The development of an integrated database of the model organism *Bacillus subtilis*

Dissertation
for the award of the degree
"Doctor rerum naturalium"
of the Georg-August-Universität Göttingen

within the doctoral program Microbiology & Biochemistry
of the Georg-August University School of Science (GAUSS)

submitted by

Raphael Michna

from Paderborn

Göttingen 2015

Reviewers

Prof. Dr. Jörg Stülke (supervisor and first reviewer)

Prof. Dr. Burkhard Morgenstern (second reviewer)

Additional members of the examination board

PD. Dr. Fabian Commichau (third thesis committee member)

Prof. Dr. Rolf Daniel

Prof. Dr. Ivo Feußner

Prof. Dr. Ralf Ficner

Date of Examination: 13.1.2016

I hereby declare that the doctoral thesis entitled, “The development of an integrated database of the model organism *Bacillus subtilis*” has been written independently and with no other sources and aids than quoted.

____

City, date, name

Raphael Michna, Göttingen

Danksagung

Im Laufe meines Lebens konnte und kann ich mich glücklich schätzen, so viele Menschen kennengelernt zu haben, die mein Leben auf eine so wundervolle Weise beeinflusst und bereichert haben. Vor allem möchte hier die Möglichkeit ergreifen um meiner Mutter zu danken, die mich mit geduldig und mit Herz aufzog und über während des Studiums stets unterstützte. Im Studium musste ich meinen Weg eigenständig finden, doch auch dort war ich nicht immer überzeugt von dem, was ich tat und wie ich es tat. Hier möchte ich speziell meiner Freundin danken, die es nicht immer leicht mit mir hatte. Durch ihre Hartnäckigkeit bewegte sie mich doch häufig zum Lernen, das ich ohne sie an einigen Stellen, denke ich, nicht getan hätte. Ich bin wirklich glücklich solche zwei Frauen zu haben, die mir den Rücken stärken und immer an mich glauben.

Im Laufe des Studiums kam noch eine weitere Person hinzu, die es verstand mich voranzutreiben und vielleicht auch schon wusste, wo man diesen leichtgläubigen und leicht verwirrten Studenten unterbringen kann. Hier möchte ich Jörg Stülke hervorheben, ohne den bereits die Bachelorarbeit gescheitert wäre. Zur Bachelorarbeit nahm er mich auf und leitete mich in das weite Feld der Bioinformatik. Ab diesem Moment unterstand ich dem Kommando von Lope, der mich mit Geduld und Hingabe in die Informatik einführte, das bis dato bekanntes Gebiet für mich war. Danken möchte ich auch ihm für seine Zeit. Jörg ermöglichte mir auch das weitere Studium im Bereich der Mikrobiologie und Biochemie, doch meine Leidenschaft war da schon die Informatik, doch im Fokus stand immer das Biologiestudium. Somit war es auch schon klar, dass ich bei Jörg die Masterarbeit schreiben wollte in genau dieser Kombination Biologie und Informatik. Zum goldenen Abschluss wurde mir auch noch die Doktorarbeit angeboten, die für mich den Höhepunkt meiner akademischen Laufbahn darstellte und ich dieses Angebot dankend annahm.

Im Laufe der Zeit habe ich natürlich auch noch einige Studenten betreut, die mich in meinem Projekt unterstützten und auch viele interessante Ideen mitbrachten. Allen voran ist hier Bingyao zu nennen, die ich während ihrer Masterarbeit betreut habe. Danken möchte ich ihr für die vielen anregenden Gespräche, die sich während ihrer Arbeit ergaben. Weitere Studenten sind Miguel, Elena und Wanwan, die sich auch mit Begeisterung in ein ihnen fremdes Gebiet stürzten.

Außerdem möchte ich den Mitgliedern der AG Stülke und AG Commichau danken, ohne die gesunde Arbeitsatmosphäre nicht möglich gewesen wäre. Das Klima innerhalb der Gruppe war auch einer der Grund sich der Gemeinschaft anzuschließen.

Schließlich möchte ich noch den meinen Freunden danken, die mich immer so genommen haben wie ich bin und mich manchmal mit ihrer Direktheit in die richtige Bahn gelenkt haben.

Vielen Dank und eine schöne Zeit

Table of contents

List of publications

In this Ph.D. thesis:

***Subti*Wiki 2.0-an integrated database for the model organism *Bacillus subtilis*.** Michna RH, Zhu B, Mäder U, Stülke J. Nucleic Acids Res. 2015 Oct 3. pii: gkv1006.

***Subti*Wiki-a database for the model organism *Bacillus subtilis* that links pathway, interaction and expression information.** Michna RH, Commichau FM, Tödter D, Zschiedrich CP, Stülke J.Nucleic Acids Res. 2014 Jan;42(Database issue):D692-D698.

Before this Ph.D. thesis:

CellPublisher: a web platform for the intuitive visualization and sharing of metabolic, signalling and regulatory pathways. Flórez LA, Lammers CR, Michna R, Stülke J. Bioinformatics. 2010 Dec 1;26(23):2997-2999.

