ready-to-use parsers to extract data from public life science data sources and stores the content in a data warehouse. Hence, BioDWH provides information about several domains such proteins, enzymes, genes, chemical compounds, diseases etc. Detailed information can be found in section 4.2.

Requirement 5.3 *Regarding the comprehensive knowledge base, provided by the data integration system, it is necessary to divide information into adequate and specific domains.*

A web-based DWH system should provide different biological domains that cover a wide range of biological knowledge. Usually, data sources such as KEGG or Transpath provide information multiple of domains. Hence, information of a data source can be assigned to different domains.

Requirement 5.4 *Each domain should provide its own query form to support queries in different granularities.*

The user should be able to formulate search criteria that are supported by the specific domain. Furthermore, examples should be given for each and every form to point out the structure of the query. Moreover, while filling a query form a list of related or similar entries should support a researcher and facilitate the selection of an entry.

Requirement 5.5 *The result page of an entry has to be well structured and clearly-arranged.*

For a better overview it is necessary to hide detailed information, that passes a certain length, and minor information. This information is displayed by request of the user. In addition, links to other domains should be available to provide related information.

Requirement 5.6 *The data warehouse system should be transparent.*

A user does not need knowledge about the structure or the database model of the data warehouse to use the system. The source of the information should be clear, because scientist often trust sources in a different way. The trust of a data source is important for research.

5.2 Web-based Data Warehouse Systems

In section 5.1 requirements of biological and life science data warehouses were discussed. Based on this requirements in section 5.1 and based on data warehouse infrastructure in the previous chapter, this section will introduce two DWH approaches. The system architecture and its components, the implementation will be explained in detail, and the structure and functionality of the web-applications will be presented.

Both web-based data warehouse approaches are based on the *BioDWH* integration toolkit described in chapter 4. CardioVINEdb as well as DAWIS-M.D. are realized as web application using *JavaServer Pages* (JSP), *Java Servlet* and *Java classes*. The technologies allow software a developer to add dynamic content to a web server using the Java platform. The generated content is commonly HTML. Hence, JSP and Java Servlets make the system dynamic and very flexible such as *PHP*, *CGI* or *ASP.NET*. As infrastructure an *Apache Tomcat* servlet container was used. Apache Tomcat is an open source software implementation of the Java Servlet and JSP technologies and provides a "pure Java" HTTP web server environment for Java code to run. Thus, the DWH information systems are platform independent, because the websites are displayable in every common web browser. Web applications offer flexible software solutions that can be provided to a large audience, because of this the dynamic browser applications become widely used.

5.2.1 A Data Warehouse Approach for Integration of Life Science Data in Cardiovascular Diseases

One of the major challenges in bioinfomatics is to integrate and manage data from different sources as well as experimental microarray data and present them in a user-friendly format. In cooperation with our partners from medicine and biology within the *Cardioworkbench* project we designed a platform-independent data warehouse system that integrates multiple heterogeneous data sources into a local database enriched with protein microarrays from human smooth muscle cells that related to cardiovascular disease. Based on the concept of *VINEdb* information system [HTBH07] the *CardioVINEdb* DWH is extended with more data sources, better data warehouse infrastructure including monitoring and microarray data. In comparison to the VINEdb system, the visualization components and web pages were upgraded for better navigation and exploration. The monitor component from BioDWH was used for cyclic updates, to ensure maximum up-to-date data in CardioVINEdb. Furthermore, the common web-based user interface provides a visualization component that allows interactive exploration of the integrated data.

5.2.1.1 System Architecture

The CardioVINEdb system consists of a 4-layer architecture that is illustrated in figure 5.1. The source layer contains the multiple data sources BRENDA, EMBL, GO, IntAct, KEGG, MINT, OMIM, PubChem, SCOP, Transfac, Transpath and UniProt. In addition to the publicly available databases, experimental microarray data of human smooth muscle cells that are associated to cardiovascular disease are integrated into the DWH. The microarray data was provided by the Italian partners of the Cardioworbench project. Most of these databases provide parseable flat files that can be processed by the BioDWH data warehouse infrastructure. A monitor component that is part of the integration layer controls the different data sources. The monitor is described in detail in section 4.2.3. It recognizes changes in the original sources and starts download if the file has changed. In a defined cycle the parser will be activated to start the ETL (Extraction-Transform-Load) process. ETL means that the data is extracted from the source data, transformed into the data warehouse schema and loaded into the DWH as described in section 4.2.4.

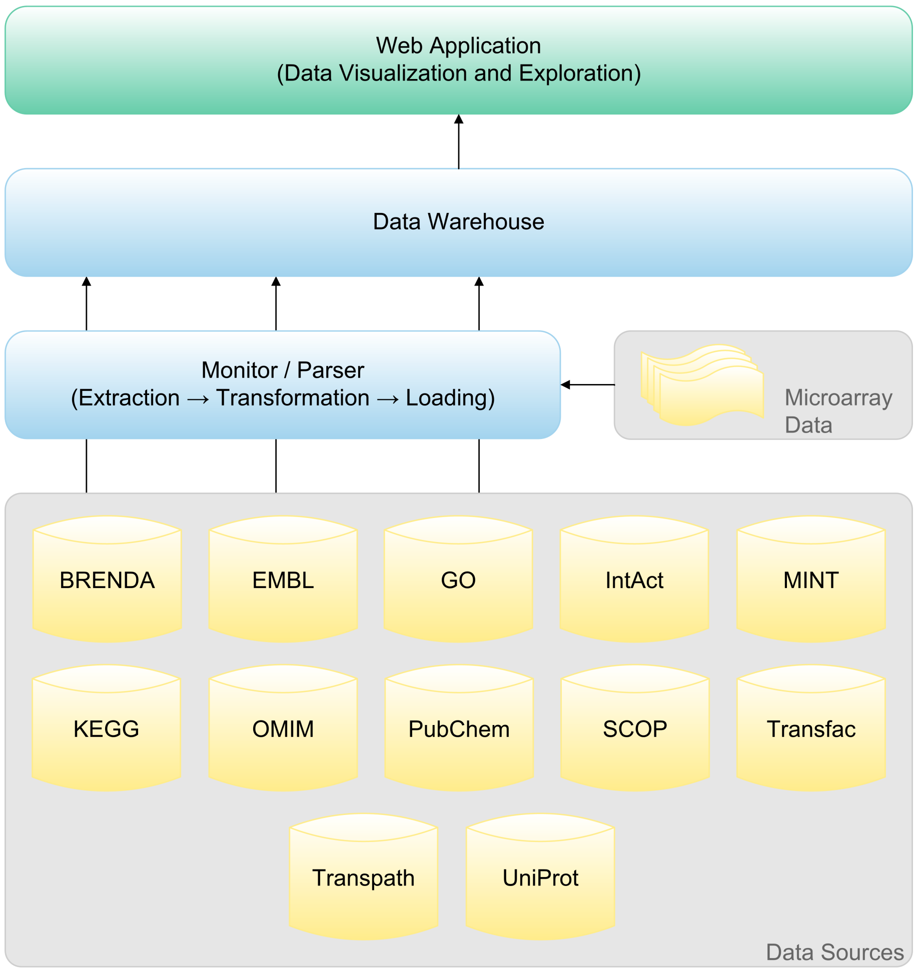


Figure 5.1: Schematic representation of the CardioVINEdb 4-layer system architecture from the original heterogeneous data sources to the web application layer.

5.2.1.2 Implementation

The web-based graphical user interface of CardioVINEdb is implemented with JSP and runs on an Apache Tomcat web server. Each data entry has detailed information and a further link to the original data source. A general search engine allows the user to find information of interest spanning multiple domains, such as proteins, enzymes, genes, compounds etc. Additionally, each domain has its own specific search engine to find required information for research. The images or the graphical representations of the relationships between the entities of the data warehouse are created by JUNG at runtime. This approach allows dynamic and interactive exploration of the data. JUNG[2] is an open source Java-based library that provides classes to describe graphs, nodes and edges with additional layout preferences. Accordingly, a graph is generated by JUNG according to the entities selected by the user. The system produces a *Portable Network Graphic* (PNG) image file with the graphical visualization of biological objects in different domains and their linkage. Finally, this image is embedded in the HTML pages and displayed by the web browser. Moreover, the dynamic component using a Java Applet works in the same manner, but in this case the graph is directly generated and displayed within the applet. For more interactive navigation and exploration the applet has a zoom function, different graph layouts and a picking function to move and select nodes within the graph. The Java applet is embedded in the HTML pages and can be displayed by the web browser if Java Runtime Environment is installed on the computer. As relational backend a MySQL database management system is used for integrated data. By using *Java Database Connectivity* (JDBC) it is possible to access the underlying RDBMS. The core of the DWH, i.e. BioDWH, as described in chapter 4 is completely implemented in Java, which ensures platform independence of the operating system. Hence, it could be used separately from the web interface.

5.2.1.3 Structure and Function of the Web-Application

In this section the functionality and the components of web application will be introduced. The function and structure of CardioVINEdb will be illustrated based on different figures.

A system administrator can easily configure the web-application via Java class *Configuration* to setup the system on different web server. Based on the configuration class the administrator is able to install the software fast and user-friendly way. Figure 5.2 illustrates a web site with a search engine of the web application. The global navigation within the web application is possible via the head of the web page. Thereby a user-friendly and dynamic web-interface is provided. The head contains a global search engine that is spanning all domains of the DWH. Furthermore, all domains of the CardioVINEdb system such as: *Transpath, Transfac, Pathway, Disease, GO, Protein, Enzyme, Reaction, Reaction Pair, Interaction, Compund, Glycan, Drug* and *Experimental Data* are listed in the head and thereby directly, intuitively useable.

Each domain provides a special search form to formulate simple queries to the database. The structure of the different search forms is similar. Therefore an example of a search form for *Protein* domain is shown in figure 5.2. For every search criterion an extra search field is provided by the web site. Furthermore, for every search item an adequate example

[2]http://jung.sourceforge.net/

is given to support the user with a search. A domain can span multiple data from different databases. Therefore the search forms are slightly different. This is necessary to provide an individual search for a specific domain related to the schema of the data warehouse.

Once a query is processed a result set of related entries will be displayed as a table in new a web site. The representation of the search results is basically similar. Every row of the table is a different entry of the database query. Each column represents a category of the entry to provide additional information. The result table has a maximum of three categories. For instance, “protein search” will show the following categories: *UniProt id, UniProt accession number,* and *organism*. Now, the user is able to select an entry of the result set.

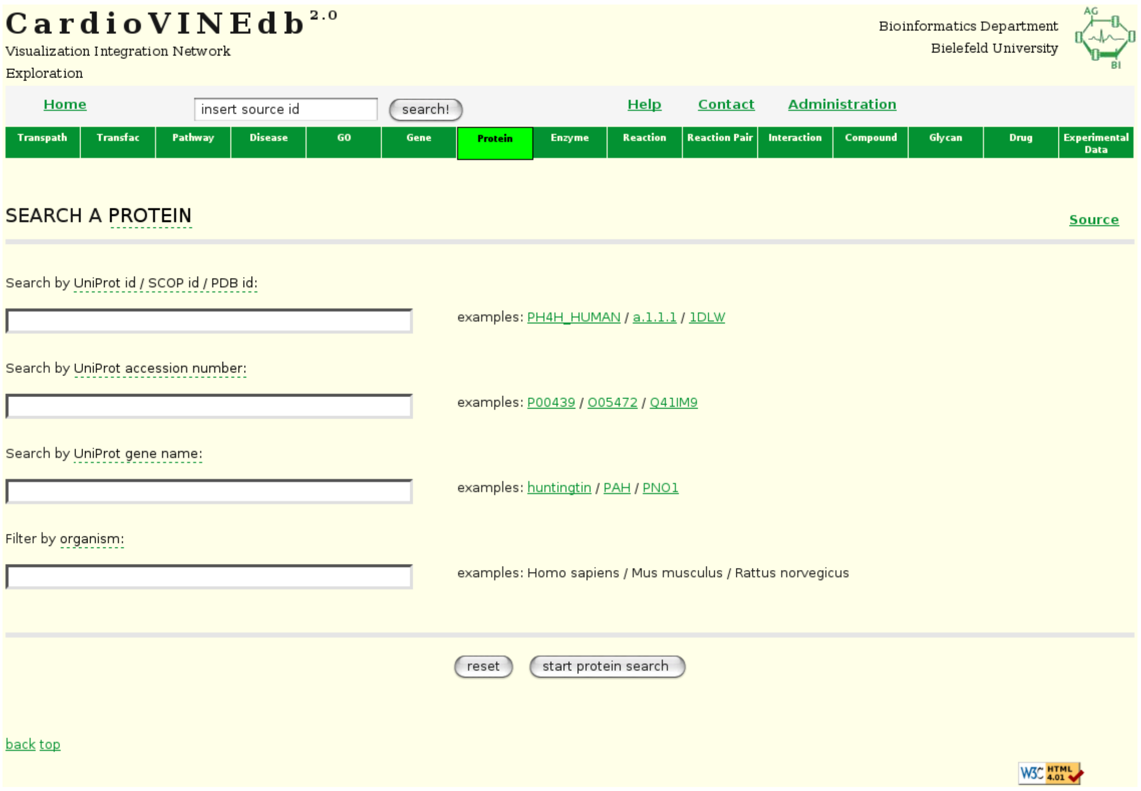
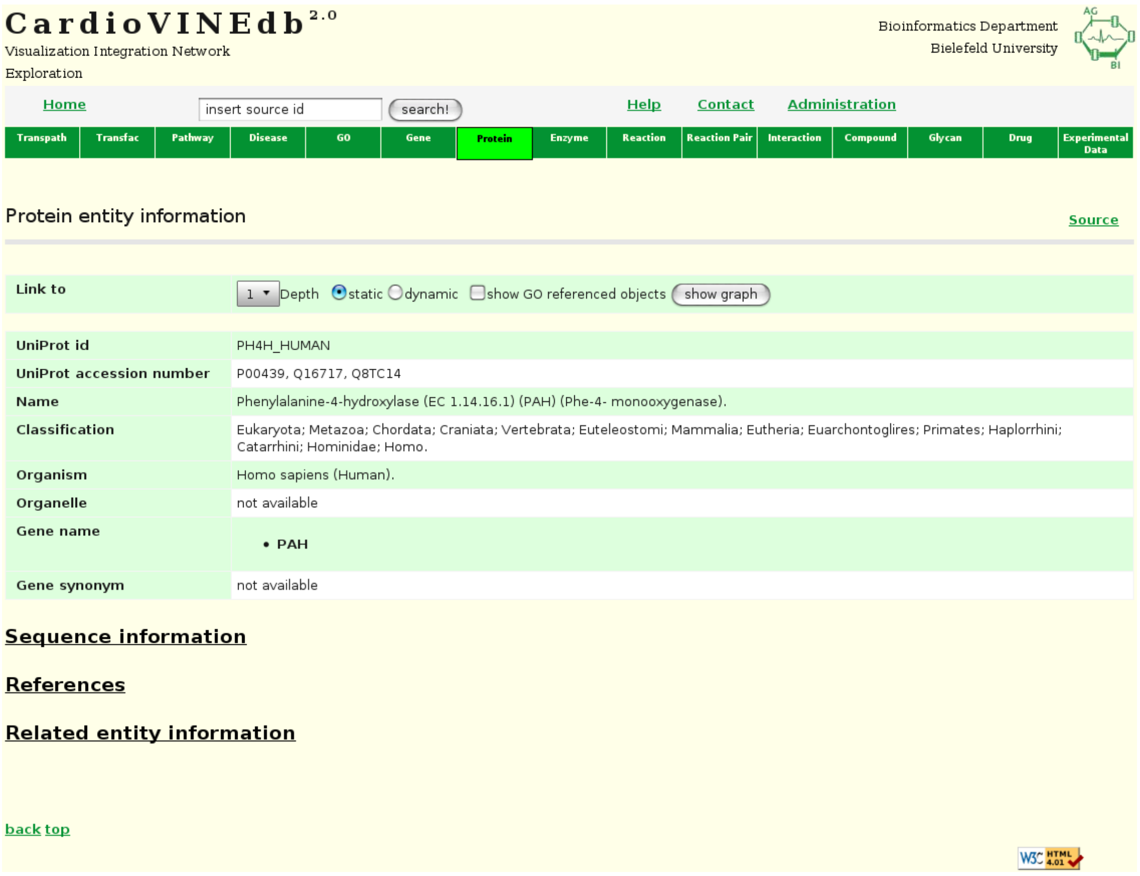

CardioVINEdb 2.0
Visualization Integration Network Exploration
Bioinformatics Department
Bielefeld University
AG BI
Home
insert source id
search!
Help
Contact
Administration
Transpath
Transfac
Pathway
Disease
GO
Gene
Protein
Enzyme
Reaction
Reaction Pair
Interaction
Compound
Glycan
Drug
Experimental Data
SEARCH A PROTEIN
Source
Search by UniProt id / SCOP id / PDB id:
examples: PH4H_HUMAN / a.1.1.1 / 1DLW
Search by UniProt accession number:
examples: P00439 / O05472 / Q41IM9
Search by UniProt gene name:
examples: huntingtin / PAH / PNO1
Filter by organism:
examples: Homo sapiens / Mus musculus / Rattus norvegicus
reset
start protein search
back top

Figure 5.2: Search form for the *Protein* domain in CardioVINEdb.

After selection of an entry, all information of this entry will be displayed on a new web page as shown in figure 5.3. Basically, the structure of the resulting page is independent and differing from the originating domain. The information is presented as a table. On the left side of the table the keyword of the information is displayed, and on the right-hand side the information itself is displayed. Every row represents a different data set of the result. For a better overview the background color differs. Moreover, a function is implemented to hide additional information to keep a well structured and clear overview of the result. Additional information will be displayed by clicking the specific keyword. In order to provide a transparent system each result page has a link to the original data source, i.e. web page of the original entry.



Protein entity information

Link to: 1 Depth [✓ static] dynamic ☐ show GO referenced objects

UniProt id	PH4H_HUMAN
UniProt accession number	P00439, Q16717, Q8TC14
Name	Phenylalanine-4-hydroxylase (EC 1.14.16.1) (PAH) (Phe-4- monooxygenase).
Classification	Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Euarchontoglires; Primates; Haplorrhini; Catarrhini; Hominidae; Homo.
Organism	Homo sapiens (Human).
Organelle	not available
Gene name	• PAH
Gene synonym	not available

Figure 5.3: Detailed result page for the protein *PH4H_HUMAN* in CardioVINEdb merging the data from different sources for that entry.

Furthermore, the system provides a "static" and "dynamic" visualization component for interactive graphical exploration of the data as introduced in section 5.2.1.2. For both visualization components a so-called "deep search" is implemented. For instance a search for the protein *PH4H_HUMAN* in *depth 1* shows all direct links to enzymes, compounds, drugs etc in a graph as illustrated in Figure 5.4. Now, if the search goes for *depth 2* all links of the enzyme, drug will show up. Hence, it is easy to generate a complex graph with links between the different domains.

5.2.1.4 Summary

CardioVINEdb provides integrated data from different popular life science databases and microarray data related to cardiovascular diseases from an EU project in a homogeneous web-based system. The system enables an intuitive search of integrated life science data, simple navigation to related information as well as visualization of biological domains and

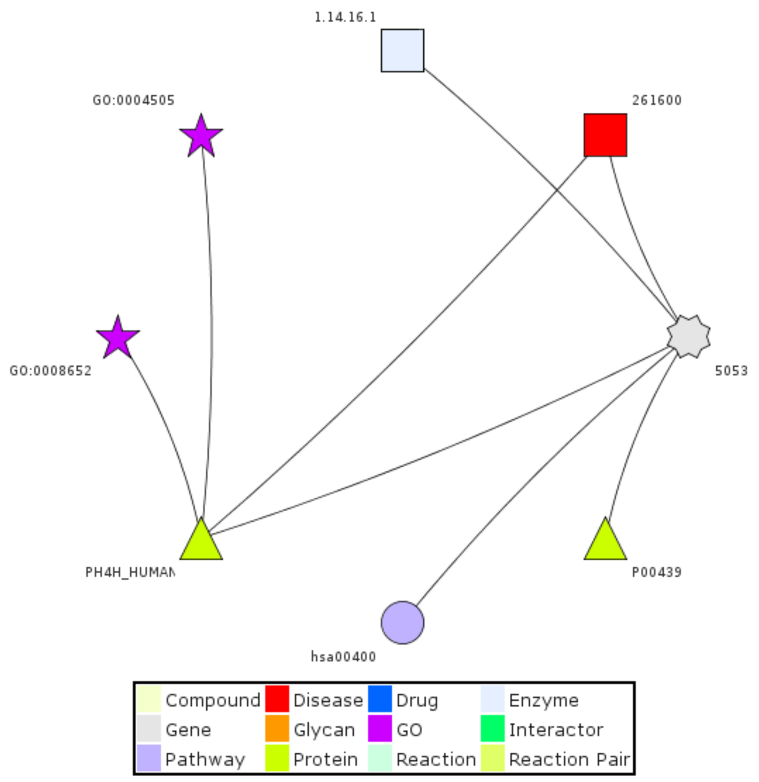


Figure 5.4: Graph representation of the *PH4H_HUMAN* protein search result with depth 2 (where star symbol represents GO annotations; rectangles are proteins; circles are pahways; red squares are genes; blue squares are enzymes).

their relationships. Equipped with a monitor component that updates the integrated data in defined update circles, it enables a way for presenting complex biological data in a very user-comprehensible manner.

5.2.2 A Data Warehouse Information System for Metabolic Data

One important goal in bioinformatics is to integrate data from disparate sources of heterogeneous biological information. Data integration allows the assembly of the targeted data reagents for bioinformatics analyses, and to discover scientific relationships between data or domains. Most public repositories of biological data focus on deriving and providing one particular type of data, such as nucleotide/amino acid sequences, protein-protein interactions, or gene expression/microarrays and so on. Integrating these heterogeneous sources of data enables researchers to discover new associations between the data, or validate existing hypotheses. Therefore, working with publicly available biological data can be challenging due to the volume and complexity of the data types. In this section an insight of system architecture, the implementation, the structure and function of *DAWIS-M.D.*, which is a acronym for **D**ata **W**arehouse **I**nformation **S**ystem for **M**etabolic **D**ata, will be given. In comparison to CardioVINEdb the DAWIS-M.D. data warehouse is a more general system that is not addressed to a specific problem such as cardiovascular diseases. Furthermore, the system is connected to the VANESA network editor (see chapter 6) to easily visualize and analyze biological networks from data of interest.

5.2.2.1 System Architecture

The DAWIS-M.D. system architecture consists of a n-layer architecture that is illustrated in figure 5.5. The source layer contains the multiple data sources: BRENDA, EMBL, GO, Enzyme, KEGG, HPRD, OMIM, SCOP, Transfac, Transpath and UniProt. Most of these databases provide parseable flat files or XML that can be processed by the BioDWH data warehouse infrastructure. A monitor component that is part of the integration layer controls the different data sources that is described in detail in section 4.2.3. It recognizes changes in the original sources and starts the download if necessary. In a defined cycle the parser will be activated to start the ETL (Extraction-Transform-Load) process. The data is extracted from the source data, transformed into the data warehouse schema and loaded into the DWH as described in section 4.2.4. Furthermore, a mapping table will be created whereby links between the domains as well as the data sources will be identified. The communication between the integration layer and the data sources is realized with a JDBC interface.

Data warehouses or data marts for specific analysis applications can easily be constructed by the database layer. To avoid dependencies between an additional layer called persistence layer is inserted. Thereby it is possible to support different DBMS such as MySQL, Oracle or PostgreSQL without changing the application. The persistence layer provides all components for the object-relational mapping whereby the application layer is independent from the database layer. The databases layer also communicates via JDBC with the persistence layer. Furthermore, the persistence layer has to communicate with the application layer. A user can interact via the application layer with the system. Therefore, the web application provides a homogenous and integrated view of the data. A web service controls the communication between the server and external tools by using SOAP. As illustrated in figure 5.5 the application layer interacts with the persistence layer.

5.2.2.2 Implementation

The web-based graphical user interface of DAWIS-M.D. is implemented with JSP that is based on the programming language Java and runs on an Apache Tomcat web server. For dynamic and interactive web sites a web server with JSP-/Servlet engine is required. The communication between client and server is realized via HTTP protocol where the client is the web browser of the user. As relational backend a MySQL database management system is used for integrated data. Other relational DBMS are also possible because of a persistence layer as described in the previous section. By using JDBC it is possible for the web application to access the underlying RDBMS. Furthermore, JDBC has to forward SQL queries to the database and return the results to the application. DAWIS-M.D. uses HQL, a SQL-like query language, to query the underlying database management system. The Hibernate framework provides this object oriented query language. By using HQL it is possible to represent connection and inheritance functionality. Each and every HQL query to the database is translated by the Hibernate framework into SQL queries. Furthermore, queries formulated in HQL are much more compact than queries in SQL.

Beside DAWIS-M.D. it is possible to access the data warehouse via a web service. The software *VANESA* that will be introduced in detail in chapter 6 is able to access the DWH via this web service. The server as well as the VANESA software use an *Apache Axis2*[3] framework. In addition, a communication between VANESA and DAWIS-M.D. is possible via the Java based Axis2 framework. In order to exchange information a specific database table is provided by the DWH. Hence, a user is able to send a single element or all elements of a DAWIS-M.D entry to the VANESA network editor. Hence, this specific database table manages the information of the user. Moreover, the VANESA software is now able to generate a network of the entry, i.e. an entry of the web application. Finally, in DAWIS-M.D the information is presented text-based and visualized as networks in VANESA. The representation of the graphs in VANESA is realized with the JUNG framework that will be explained in more detail in the next chapter. Moreover, the JUNG framework is also used to generate the protein classification in the web application DAWIS-M.D (see figure B.1). The classification of the proteins is based on the data of the SCOP database. This information is available in the *Enzyme* and *Protein* domains of the information system. The protein classification is displayed as a static image within the web page.

Finally, the DAWIS-M.D. web application provides a statistical tool. The statistics are

[3]http://ws.apache.org/axis2/

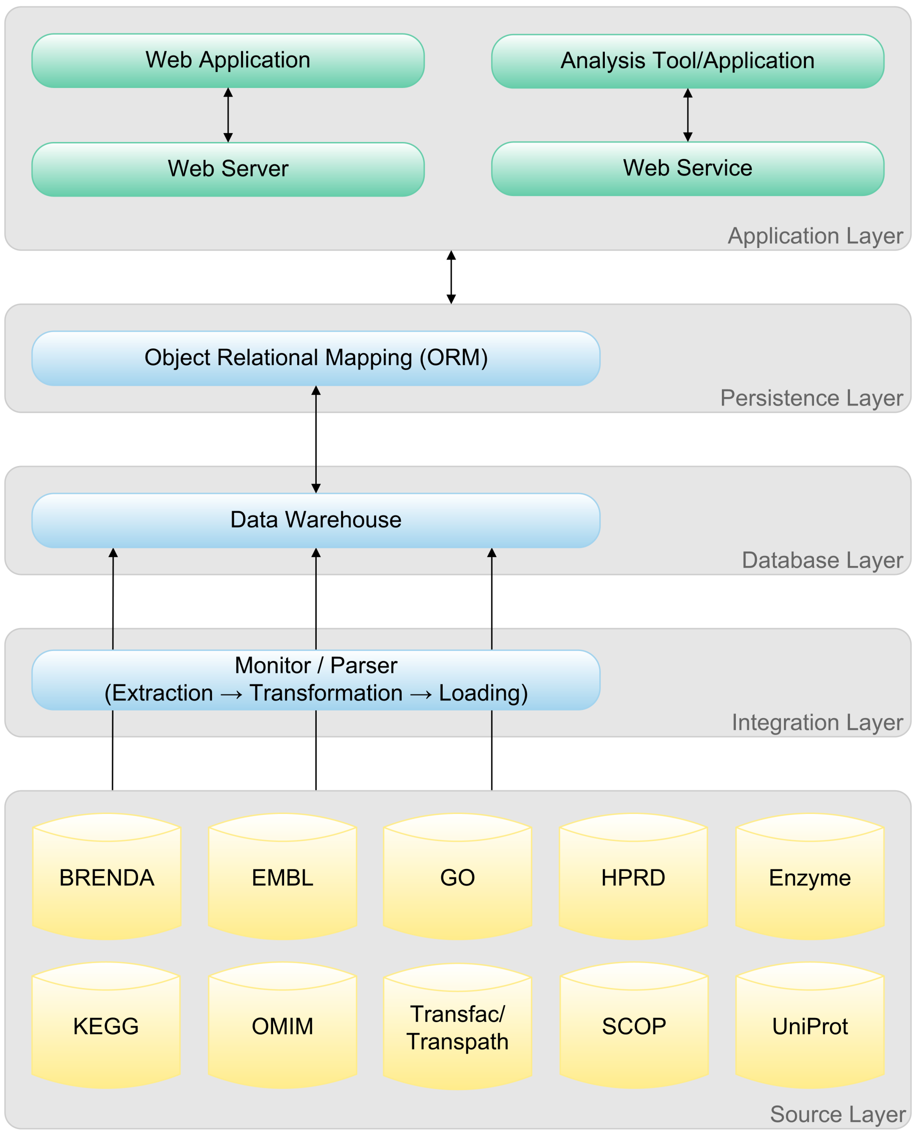


Figure 5.5: Schematic representation of the DAWIS-M.D. n-layer system architecture from the original heterogeneous data sources to the web application layer.

realized by using the *JFreeChart*[4] framework. Therefore, the data for the statistics are provided by the DWH and displayed as charts by the framework within the web page.

5.2.2.3 Structure and Function of the Web Application

In this section the functionality and the components of the web-application will be introduced. The function and structure of DAWIS-M.D. will be illustrated based on different figures.

A system administrator is able to configure the web application via XML configuration file so that DAWIS-M.D. can be run on a different web server. Therefore, an administrator is able to install and adjust the software fast and easily.

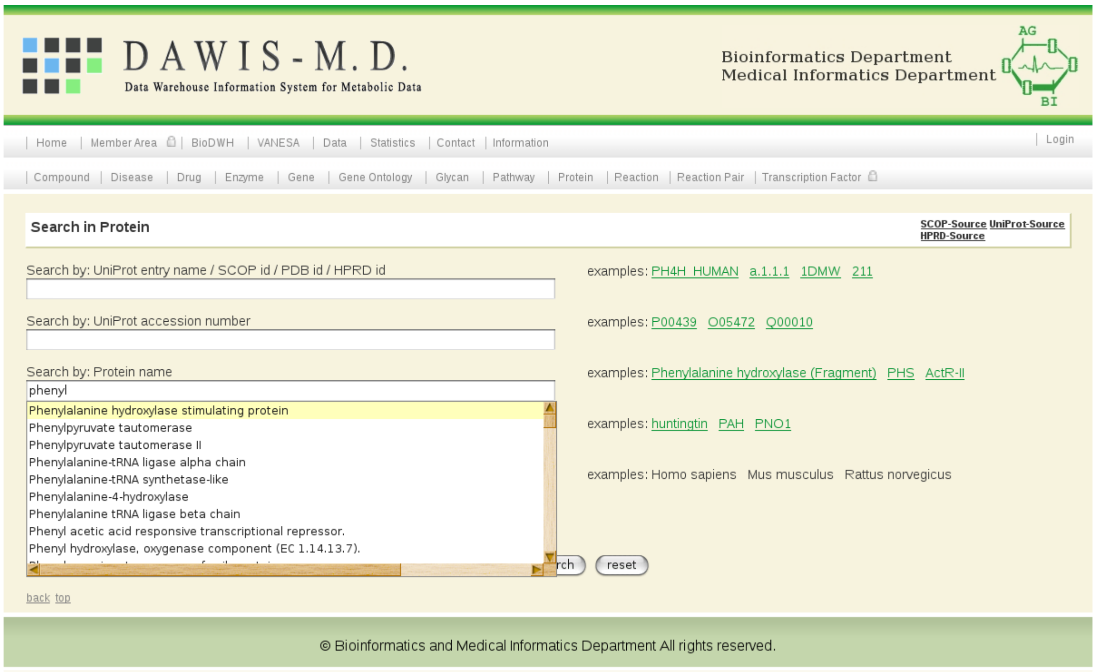
Figure 5.6 illustrates a web page of the application. Based on this screenshot it is clear that the global navigation is possible via the head of the web page. The head of the web page is displayed on each and every web page of the application whereby a user-friendly and dynamic navigation system is provided. Furthermore, some of the domains have a "lock" symbol, such as *Transcription Factor*, so that only registered user have access to this domains. This is necessary because of the terms of license for some databases that are usually commercial or not publicly available. The homepage of DAWIS-M.D. displays a diagram with connections of the different domains among each other. This diagram contains hidden links to the different domains, so that a user has an intuitive access to a specific domain.

Each domain provides a special search form to formulate simple queries to the data warehouse. Figure 5.6 illustrates one of the search forms, in this case for the *Protein* domain. In general, the structure of the search form is identical. Based on figure 5.6 it is clear that each search criterion has its own search field. Furthermore, to simplify the search for a user, for each search form an example is given. Moreover, every search form has an so-called "auto-complete" function whereby related and additional suggestions are presented in a list, as shown in figure 5.6. A domain can span multiple data from different databases. For this reason the search forms are slightly different. This is necessary to provide an individual search for a specific domain, related to the schema of the data warehouse. The web application provides search forms for the: *Compound, Disease, Drug, Transcription Factor, Enzyme, Gene, Glycan, Gene Ontology, Pathway, Protein, Reaction* and *Reaction Pair* domain.

The results of a query to the database are presented on a new web site. In general the result pages look identical and are displayed as a table whereas each row represents a new result. Each column represents a category of the entry to provide additional information. For instance, the result of a protein search contains following categories: *Identifier, Protein name,* and *Gene name*. Then the user is able to select an entry of the result set. Furthermore, the selected result has another background color to highlight the entry for a better overview. It is possible that a selected entry is protected, therefore the user will be forwarded to a

[4]http://www.jfree.org/jfreechart/

login page. If the login is successful the user will be forwarded to the specific result web page. For instance, a user has to login to get results or information from Transfac and Transpath.


D A W I S - M. D.
Data Warehouse Information System for Metabolic Data

Bioinformatics Department
Medical Informatics Department

AG
BI
| Home | Member Area | BioDWH | VANESA | Data | Statistics | Contact | Information
| Login
| Compound | Disease | Drug | Enzyme | Gene | Gene Ontology | Glycan | Pathway | Protein | Reaction | Reaction Pair | Transcription Factor
rch
reset
back top

Search in Protein

SCOP-Source UniProt-Source HPRD-Source

Search by: UniProt entry name / SCOP id / PDB id / HPRD id — examples: PH4H_HUMAN a.1.1.1 1DMW 211

Search by: UniProt accession number — examples: P00439 O05472 Q00010

Search by: Protein name — phenyl — examples: Phenylalanine hydroxylase (Fragment) PHS ActR-II

Phenylalanine hydroxylase stimulating protein
Phenylpyruvate tautomerase
Phenylpyruvate tautomerase II
Phenylalanine-tRNA ligase alpha chain
Phenylalanine-tRNA synthetase-like
Phenylalanine-4-hydroxylase
Phenylalanine tRNA ligase beta chain
Phenyl acetic acid responsive transcriptional repressor.
Phenyl hydroxylase, oxygenase component (EC 1.14.13.7).

examples: huntingtin PAH PNO1

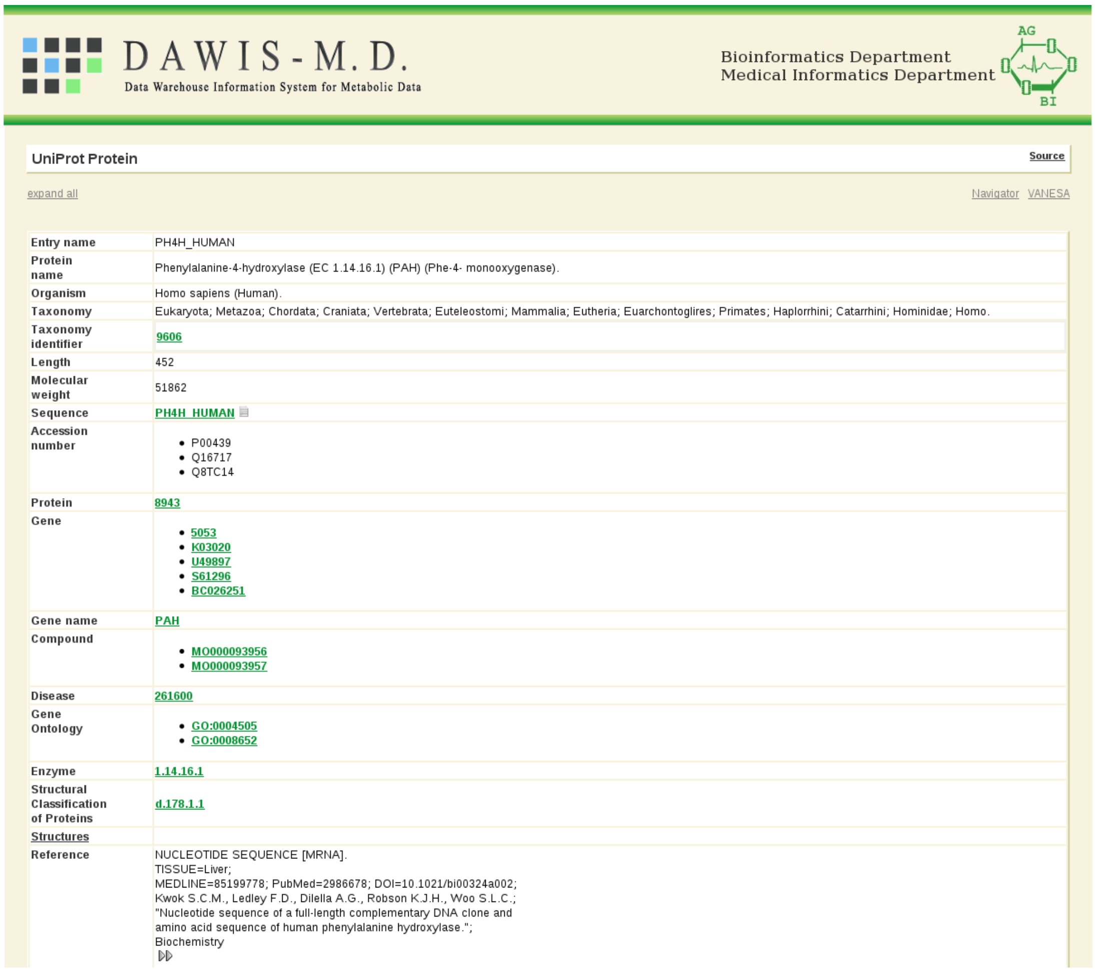
examples: Homo sapiens Mus musculus Rattus norvegicus

© Bioinformatics and Medical Informatics Department All rights reserved.

Figure 5.6: Search form for the *Protein* domain in DAWIS-M.D. with suggestion box for related protein entries.

All information of this entry will be displayed on a new web page. Basically, the structure of the resulting page is independent and differing from the originating domain. The information is presented as a table. On the left side of the table the keyword of the information is displayed, and on the right-hand side the information itself is displayed. For a better overview the selected information is highlighted in a different background color. Links to other domains or databases are highlighted in different color. Hence, a link is directly identifiable for a user. Furthermore, information that is larger than a certain length is hidden to keep a clear and well structured overview. This information will be displayed by clicking a specific keyword. Using this function it is possible to show important information and to hide information that is not of interest. Moreover, the web application provides a local navigation bar. Hence, it is possible to navigate easily and fast within the result page. But the navigation bar is only displayed for large results.

One of the most important features of DAWIS-M.D. is the interaction with the network editor VANESA. The web application provides two different views for the interaction. Therefore, a *Standard* and *Expert* view is implemented. Using the standard view a user is able to display the root element of the entry in the web application as a node in the network editor. In addition, it is possible to generate a complex network in VANESA based on the information of an entry in the web application. Moreover, a *Session* provided by


DAWIS-M.D.
Data Warehouse Information System for Metabolic Data
Bioinformatics Department
Medical Informatics Department
AG
BI
Source
expand all
Navigator
VANESA

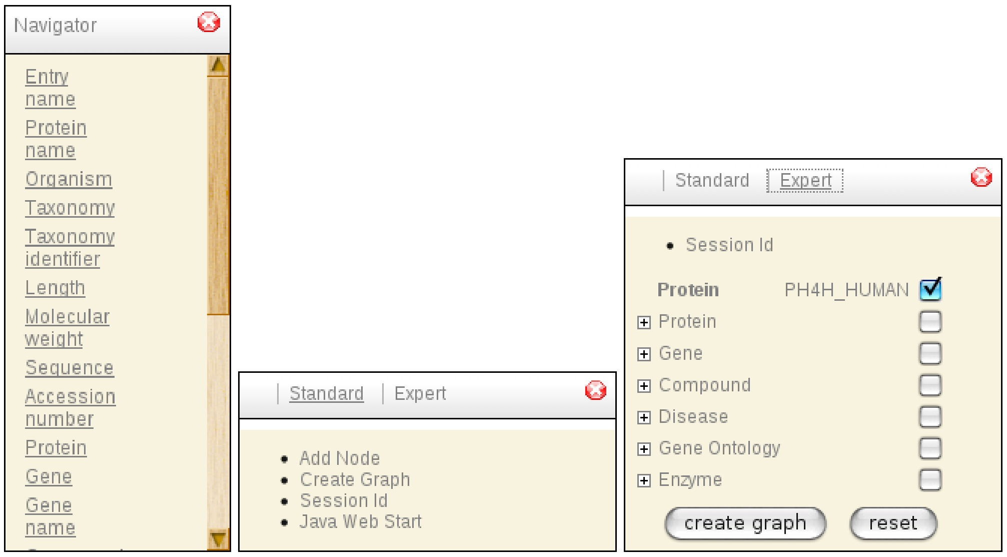
UniProt Protein

Entry name	PH4H_HUMAN
Protein name	Phenylalanine-4-hydroxylase (EC 1.14.16.1) (PAH) (Phe-4- monooxygenase).
Organism	Homo sapiens (Human).
Taxonomy	Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Euarchontoglires; Primates; Haplorrhini; Catarrhini; Hominidae; Homo.
Taxonomy identifier	9606
Length	452
Molecular weight	51862
Sequence	PH4H_HUMAN
Accession number	• P00439 • Q16717 • Q8TC14
Protein	8943
Gene	• 5053 • K03020 • U49897 • S61296 • BC026251
Gene name	PAH
Compound	• MO000093956 • MO000093957
Disease	261600
Gene Ontology	• GO:0004505 • GO:0008652
Enzyme	1.14.16.1
Structural Classification of Proteins	d.178.1.1
Structures	
Reference	NUCLEOTIDE SEQUENCE [MRNA]. TISSUE=Liver; MEDLINE=85199778; PubMed=2986678; DOI=10.1021/bi00324a002; Kwok S.C.M., Ledley F.D., Dilella A.G., Robson K.J.H., Woo S.L.C.; "Nucleotide sequence of a full-length complementary DNA clone and amino acid sequence of human phenylalanine hydroxylase."; Biochemistry

Figure 5.7: Detailed result page for the protein *PH4H_HUMAN* in DAWIS-M.D. Information that is larger than certain length is hidden to keep a clear and well structured overview. Using the hide/show function it is possible to make important information visible and to hide information that is not for interest.

DAWIS-M.D. is required for the communication with the network editor, so that VANESA is able to display the element or the network. The functionality of the expert view is nearly the same, but in this case the user is able to select elements more precisely. Hence, it is possible to create user-defined networks in VANESA to the current entry. In the expert view the elements are organized in a tree structure to maintain a clear and well structured overview. However, the communication between the web application and the network editor is realized via a web service of the data warehouse. The user-specific information is stored in the database of the web application via the GUI. Thus, VANESA is able to visualize the information, that is stored in the database, as a network via the web service. Figure 5.8 illustrates the communication bridge between DAWIS-M.D. and VANESA.