List of abbreviations

AJAX – Asynchronous JavaScirpt and XML

API – Application Programming Interface

ATP – Adenosine-triphosphate

BLAST – Basic Local Alignment Search Tool

CSS – Cascading Style Sheet

DNA – DesoxyriboNucleic Acid

FAQ – Frequently Asked Questions

GUI – Graphical User Interface

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

JSON – JavaScript Object Notation

KEGG – Kyoto Encyclopedia of Genes and Genomes

LIMP – Linux MySQL PHP [Perl, Python]

mRNA – messenger RiboNucleic Acid

MRSA – Multi Resistant Staphylococcus Aureus

NAR – Nucleic Acids Research

NCBI – National Center for Biotechnology Information

PHP – Pre Hypertext Processor

re – regular expressions

RNA – RiboNucleic Acid

SPINE – Strep-protein interaction experiment

SQL – Structured Query Language

SVG – Scalable Vector Graphics

TCA – Tricarboxylic Acid

XML – Extensible Markup Language

zetta – 10^{21}

1. Summary

In the era of “Omics” approaches, the available data is exponentially increasing and new methods must be invented to handle this data. For this purpose, databases store, maintain and provide the data, and algorithms provide the possibility for further analyzing, clustering or organizing the datasets. *Bacillus subtilis* is a well-studied model bacterium which is a prerequisite for further scientific research. *Subti*Wiki is a database storing the annotated information on genes of *B. subtilis*. This database is curated by the *Bacillus* community which keeps it up to date. Unfortunately, the mediawiki engine provides the information in raw text and the properties of a gene are not directly usable in bioinformatic approaches. The main properties of the gene are hidden in raw text of the *Subti*Wiki pages. Here we altered the layout of the database, extracted the information from the wiki pages and created a completely new website. In the beginning, the relevant information was filtered out from the wiki pages via text mining. In parallel, the extracted properties of a gene were organized and then inserted in the new database structure. The content of the applications (*Subt*Interact, *Subti*Pathways, *Subti*Express) was adapted to the new database layout to extend the information of the gene. Finally, the web sites obtained a new design. As a result, *Subti*Wiki is not a classical “Wiki” anymore but the idea of fast editing and community curation remains. Thus, this database moved to a classical database structure which can be used for bioinformatic approaches and it simplifies maintenance of the data. *Subti*Wiki becomes a powerful database covering genes, pathways, expression-levels and protein-protein interactions of *B. subtilis* but the analysis and comparison of the properties and networks recently began. The newly constructed database and the new web performance facilitate new insights in *B. subtilis*. Moreover, the tools for the analysis and especially visualization were adapted and improved to fit to the new database layout. The tools and the data are now easily accessible for the user and *Subti*Wiki is prepared to integrate high-throughput data.

2. Introduction

In the era of high-throughput data and technological approaches the data increases exponentially. By the enormous increase of data, technologies had to be developed to compare, maintain and analyze the content. This cannot be done by humans due to the plentitude of data. For this purpose, analyzing tools and databases are provided to support researchers. These tools are realized by web applications to make the data easily accessible. There are different categories of databases available working on a broad spectrum of topics. Some databases are specialized on a single topic whereas others are large databases storing large repositories of data.

2.1 Big data

Big data defines a huge set of data that is too large and too complex to manage by hand. For this purpose, large computer systems with large memory are created to store the incredible datasets of information. Big data is a new challenge in all fields of life. Researchers are developing tools to derive a model from tremendous datasets. Today big data is more and more associated with observation and security due to the data of a user which is collected. This data is further analyzed. Some companies use the data to analyze the behavior of the user during the internet session.

In the field of biology big data is the information created by high-throughput methods like genomics, proteomics, metabolomics, transcriptomics and many more. These methods produce large sets of data points. These experimental data has to be analyzed by bioinformatic tools which have to be developed additionally. The amount of data is exponentially increasing (see figure 2.1) and therefore databases are necessary tools to store and to provide biological data. The need of these databases is undoubted due to the fact the amount of databases is also exponentially increasing. Over time, many databases were invented which specialized in a certain topic.

2.2 High-throughput data

In the field of biology there are many methods available measuring the amount of biological units like transcripts, proteins and metabolites. The most popular ones are genomics, proteomics, transcriptomics and metabolomics (Tang 2011). These methods ending on “omics” are widely used in determining the microbial composition of an environmental sample. These methods are then called metaproteomics, metatranscriptomics and metagenomics. For example, in contrast to genomics the metagenomic approach is determining the DNA sequences of all species in the sample instead of a single species. Furthermore, the interrelationships between the species can be determined by using all the other "meta-omics" approaches (Abram, 2014).