Furthermore, DAWIS-M.D. provides a user and system administration. An administrator is able to add a new user, edit or delete an existing user. In addition, it is possible to edit information of integrated databases. If new databases have been integrated into the system, information from these databases can be added by the administrator. Finally, the data warehouse information system provides additional information about the integrated databases such as release information or update information. The statistics are presented as table or diagram. Based on the statistics it is clear how the database of DAWIS-M.D. is composed.



(a) Navigation bar. (b) Standard view. (c) Expert view.

Figure 5.8: Graphical representation of the navigation and the communication bridge between DAWIS-M.D. and VANESA.

5.3 Summary

In this chapter the building of web-based life science data warehouses was discussed on the bases of two data warehouse approaches *DAWIS-M.D.* and *CardioVINEdb*. Particularly, the function and the structure were presented in detail as well as the system architecture and implementation. Both systems were implemented based on the concepts in section 5.1. The integration and update process for both systems is realized with the *BioDWH* data warehouse infrastructure.

The CardioVINEdb data warehouse approach provides integrated data from different popular life science databases, such as UniProt, KEGG, GO etc., and microarray data related to cardiovascular diseases from the *Cardioworkbench* EU project (see chapter 7) in a homogeneous web-based system. The system enables an intuitive search of integrated life science data, simple navigation to related information as well as visualization of biological domains and their relationships. Therefore, a "static" and "dynamic" visualization component for interactive graphical exploration of the data was introduced. Equipped with a monitor component that is part of the BioDWH infrastructure, updates of the integrated data will be executed in defined update circles. Furthermore, it enables a way for presenting complex biological data in a very user-comprehensible manner.

The DAWIS-M.D. data warehouse information system provides integrated data from eleven different molecular biological databases. The BioDWH infrastructure manages and updates the integrated data of the DWH. A web-based graphical user interface enables a simple and intuitive search of the integrated biological data as well as a simple navigation component within the data. Furthermore, the system provides a communication bridge to a powerful network editor and visualization tool *VANESA* that will be introduced in the next chapter. Using the DAWIS-M.D. web application and the network editor makes it possible for scientists and researchers to construct user-specific complex biological network in a comprehensive manner.

6 Modeling and Visualization of Biological Networks

This chapter gives an insight into the concept of modeling and visualization of biological networks based on life science data integration. The requirements for building modeling and visualization applications are introduced in section 6.1. This section gives an enumeration of important requirements. In section 6.2 modeling and visualization software approaches, called *MoVisPP* [CHH+09] and *VANESA* [JKT+10], will be introduced. The concepts of modeling, of visualization, the integrated databases, and the system architecture and its components will be introduced in detail. Based on screenshots of the applications the structure and function of each system will be presented. Finally, a summary of the chapter is given in section 6.3.

6.1 Requirements for Visualization of Biological Data

The importance of biological network modeling and visualization has already been discussed in chapter 3. Basically, the software is supposed to automate parts of the process in understanding and exploring biochemical networks in different organisms. Scientists are performing many different laboratory experiments to understand the way biological systems, metabolic pathways or signaling pathways work and how these pathways and networks might appear. The software solution is meant to support the scientists during their working procedures and decision-making process while exploring and understanding biological systems. Therefore, following a discussion with the partners from the Cardioworkbench project outlining the main requirements, the following general software specifications were conceived.

Requirement 6.1 *The biological network visualization software should be implemented as a platform-independent system.*

The main goal is to establish a novel bioinformatics data modeling and visualization software framework that integrates biological information. To attract a wide range of users the system should be platform-independent.

Requirement 6.2 *Information needs to be visualized in a clear and understandable manner to deliver a well defined overview of complex biological networks.*

A biological network consists of a set of diverse biological elements. A biological element needs to be described by a name, a label, a shape, by color and the network position. Each of these elements has certain properties that specifies them in detail and makes it easier to distinguish them within a network. Biological elements are represented as vertices and relations between elements are represented as edges. The graph layout should be realized by several layout algorithms.

Requirement 6.3 *A graphical user interface (GUI) should support a user to easily build and model biological networks.*

The application should consist of a graphical user interface which the user can use with any common system. The client application connects with the server to receive relevant information that is related to the biological network or model.

Requirement 6.4 *The tool should also have the capacity to share and export the biological networks with suitable and standardized file formats.*

The user should be able to store the graph in a general file format which is widely accepted by the biological community and which can also be used in other modeling and simulation environments.

Requirement 6.5 *Integration of selected biological and medical data sources to search the database for relevant information that might be helpful in understanding biological and medical processes.*

The user should have the possibility to search through biological and life science data sources for relevant information. Search forms should enable the user to access information that meet his criteria and might be helpful in understanding biological and medical processes. Moreover, the data warehouse that contains life science and biological information runs on the server side.

6.2 Modeling and Visualization Approaches

The accurate representation of research data is one of the main tasks in computer science. Information should be visualized in a clear and understandable manner to meet the aims of research activities, otherwise important information could get lost. One of the most important tasks is the integration of data as well as the development of a graphical user interface by which the researcher is able to create, edit and examine biological networks. In addition, a user should be able to handle any kind of biological network including its biological and medical details.

Over many years research activities took place on biological questions with the goal to discover their functions and gain insight. Software solutions which provide visualization, analysis services and an information management framework are highly demanded among scientists. It is not surprising that many bioinformatics groups have contributed to the task of developing modeling and visualization software frameworks as described in chapter 3. But usually each software approach focuses on a particular problem. And none of these approaches is so powerful that it would be able to address to all problems. The aim is to provide a software framework that gives the biological scientists the opportunity to model their biological networks and systems with all its relevant details. In the following sections two novel approaches for the visualization of biological networks will be presented.

6.2.1 A Web-based Tool to Model and Visualize Biochemical Pathways as Petri Nets

Today, scientific research needs a combination of different data sources as delineated in the previous chapters. The existing pathway databases have a number of limitations, because they do not contain comprehensive information about metabolic pathways, such as physical and chemical properties of the enzymes involved. Furthermore, these data and biological information have to be visualized in a clear and understandable manner. Therefore, in this section the MoVisPP tool and its concept of modeling and visualization will be introduced. MoVisPP, an acronym for **Mo**deling and **Vis**ualization of **P**athways using **P**etri nets, is a web-based tool for modeling and visualizing of biochemical pathways. This web application consists of a data warehouse approach spanning multiple databases containing biochemical and metabolic information from Enzyme, MINT, OMIM, PDB, iProClass, GO and KEGG. The MySQL relational database management system as backend contains all necessary information. For visualization of Petri nets as pathways, metabolic and biochemical information databases are required, as mentioned previously. The integration of data is carried out semi-automatically by using a BioDWH prototype that provides Java-based parsers for each database. Hence, transformation and merging of data is done manually. For visualization of the pathway as Petri nets the JUNG graph library framework is used. A web-based graphical user interface is implemented in JavaServer Pages (JSP) that can be accessed with a common web browser.

In the following section, an introduction of modeling biological networks using MoVisPP is given. Finally, it is possible to export the networks to a simulation environment such as Cell Illustrator by using standardized export formats like Systems Biology Markup Language (SBML), Petri Net Markup Language (PNML) or Cell System Markup Language (CSML).

6.2.1.1 Concept of Modeling and Visualization

Basically, data from KEGG is used in the MoVisPP approach to model and visualize the metabolic pathways as Petri nets. First, the concept of handling the data from KEGG will be introduced. As already mentioned in chapter 3.4.1, places establish the state and transitions represent the system activity. The graphical representation follows the Petri net model, i.e. places are shown as circles and transitions as rectangles. Dependencies are modeled as directed arcs.

Therefore, genes, proteins, enzymes, ordinary chemical compounds, glycans, active components and pathways are shown as circles. Reactions and relations from KEGG are modeled as transitions. MoVisPP follows the concept of qualitative modeling, so kinetic aspects and others are not taken into account, because in most cases kinetic data is lacking or not available. Consequently, it is not useful to give places an initial marking. But on the other hand, the user is able to give values to the places and functions to the arcs by using secondary literature after importing into a simulation environment such as Cell Illustrator.

Proteins and enzymes are shown as blue-colored places. See figure 6.1(b). More detailed information about the protein and protein-related genes is also available from the integrated databases. The existence of gene information is represented graphically as a thick, red border. Furthermore, information about multiple enzymes, proteins and genes is also available. This is useful to obtain an overview and understanding of missing modeling details such as gene expression data.

For modeling reactions in a pathway, information about fitting enzymes, substrates and products is essential. The integrated KEGG database provides all relevant information from different tables. For a clear overview, other chemical compounds such as water or ATP are not visualized, since these components typically are ubiquitous and approximately involved in each reaction. Co-enzymes are also not drawn in the graph. More detailed information about the reactions is available in text form within the tool. Enzymes act as catalysts in a reaction and are not consumed. Accordingly, from a place "enzyme" to a transition "reaction", an arc is drawn in both directions. If a reaction is reversible, an arc is drawn in our model from place "substrate" to transition "reaction", from there to another place "product" and, of course, vice versa.

Differing from the definition of Petri nets, a reaction and its reaction number from KEGG (e.g. R01087) are labeled at the arc between product or substrate and the transition. This is suitable for obtaining a better overview and of course not an arc weight or mathematic function.

In KEGG, there are five different kinds of relations[1]: interaction within gene expression (GErel), protein-protein interaction (PPrel), interaction between protein and chemical compound (PCrel), enzyme-enzyme relation (ECrel), as well as a link to a pathway map (Maplink).

[1]http://www.genome.jp/kegg/docs/xml/

⋆ *Maplinks* are connections to other pathways, that means they show which products of the current pathway are processed in another pathway. Typically, these relations are a subclass of the compound. An example is illustrated in figure 6.1(a).

⋆ *ECrels* are always connected via chemical compounds. Enzymes, as described in section 2.1.2, act as catalysts and are not consumed. Hence, each place has arcs in both directions as shown in 6.1(b). Furthermore, another arc goes to the place of the chemical compound.

⋆ *PCrel* always shows arcs in one direction from the pre-condition through a transition to the place of the post-condition. The transition is colored red for better differentiation (see figure 6.1(c)).

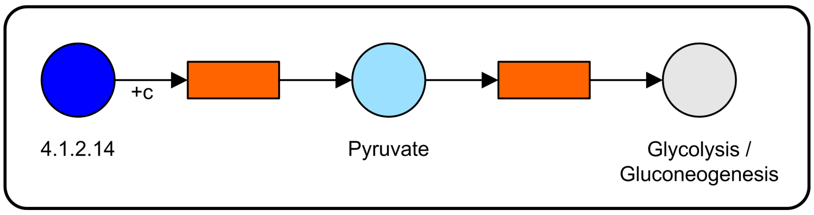
⋆ *PPrels* are usually modeled with an arc from the pre-condition via a transition to the post-condition. If they have an intermediate compound, an additional pre-condition and post-condition will be included into the model at the same time. A special feature is a so-called "inhibitory arc", which is defined as HFPN in section 3.4.2. Inhibitory arcs are marked with a circle at the end of an arrow on the transition side. Protein-protein interactions are yellow-colored in the Petri net model as described in figure 6.1(d).

⋆ *GErels* have three subdomains: expression, repression and indirect effects. In principle, they are connected via a transition and if genetic information such as DNA is present, this information is also connected to the transition. Repressions are indicated by a circle at the end of an arc (inhibitory arc). Gene expression relations are colored light-red in the visualization (see figure 6.1(e)).

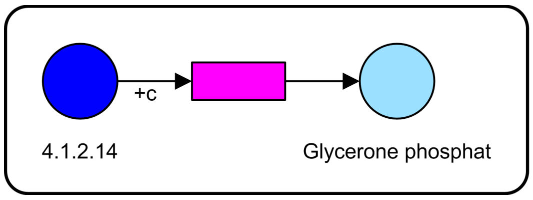
6.2.1.2 Implementation

MoVisPP web application consists of a graphical user interface which the user can use with any common browser. Moreover, the application logic and data warehouse run on the server side. The application logic is implemented in Java, the platform-independent programming language. Figure 6.2 illustrates system architecture of the MoVisPP software.

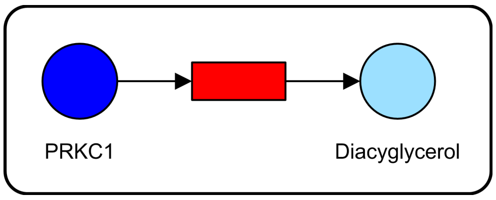
For the web application, MySQL is used as a database system for the server. A Java Database Connectivity (JDBC) driver for MySQL manages access to the relational database from the Java classes. JavaServer Pages (JSP) contains Java code that can be included via JSP elements in static HTML code. Depending on the behavior of the user, the static HTML parts can be dynamically complemented. Whenever a JavaServer page is opened in a web browser, a request is sent from the client to the server. The servlet container from the web server creates, compiles and/or executes a Java servlet. A response in form of a generated static HTML website is sent to the client from the server. MoVisPP uses an Apache Tomcat web server to generate HTML code.



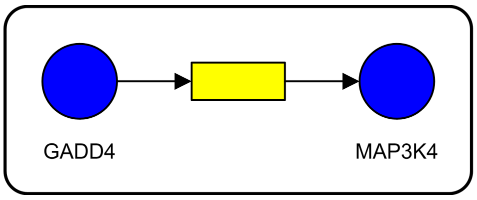
(a) Maplink relation.



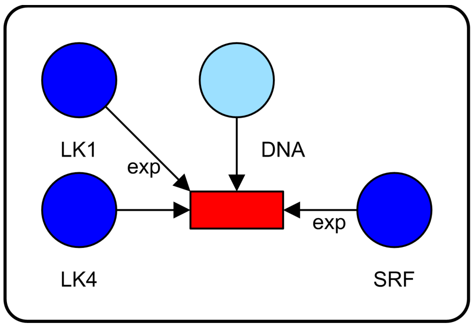
(b) Enzyme-Compound relation



(c) Protein-Compound relation



(d) Protein-Protein relation



(e) Gene-Enzyme relation

Figure 6.1: Graphical representation of KEGG relations as Petri net used in MoVisPP.

The user is able to choose different options for the modeling and visualization of the pathways as Petri nets within the web application. It only takes a few steps to generate a Petri net. First, the user selects one of the KEGG pathways, afterwards a specific organism. Furthermore, the user has the opportunity to decide if KEGG reactions, different KEGG relations, interactions from MINT or data from OMIM should be modeled and visualized. Moreover, it is possible to modify basic settings related to the size of places, transitions and the whole image of the Petri net.

Finally, a website with an image of the generated pathway as Petri net, based on the user selection, is displayed. A so-called “image map” over the pathway image allows the user to obtain detailed information about all the elements of the Petri net.

6.2.1.3 Structure and Function of the Web-Application

In this section the functionality and the components of web application will be introduced. The function and structure of MoVisPP will be illustrated based on different figures.

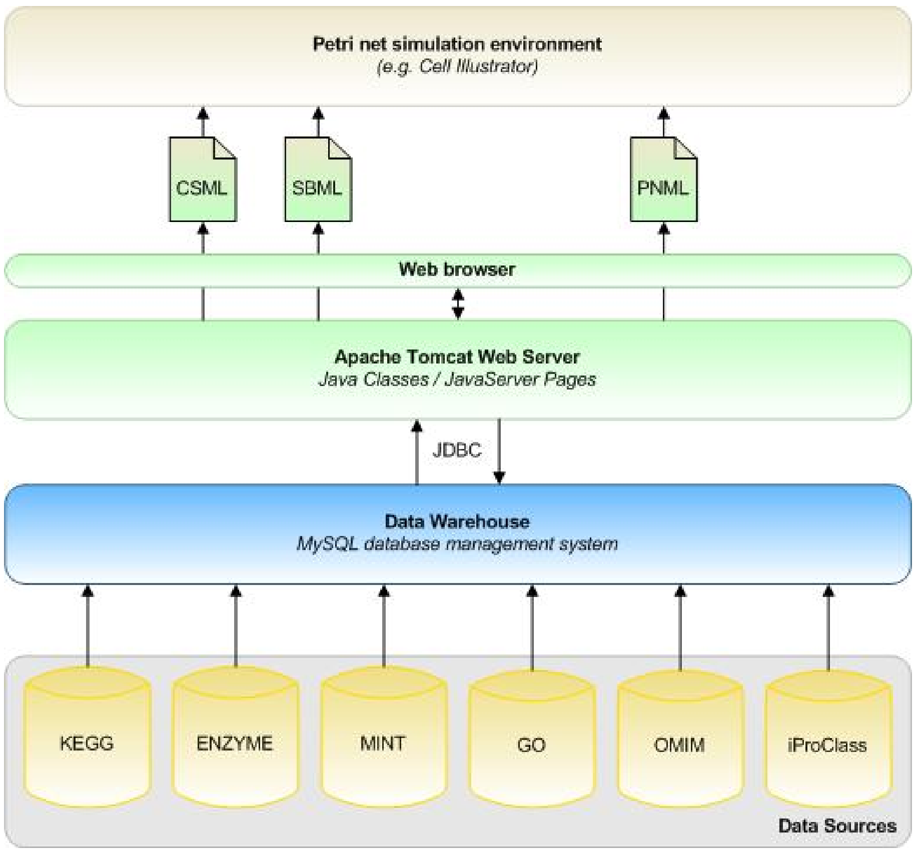
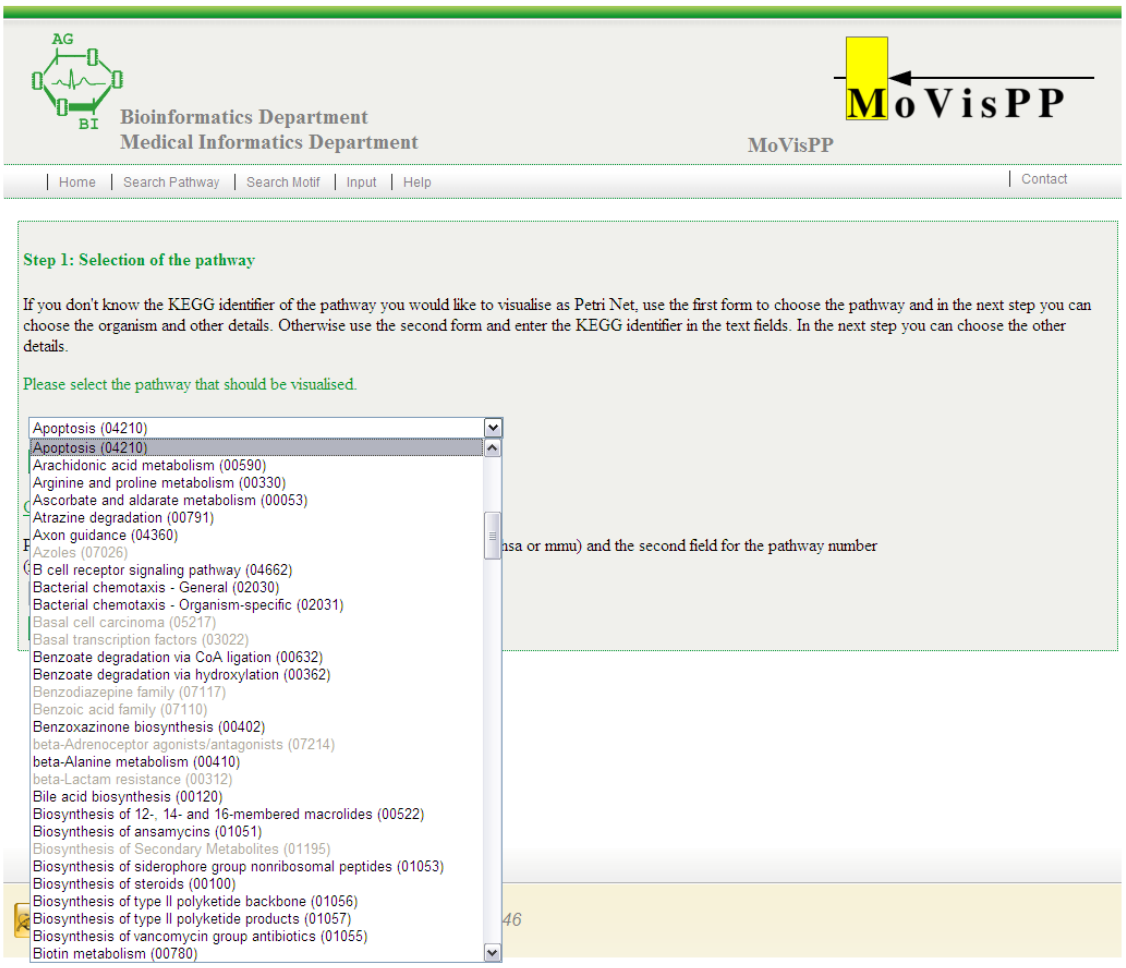


Figure 6.2: System architecture of MoVisPP system.

Figure 6.3 illustrates a website with a search engine of the web application. The global navigation within the web application is possible via the head of the web page. In general MoVisPP is divided into three parts, two search engines and an input interface. It is possible to search a pathway by name or KEGG identifier by clicking *Search Pathway*. On the other hand, it is possible to search pathways in the KEGG pathway database with a sequence of proteins which are similar to the given input motif by using the *Search Motif* engine. Moreover, it is possible to create user-defined Petri net pathways by using the *Input* interface.

First of all, to visualize a pathway the user can choose a specific pathway from the pathways in the database by a drop-down menu as described in figure 6.3. Alternatively, it is possible to input the shortcut of the organisms of the pathway and the pathway number, i.e. identifier of the pathway (for instance *hsa* and *0010*). Next, the user has to specify the organism of the pathway, if a pathway was only selected in the first form. Now it is possible to configure the reaction and relation types of the KEGG pathway that should be modeled. Hereby the user is enabled to choose between organism specific reactions and relations or relations and reactions of the reference pathway. Additionally, five different types of relations, such as *Protein-protein interactions*, *Protein-compound relations*, *Enzyme-enzyme relations*, *Gene-expression interactions* and *Links to other maps*, can be modeled and visualized, as described in section 6.2.1.1. Accordingly, MoVisPP is able to provide specific gene information to an organism and on the other hand all reactions and relations that are available and known for the reference pathway could be visualized. Sometimes there is little or no reaction and relation for organism specific pathways available. In organism specific KEGG pathway maps connections of the reference pathway are displayed. Furthermore, protein-protein interactions from MINT and information from OMIM diseases can be modeled in the Petri net. But this is only possible for the *homo sapiens (human)* organism. Finally, the size of the picture and the size of the transition and the places can be configured. The picture can be scaled by factor 1 up to 3 and the transition and places can be displayed "default" or "bigger".