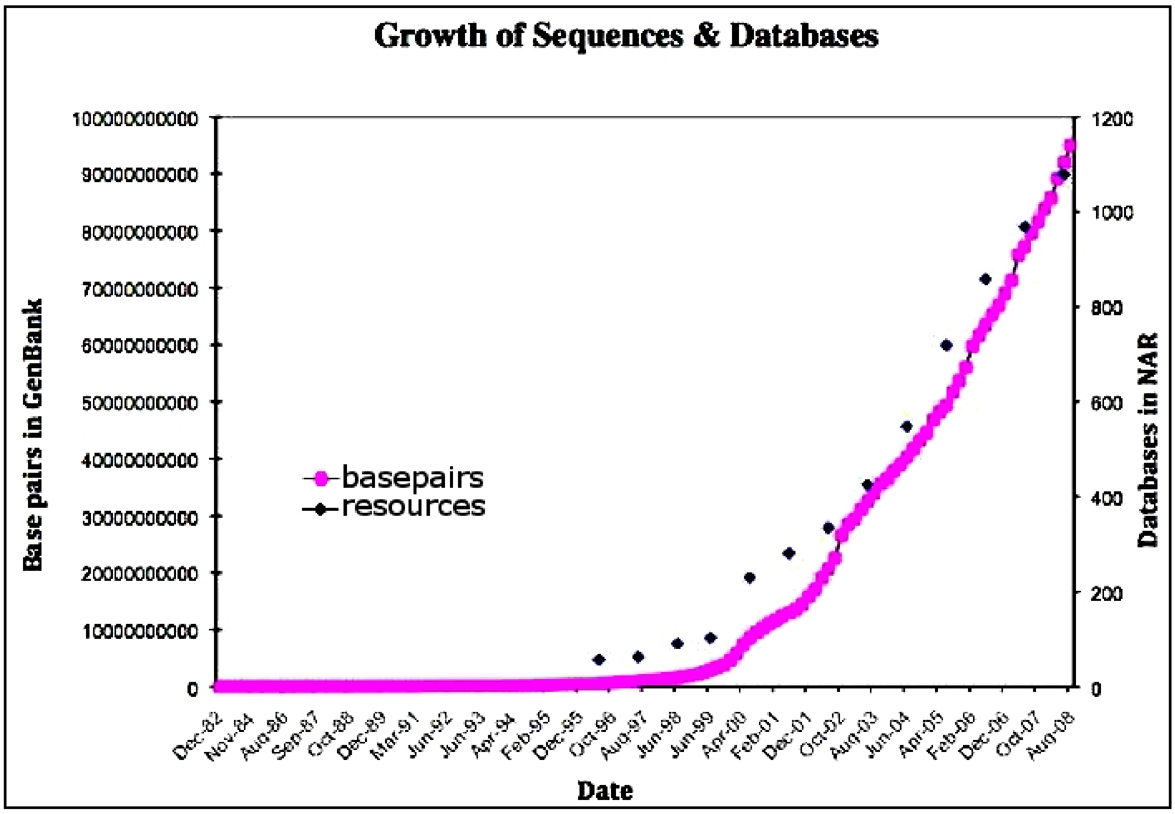


Figure 2.1: The increase of databases and base pairs. The graph shows the exponential increase of the resources mentioned in NAR and the base pairs available in GenBank.

With proteomics approach the amount of cytosolic protein is measured at a defined time point. The time-dependent snapshot provides a closer insight into the cell. Unfortunately, proteomics (Maass et al., 2011) cannot or purely measure the membrane-bound proteins which lead to an incorrect result. Here, the technology has to be improved to solve this problem. Additionally, at this point the state of the protein is unknown due to missing information on post-translational modifications. In the transcriptomics approach (Nicolas et al., 2012) the expression levels of transcripts are measured at a defined time point. This snapshot provides information on the transcripts present in the cell at a certain point. Still with this information it is not known if a totally functioning protein is produced or the mRNA is directly digested. But it can be observed that the gene transcription is activated or repressed. In this large datasets it is hard to identify the up or down-regulated gene cluster of genes by human eye. Therefore, new approaches in computer science support the scientist to analyze the datasets. Visualization is an often used tool to cluster genes. Moreover, different sets can be compared to find equalities or differences between organisms. In this field mathematic models play an important role to produce a representative result.

It was stated out that only sequence data will grow ~ 1 zettabase/year by 2025 (Stephens et al., 2015). This enormous number offers the opportunities for novel mining algorithms. Classification algorithms, frequent pattern algorithms , clustering algorithms and graph and network algorithms evolved due to the current challenges to capture the mass of information which not readable for humans anymore.

Especially, transcriptomic and proteomic data provide closer insights into the availability of proteins and mRNA. In a context-free environment this data is not more than numbers but in the context of metabolic or regulatory networks it could support the understanding of whole networks in the cell. With the data a second layer of data can be added to the existing information to enrich a metabolic or regulatory pathway or an interaction profile.

2.3 Databases

The general idea of a database is that the researcher is able to easily find and access the relevant data. A database is a well organized collection of related data working on a specific topic. Over time, in the area of biology many types of databases evolved which archive, store, maintain and provide information. Some sources are specified on single topics. PDB (http://www.rcsb.org/pdb/home/home.do) – Protein Data Bank – is a database storing information on 3 D structures of proteins, nucleotides and complex assemblies (Berman, 2008). Another web source is Pfam (http://pfam.xfam.org/) providing information on protein domains (Finn et al., 2014) which is divided into two parts Pfam-A and Pfam-B. Pfam-A contains the well studied domains whereas Pfam-B contains the domains with unknown function. With the pattern recognition domains can be identified in new protein sequences. Furthermore, there are many more databases specialized on reactions, interactions, metabolites, pathways, expression profiles and many more. Due to that the list of specialized databases is quite long and the number of resources is constantly increasing.