Step 1: Selection of the pathway

If you don't know the KEGG identifier of the pathway you would like to visualise as Petri Net, use the first form to choose the pathway and in the next step you can choose the organism and other details. Otherwise use the second form and enter the KEGG identifier in the text fields. In the next step you can choose the other details.

Please select the pathway that should be visualised.

Apoptosis (04210)

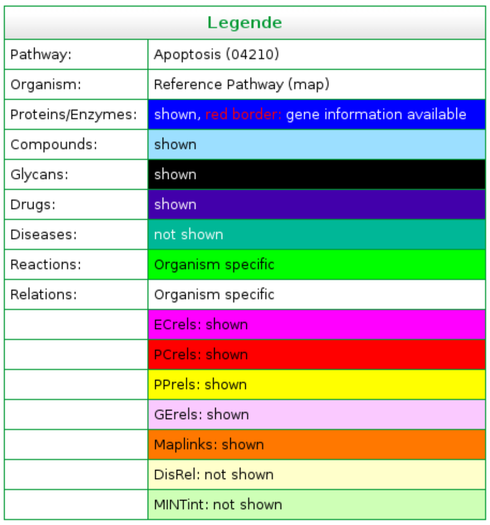
Apoptosis (04210)
Arachidonic acid metabolism (00590)
Arginine and proline metabolism (00330)
Ascorbate and aldarate metabolism (00053)
Atrazine degradation (00791)
Axon guidance (04360)
Azoles (07026)
B cell receptor signaling pathway (04662)
Bacterial chemotaxis - General (02030)
Bacterial chemotaxis - Organism-specific (02031)
Basal cell carcinoma (05217)
Basal transcription factors (03022)
Benzoate degradation via CoA ligation (00632)
Benzoate degradation via hydroxylation (00362)
Benzodiazepine family (07117)
Benzoic acid family (07110)
Benzoxazinone biosynthesis (00402)
beta-Adrenoceptor agonists/antagonists (07214)
beta-Alanine metabolism (00410)
beta-Lactam resistance (00312)
Bile acid biosynthesis (00120)
Biosynthesis of 12-, 14- and 16-membered macrolides (00522)
Biosynthesis of ansamycins (01051)
Biosynthesis of Secondary Metabolites (01195)
Biosynthesis of siderophore group nonribosomal peptides (01053)
Biosynthesis of steroids (00100)
Biosynthesis of type II polyketide backbone (01056)
Biosynthesis of type II polyketide products (01057)
Biosynthesis of vancomycin group antibiotics (01055)
Biotin metabolism (00780)

hsa or mmu) and the second field for the pathway number

46

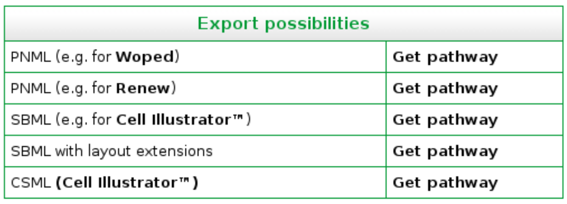
Figure 6.3: Screenshot of the MoVisPP web-application while using the pathway top-down menu.

A website with an image of the generated pathway as Petri net, based on the user selection, is generated. A "image map" over the pathway image allows the user to obtain detailed information about all the elements of the Petri net. Furthermore, a legend of the different colors is available and will appear as illustrated in figure 6.4(a). But the most important feature are the different *Export* functions using the Petri net pathways in several simulation environments. Figure 6.4(b) illustrates the different export formats *PNML*, *SBML* and *CSML*. There are two possibilities to export the PNML file, because the programs WoPeD (Workflow Petri Net Designer)[2] and Renew[3] handle PNML in different ways. The implementation of the visualization in PNML is very easy, because PNML was specially developed for the description of Petri nets. For the *P/T nets* (Place/Transition nets) the PNTD (Petri Net Type Definition) from the PNML website[4] was used. Furthermore, two different SBML formats are available, as well. The reason is that SBML does not support graphical positioning which is essential for the reconstruction of biological and most specifically metabolic pathways. Most of the programs that work with SBML are able to layout simple networks, but they are not able to layout complex networks, such as the KEGG pathway maps, in a clear way. An SBML layout extension by Gauges [GRSW05] already exists. This should be possible with the *libSBML*[5] in version 3.0. Finally it is possible to export the networks into the CSML format that is supported by the Cell Illustrator (see section 3.4.3) Petri net simulation environment. The implementation of the visualization in CSML is very easy, because CSML was specifically developed for the description of Petri nets and the Cell Illustrator.



Legende	
Pathway:	Apoptosis (04210)
Organism:	Reference Pathway (map)
Proteins/Enzymes:	shown, red border: gene information available
Compounds:	shown
Glycans:	shown
Drugs:	shown
Diseases:	not shown
Reactions:	Organism specific
Relations:	Organism specific
	ECrels: shown
	PCrels: shown
	PPrels: shown
	GErels: shown
	Maplinks: shown
	DisRel: not shown
	MINTint: not shown

(a) Colored legend about the places and transitions of the Petri net.



Export possibilities	
PNML (e.g. for **Woped**)	**Get pathway**
PNML (e.g. for **Renew**)	**Get pathway**
SBML (e.g. for **Cell Illustrator™**)	**Get pathway**
SBML with layout extensions	**Get pathway**
CSML **(Cell Illustrator™)**	**Get pathway**

(b) Supported export formats for the Petri nets.

Figure 6.4: MoVisPP visualization legend and export window.

Another option is to search pathways in the KEGG pathway data by a sequence of pro-

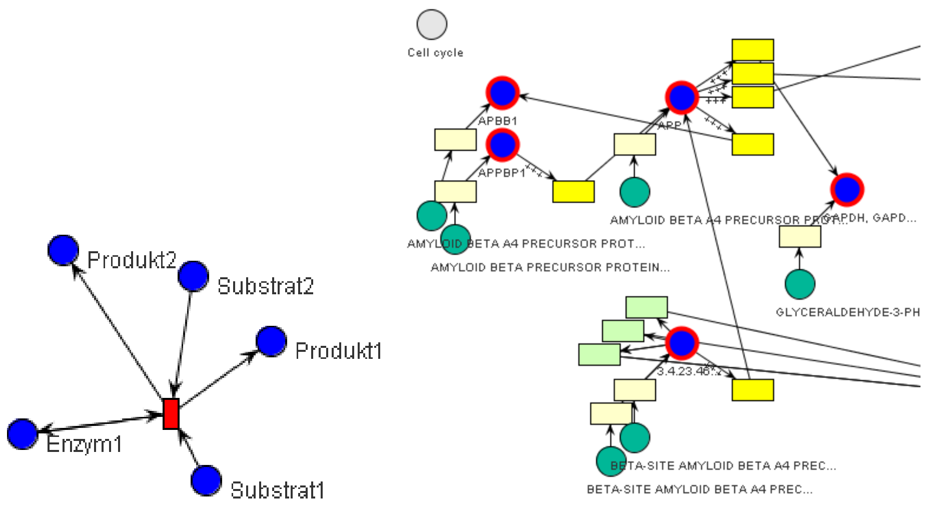
[2]http://www.woped.org/

[3]http://www.renew.de/

[4]http://www2.informatik.hu-berlin.de/top/pnml/pntd.html

[5]http://www.sbml.org/software/libsbml/

teins which are similar to the given input motif. Any variety of sequencing numbers from KEGG, PDB, UniProt and enzymes will lead to the corresponding reference KEGG maps. The multiple alignment algorithm against the databases is described in section 6.2.1.4. A list of similar and potential KEGG maps ordered by hit score is displayed on a new website. Furthermore, information about the input proteins including the related pathway is given. After the selection of the pathway it is possible to configure the pathway visualization as described in the paragraph above. An enzymatic reaction with two substrates, an enzyme that catalyzes the substrate into two products as input looks like this `[Substrat1,Substrat2];[Enzym1];[Produkt1,Produkt2]`. The Petri net of this reaction is shown in figure 6.5(a). Furthermore, it is possible to upload a file to visualize complex networks that include different kinds of reactions and relations. Finally, it is possible to create user-defined pathways as Petri net and visualize them in one step. A detail of a pathway as Petri net is shown in figure 6.5(b).


Produkt2
Substrat2
Produkt1
Enzym1
Substrat1
Cell cycle
APBB1
APPBP1
APP
AMYLOID BETA A4 PRECURSOR PROT...
AMYLOID BETA PRECURSOR PROTEIN...
AMYLOID BETA A4 PRECURSOR PROT...
GAPDH, GAPD...
GLYCERALDEHYDE-3-PH
3.4.23.46
BETA-SITE AMYLOID BETA A4 PREC...
BETA-SITE AMYLOID BETA A4 PREC...

(a) MoVisPP input visualization of a reaction as Petri net.

(b) MoVisPP detail visualization of the Alzheimer's disease pathway as Petri net.

Figure 6.5: MoVisPP visualization as Petri net.

6.2.1.4 Multiple Alignment Algorithm for Motif Search in MoVisPP

As mentioned in the previous section it is possible in MoVisPP to search pathways in the KEGG pathway data by a sequence of proteins which are similar to the given input motif. Therefore, a multiple alignment algorithm against the databases was developed. For the multiple alignment of linear interactions and reactions a classical sequence analysis algorithm is adapted. In detail a modified version of the *Needleman-Wunsch algorithm* [NW70] is used. The Needleman-Wunsch algorithm performs a global alignment on two protein or nucleotide sequences. Therefore, a so-called *dynamic programming matrix* is constructed and the alignment is found by a *traceback* algorithm [Dur07].

The dynamic programming matrix F has the dimension $(n + 1) \times (m + 1)$, where n has the length of sequence X and m has the length of sequence Y. The two sequences indicate by i respectively j, so that $x_1x_2 \ldots x_i \ldots x_n$ and $y_1y_2 \ldots y_j \ldots y_m$. Accordingly, the matrix F covers the optimal *score* of the alignment, between the sequences $x_1 \ldots x_i$ and $y_1 \ldots y_j$, at position $F(i, j)$. Beginning by initializing F with $F(0, 0) = 0$,$F(0, j) = j \times gp$ and $F(i, 0) = i \times gp$. The best score up to (i, j) will be the largest for the following three options. Therefore, it is

$$F(i,j) = max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) + gp \\ F(i, j-1) + gp \end{cases} \tag{6.1}$$

Thereby gp is a *gap* and gets a penalty with a predefined value called *gap-penalty*. The score $s(x_i, y_j)$ describes a *match* between two proteins of the sequences and is defined by their similarity. For the calculation of the similarity between two proteins two algorithms can be used.

1. The comparison of two proteins is realized via the *EC number* similar to the *PathAligner* algorithm [CH04] or the algorithm described by Pinter [PRYLZU05]. If the first digit of the EC number is equal the similarity score s is 0.25, if the second digit is also equal it is $s = 0.50$, and if the third digit is equal the score is $s = 0.75$. Hence, if both EC numbers are equal the similarity score is $s = 1.0$.

2. Using the protein classification databases SCOP and CATH [OMJ+97] proteins can be compared based on their evolutional respectively structural similarities. For instance the similarity of two proteins is calculated by using SCOP as follows.

$$s(p_1, p_2) = 1 - \frac{log(\Delta_{SCOP}(p_1, p_2))}{total} \tag{6.2}$$

   Therefore Δ_{SCOP} is the number of proteins that have the same classification in the SCOP database as protein p_1 and p_2. Hence, the $total$ is the overall number of classes in SCOP. The calculation for the similarity score of CATH is analog.

The final alignment is calculated by the *traceback* algorithm. For that purpose the algorithm starts at the matrix position $F(n, m)$ and calculates the path back to the beginning $F(0, 0)$. The classical Needleman-Wunsch algorithms has a time and space complexity of $O(nm)$.

For a multiple alignment of multiple sequences a progressive alignment method of Feng and Doolittle [FD87] is used. Therefore, calculate a diagonal matrix of $N(N - 1)/2$ distances between all pairs of N sequences by standard pairwise alignment and convert the raw alignment scores into approximate pairwise distances. Then the pair with the highest alignment score is merged into one single alignment. Both sequences will be removed from diagonal matrix and the new alignment will be added. This step will be repeated until all sequences are merged into a multiple alignment. In the following, the formula for converting alignment scores to distances values is given.

$$D = -logS_{eff} = -log\frac{S_{obs} - S_{rand}}{S_{max} - S_{rand}} \quad \text{where} \tag{6.3}$$

S_{obs} observed pairwise alignment score

S_{max} maximum score, the average of the score of aligning either sequence to itself

S_{rand} score of a random alignment

S_{eff} normalized percentage similarity (decreasing fast to zero with increasing evolutionary distance)

The log function is used to measure the distance more linearly. This approach is similar to the alignment algorithm of the *SignAlign* prototype [HTOH08]. More detailed information about the presented alignment algorithms could be found in [Dur07].

6.2.1.5 Summary

In this subsection an application of Petri nets for modeling of biological networks was delineated. Furthermore, the concept of modeling and visualization based on relevant metabolic databases such as KEGG, BRENDA, etc. were presented in detail. Based on this integration process, the system supports semi-automatic generation of the correlated hybrid Petri net model. The networks created from MoVisPP can be exported and used by several other applications that support Petri nets e.g. WoPeD, Renew, Cell Illustrator. Moreover, the tool provides different export possibilities such CSML, PNML and SBML. Finally, a multiple alignment algorithm for motif search engine in MoVisPP was introduced in detail.

In comparison to other tools, MoVisPP runs in common web browsers without any additional software platforms such as Java. An easy-to-use web interface makes it possible to create a biological network within two steps. Furthermore, the application is able to visualize on-the-fly, a clearly arranged network, enriched with integrated external database information. Finally, it is possible to use generated networks in external simulation environments for qualitative and quantitative simulation.

6.2.2 An Integrative Bioinformatics Approach to Visualize and Model Biological Networks

Every living cell is made of a vast and complex network of interacting proteins, enzymes, DNA, RNA, metabolites, chemical compounds and other molecules. There are many biochemical processes involved in a cell. Typically, in literature those processes with their elements and interactions are captured as networks. During research, enormous amounts of biological data are produced by high-throughput sequence investigation tools, array technologies for genes and proteins and the expanding electrical equipment. Information from different fields of studies is brought together in order to examine and analyze quantities and relationships. Now, the fundamental question is how to model and simulate complex biological networks. Consequently, an approach of extracting, analyzing and modeling meaningful biological and metabolic networks enriched with biological data from heterogeneous data sources is required. This challenge is one of the main tasks in bioinformatics. Therefore, a new editor, including an information system that combines heterogeneous and diverse data, called *VANESA* (**V**isualization and **An**alysis of **Ne**tworks in **S**ystem Biology **A**pplications) will be introduced in this subsection. This approach shows how to combine different fields of studies such as life science database integration, modeling, visualization and simulation for a semi-automatic reconstruction of the complex networks. In comparison to MoVisPP, the VANESA application is more dynamic because of the implementation as Java application. Therefore, it is easier to create and manipulate biological networks that are addressed to a specific problem. In addition, the architecture of the system makes it easy to plug-in new and complex analysis tools.

6.2.2.1 System Architecture

The network editor VANESA consists of several modules which will be described in this subsection. The main components are appropriate for data modeling, loading and transformation. Consequently, those modules build the framework of the application. On the one hand they realize the representation of the network models and on the other hand they make a data and information retrieval and transformation of the integrated databases possible. The communication and query processing component is responsible for the communication between the data loading and transformation module. Moreover, it handles the communication to the external data sources and the DAWIS-M.D. data warehouse. The connection to the external data sources BRENDA, KEGG and DAWIS-M.D. is realized by an Axis2 web service. Therefore, the service connects to the data warehouse and the underlying data sources to extract biological and medical information of interest. The user interacts with the software through the graphical user interface that consists of different visualization and graph manipulation components. Using these GUI components the researcher has the possibility to create, edit and manipulate the network. At the minimum, VANESA supports different export and import functions. Those functions realize the data exchange of network models among different users and different visualization and modeling tools, such the Cell Illustrator that is described in section 3.4.3.

The module design of the system as shown in figure 6.6 makes clear that the software consists of different individual modules. Moreover, the system consists of independent modules that fulfill different tasks in order to solve the diverse questions. In the following a detailed description of different modules in VANESA is given.

⋆ *Data modeling module:* It is the central module of the application that is appropriate for the interpretation of obtained information. The interpretation is supposed to result in a semantically correct biological model. This model includes any kind of biological elements, such as genes, enzymes, chemical compounds or other molecules. In addition, the modeling module is able to find relationships between the different biological elements in order to construct a metabolic or biological network.

⋆ *Graph calculation and representation module:* Typically, the best way to handle networks or pathways is to transform them into a mathematical model. This module calculates a mathematical graph out of a biological network or pathway. A mathematical model has the advantage of being able to transform and manipulate a network in an easy and efficient way. Moreover, a mathematical graph model has the ability to apply any kind of graph theory on it. Therefore, it provides the opportunity to examine a biological network with mathematical rules and functions.

⋆ *Graph visualization:* This module provides the graphical representation of created networks to the user. The representation of the networks as a graph is closely related to the mathematical specifications. Biological elements are represented as vertices and relations/reactions between elements are illustrated as edges. For a clear overview and a simple representation, all vertices and edges corresponding to their biological origin are specifically designed in different shapes and highlighted in different colors. The graph model is a directed graph, so that the direction of an edge, i.e. a relation or reaction, is illustrated by an arrowhead at the end of the edge.

⋆ *Graph layout algorithms:* The graph layout is realized by several standard layout algorithms, such as *Circle Layout*, *FRLayout*, *Spring Layout* and so on. Additional layouts such as *GEMLayout* based on [FLM95] are also implemented or under construction. The plug-in structure of the module does support the possibility of new graph layout integrations. This module supports the user with layout and transformation algorithms to represent a graph in a clear and user-friendly way.

⋆ *Graph drawing and editing module:* Biological networks can be created either manually by the researcher or it can be generated automatically out of integrated databases or the data warehouse DAWIS-M.D. If, because of lack of information, it is not possible to extract a network from the data sources, the user has the possibility to create his own model manually. The module supports the user with a graphical user interface to create, move, drag and drop any kind of biological components on the workspace. The different tools and functions of the module support the user to model or create semantically correct biological networks in a quick and efficient way.

⋆ *Auto suggest and correction functions:* VANESA supports the user with so-called "auto suggest" and "correcting functions". During the modeling process this module verifies the nomenclature of each biological element on the workspace. This is necessary to build a correct biological model in terms of a nomenclature that is used by the biological community. The naming function is supported by auto suggests that correct the naming if necessary. Therefore, information from the data warehouse as well as from the biological data sources KEGG and BRENDA are used.

⋆ *Search mask:* A user should be able to query external data source for information of interest. This module provides a researcher with search forms for KEGG, BRENDA and DAWIS-M.D. to receive any kind of biological data. Furthermore a protein-protein interaction form is available to build or predict protein-protein interactions networks. Using these forms the user is able to browse the integrated biological data sources for required information.

⋆ *Data sources and communication interface:* This module handles the connections and queries to the DAWIS-M.D. data warehouse, KEGG and BRENDA to collect information from different databases. The communication of the VANESA client and the data sources is realized via web service.

⋆ *Analysis and comparison tools:* The module provides the user with useful analysis and comparison functions, such as *shortest path* or *node ranking* algorithms. It is also possible to compare different kind of networks with each other as well as to investigate and analyze networks concerning particular questions.

⋆ *Data export and import:* The tool is able to export and import the created networks by selected file formats. In the first place, the choice was to make use of the SBML file format. The export is also realized via a web service to support valid SBML files. Therefore, it is not necessary to install SBML libraries on the client computer. In addition, the user is able to export his model as a GraphML file.

In summary, the main advantage of the software lies in the interaction of the different modules that can easily be extended, so that it is easy to enhance the existing modules or even integrate new ones. The flexible system architecture and plug-in architecture supports these kinds of changes. Moreover, the use and combination of modules exposes the user to a wide range of possibilities to investigate and explore networks, in which the user is interested. To provide a wide range of biological knowledge VANESA is based on the DAWIS-M.D. data warehouse which integrates the most important and popular life science databases.

6.2.2.2 Implementation

The software framework is developed in the programming language Java. Java is platform independent and is available for a wide variety of computing platforms. Furthermore, Java is often used in web servers and enterprise applications. Hence, Java applications have a high degree of platform independence. Moreover, Java applications offer flexible software solutions that can be provided to a large audience. In this way the software solutions can become widely used. Figure 6.7 illustrates the system architecture of the VANESA software.