The KEGG (http://www.kegg.jp) is database covering biological systems including the cell, the organism and the ecosystem. KEGG does also offer analyzing tools as “KEGG Mapper” and “KEGG Atlas” which are tools to explore metabolic pathways and further analyze pathways by including “omics” data (Kanehisa et al., 2014).

The NCBI (http://www.ncbi.nlm.nih.gov/) database provides publically available nucleotide sequences of more than 260,000 species (Benson et al., 2013). Additionally, it is also a resource for literature known as PubMed (http://www.ncbi.nlm.nih.gov/pubmed/). It is beyond all questions that NCBI is the main resource for all researchers working in molecular biology. The main goal of these efforts is to facilitate research and comparative studies. Moreover, the NCBI is the main access point to start collecting data or to start analyzing it. One of the most prominent tools which the NCBI provides is BLAST (Mount, 2007). The tool allows the comparison of a gene sequence to all available sequences in a database. In contrast to the FASTA (Pearson et al., 1988) algorithm BLAST also searches for small common patterns in the sequence but the BLAST algorithm much faster due to the specified search on rare patterns. The comparison of sequences makes it possible to deduce the functionality due to the similarity of a gene/ protein in another organism. The NCBI also maintains the Sequence Read Archive (Kodama et al., 2012) which provides data next generation sequence analysis.

Another popular and powerful resource is the EMBL-EBI (Brooksbank et al., 2014) providing also databases and tools for the genomes, genes, proteins, transcripts, metabolites and networks like reactions and interactions. The most popular member of EMBL-EBI is the UniProt database containing comprehensive data of protein sequence and protein annotation data (Magrane & Consortium U., 2011).

In the end there are the smaller databases curating information on a single organism. These databases are initiated by smaller groups. Mostly, these databases are “community curated” but unfortunately there are only a few users updating information to the database. But one of the main problems in the idea is the editing of the web page. The user should be allowed to edit the web page but therefore the One community curation solution is the "wikification" (Hu et al., 2008) of data update and curation, where the researcher update their information. A famous example of wiki based database is *Ecoli*Wiki which provides all kind of information on the well studied model organism *Escherichia coli*. In contrast to *Subti*Wiki, *Ecoli*Wiki (http://ecoliwiki.net/colipedia/index.php/Welcome_to_EcoliWiki) provides not only information on a single strain. The mass of information make the appearance slightly confusing (McIntosh et al., 2012) and the content could overwhelm the user. The advantage of the wiki is the fast editing process but it is only effective if the community participates in updating. Otherwise, there is no quality proof of the content the user is updating. As a result, the content quality can be diminished and a consortium of a few persons is proving the quality of an edit.

There are many other different databases curating the knowledge on *B. subtilis* or other organism. BsubCyc (http://bsubcyc.org) is one of the online platforms providing information on the genes/ proteins on *B. subtilis* (Karp et al., 2014). It offers a lot of information on the single gene, but also on regulatory and metabolic networks as well as genetic arrangement. Furthermore, there are quite handy tools as the “smart tables” where the user can generate, arrange or alter lists. Due to the nice structure *Bsub*Cyc is not a community curated database and is only updated by few administrators what makes it not totally up-to-date.

Another database providing information on *B. subtilis* is *Subti*List (http://genolist.pasteur.fr/SubtiList/). (Moszer et al., 2002) This database was not updated since 2001 which means that the information on this database is not up-to-date. Due to missing constant updates the names or other properties could differ from the ones annotated in *Subti*Wiki or *Bsub*Cyc.

2.4 Text mining

In computer science it is also known as text data mining meaning that a script runs through a collection of documents written in a natural language. The main aim is to extract the relevant information from raw text. Text mining typically involves the following steps: structuring of the raw text, deriving a structure from it and finally evaluating and interpreting the output. In the end text mining should essentially turn pure text into data for analysis. Text written in a natural language is not suitable for comparative analyses.

The database PubMed (https://www.ncbi.nlm.nih.gov/pubmed/) is providing a large archive of scientific publications. This database is a first source to start the scientific investigation. Due to publications that are written in a natural language and the exponentially increasing number of publications it is of high interest to extract automatically the relevant content (Cohen & Hunter, 2008; Rzhetsky et al., 2009). The automatic methods are widely used in science to search for patterns and to filter out the relevant significant content without noise. This filtered content is then further processed to fit to the desired format.

2.5 Web application

Server-side scripting defines the scripts that are executed on the server and then the result is sent to to the client. For the client the script that was executed on the server is not visible. Usually the result is presented in the browser. The tool must be as easy and as intuitive as possible so that the scientist could find the relevant information as fast as possible. The languages used on the server-side are PHP, perl, python and many more. On the server side MySQL offers the information and PHP is preparing it for the presentation on the screen or PHP simply delivers it to the client (see figure 2.2).

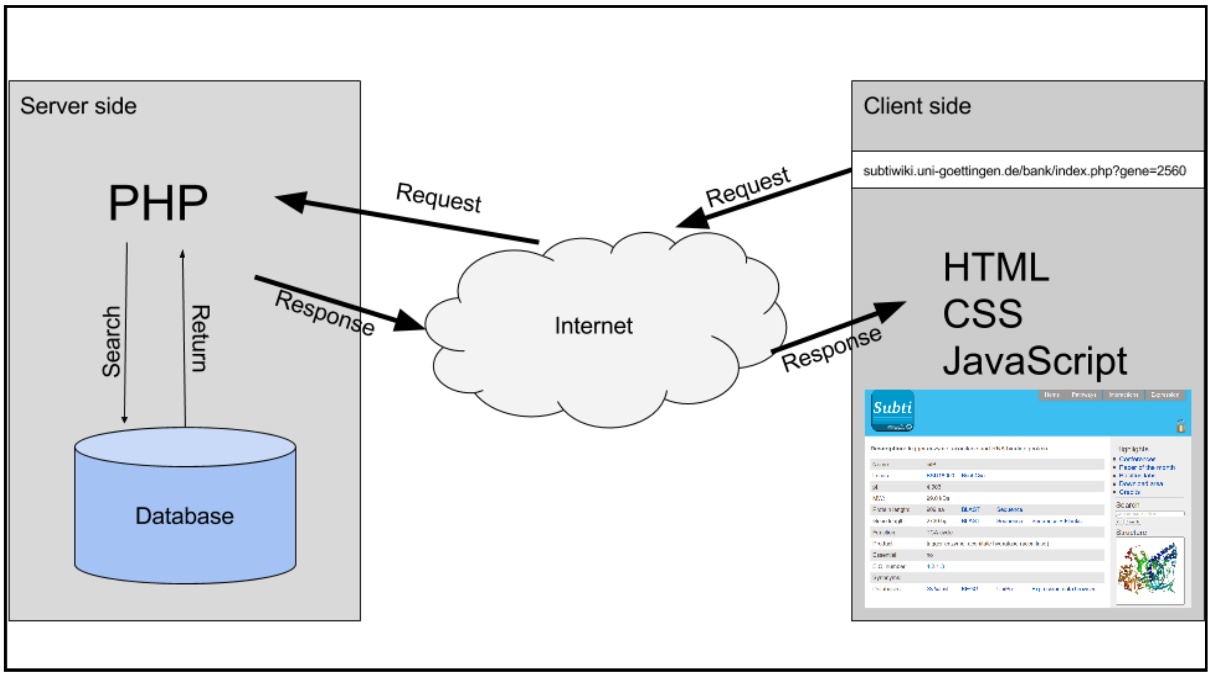


Figure 2.2: The interaction between client and server. The client sends a request to the server via the internet. A php scirpt processes the request. MySQL searches for the information in the database and if the request was successful php creates the static page and the server sends back a response to the client via the internet and the generated HTML page is then presented in the browser.

The client-side is different from the server-side. Thereby, the client is requesting a page from the server and receives the desired response. The difference is now that not the server is executing the script. Now, the requested data is sent to the client and directly executed on the client side using a different set of programming languages. The most prominent one is JavaScript which can interactively alter the web site. Due to that JavaScript is able to manipulate the appearance and the styling of the page as well as calculations that are executed on the page. AJAX is a programming concept to request the information from the browser without loading the whole page again. The client is requesting a snippet of information to interactively update parts of the page instead of the whole one.

2.6 *Bacillus subtilis*

Bacillus subtilis is a gram positive soil bacterium which belongs to the phylum of the *Firmicutes*. The rod-shaped model organism is well studied and the genome was totally sequenced in 1997 (Kunst et al., 1997). *B. subtilis* is non-pathogenic but it is closely related to pathogenic species of this phylum like *Staphylococcus aureus*. The close relation makes the study of *B. subtilis* so important. MRSA (multi resistant *Staphylococcus aureus*) is a mutant strain of *S. aureus* which can be dangerous to people with a weak immune system and it can cause incredible damages to the patient which could finally lead to death. Additionally, *B. subtilis* is also used in biotechnology for a wide range of applications. It is used for vitamin and enzyme production (Schallmey et al., 2004). The genome of *B. subtilis* is easy to manipulate. It possesses all necessary enzymes for a homologous recombination enabling the targeted DNA integration into the genome (Fernández et al., 2000).

2.7 Minibacillus

In a cell the genes are the smallest building blocks which determine the properties of a cell like shape and metabolic activity. Without some of the genes the cell cannot live. Those genes are called essential. The research on the essential gene set of *B. subtilis* is the main target of the Mini*bacillus* project (http://minibacillus.org). Due to experimental (Baba et al., 2006; Boutros & Ahringer, 2008; Christen et al., 2011; de Berardinis et al., 2008; French et al., 2008; Kato & Hashimoto, 2007; Kobayashi et al., 2003; Langridge et al., 2009) and computational approaches (Juhas et al., 2011, 2014; McCutcheon & Moran, 2010; Moya et al., 2009) the study and the identification of essential genes were improved. Unfortunately, the identification and prediction of the essential gene set which is universally required by all organisms is still impossible. Bacteria are adapted to a wide range of environments which lead to a different set of genes. Different genes are needed to use the nutrients of the current habitat. Over time, the genome evolves and the idle and unprofitable genes are deleted or silenced. Mutations and DNA uptake could lead to new properties improving but even diminishing the functionality of a protein. Therefore, the behavior of the cell could be altered positively or negatively. Finally, the better adapted population will suppress the other ones (Stannek et al., 2014). This phenomenon is known as survival of the fittest population.