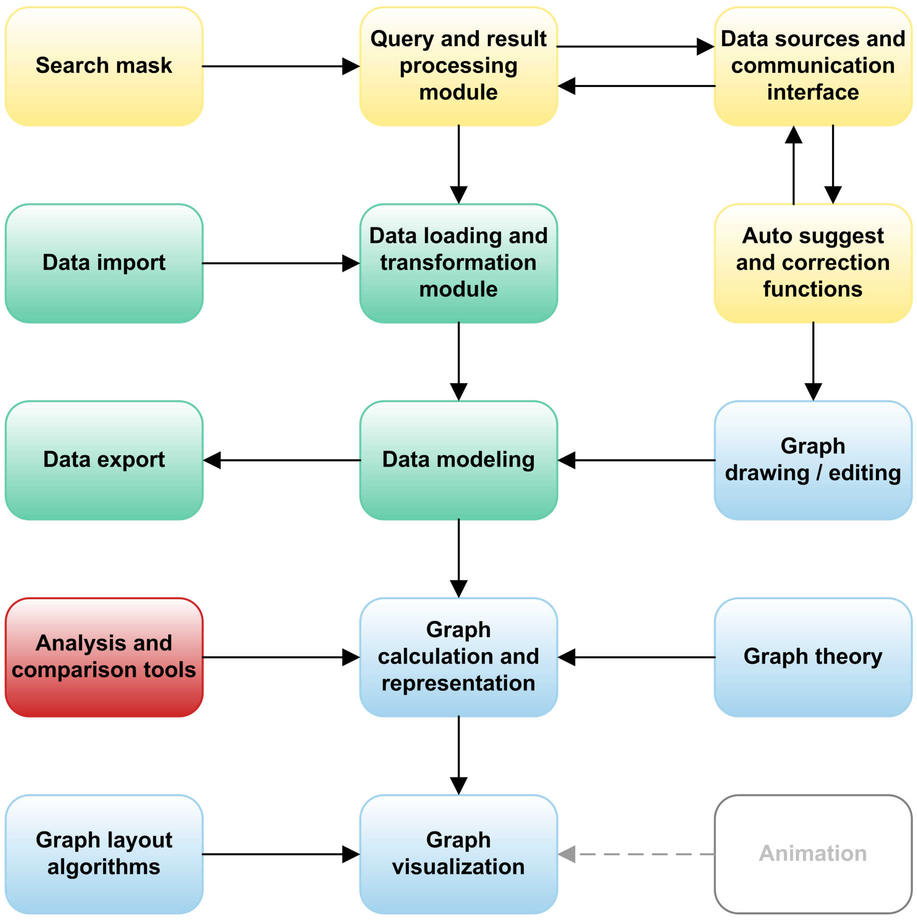

Search mask
Query and result processing module
Data sources and communication interface
Data import
Data loading and transformation module
Auto suggest and correction functions
Data export
Data modeling
Graph drawing / editing
Analysis and comparison tools
Graph calculation and representation
Graph theory
Graph layout algorithms
Graph visualization
Animation

Figure 6.6: Software module design of VANESA.

The graphical network representation is based on the JUNG graph library version 1.7.5. JUNG is an open source Java-based library that provides classes to describe graphs, nodes and edges with additional layout preferences. Moreover, JUNG provides a variety of entities and relations such as directed and undirected graphs. It also provides a visualization framework that makes it easier to implement tools for an interactive exploration of networks. The benefit of this framework is the ability to integrate user-specific functions, representations and analysis tools. For VANESA JUNG is used to represent a graph structure. Finally, the library was adapted to the requirements of VANESA and changed in such a way that it is possible to represent complex biological network models.

The communication between VANESA client and the data sources is realized with *data sources and communication interface* as mentioned in the previous section. As relational backend a the MySQL database management system is used for integrated data. On the one hand, it is possible to access the underlying databases by using Java Database Connectivity (JDBC) directly. Hence, the KEGG, BRENDA and DAWIS-M.D. data sources need to be installed locally. On the other hand, a web service controls the communication between the server and client by using SOAP. This is much more convenient and does not need a local installation of data sources. The user only needs an ordinary internet connection.

6.2.2.3 Structure and Function of the Application

Figure 6.8 illustrates the graphical user interface of the VANESA software. The GUI can be divided into four parts. The most important component of the tool is the network modeling workspace. The toolbar is located on the right-hand side of the main window while on the left-hand side the database search forms, the network information and analysis panels are arranged. The menu bar is located at the top of the main window.

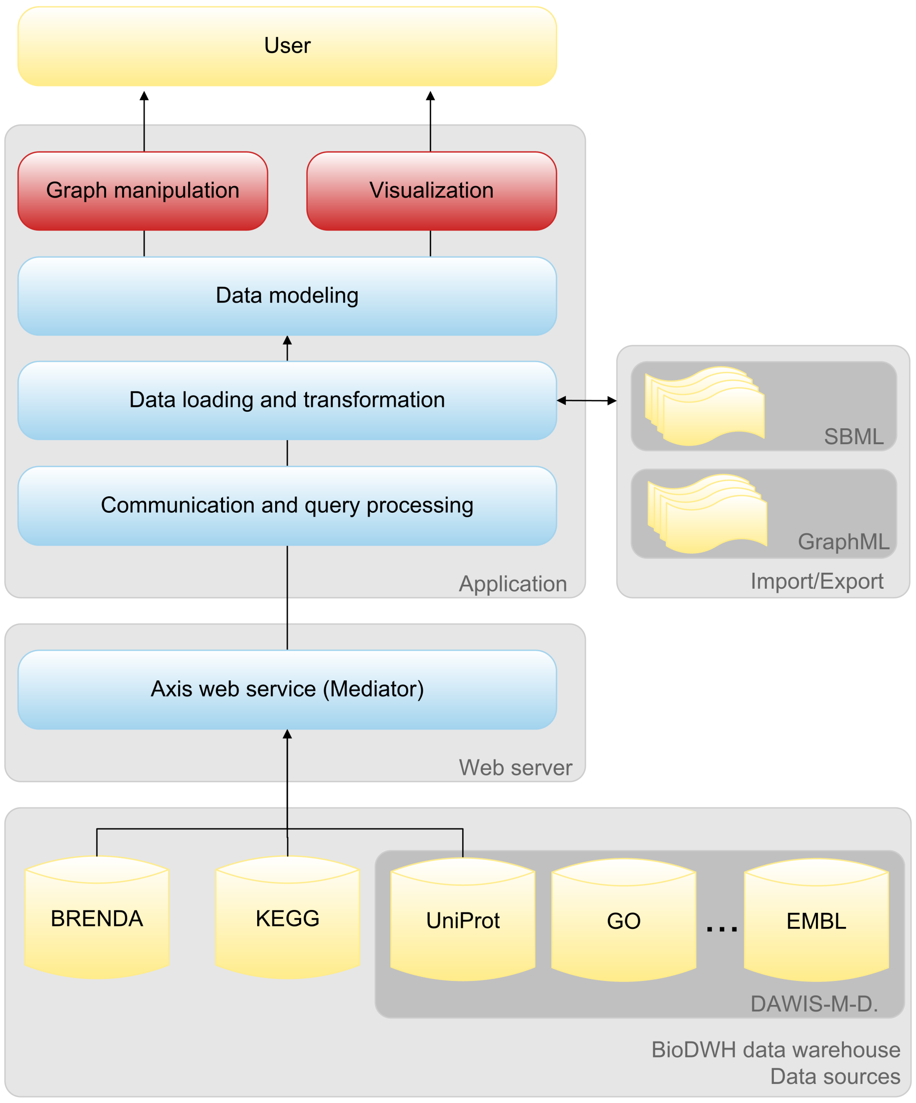
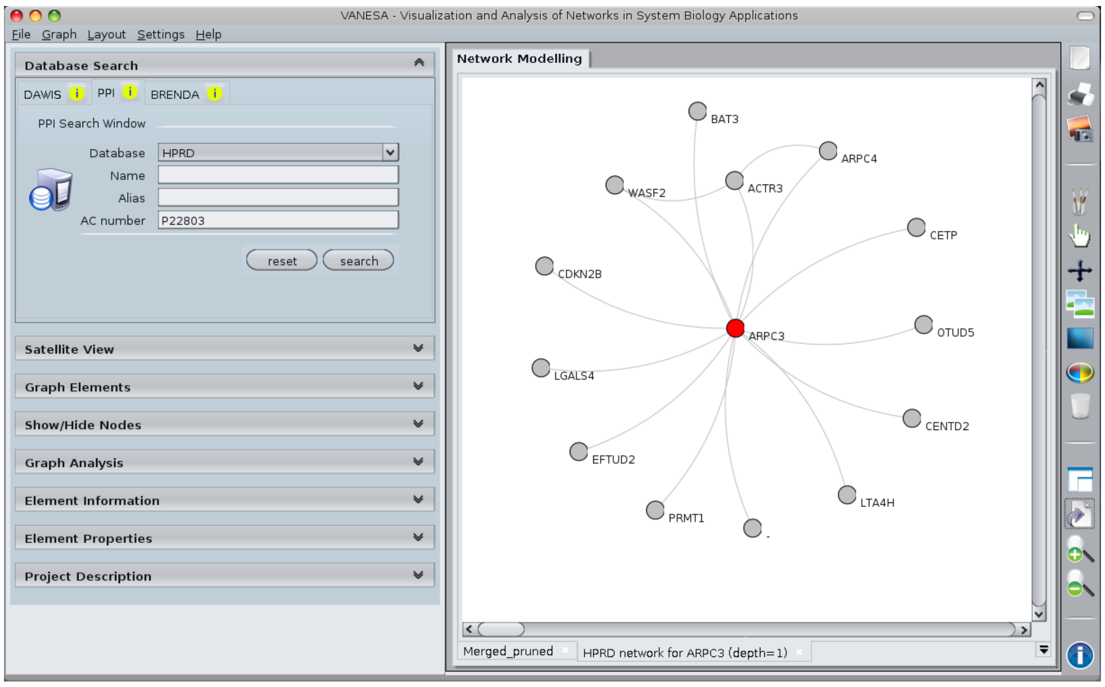


Figure 6.7: System architecture of VANESA system.



Database: HPRD
Name:
Alias:
AC number: P22803

Figure 6.8: Screenshot of VANESA graphical user interface.

The Network modeling workspace displays a graphical representation of the network model. The workspace can be used in three different software modes. The editing mode gives the user the possibility to create and edit biological networks, the picking mode gives the user the facility to pick certain elements in order to view the element properties, and the transforming mode enables the user to move, drag, drop and transform the network within the visualization panel. In the following, the modules are described in detail.

⋆ *Editing mode:* Gives the user the opportunity to create and edit biological network models on the workspace. Furthermore, it is possible to load, import and save any kind of biological network. By clicking the left mouse button on the workspace a creation dialog will appear. Now, the user is able to add a biological element to the network. A relation between two elements can be created by a drag from one element to another. While creating a relation, a dialog will be displayed that gives the user the possibility to specify the kind of relation, such as *Activation, Repression, Phosphorylation, Methylation* and so on. Another feature of the software is the *auto-suggest* function, as already mentioned in the previous section, which supports the user during the element-naming. While typing in the element name, VANESA checks names and synonyms in the external biological databases and the data warehouse. If the input name can be found in one of the data sets, a recommended biological term will be suggested to the researcher. Using this function the researcher should be able to name the biological elements easily, within his defined network, which generally accepted and universalized names from the integrated databases.

Those names are familiar to the scientific community. A biological network that is described by well-defined biological term, is understandable to every biological scientist. Hence, it is easier to discuss and investigate the properties of the network model among different scientists.

- ⋆ *Picking mode:* The user is able to select one or multiple elements of interest from the network. Several picking functions are provided by the software. Moreover, selected elements and their immediate neighbors are emphasized. Each and every selected element, including their neighbors, is highlighted by more intensive colors whereas the remaining elements are pale in their appearance. Once an element or edge is selected, further information spanning multiple attributes can be given in more detail. Therefore, on the left-hand side of the main window an *Element properties panel* is available by which the user can revise the label, the name, the compartment within the cell and of course, the color of the selected element.

- ⋆ *Transforming mode:* The user is able to move, transform and rearrange the graph to get a closer look on elements of interest or a better overview of sub-graphs. Furthermore, the user is able to manipulate, rotate and arrange the graph to his preferred view. The network layout transformations are realized by mathematical matrix operations as described in the previous section. Accordingly, the network can be transformed into any kind of form using these mathematical operations.

The *toolbar* provides the most important functions that are needed in an easily accessible manner. Furthermore, it is possible to switch between the different network modes as describes above.

- ⋆ *Centre button*: The user can easily center the network within the visualization panel.

- ⋆ *Maximization function*: Extends the network modeling workspace, because at times it is necessary to extend the network in order to represent the whole picture of the biological system.

- ⋆ *Zoom-in and zoom-out functions*: The user is able to zoom-in to get a closer view or zoom-out to get a brighter overview of the network. Alternatively, the user can make use of the mouse wheel to zoom.

In summary, these functions make it easier to distinguish elements from each other that may be clustered, not well laid-out or arranged. All these transformation functions to transform the network are based on standard mathematical network manipulations algorithms. That makes it possible to transform a view of a network in a deterministic way that can be changed.

The menu bar contains additional functions that can be applied on the graphical network model. It is possible to save a graph as SBML, GraphML or image file. Moreover, the current viewport of the network modeling workspace can be printed out. Another important function the ability is to highlight certain properties within the network using different graph layout algorithms. At present VANESA provides seven different graph layout algorithms. Each layout has its certain properties and advantages. The use of different layout algorithms might result in a clear and better overview of the network model.

The operating panel contains all relevant functions to characterize and analyze a biological network. It is designed as a so-called "task pane" to collapse and show the available functions. Using this structured design, the user has a clear and well-arranged view of the network without being irritated by too much information. The operating panel contains the following elements.

- ⋆ *Database Search Panel:* The user is able to query KEGG, BRENDA, PPI databases and the DAWIS-M.D. data warehouse for information of interest.
- ⋆ *Satellite View Panel:* The satellite panel displays a minimized view on the network model. The user can roughly navigate through the biological network.
- ⋆ *Graph Elements Panel:* All network elements are listed in alphabetical order, so that the user has an overview of the elements which are available in the network model. The user chooses one element from the list. Then, this children element is highlighted in yellow on the map. All other related elements will consequently be highlighted as well, not necessarily in yellow.
- ⋆ *Show/Hide Nodes Panel:* This panel provides filter options to show and hide biological elements within the system.
- ⋆ *Graph Theory Panel:* In this panel the user is able to visualize the shortest path between two elements.
- ⋆ *Element Information Panel:* User-specific information and database information on the selected element are displayed in detail.
- ⋆ *Element Properties Panel:* The user is able to change the appearance of selected network elements. In addition, it is possible to change the name, label, compartment and the color of the selected element.
- ⋆ *Project Description Panel:* This panel describes the network model and gives information about the author, version and date of the created network.

6.3 Summary

In this chapter the web-based modeling tool *MoVisPP* and a Java-based network editor *VANESA* was discussed. Particularly, the function and the structure were presented in detail as well as the system architecture and implementation. Both systems were implemented based on the concepts in section 6.1. The integration and update process for both systems are realized with the *BioDWH* data warehouse infrastructure.

In section 6.2.1 the MoVisPP application for modeling of biological networks as Petri nets was motivated. Furthermore, the concept of modeling and visualization based on relevant metabolic databases such as KEGG, BRENDA, etc. were presented in detail. Based on this integration process, the system supports semi-automatic generation of hybrid Petri net models. The networks created from MoVisPP can be exported and used by several other applications that support Petri nets such as the Cell Illustrator. Moreover, the tool provides different export possibilities as described in section 6.2.1. Finally, a multiple alignment algorithm for a motif search in MoVisPP was introduced. MoVisPP runs in common web browsers without any additional software platforms such as Java. An easy-to-use web interface makes it possible to create a biological network within two steps. The tool is able to visualize on-the-fly, a clearly-arranged network enriched with integrated external database information. Finally, it is possible to use generated networks in external simulation environments, such as Cell Illustrator, for qualitative and quantitative simulation. An application within the *Cardioworkbench* project will be given in the next chapter.

In section 6.2.2 the development and implementation of the network editor VANESA was introduced in detail. The software allows the modeling and editing of pathways and networks. The major idea of this system is to offer a flexible and easy-to-use molecular network editor to laboratory researchers. Therefore, the biological user is able to edit the project data and knowledge including information from literature. Furthermore, the scientist has the possibility to use the *extending function* of VANESA to realize the semi-automatic extension of their networks. VANESA is based on the DAWIS-M.D. data warehouse which provides some of the most widely-used life science databases. As already mentioned in chapter 5.2.2, the integration of the data was realized with the BioDWH data warehouse infrastructure. The data from the DAWIS-M.D. data warehouse provides an established basis for the modeling and characterization of a biomedical system. An important aspect of visualization is the consideration of multi-dimensional data annotations in a way that it is suitable for the information discovery process.

7 Application Case

This chapter will present two applications for semi-automated reconstruction of biological networks based on the software applications introduced in the previous chapters. First, in section 7.1 an overview of the *Cardioworkbench* project is given, because most of the presented applications and the results are produced within this project. In section 7.2 a system for semi-automatic generation of the correlated hybrid Petri net model [CHH+09] based on the MoVisPP software application will be presented. Furthermore, in section 7.3 a case study of a cardiovascular-disease related gene-regulated biological network based on the VANESA will be also illustrated [KJT+09]. This application of integrative bioinformatics is based on high throughput proteomic data from our partners of the Cardioworkbech project.

7.1 Cardioworkbench - Drug design for Cardiovascular Diseases: Integration of in silico and in vitro Analysis

The *Cardioworkbench* project is an EU-funded project of the "Sixth Framework Programme". The Bioinformatics and Medical Informatics department and participants from Italy, The United Kingdom, Estonia, Austria, Sweden, Finland and Germany are involved in the project.

Recent technological advances have greatly increased the capability to investigate complex molecular interactions that occur at the onset of diseases in order to identify genetic alterations and to screen new drugs. The first critical step of the drug development process is the identification and the validation of drug targets (e. g. proteins, nucleic acids) for therapeutic applications. At present about 500 molecular targets have been identified and validated. The target identification is essential to screen and synthesize new bioactive molecules that can interact with them. Recent advances have greatly expedited the screening of new drugs, however the number of resulting molecules which are not pharmacologically active is still quite large. The optimization of screening depends on the integration of *in silico* and *in vitro* analyses. The screening of targets is extremely difficult for multifactorial diseases, which are characterized by complex synergies between genetic and environmental factors. Cardiovascular diseases (CAD) are one of the major groups of multifactorial diseases and are one of the main causes of mortality and morbidity in most developed countries.

The aim of this project is to improve the target selection/validation process and optimize drug design for cardiovascular diseases. In detail, the project focused on atherosclerosis. In recent years research in this field has acquired a multidisciplinary style, due to the increase in knowledge about cellular and molecular mechanisms involved in the onset and progress of atherosclerosis[1]. Cardiovascular diseases provide a relevant domain for the development of a new multidisciplinary approach to target selection and validation. In the Cardioworkbench project groups with complementary expertise are involved, specifically, computer scientists, clinicians, molecular/cellular biologists and pharmacologists will combine their expertise.

In the following two sections a case study of the cardio-disease related gene-regulated biological network as a result of the Cardioworkbench will be presented. Both case studies are based on the tools presented in the last two chapters.

7.2 Petri Net Models for the Semi-automatic Construction of Biological Networks

Typically the construction of biological and metabolic pathways using common Petri net tools is very time-consuming and manually intensive. On the other hand, a semi-automatic method can construct a Petri net based network using information from different heterogeneous data comparatively faster. Moreover, using MoVisPP for semi-automated networks it is possible to simulate the pathways by using external commercial (e.g. Cell Illustrator) and non-commercial Petri net-based tools (e.g. Pipe2 or WoPeD). This is the main advantage of our approach in comparison to currently available approaches and methods of constructing biological pathways.

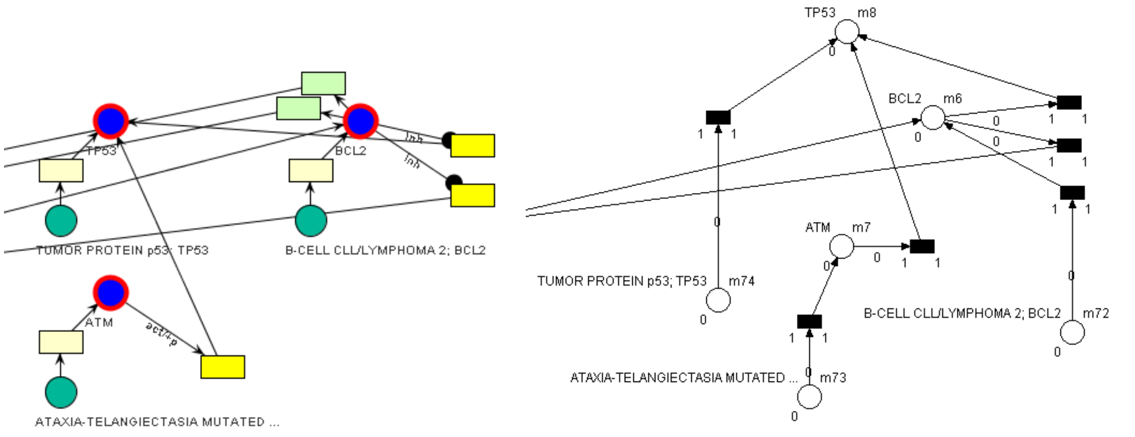
The databases introduced in chapter 3 contain much information about different pathways that are associated with several diseases. In KEGG for instance, pathways are stored as maps that show the connection between the protein, gene, enzyme etc. including the involved reactions. If a KEGG pathway contains all this features, it will be possible to construct a qualitative network model. Therefore, providing the user with a Petri net model of the KEGG-based network taking into account additional information from other databases can be very useful for research and studies of biological networks.

Gene and protein expression profile studies of *Bcl2* and *Tp53*, that have been studied among other things in the Cardioworkbench project, show their role in cardio-vascular diseases. Furthermore, both genes play a major role in neurodegenerative diseases and in apoptosis. A search for these genes in the KEGG database returns a list of the pathways in which they are involved. Figure 2.4 in chapter 2 illustrates an apoptosis pathway map from KEGG that indicates the participation Bcl2 and Tp53 in this pathway. The main dis-

[1]also known as Arteriosclerotic Vascular Disease (ASVD)

advantage of KEGG maps is that they are static and only provide links to the information within the database.

As already mentioned in section 6.2.1, MoVisPP is an easy-to-use web application for generating biological networks as Petri net. It allows the user to choose a pathway from a list of available pathways and then enrich the pathway with additional information to construct a user-specific network. Figure B.3 shows the apoptosis network as Petri net generated by the MoVisPP. As described in section 6.2.1, it is possible to add information about involved pathways, details of the organism, reactions, relations, disease and protein interaction along with the stretch factor as well as the size of transition and places.