Furthermore, the genes are strongly connected to each other via highly complex metabolic and regulatory networks which are in detail not yet fully understood. *B. subtilis* harbors a set of regulators which respond to certain environmental changes. Not only proteins regulate the transcription or translation but also the small RNAs (Mars et al., 2015; Maaß et al., 2014; Muntel et al., 2014) are involved. The high complexity makes it difficult to identify not relevant genes because an identified gene could have an additional functionality. Such genes are called moonlighting proteins which harbor more than one catalytic function (Commichau et al., 2009). For example the deletion of a gene can lead to an accumulation of toxic compounds but if the whole metabolic pathway is deleted the organism survives. In conclusion, the deletion of a gene can lead to an unpredictable and unprofitable behavior of the metabolism if the context and the role of the gene are not fully understood.

The dynamics of the constant evolution of the genome (Yu et al., 2015) to the conditional circumstances and the dependency network of the gene set leads to a distorted idea of the "universal essential gene set". The shapes of the organisms differ and also the metabolic processes are adapted to the varying environments. In the mentioned paper the group identified diverging *B. subtilis* strains. The whole genome sequencing of two isolated strains revealed a large variation including deletions and insertions.

B. subtilis is a well studied organism which is used in biotechnology as a cell factory to produce a specific range of substances. Therefore, the research on the *B. subtilis* genome is of high importance. The minimal genome could be used as the chassis and the desired genes would be additionally introduced to a fully categorized and defragmented cell. As a result, the production of a specific compound will be improved without any accumulation of side products. Otherwise, one of the disadvantages of the stepwise minimization could affect the growth rate of the cell.

2.8 ***Subti*Wiki**

*Subti*Wiki (http://subtiwiki.uni-goettingen.de) is a web platform of the model organism *B. subtilis* and a frequently used web interface for the *B. subtilis* community (Mäder et al., 2012). The concept of the web platform is that each page contains information on one gene/ protein. Furthermore, there are also special pages describing categories, regulons and further properties. Each page of *Subti*Wiki is built up by the following structure: At the top of the page a table is shown where the main information like name, synonyms, molecular weight, isoelectric point, DNA and protein sequences can be found. Additionally, the genomic arrangement is presented on each site. Then the page shows the categories and regulons the gene/ protein belongs to. Moreover, the user can find information on the protein, as well as on expression and regulation, on biological materials, on labs and in the end references. *Subti*Wiki runs on the mediawiki (https://www.mediawiki.org/wiki/MediaWiki) engine to store the information. A popular web site using the mediawiki engine is WikiPedia (https://www.wikipedia.de). Over time many small groups working on a single organism preferred to use this engine due to fast implementation and the editing of the pages is not restricted to a handful administrators.

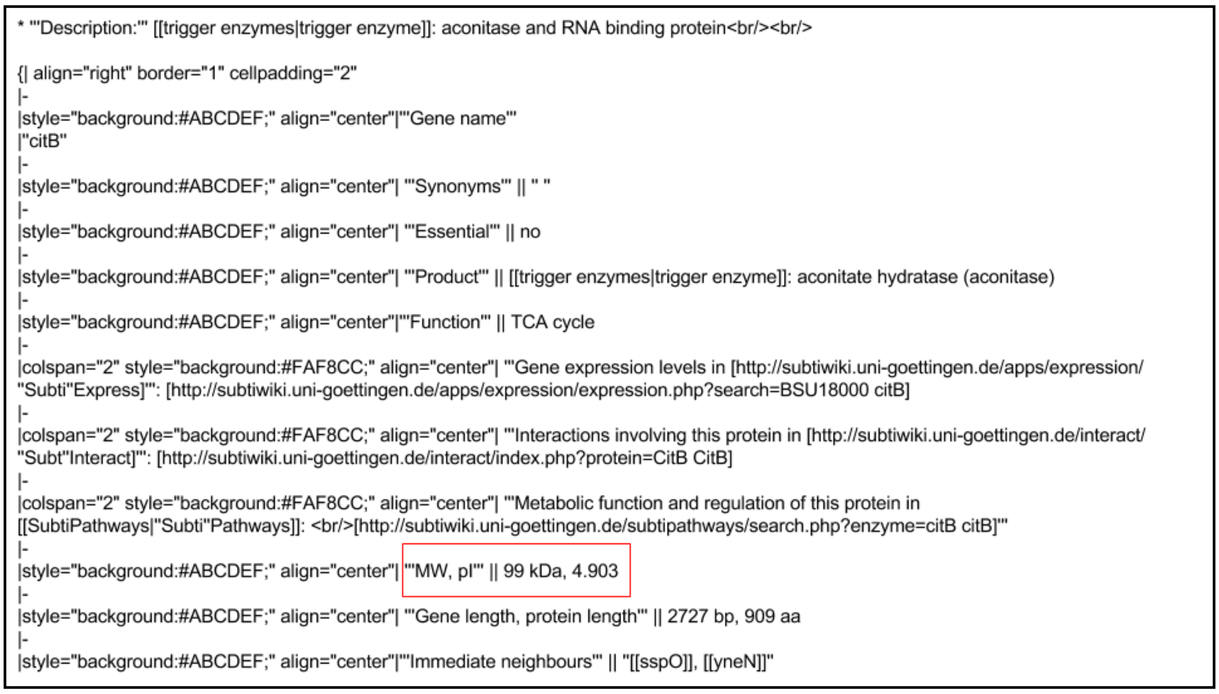
Over time information on metabolic and regulatory actions of a protein were added but that information was not significant in raw text. For this purpose, new applications were developed to visualize the information and to draw the reactions in metabolic pathway. First *Subti*Pathways (Lammers et al., 2010) was developed to display metabolic pathways and regulatory mechanisms. Actually, 50 interactive pathways are available. *Subti*Pathways utilizes the Google maps API to create a zoomable user interface to explore the metabolic pathway and to simulate interactivity. It is not totally interactive as the background is a static image where markers are positioned dynamically. The markers are clickable and offer the necessary information on the protein or the metabolite. Furthermore, due to increasing information on SPINE and yeast-two-hybrid experiments (Volker et al., 1996) new protein-protein interactions appeared and *Subt*Interact was developed to visualize these interactions. Actually, *SubtI*nteract encounters 1936 interactions and 952 involved proteins. *Subti*Express show the expression of genes under 104 conditions and the quantitative analysis of protein amount under 16 conditions.