TP53
BCL2
inh
inh
TUMOR PROTEIN p53; TP53
B-CELL CLL/LYMPHOMA 2; BCL2
ATM
act+p
ATAXIA-TELANGIECTASIA MUTATED ...
TP53 m8
BCL2 m6
ATM m7
TUMOR PROTEIN p53; TP53 m74
B-CELL CLL/LYMPHOMA 2; BCL2 m72
ATAXIA-TELANGIECTASIA MUTATED ... m73

(a) MoVisPP detail visualization of the apoptosis pathway as Petri net.

(b) Cell Illustrator detail visualization of the apoptosis pathway as Petri net.

Figure 7.1: Graphical representation of KEGG relations as Petri net used in MoVisPP.

In comparison to conventional modeling tools which require quantitative data, such as kinetic parameters etc., the networks generated by MoVisPP are not dependent on the quantitative data. It is designed to allow qualitative modeling (see Figure 7.1) of pathways as Petri nets. Furthermore, the networks created by MoVisPP can be exported and used by several other applications that support Petri nets e.g. Cell Illustrator as described in 6.2.1. To export a network into the Cell Illustrator environment, a CSML based output file is generated. Hence, the generated network when imported into Cell Illustrator, allows a more dynamic, qualitative and user-friendly simulation. Figures 7.1(a) and 7.1(b) illustrates snapshots of a detailed view of the KEGG pathway-based apoptosis network generated by the MoVisPP web application. The illustrated apoptosis network is enriched information from the integrated databases in MoViSPP such as Enzyme, MINT, OMIM, PDB, iProClass, GO and KEGG (see section 6.2.1).

Moreover the Petri net based apoptosis network shown in figure B.3 can be modeled as quantitative network including the kinetic information of biochemical reactions that are partly available in the Brenda database. Finally, in this application, only a qualitative network could be presented, because of the lack of information and data.

7.3 Identification of Altered Molecular Pathways in Cardiomyopathy

Cardiovascular diseases (CVDs) are the leading cause of death in the developed world. Recently much attention has been focused on the metabolic aspects of these multifactorial diseases. The discovery of new CVDs specific molecular targets promoted the investigation of proteins' functional roles in their specific pathways. To evaluate the weight of each trigger factor such as metabolism, hormones, exogenous factors, etc on cardiovascular disease is a complex problem. Typically, epidemiological studies are the starting point for molecular medicine screening. In recent years high-throughput analytical techniques such as DNA chips, protein arrays, and molecular imaging have improved the capability to screen new target genes or proteins. The relations between the cytoplasmic proteome profile and the metabolic pathways (coming from patients of the Cardioworkbench project affected by dilatative cardiomyopathy (DCM)) have been studied in this work. The proteome analysis has been done by using double channel *Clontech Ab Microarray™*. To characterize the specific functional pathways that are dysregulated in each pathological sample, the DAWIS-M.D. and VANESA system were used. Accordingly, differences in significant pathways have been pointed out. Furthermore, the effect of concomitant specific pathologies like diabetes, renal and pulmonary insufficiency on metabolic and genetic pathways alteration has been emphasized.

Typically, genomic, transcriptomic and proteomic analyses produce a large amount of expression profile data for heart and vascular tissues. Furthermore, the availability of new protein microarrays technology has enhanced the capability to screen hundreds and thousands of proteins. The array technology will support research for different fields such as diagnostic, biomarker and drug target discovery. Therefore, the large amount of experimental data can be efficiently managed and analyzed only by comprehensive system biology tools or data warehouses systems. Accordingly, the number of biological databases is increasing significantly as already mentioned in section 2.2. The major challenge of database integration is to combine heterogeneous and multiple data and to bring them into a homogenous and consistent state. The identification of protein-protein and gene-gene interactions is able to support the discovery of novel biomarkers and the identification of unknown and unexplored therapeutic targets. Therefore, the use of proteomic data for pathway modeling can give insight into the biological function of a gene or a protein in its specific genetic or functional pathway. Finally, the integration of proteomic analysis, based on array technologies, using bioinformatics workflows is able to improve molecular medicine diagnostics.

Sample	**Sex**	**Age**	Associated pathology
S1	female	52	Dilatative cardiomyopaty with renal insufficiency
S2	male	52	Dilatative cardiomyopathy and Type 2 diabetes
S3	male	52	Dilatative cardiomyopathy with renal insufficiency and pulmonary disease

Table 7.1: Pathological characteristics of the samples by using proteome analysis.

7.3.1 Materials and Methods

The analysis has been performed on three cytoplasmic samples (S12 fraction) of cultured aortic smooth muscle cells that are extracted from three different dilatative cardiomyopathy patients. Those patients were two males, (59 and 57 years old) and one female (52 years old) person with an established diagnosis of severe atherosclerosis. Therefore, three DCM samples were selected taking into account the presence of overlapping pathologies. Moreover, the influence of additional pathological conditions on dysregulated pathway was investigated. Table 7.1 illustrates the pathological characteristics of the samples by using proteome analysis. The main aim of this work was to investigate the proteomic profile of each sample in order to highlight its associated dysregulated pathways. Figure B.2 shows an image of the Abs-microarray.

Based on the data content of the BioDWH data warehouse, the VANESA software framework was used for modeling and visualization. With the use of VANESA, it was possible to model and visualize the most important pathways based on the proteins in the different microarray samples that are shown in table 7.3. Figure 7.2 illustrates the *Tight junction* signaling pathway with the *multiple PDZ domain protein* as a relevant protein from the first sample (S1). In addition to the general pathway, the network is enriched with direct protein-protein interactions based on database information of the data warehouse. Besides the *Tight junction* signaling pathway, the pathway of the *Regulation of the actin cytoskeleton* based on sample 2 in figure 7.3 and the *Calcium signaling pathway* based on sample 3 (figure 7.4) were also predicted by VANESA software.

7.3.2 Results

Based on the previous considerations, 36 KEGG pathways having a correspondence with a single dysregulated protein screened by Abs-microarray have been identified. Table 7.2 shows the relative number of pathways in which a single dysregulated protein is involved for each sample. Hence, previously defined pathways have been identified by intersection as potentially disordered pathways in all samples. Then, the pathways specifically dysregulated in each sample have been identified. There is a set of pathways which possess the highest specificity for being targeted. In fact, in a pharmacological perspective a protein involved in multiple pathways could not be a DCM specific dysregulated element and its targeting can trigger the emergence of a specific side effects. This analysis enables the

Sample	**Number***	**Associated pathology**
S1	23/36	Dilatative cardiomyopaty with renal insufficiency
S2	24/36	Dilatative cardiomyopathy and Type 2 diabetes
S3	23/36	Dilatative cardiomyopathy with renal insufficiency and pulmonary disease

Table 7.2: The table outlines for each sample the relative number of pathways in which a single dysregulated protein is involved.
(*Number of biunivocal correspondences single pathway-single protein)

determination of key elements that are dysregulated in each idiotypic proteomic sample.

The analysis of Abs-microarrays differs from transcript array analysis in term of fluorescence range. In fact, the fluorescence intensity of the Abs-microarrays has shown a narrow range of variability. This situation has affected the standard statistical procedures normally used for the exploratory analysis of transcript microarrays. Table 7.3 gives an outline names and of the Swissprot codes of the proteins, screened by Abs-microarray, involved in the common KEGG functional networks.

According to the list of proteins mapped onto Abs-microarray, the functional pathways common to all our samples are associated with the following cellular functions.

1. Genetic Information Processing (three out of twelve)
2. Cellular Processing (four out of twelve)
3. Environmental Information Processing (two out of twelve)
4. Metabolism (two out of twelve)
5. Human-disease specific pathway (one pathway).

Finally, the high-throughput proteomic data has pointed out a general framework of functional dysregulations associated with DCM in the presence of concomitant pathologies. Our approach can be useful to support clinicians in order to select the more appropriate therapy for a patient.

7.4 Summary

Data integration is one of the major problems in bioinformatics that primarily addresses the heterogeneity and increasing volume of information in biological databases. Based on life science data from different molecular biology databases and microarray experiments a comprehensive system for integration was built. This software system is based on BioDWH infrastructure for data integration.

Pathway	Nr	Protein names	Swissprot code
hsa00350 Tyrosin metabolism	1	catechol-O-methyltransferase	P21964
hsa03022 Basal Transcription factor	2	general transcription factor II, I general transcription factor IIB	O15359 Q00403
hsa03060 Protein Export	1	signal recognition particle 54kDa	P13624
hsa04010 MAPK signalin pathway	1	arrestin, beta 1	P49407
hsa04110 Cell cycle	4	cyclin A1 MCM6 minichromosome maintenance deficient 6 (MIS5 homolog, S. pombe) (S. cerevisiae) cyclin-dependent kinase 7 (MO15 homolog, Xenopus laevis, cdk-activating kinase) cyclin-dependent kinase inhibitor 1C (p57, Kip2)	P20248 Q14566 P50613 P49918
hsa4120 Ubiquitin mediated proteolysis	1	ubiquitin-conjugating enzyme E2I (UBC9 homolog, yeast)	P50550
hsa04510 Focal adhesion	1	caveolin 1, caveolae protein, 22kDa	Q03135
hsa04612 Antigen processing and Presentation	2	heat shock 90kDa protein 1, alpha calnexin	P07900 P27824
hsa05222 Small cell lung cancer	1	TNF receptor-associated factor 4	Q14848
hsa00860 Porphyrin and chlorophyll Metabolism	1	heme oxygenase (decycling) 1	P09601
hsa04080 Neuroactive ligand-receptor interaction	1	nuclear receptor subfamily 3, group C, member 1 (glucocorticoid receptor)	P04150
hsa04910 Insulin signaling pathway	1	flotillin 2	Q14254

Table 7.3: The table outlines for each sample the relative number of pathways in which a single dysregulated protein is involved.

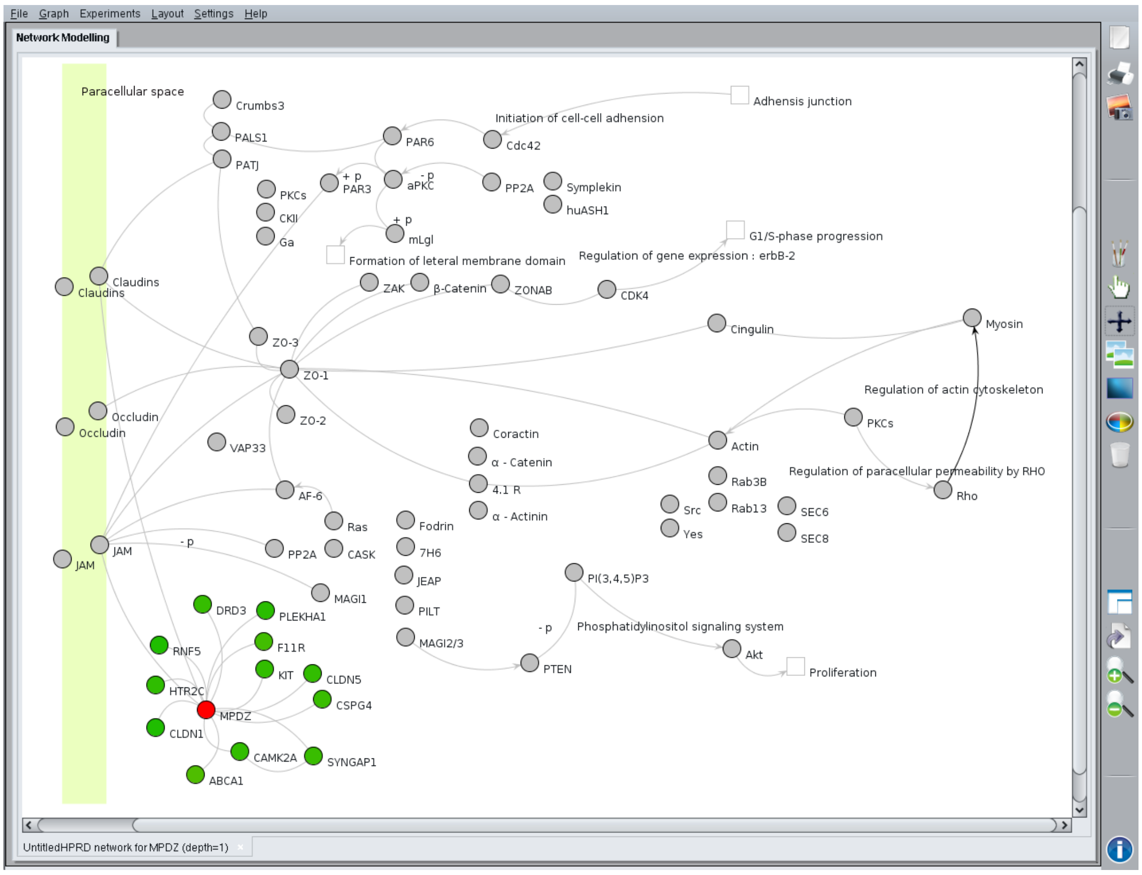


Figure 7.2: Visualization of the Tight junction signaling pathway (hsa04530) by VANESA. The red-marked place is the relevant protein of the microarray sample.

Based on the data integration system the application of Petri nets for modeling and simulation of biological networks (MoVisPP) in section 7.2 was motivated. The system supports semi-automatic generation of the correlated hybrid Petri net model. A case study of the cardiovascular-disease related gene-regulated biological network based on results of the Cardioworkbench project was also presented. The work was focused on the research of the project, which presents a novel tool for the automatic generation of models based on the relevant databases (e.g. KEGG, BRENDA) and experimental microarray data. As discussed earlier, this cannot be done automatically because the relevant kinetic data (catalyses or gene regulation) is not available or complete.

Furthermore, in section 7.3 an approach for modeling and analysis of biological networks was presented. Based on the BioDWH approach the DAWIS-M.D. for web-based data access and VANESA for modeling and analyzing biological network data were motivated. Therefore, to explore and compare the heterogeneous datasets provided by the DAWIS-M.D. system the software application VANESA was implemented. VANESA provides different bioinformatics methods and visualization approaches to analyze biological and protein-protein interaction networks. The data from the DAWIS-M.D. system is ana-

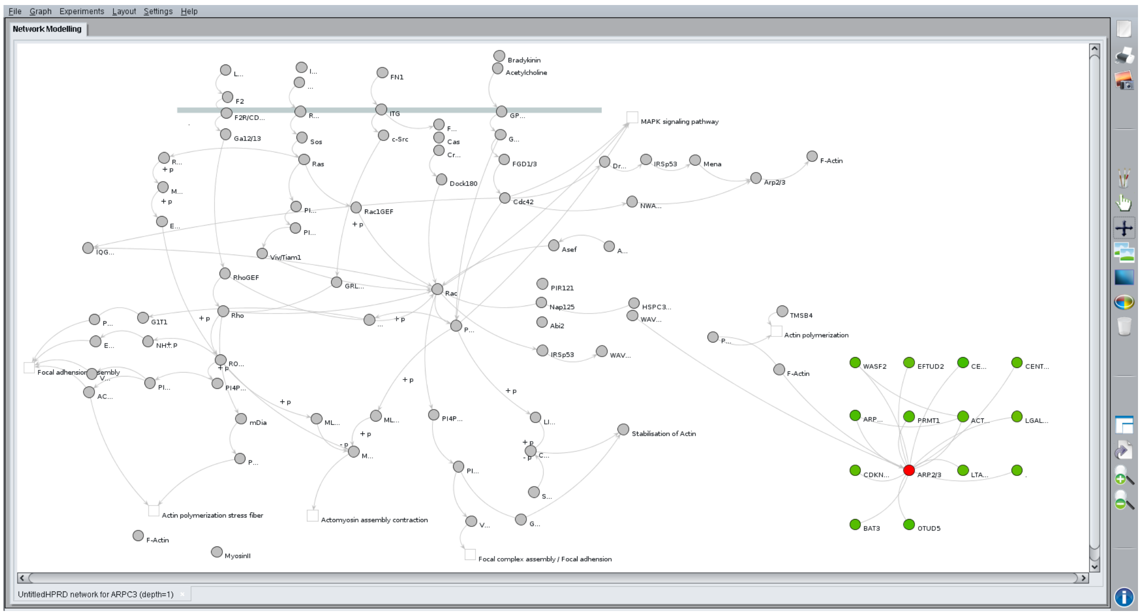


Figure 7.3: Visualization of the regulation of the actin cytoskeleton (hsa04810) based on microarray sample 2. The red-marked place is the relevant protein on the microarry sample.

lyzed on a large scale and visualized in a biological manner. The important aspect of the visualization is to consider multi-dimensional data annotations in a clear and convenient way, suitable for the knowledge discovery. With the software application it was possible to trim down the data to a manageable size. Moreover, it was possible to analyze and identify new interaction patterns. Summarizing, the integrative bioinformatics approach in section 7.3 on high-throughput proteomic data has pointed out, for the first time, a general framework of functional dysregulations associated with DCM in the presence of concomitant pathologies. This proposed integrative approach can be a helpful support to clinicians in order to select the more appropriate therapy for a patient, taking a wide survey of metabolism into account.

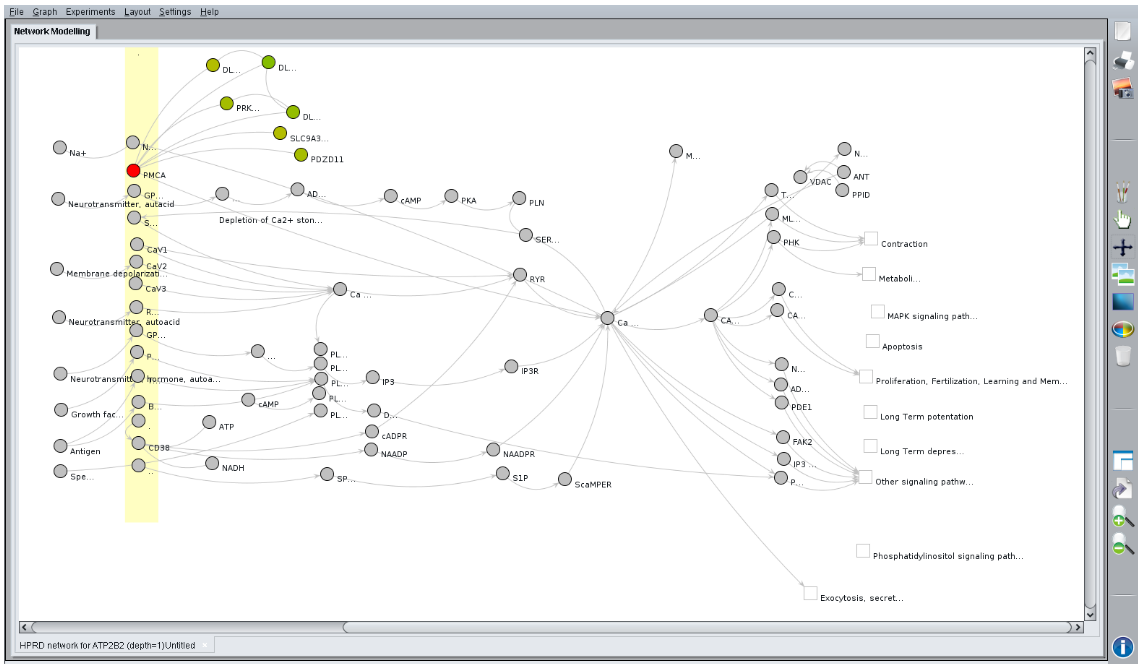


Figure 7.4: Calcium signaling pathway (hsa04020) based on sample 3 visualized by VANESA. The red marked place is the relevant protein on the microarry sample.

8 Conclusion

Today, in life science and biological research different database and information systems are utilized to give insight into diverse biological aspects, starting from genomic sequences via gene regulatory networks up to clinical data. For a comprehensive and efficient usage of this important data it is necessary to integrate the distributed and heterogeneous data and provide them for further analysis to the researcher. Beside data integration the user needs to be supported by applicable tools for navigation within the integrated data sets that support an efficient and precise processing of the data.

The number of molecular databases has been continuously increasing in the last decade. Today, approximately 1170 publicly available databases and information systems for life science data are listed in the NAR catalogue [GC09], whereas research groups usually use several data sources simultaneously. Therefore, data integration is necessary to provide a consistent view on the distributed, heterogeneous and redundant data. Typically, problems in data integration are different heterogeneities on different levels as described in section 2.2.1.

Classical requirements of integration such as transparency, integrity, semantic correctness and non-redundancy are important. However, other requirements gain importance in life science data integration, such as an efficient access to the increasing amount of data which should be, but is not always up-to-date. In addition, solutions for complex and changing schemata in life science data are required. These requirements are illustrated by several integration approaches. The use of federated database systems is augmented by data warehouses, which are gaining wider usage. But until now, none of these approaches, such as Columba [TRM$^+$05], BioWarehouse [LPW$^+$06] or ONDEX [KBT$^+$06] has become widely accepted.

The already existing life science databases and available DWH approaches in the field of database integration and the still unanswered questions in the development of such systems are the starting point of this work. Together with the partners from the *Cardioworkbench* project (see section 7.1) several molecular biological databases were analyzed and investigated. Therefore, widely used life science databases such as BRENDA, EMBL, GO, IntAct, KEGG, MINT, OMIM, PubChem, SCOP, Transfac, Transpath and UniProt are necessary for research. By using the information from different databases it is possible to model, investigate and analyze complex biological systems on different levels and on diverse biological aspects.