2.9 Objectives

The structure of the *Subti*Wiki database was not appropriate for any further simple analysis or a comparison of gene properties. The simple comparison of the size of all genes was not easily possible due to the idea of the wiki structure. The properties of a gene were represented on a page in plain text. This plain text contained all the necessary information but the desired properties were cryptically hidden (see figure 2.3). Additionally, the manually curated pages make an offer for typing errors. Possibly, an implemented algorithm would not find the desired property due to a misspelling. This would lead to a loss of information. The sites also contain a large amount of styling information which burdens the search through the text. The advantage of this page is that the user can easily change the text and the site appearance.

Otherwise, excel sheets already exist with columns describing the properties of the gene. Unfortunately, the sheets are also manually modified. If something changes on the gene site, the excel sheet has to be additionally altered. The maintenance of the pages is an exhausting challenge. A drastic example is the change of the name where the user has to add the name to the synonyms, to change all excel sheets, to change the names in the applications and last but not least to change the names appearing on all special pages of the wiki if it should be up to date.

In this work the idea was to keep some of the flexibility in editing but most of the properties should be integrated in a well defined database structure to allow simple analysis and to simplify maintenance. Besides of the main properties of the gene (see table 2.1) there is also secondary information covering the categories, regulons, gene, protein, regulation, expression, biological materials, labs working on this topic and references. This information is not directly describing a gene/ protein itself (see figure 2.4). Furthermore, the repository of the properties should be extended. The DNA and protein sequence as well as the positioning on the genome and several database entries are added. These properties were already available but not yet properly integrated in the structure of *Subti*Wiki. To provide the raw sequences as text on the page would not be the accurate way. The raw sequence would overflow the content of the gene page. Eventually, it is possible to create an interactive genome browser and to do simple alignment analysis using the BLAST engine.