Accordingly, the challenge was to combine diverse and multiple data and to bring them into a homogenous, consistent state. Consequently, the new system should be flexible and also applicable in general for any other project. As discussed in chapter 3 none of the existing data warehouse approaches meet these requirements. For that purpose, *BioDWH* data warehouse software kit is developed as a Java-based open source application for building life science data warehouses using common relational database management systems. By using the object-relational mapping (ORM) technology it is no longer necessary to select the local database management system based on the restrictions of the integration software. BioDWH provides a number of ready-to-use parsers to extract data from public life science data sources and to store the content in a data warehouse. The integration process is supported by an easy-to-use graphical user interface that makes it possible to integrate any supported database in a few steps into a local database. Hence, it enables the configuration of the monitor and parser for the different public life science data sources as well as the local database management system. In comparison to other data warehouse systems, such as Atlas and Columba, BioDWH is not restricted to a specific biological question. Moreover, by using object-relational mapping the system is not restricted to a specific database management system such as MySQL, PostgreSQL or Oracle. BioWarehouse and Atlas need a local installation on a UNIX/Linux-based operation system that is very time-consuming. By using Java Runtime Environment the BioDWH system is platform independent. Most of these data warehouse approaches are not up-to-date or need to be manually updated. Hence, a monitoring utility is needed to keep the system up-to-date. Therefore, the BioDWH architecture provides a monitoring component that observes multiple data sources, so that different update strategies are configurable. The monitor component is configured using XML.

Now, the accurate representation of the integrated research data as well as experimental microarray data in a user-friendly format is a major task in computer science. Information must be visualized in a clear and understandable manner to support research activities. Otherwise important information can get lost. None of the presented data warehouse approaches in chapter 3 meet the aims of these requirements. After a discussion with associated partners from the Cardioworkbench project, it was determent that the most important task is on the one hand, the data integration. And on the other hand the development of a graphical user interface by which the scientist is able to navigate and browse through the data. Therefore, *CardioVINEdb* was presented in section 5.2.1, a data warehouse approach developed to browse and explore life science data. The data warehouse architecture provides a platform independent web interface that can be used with any common web browser. The system enables intuitive search of integrated life science data, simple navigation to related information as well as visualization of biological domains and their relationships. To ensure maximum up-to-dateness of the integrated data the BioDWH data warehouse infrastructure including a monitor component is used. Furthermore, the common web-based user interface provides a "static" and "dynamic" visualization component that allows interactive exploration of the integrated data. In comparison to other systems, such as Atlas, Columba and Systomonas, the key task was not only to integrate and manage the data from different sources by creating a data warehouse, but also to create a method for visually representing the integrated data in a simple way to understand the underlying biological complexity with ease.

Software solutions which provide visualization, analysis services and an information management framework are in high demand among scientist as already discussed. It is not surprising that many groups the world over have contributed to the task of developing such software frameworks. But typically, all these software approaches focus on one particular problem. And none is powerful enough to address to all problems. In addition, no software solution is able to handle and analyze the integrated data sets from our biological experiments and different databases. Therefore, a DWH system to search integrated life science data and simple navigation called *DAWIS-M.D.* and a system for modeling and visualization called *VANESA* were implemented. The DAWIS-M.D. data warehouse incorporates the advantage of a navigation and information system, such as CardioVINEdb, and builds a bridge to the network editor approach VANESA. Hence, it is possible to browse through the integrated life science data and bring the information directly into a modeling and visualization system. VANESA is a software framework for visualizing and modeling biological networks. The user is able to create a user-specific pathway without any restrictions. In contrast to other software solutions, the user can make use of a wide set of biological elements and relations to create his own biological system. Almost all relevant biological elements and relations can be represented and visualized by VANESA. Moreover, the editor consults the data warehouse DAWIS-M.D. that is based on the data warehouse infrastructure BioDWH for biomedical information. BioDWH provides a wide range of biomedical data sources and continuously initiates updates and changes. This database information is helpful to identify relationships between genes, proteins, chemical compounds and other molecules that can occur in cells. The consulting of the KEGG and BRENDA databases as well as the DAWIS-M.D. data warehouse can give additional information on biological phenomena and unknown elements. Consequently, it could aid the scientist to a better understanding of the biological pathway or network. In comparison to other software tools, such as Cytoscape or VisANT, VANESA is able to formulate complex queries on the data sources in order to find specific information and data sets. Furthermore, it is possible to save and load networks in SBML files. SBML files are widely used and accepted as a standard data language for biological network models. Hence, many software approaches in this field make use of the SBML data format.

As already discussed in the previous paragraph, modeling and visualization is a major task in bioinformatics. Therefore, in section 6.2.1 the web-based *MoVisPP* application for the modeling of biological networks as Petri nets was motivated. Based on a data integration process by using the BioDWH infrastructure, the system supports semi-automatic generation of hybrid Petri net models. The web application consists of a data warehouse approach spanning multiple databases containing biochemical and metabolic information from Enzyme, MINT, OMIM, PDB, iProClass, GO and KEGG. The networks created from MoVisPP can be exported and used by several other applications that support Petri nets such as the Cell Illustrator. Furthermore, MoVisPP provides a multiple alignment algorithm for motif search in KEGG pathways. In comparison to other tools, MoVisPP runs in common web browsers without any additional software platforms such as Java.

An easy-to-use web interface makes it possible to create a biological network as Petri net within two steps. Then, it is possible to use generated networks in external simulation environments for qualitative and quantitative simulation.

Finally, two case studies ([KJT+09],[CHH+09]) of the cardio-disease related gene-regulated biological networks based on results of the Cardioworkbench project were presented. The work was focused on the current research of the project, which uses the approaches presented in this thesis, for the semi-automatic construction of biological networks based on widely used life science databases and experimental microarray data. Figure 8.1 shows an overall picture of the presented software applications in this thesis and how they are related to each other. However, by using these tools it was possible to analyze and identify new interaction patterns as well as new pathway models and networks. Finally, these proposed integrative approaches can be helpful in supporting scientists to select more appropriate pathways, taking a wide survey of metabolisms into account.

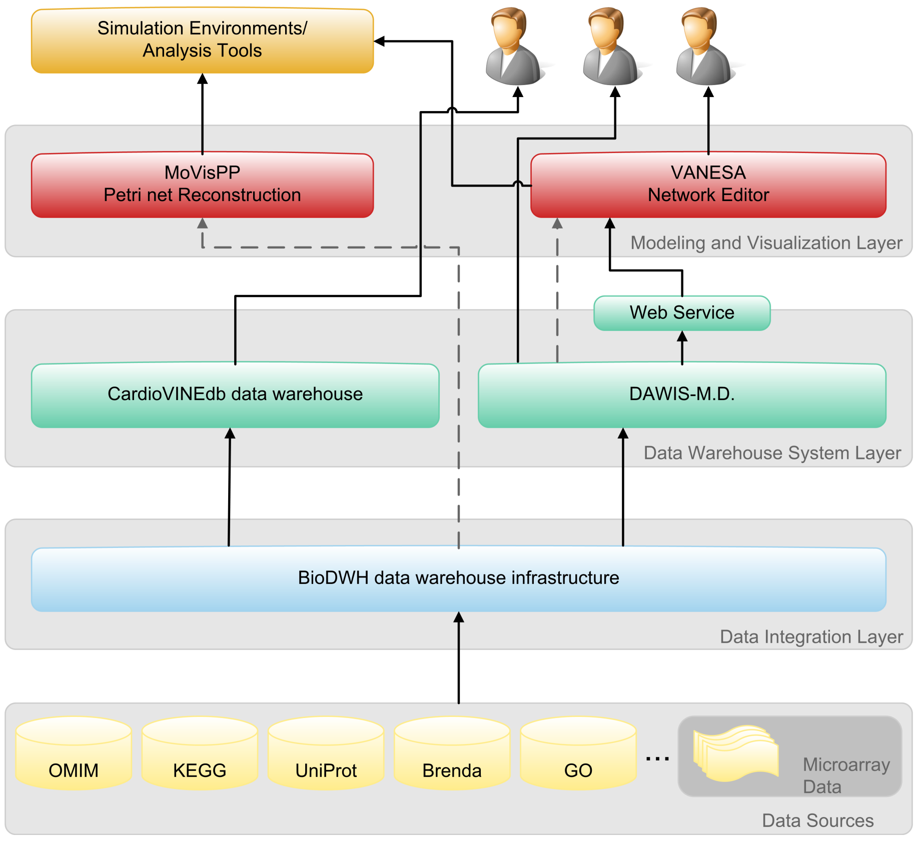

Simulation Environments/ Analysis Tools
MoVisPP
Petri net Reconstruction
VANESA
Network Editor
Modeling and Visualization Layer
Web Service
CardioVINEdb data warehouse
DAWIS-M.D.
Data Warehouse System Layer
BioDWH data warehouse infrastructure
Data Integration Layer
OMIM
KEGG
UniProt
Brenda
GO
...
Microarray Data
Data Sources

Figure 8.1: Overview of the applications for the reconstruction of biological networks based on life science data integration and their relation.

In the future, additional databases could be integrated into the different systems. Therefore, it is necessary to develop new parser for the BioDWH toolkit. Hence, the data warehouse infrastructure becomes more interesting for projects in different research fields.

Then it will be easier to integrate additional information into the DAWIS-M.D. information system as well as to the VANESA network editor. This leads to more comprehensive knowledge in different topics and more complex biological networks as well. It is also planned to integrate text mining approaches and full a text indexing system in the near future. The idea behind the text mining approach is to filter the large amount of data in order to gather the most relevant information. The focus lies on the literature database *PubMed*. PubMed contains over 18 million citations from *MEDLINE* and other life science journals. Furthermore, it is planned to automatically integrate PCR and microarray data into a local database. With this feature the user should have the possibility to browse, analyze and store experimental data in a data warehouse. Furthermore, the researcher is able to include experimental data directly into his network model by using VANESA. Finally, other research groups from Bielefeld University and external groups will be invited to join our project on data integration, life science information systems and network modeling and visualization.

In conclusion, this work presents a powerful and flexible data warehouse infrastructure *BioDWH* that can be used for building project-specific information systems, such as *CardioVINEdb*, as well as general data warehouse systems like *DAWIS-M.D.*. Furthermore, the system is the basis for the network modeling application *VANESA* and the pathway reconstruction tool *MoVisPP*. Finally, this work shows the semi-automated reconstruction of biological networks based on life science data integration supported by the presented tools. The presented applications are already in use within the *Cardioworkbench* project as well as in ongoing in-house projects.

Acknowledgment

The present thesis come into existence during my work at the Technical Faculty at Bielefeld University in the context of the *Cardioworkbench* project which was funded by "Sixth Framework Programme" of the European Union.

This work could not have been carried out without the support of many people. First of all, I would like to thank Prof. Dr. Ralf Hofestädt for supervising and many of helpful discussions. My sincere gratitude also goes to Prof. Dr. Stefan Conrad from the Institute of Computer Science at Heinrich-Heine-Universität Düsseldorf for external review of the thesis. Furthermore, I would like to express thanks to Prof. Dr. Philipp Cimiano for leading the examination board and Dr. Stefan Kopp for participating the examination board. Moreover, sincere thanks go to Dr. Thoralf Töpel for the close collaboration, helpful discussions and proof-reading of the manuscript. Barbara Davis and Joe Jungeblodt corrected the English of the thesis very carefully.

I wish to acknowledge my colleagues at the Bioinformatics/Medical informatics department David Braun, Björn Sommer, Sridhar Hariharaputran, Hang-Mao Lee, Aljona Oprawchat, Alban Shoshi, Klaus Kulitza and Sabine Klusmann for their help and collaboration. In addition, I want to thank my former students Sarah Spangardt, Corinna Fietz, Andreas Klassen and Nils Oppermann for their work. Moreover, I am deeply grateful to Klaus Hippe and Sebastian Janowski for the close collaboration and their help in implementing the different software tools that are part of this thesis.

Most of all, I want to thank my parents, Friedhelm and Dietlinde, for giving me the opportunity to study and for all their support and encouragement. And of course, my sister Kristina and my grandmother Erna as well.

A Useful WWW Links

Following overview shows useful links of presented life science data sources, molecular biology data warehouses, modeling and visualization tools as well as links of used frameworks.

Life science data sources

Biocarta	http://www.biocarta.com/
BRENDA	http://www.brenda-enzymes.org/
DBGET	http://www.genome.jp/dbget/
EMBL-Bank	http://www.ebi.ac.uk/embl/
Enzyme	http://www.expasy.ch/enzyme/
GO	http://www.geneontology.org/
HPRD	http://www.hprd.org/
iProClass	http://pir.georgetown.edu/iproclass/
IntAct	http://www.ebi.ac.uk/intact/
Kyoto Encyclopedia of Genes and Genomes	http://www.genome.ad.jp/kegg/
MINT	http://mint.bio.uniroma2.it/mint/
OMIM	http://www.ncbi.nlm.nih.gov/omim/
PDB	http://www.rcsb.org/
PubChem	http://pubchem.ncbi.nlm.nih.gov/
SCOP	http://scop.mrc-lmb.cam.ac.uk/scop/
UniProt	http://www.uniprot.org/
TRANSFAC®/TRANSPATH®	http://www.biobase-international.com/
	http://www.gene-regulation.com/pub/databases.html

Life science data warehouses and information systems

Atlas	http://bioinformatics.ubc.ca/atlas/
BioRS	http://www.biomax.de/products/biors.php
BioWarehouse	http://biowarehouse.ai.sri.com/
Columba	http://www.columba-db.de/
CoryneRegNet	http://www.coryneregnet.de/
DiscoveryLink	http://www.ibm.com/
Entrez	http://www.ncbi.nlm.nih.gov/Entrez/
ONDEX	http://www.ondex.org/
SYSTOMONAS	http://www.systomonas.de/
TAMBIS	http://www.cs.man.ac.uk/ stevensr/tambis/index.html

Modeling and simulation environments

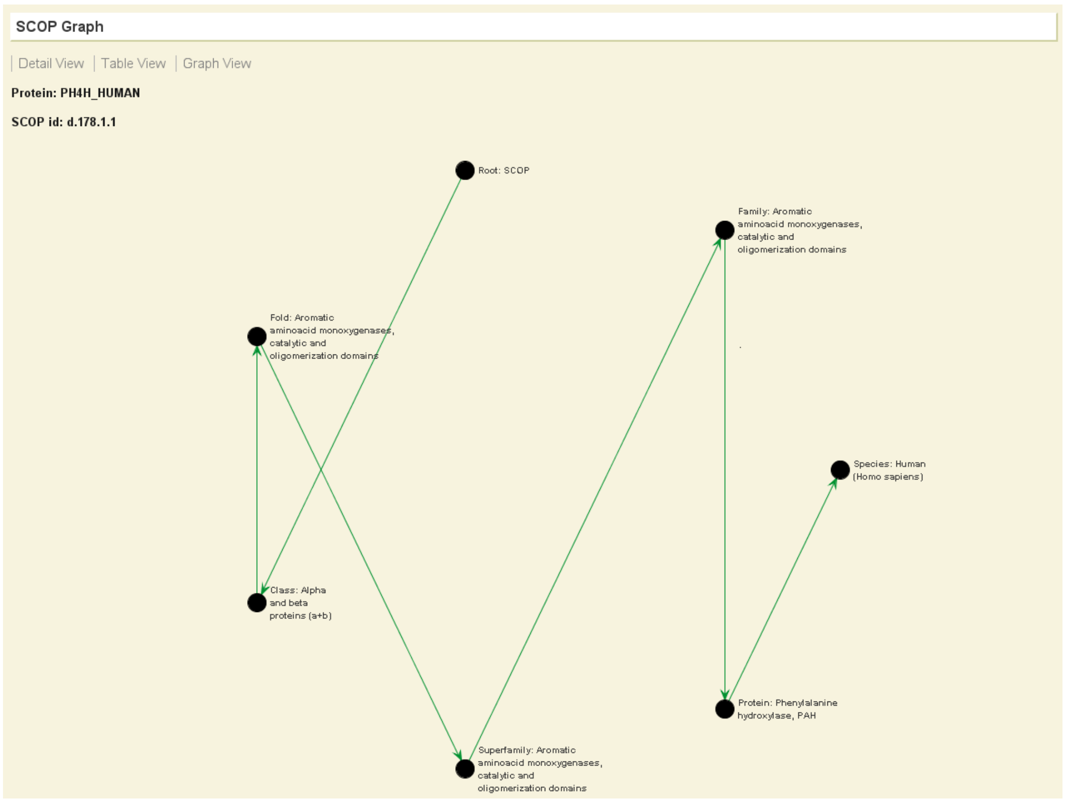
Cell Illustrator	http://www.cellillustrator.com/
Cytoscape	http://www.cytoscape.org/
Renew	http://www.renew.de/
VisANT	http://visant.bu.edu/
VANTED	http://vanted.ipk-gatersleben.de/
WoPeD	http://www.woped.org/

Implemented systems

BioDWH	http://sourceforge.net/projects/biodwh/
CardioVINEdb	http://agbi.techfak.uni-bielefeld.de/CardioVINEdb/
DAWIS-M.D.	http://agbi.techfak.uni-bielefeld.de/DAWISMD/
MoVisPP	http://agbi.techfak.uni-bielefeld.de/movispp/
VANESA	http://sourceforge.net/projects/vanesa/

B Supplementary Material

Supplementary material of chapter 5:


SCOP Graph

Protein: PH4H_HUMAN

SCOP id: d.178.1.1

Detail View
Table View
Graph View
Root: SCOP
Family: Aromatic aminoacid monoxygenases, catalytic and oligomerization domains
Fold: Aromatic aminoacid monoxygenases, catalytic and oligomerization domains
Species: Human (Homo sapiens)
Class: Alpha and beta proteins (a+b)
Protein: Phenylalanine hydroxylase, PAH
Superfamily: Aromatic aminoacid monoxygenases, catalytic and oligomerization domains

Figure B.1: Protein classification of the enzyme PAH in DAWIS-M.D. using SCOP hierarchy.

Supplementary material of chapter 7:

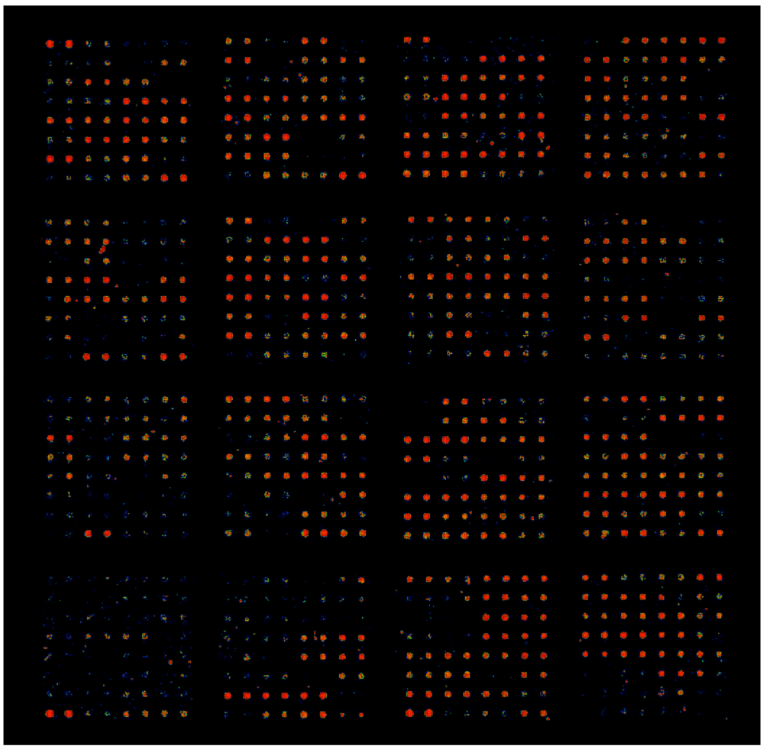


Figure B.2: Example of Antibody microarray results as obtained from SMC purified from human aorta biopsy.

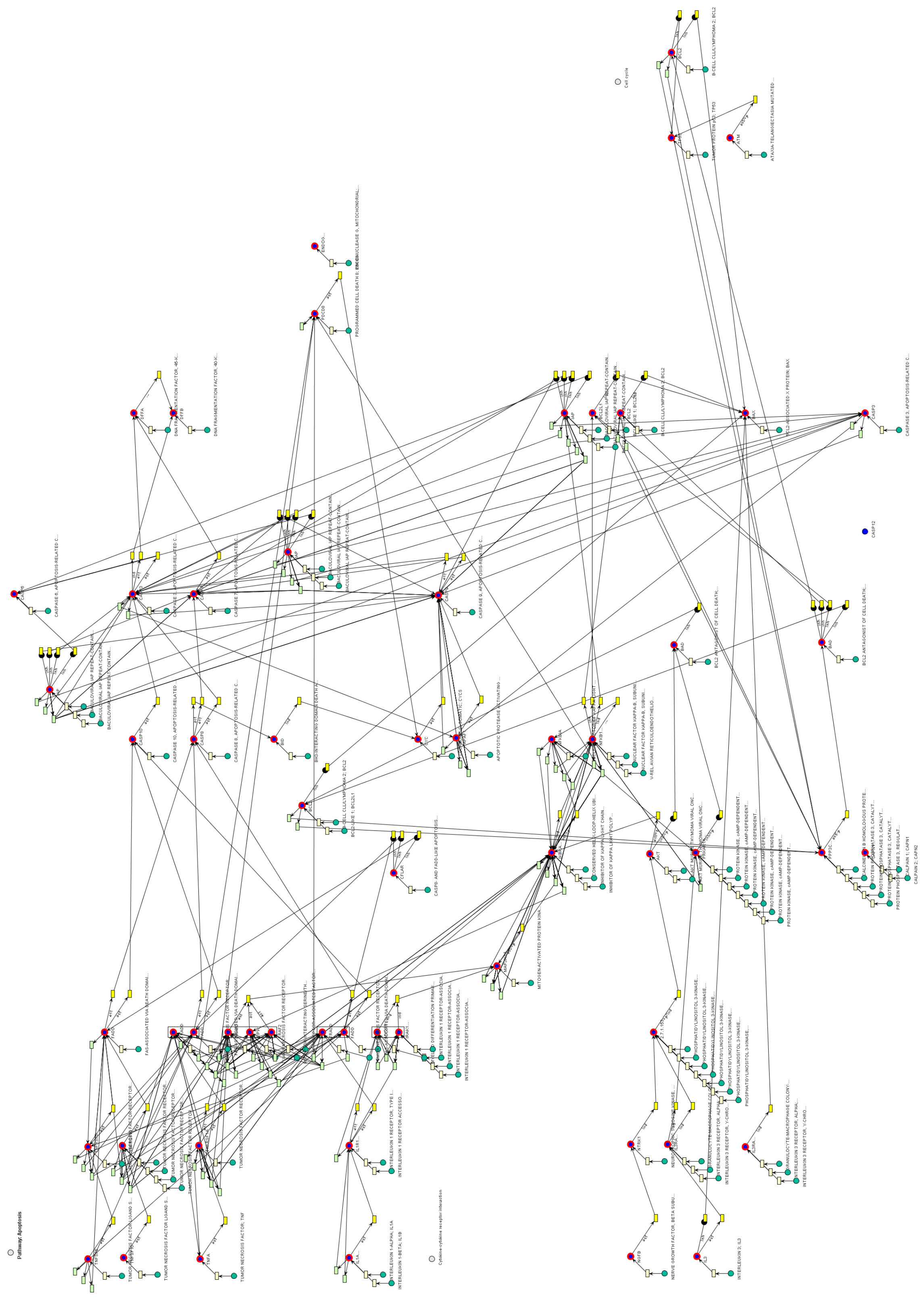


Figure B.3: Visualization of the Apoptosis pathway (hsa04210) visualized by MoVisPP. A full picture can be found at http://agbi.techfak.uni-bielefeld.de/movispp/publication/MoVisPP_Apoptosis.png.

Bibliography

[ABW$^+$04] R. Apweiler, A. Bairoch, C. H. Wu, W. C. Barker, B. Boeckmann, S. Ferro, E. Gasteiger, H. Huang, R. Lopez, M. Magrane, M. J. Martin, D. A. Natale, C. O'Donovan, N. Redaschi, and L. S. Yeh. UniProt: the Universal Protein knowledgebase. *Nucleic Acids Res.*, 32:D115–119, Jan 2004.

[AD98] H. Alla and R. David. Continuous and hybrid petri nets. *Journal of Circuits Systems Computers*, 8(1):159–188, 1998.

[Bai00] A. Bairoch. The ENZYME database in 2000. *Nucleic Acids Res.*, 28:304–305, Jan 2000.

[Bar97] A. J. Barrett. Nomenclature Committee of the International Union of Biochemistry and Molecular Biology (NC-IUBMB). Enzyme Nomenclature. Recommendations 1992. Supplement 4: corrections and additions (1997). *Eur. J. Biochem.*, 250:1–6, Nov 1997.

[Bau07] J. Baumbach. CoryneRegNet 4.0 - A reference database for corynebacterial gene regulatory networks. *BMC Bioinformatics*, 8:429, 2007.

[BBA$^+$09] A. Bairoch, L. Bougueleret, S. Altairac, V. Amendolia, A. Auchincloss, G. Argoud-Puy, K. Axelsen, D. Baratin, M. C. Blatter, B. Boeckmann, J. Bolleman, L. Bollondi, E. Boutet, S. B. Quintaje, L. Breuza, A. Bridge, E. deCastro, L. Ciapina, D. Coral, E. Coudert, I. Cusin, G. Delbard, D. Dornevil, P. D. Roggli, S. Duvaud, A. Estreicher, L. Famiglietti, M. Feuermann, S. Gehant, N. Farriol-Mathis, S. Ferro, E. Gasteiger, A. Gateau, V. Gerritsen, A. Gos, N. Gruaz-Gumowski, U. Hinz, C. Hulo, N. Hulo, J. James, S. Jimenez, F. Jungo, V. Junker, T. Kappler, G. Keller, C. Lachaize, L. Lane-Guermonprez, P. Langendijk-Genevaux, V. Lara, P. Lemercier, V. Le Saux, D. Lieberherr, T. d. e. O. Lima, V. Mangold, X. Martin, P. Masson, K. Michoud, M. Moinat, A. Morgat, A. Mottaz, S. Paesano, I. Pedruzzi, I. Phan, S. Pilbout, V. Pillet, S. Poux, M. Pozzato, N. Redaschi, S. Reynaud, C. Rivoire, B. Roechert, M. Schneider, C. Sigrist, K. Sonesson, S. Staehli, A. Stutz, S. Sundaram, M. Tognolli, L. Verbregue, A. L. Veuthey, L. Yip, L. Zuletta, R. Apweiler, Y. Alam-Faruque, R. Antunes, D. Barrell, D. Binns,

L. Bower, P. Browne, W. M. Chan, E. Dimmer, R. Eberhardt, A. Fedotov, R. Foulger, J. Garavelli, R. Golin, A. Horne, R. Huntley, J. Jacobsen, M. Kleen, P. Kersey, K. Laiho, R. Leinonen, D. Legge, Q. Lin, M. Magrane, M. J. Martin, C. O'Donovan, S. Orchard, J. O'Rourke, S. Patient, M. Pruess, A. Sitnov, E. Stanley, M. Corbett, G. di Martino, M. Donnelly, J. Luo, P. van Rensburg, C. Wu, C. Arighi, L. Arminski, W. Barker, Y. Chen, Z. Z. Hu, H. K. Hua, H. Huang, R. Mazumder, P. McGarvey, D. A. Natale, A. Nikolskaya, N. Petrova, B. E. Suzek, S. Vasudevan, C. R. Vinayaka, L. S. Yeh, and J. Zhang. The Universal Protein Resource (UniProt) 2009. *Nucleic Acids Res.*, 37:D169–174, Jan 2009.

[BBC$^+$06] J. Baumbach, K. Brinkrolf, L. F. Czaja, S. Rahmann, and A. Tauch. CoryneRegNet: an ontology-based data warehouse of corynebacterial transcription factors and regulatory networks. *BMC Genomics*, 7:24, 2006.

[BG04] A. Bauer and H. Günzel. *Data-Warehouse-Systeme.* dpunkt-Verl., Heidelberg, 2004.

[BK07] Christian Bauer and Gavin King. *Java Persistence mit Hibernate.* Hanser, München [u.a.], 2007.

[CFG$^+$05] K. Cartharius, K. Frech, K. Grote, B. Klocke, M. Haltmeier, A. Klingenhoff, M. Frisch, M. Bayerlein, and T. Werner. MatInspector and beyond: promoter analysis based on transcription factor binding sites. *Bioinformatics*, 21:2933–2942, Jul 2005.

[CH04] M. Chen and R. Hofestädt. PathAligner: metabolic pathway retrieval and alignment. *Appl. Bioinformatics*, 3:241–252, 2004.

[CH08] A. Crombach and P. Hogeweg. Evolution of evolvability in gene regulatory networks. *PLoS Comput. Biol.*, 4:e1000112, 2008.

[CHH$^+$09] M. Chen, S. Hariharaputran, R. Hofestädt, B. Kormeier, and S. Spangardt. Petri net models for the semi-automatic construction of large scale biological networks. *Natural Computing*, 2009.

[CKK$^+$04] C. Choi, M. Krull, A. Kel, O. Kel-Margoulis, S. Pistor, A. Potapov, N. Voss, and E. Wingender. TRANSPATH-A High Quality Database Focused on Signal Transduction. *Comp. Funct. Genomics*, 5:163–168, 2004.

[CM95] I.-Min A. Chen and Victor M. Markowitz. An overview of the object protocol model (opm) and the opm data management tools. *Inf. Syst.*, 20(5):393–418, 1995.

[CML$^+$07] C. Choi, R. Munch, S. Leupold, J. Klein, I. Siegel, B. Thielen, B. Benkert, M. Kucklick, M. Schobert, J. Barthelmes, C. Ebeling, I. Haddad, M. Scheer, A. Grote, K. Hiller, B. Bunk, K. Schreiber, I. Retter, D. Schomburg, and

D. Jahn. SYSTOMONAS–an integrated database for systems biology analysis of Pseudomonas. *Nucleic Acids Res.*, 35:D533–537, Jan 2007.

[Con97] S. Conrad. *Föderierte Datenbanksysteme - Konzepte der Datenintegration*. Springer, Berlin [u.a.], 1997.

[CSG^{+}09] A. Chang, M. Scheer, A. Grote, I. Schomburg, and D. Schomburg. BRENDA, AMENDA and FRENDA the enzyme information system: new content and tools in 2009. *Nucleic Acids Res.*, 37:D588–592, Jan 2009.

[Dra98] R. Drath. Hybrid object nets: an object oriented concept for modeling complex hybrid systems. *Proc. Hybrid Dynamical Systems, 3rd International Conference on Automation of Mixed Processes, ADPM'98*, pages 437–442, 1998.

[Dur07] R Durbin. *Biological sequence analysis*. Cambridge Univ. Press, Cambridge [u.a.], 2007.

[EUA96] T. Etzold, A. Ulyanov, and P. Argos. SRS: information retrieval system for molecular biology data banks. *Meth. Enzymol.*, 266:114–128, 1996.

[FD87] D. F. Feng and R. F. Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, 25:351–360, 1987.

[FGM^{+}98] W. Fujibuchi, S. Goto, H. Migimatsu, I. Uchiyama, A. Ogiwara, Y. Akiyama, and M. Kanehisa. DBGET/LinkDB: an integrated database retrieval system. *Pac Symp Biocomput*, pages 683–694, 1998.

[Fis94] E. Fischer. Einfluss der Configuration auf die Wirkung der Enzyme. *Ber. Dt. Chem. Ges.*, 27:2985 – 93, 1894.

[FLM95] A. Frick, A. Ludwig, and H. Mehldau. A fast adaptive layout algorithm for undirected graphs. In *GD '94: Proceedings of the DIMACS International Workshop on Graph Drawing*, pages 388–403, London, UK, 1995. Springer-Verlag.

[FMKL99] D. Fambrough, K. McClure, A. Kazlauskas, and E. S. Lander. Diverse signaling pathways activated by growth factor receptors induce broadly overlapping, rather than independent, sets of genes. *Cell*, 97:727–741, Jun 1999.

[Gar69] D. Garfinkel. Construction of biochemical computer models. *FEBS Lett.*, 2 Suppl 1:S9–S13, Mar 1969.

[GC09] M. Y. Galperin and G. R. Cochrane. Nucleic Acids Research annual Database Issue and the NAR online Molecular Biology Database Collection in 2009. *Nucleic Acids Res.*, 37:1–4, Jan 2009.

[GLL$^+$03] G. Grillo, F. Licciulli, S. Liuni, E. Sbisá, and G. Pesole. PatSearch: A program for the detection of patterns and structural motifs in nucleotide sequences. *Nucleic Acids Res.*, 31:3608–3612, Jul 2003.

[GRSW05] R. Gauges, U. Rost, S. Sahle, and K. Wegner. Including layout information in sbml files version 2.2. 2005. http://projects.eml.org/bcb/sbml/level2/20050425/SBMLLayoutExtension-20050425.pdf.

[HCI$^+$04] M. A. Harris, J. Clark, A. Ireland, J. Lomax, M. Ashburner, R. Foulger, K. Eilbeck, S. Lewis, B. Marshall, C. Mungall, J. Richter, G. M. Rubin, J. A. Blake, C. Bult, M. Dolan, H. Drabkin, J. T. Eppig, D. P. Hill, L. Ni, M. Ringwald, R. Balakrishnan, J. M. Cherry, K. R. Christie, M. C. Costanzo, S. S. Dwight, S. Engel, D. G. Fisk, J. E. Hirschman, E. L. Hong, R. S. Nash, A. Sethuraman, C. L. Theesfeld, D. Botstein, K. Dolinski, B. Feierbach, T. Berardini, S. Mundodi, S. Y. Rhee, R. Apweiler, D. Barrell, E. Camon, E. Dimmer, V. Lee, R. Chisholm, P. Gaudet, W. Kibbe, R. Kishore, E. M. Schwarz, P. Sternberg, M. Gwinn, L. Hannick, J. Wortman, M. Berriman, V. Wood, N. de la Cruz, P. Tonellato, P. Jaiswal, T. Seigfried, and R. White. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.*, 32:D258–261, Jan 2004.

[HKHR08] T. Helikar, J. Konvalina, J. Heidel, and J. A. Rogers. Emergent decision-making in biological signal transduction networks. *Proc. Natl. Acad. Sci. U.S.A.*, 105:1913–1918, Feb 2008.

[HM85] D. Heimbigner and D. McLeod. A federated architecture for information management. *ACM Trans. Inf. Syst.*, 3(3):253–278, 1985.

[HMW$^+$05] Z. Hu, J. Mellor, J. Wu, T. Yamada, D. Holloway, and C. Delisi. VisANT: data-integrating visual framework for biological networks and modules. *Nucleic Acids Res.*, 33:W352–357, Jul 2005.

[Hof94] R. Hofestädt. A petri net application to model metabolic processes. *Syst. Anal. Model. Simul.*, 16(2):113–122, 1994.

[HSA$^+$05] A. Hamosh, A. F. Scott, J. S. Amberger, C. A. Bocchini, and V. A. McKusick. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. *Nucleic Acids Res.*, 33:D514–517, Jan 2005.

[HSK$^+$01] L. M. Haas, P. M. Schwarz, P. Kodali, E. Kotlar, J. E. Rice, and W. C. Swope. Discoverylink: a system for integrated access to life sciences data sources. *IBM Syst. J.*, 40(2):489–511, 2001.

[HT98] R. Hofestädt and S. Thelen. Quantitative modeling of biochemical networks. *In Silico Biol. (Gedruckt)*, 1:39–53, 1998.

[HTBH07] S. Hariharaputran, T. Töpel, B. Brockschmidt, and R. Hofestädt. Vinedb: a data warehouse for integration and interactive exploration of life science data. *Journal of Integrative Bioinformatics - JIB*, 4(3):63, 2007.

[HTOH08] S. Hariharaputran, T. Töpel, T. Oberwahrenbrock, and R. Hofestädt. Alignment of linear biochemical pathways using protein structural classification. *Nature Precedings*, 2008.

[Inm96] W H. Inmon. *Building the data warehouse.* Wiley, Indianapolis, Ind., 1996.

[JI98] J. D. Jordan and R. Iyengar. Modes of interactions between signaling pathways. *Biochem. Pharmacol.*, 55:1347–1352, May 1998.

[JKS06] B. H. Junker, C. Klukas, and F. Schreiber. VANTED: a system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics*, 7:109, 2006.

[JKT$^+$10] S. Janowski, B. Kormeier, T. Töpel, K. Hippe, R. Hofestädt, N. Willassen, R. Friesen, S. Rubert, D. Borck, P. Haugen, and M. Chen. Modeling of cell-cell communication processes using petri nets in the example of quorum sensing. *In Silico Biology 10, 0003*, 2010.

[JLI00] J. D. Jordan, E. M. Landau, and R. Iyengar. Signaling networks: the origins of cellular multitasking. *Cell*, 103:193–200, Oct 2000.

[JM61] F. Jacob and J. Monod. Genetic regulatory mechanisms in the synthesis of proteins. *J. Mol. Biol.*, 3:318–356, Jun 1961.

[JS08] B. H. Junker and F. Schreiber. *Analysis of Biological Networks.* Wiley & Sons, 2008.

[KAG$^+$08] M. Kanehisa, M. Araki, S. Goto, M. Hattori, M. Hirakawa, M. Itoh, T. Katayama, S. Kawashima, S. Okuda, T. Tokimatsu, and Y. Yamanishi. KEGG for linking genomes to life and the environment. *Nucleic Acids Res.*, 36:D480–484, Jan 2008.

[KBT$^+$06] J. Köhler, J. Baumbach, J. Taubert, M. Specht, A. Skusa, A. Rüegg, C. Rawlings, P. Verrier, and S. Philippi. Graph-based analysis and visualization of experimental results with ONDEX. *Bioinformatics*, 22:1383–1390, Jun 2006.

[KFQW99] A. Klingenhoff, K. Frech, K. Quandt, and T. Werner. Functional promoter modules can be detected by formal models independent of overall nucleotide sequence similarity. *Bioinformatics*, 15:180–186, Mar 1999.

[KHK$^+$05] E. Klipp, R. Herwig, A. Kowald, C. Wierling, and H. Lehrach. *Systems Biology in Practice.* WILEY-VCH Verlag GmbH & Co. KGaA, 2005.

[KHTH09] B. Kormeier, K. Hippe, T. Töpel, and R. Hofestädt. CardioVINEdb: a data warehouse approach for integration of life science data in cardiovascular diseases. In *Im Focus das Leben. Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V. (GI)*, pages 40,704–708, Bonn, 2009. S. Fischer et al. (Hrsg.), INFORMATIK 2009, Koellen Druck+Verlag.

[KJT$^+$09] B. Kormeier, S. Janowski, T. Töpel, K. Hippe, P. Arrigo, and R. Hofestädt. Reconstruction of networks based on life science data integration. In *2009 International Workshop on Computational and Integrative Biology*, China, 2009.

[KPV$^+$06] M. Krull, S. Pistor, N. Voss, A. Kel, I. Reuter, D. Kronenberg, H. Michael, K. Schwarzer, A. Potapov, C. Choi, O. Kel-Margoulis, and E. Wingender. TRANSPATH: an information resource for storing and visualizing signaling pathways and their pathological aberrations. *Nucleic Acids Res.*, 34:D546–551, Jan 2006.

[LMR90] W. Litwin, L. Mark, and N. Roussopoulos. Interoperability of multiple autonomous databases. *ACM Comput. Surv.*, 22(3):267–293, 1990.

[LN07] U. Leser and F. Naumann. *Informationsintegration*. dpunkt Verlag, 2007.

[LPW$^+$06] T. J. Lee, Y. Pouliot, V. Wagner, P. Gupta, D. W. Stringer-Calvert, J. D. Tenenbaum, and P. D. Karp. BioWarehouse: a bioinformatics database warehouse toolkit. *BMC Bioinformatics*, 7:170, 2006.

[LR03] U. Leser and P. Rieger. Integration molekularbiologischer Daten. *Datenbank-Spektrum*, 3(6):56–66, 2003.

[MDNM00] H. Matsuno, A. Doi, M. Nagasaki, and S. Miyano. Hybrid Petri net representation of gene regulatory network. *Pac Symp Biocomput*, pages 341–352, 2000.

[MHO01] S. M. Maurer, P. B. Hugenholtz, and H. J. Onsrud. Intellectual property. Europe's database experiment. *Science*, 294:789–790, Oct 2001.

[Mic99] G. Michal. *Biochemical Pathways: An Atlas of Biochemistry and Molecular Biology*. Wiley & Sons, 1999.

[MKMF$^+$06] V. Matys, O. V. Kel-Margoulis, E. Fricke, I. Liebich, S. Land, A. Barre-Dirrie, I. Reuter, D. Chekmenev, M. Krull, K. Hornischer, N. Voss, P. Stegmaier, B. Lewicki-Potapov, H. Saxel, A. E. Kel, and E. Wingender. TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. *Nucleic Acids Res.*, 34:D108–110, Jan 2006.

[NDMM05] M. Nagasaki, A. Doi, H. Matsuno, and S. Miyano. Petri net based description and modeling of biologinal pathways. *Algebraic Biology 2005 - Computer Algebra in Biology*, pages 19–.31, 2005.

[NW70] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, Mar 1970.

[OMJ$^+$97] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton. CATH–a hierarchic classification of protein domain structures. *Structure*, 5:1093–1108, Aug 1997.

[PE02] T. D. Pollard and W. C. Earnshaw. *Cell biology*. Saunders, Philadelphia, Pa. [u.a.], 2002.

[Pet62] C. A. Petri. *Kommunikation mit Automaten.* Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962.

[Pet66] C. A. Petri. Kommunikation mit Automaten. *New York: Griffiss Air Force Base, Technical Report RADC-TR-65–377*, 1:1–Suppl. 1, 1966. English translation.

[Pet77] J. L. Peterson. Petri nets. *ACM Comput. Surv.*, 9(3):223–252, 1977.

[PRYLZU05] R. Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21:3401–3408, Aug 2005.

[PW03] L. Priese and H. Wimmel. *Theoretische Informatik, Petri-Netze.* Springer, Berlin [u.a.], 2003.

[RML93] V. N. Reddy, M. L. Mavrovouniotis, and M. N. Liebman. Petri net representations in metabolic pathways. *Proc Int Conf Intell Syst Mol Biol*, 1:328–336, 1993.

[SBB$^+$00] R. Stevens, P. Baker, S. Bechhofer, G. Ng, A. Jacoby, N. W. Paton, C. A. Goble, and A. Brass. TAMBIS: transparent access to multiple bioinformatics information sources. *Bioinformatics*, 16:184–185, Feb 2000.

[SCE$^+$04] I. Schomburg, A. Chang, C. Ebeling, M. Gremse, C. Heldt, G. Huhn, and D. Schomburg. BRENDA, the enzyme database: updates and major new developments. *Nucleic Acids Res.*, 32:D431–433, Jan 2004.

[SHX$^+$05] S. P. Shah, Y. Huang, T. Xu, M. M. Yuen, J. Ling, and B. F. Ouellette. Atlas - a data warehouse for integrative bioinformatics. *BMC Bioinformatics*, 6:34, 2005.

[SL90] A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, 1990.

[SMO$^+$03] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, 13:2498–2504, Nov 2003.

[TKKH08] T. Töpel, B. Kormeier, A. Klassen, and R. Hofestädt. BioDWH: A data warehouse kit for life science data integration. *Journal of Integrative Bioinformatics - JIB*, 5(2):93, 2008.

[TRM$^+$05] S. Trissl, K. Rother, H. Müller, T. Steinke, I. Koch, R. Preissner, C. Frömmel, and U. Leser. Columba: an integrated database of proteins, structures, and annotations. *BMC Bioinformatics*, 6:81, 2005.

[vMKS$^+$02] C. von Mering, R. Krause, B. Snel, M. Cornell, S. G. Oliver, S. Fields, and P. Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417:399–403, May 2002.

[WC53] J. D. Watson and F. H. Crick. Genetical implications of the structure of deoxyribonucleic acid. *Nature*, 171:964–967, May 1953.

[Win04] E. Wingender. TRANSFAC, TRANSPATH and CYTOMER as starting points for an ontology of regulatory networks. *In Silico Biol. (Gedruckt)*, 4:55–61, 2004.

[Win08] E. Wingender. The TRANSFAC project as an example of framework technology that supports the analysis of genomic regulation. *Brief. Bioinformatics*, 9:326–332, Jul 2008.