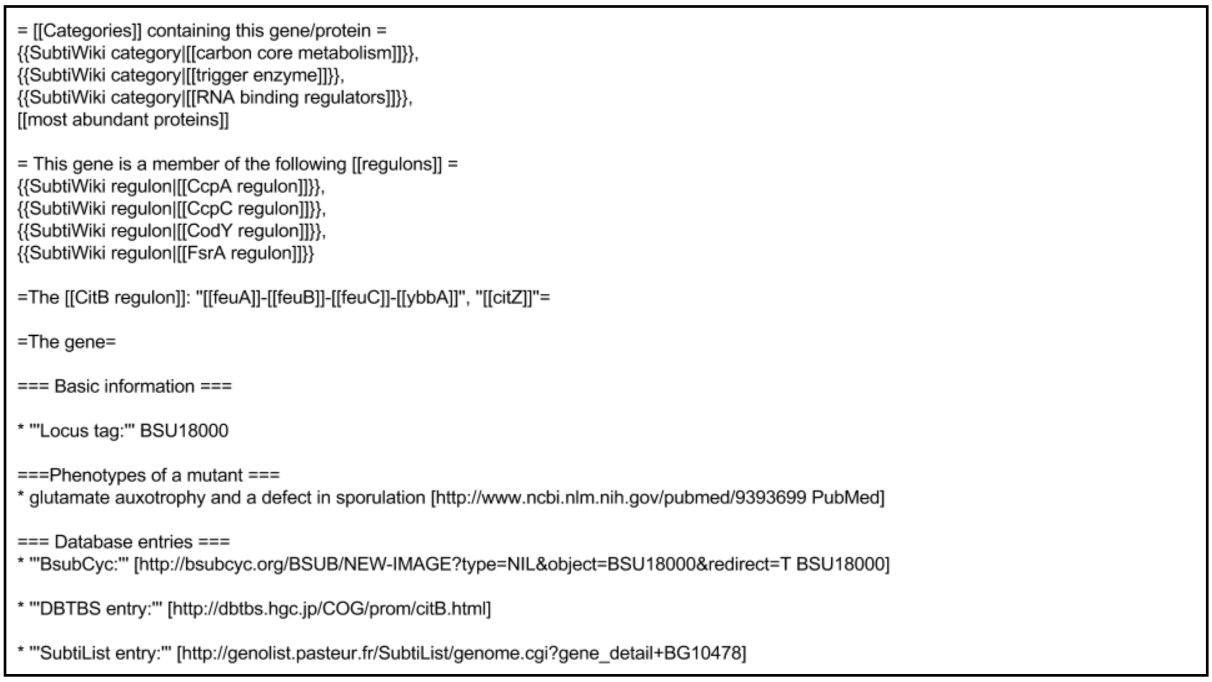
```
* '''Description:''' [[trigger enzymes|trigger enzyme]]: aconitase and RNA binding protein<br/><br/>

{| align="right" border="1" cellpadding="2"
|-
|style="background:#ABCDEF;" align="center"|'''Gene name'''
|''citB''
|-
|style="background:#ABCDEF;" align="center"| '''Synonyms''' || '' ''
|-
|style="background:#ABCDEF;" align="center"| '''Essential''' || no
|-
|style="background:#ABCDEF;" align="center"| '''Product''' || [[trigger enzymes|trigger enzyme]]: aconitate hydratase (aconitase)
|-
|style="background:#ABCDEF;" align="center"|'''Function''' || TCA cycle
|-
|colspan="2" style="background:#FAF8CC;" align="center"| '''Gene expression levels in [http://subtiwiki.uni-goettingen.de/apps/expression/
''Subti''Express]''': [http://subtiwiki.uni-goettingen.de/apps/expression/expression.php?search=BSU18000 citB]
|-
|colspan="2" style="background:#FAF8CC;" align="center"| '''Interactions involving this protein in [http://subtiwiki.uni-goettingen.de/interact/
''Subt''Interact]''': [http://subtiwiki.uni-goettingen.de/interact/index.php?protein=CitB CitB]
|-
|colspan="2" style="background:#FAF8CC;" align="center"| '''Metabolic function and regulation of this protein in
[[SubtiPathways|''Subti''Pathways]]: <br/>[http://subtiwiki.uni-goettingen.de/subtipathways/search.php?enzyme=citB citB]'''
|-
|style="background:#ABCDEF;" align="center"| '''MW, pI''' || 99 kDa, 4.903
|-
|style="background:#ABCDEF;" align="center"| '''Gene length, protein length''' || 2727 bp, 909 aa
|-
|style="background:#ABCDEF;" align="center"|'''Immediate neighbours''' || ''[[sspO]], [[yneN]]''
```

Figure 2.3: An extract of the *citB* page. To find the property one has to run through the cryptic text. The text contains styling information. The box marked in red shows the area were the molecular weight and the isoelectric point are placed.

Table 2.1: The main properties of *citB*. The table shows the main properties of gene explained with gene *citB*.

Property	**Descripiton**
Name	citB
Synonyms	-
Molecular weight	99.0
Isoelectric point	4
Protein length	909
DNA length	2730
Function	TCA cycle
Product	Trigger enzyme: aconitate hydratase (aconitase)
Essential	no
EC Number	4.2.1.3

It was not the purpose to totally erase the whole wiki due to its flexibility in editing. The wiki engine is a nice tool to present not strictly organized which does not describe a gene directly. Data of labs, methods, plasmids or events is nothing which can be used for the analysis of the gene set *B. subtilis*. This data is more important for the researcher or user using a method or searching for the contact to another group. As mentioned this kind of information should not be integrated in the new structure of the database.



```
= [[Categories]] containing this gene/protein =
{{SubtiWiki category|[[carbon core metabolism]]}},
{{SubtiWiki category|[[trigger enzyme]]}},
{{SubtiWiki category|[[RNA binding regulators]]}},
[[most abundant proteins]]

= This gene is a member of the following [[regulons]] =
{{SubtiWiki regulon|[[CcpA regulon]]}},
{{SubtiWiki regulon|[[CcpC regulon]]}},
{{SubtiWiki regulon|[[CodY regulon]]}},
{{SubtiWiki regulon|[[FsrA regulon]]}}

=The [[CitB regulon]]: "[[feuA]]-[[feuB]]-[[feuC]]-[[ybbA]]", "[[citZ]]"=

=The gene=

=== Basic information ===

* '''Locus tag:''' BSU18000

===Phenotypes of a mutant ===
* glutamate auxotrophy and a defect in sporulation [http://www.ncbi.nlm.nih.gov/pubmed/9393699 PubMed]

=== Database entries ===
* '''BsubCyc:''' [http://bsubcyc.org/BSUB/NEW-IMAGE?type=NIL&object=BSU18000&redirect=T BSU18000]

* '''DBTBS entry:''' [http://dbtbs.hgc.jp/COG/prom/citB.html]

* '''SubtiList entry:''' [http://genolist.pasteur.fr/SubtiList/genome.cgi?gene_detail+BG10478]
```

Figure 2.4: The flexible lower part of the *citB* page. This is an extract of the lower part of the page showing the categories, regulons and basic information of the gene.

As mentioned in previous chapters the applications *Subti*Pathways, *Subti*Express, *Subt*Interact were developed afterwards and totally new databases were created. These applications were totally separated from the *Subti*Wiki database. The maintenance was inconvenient due to the unrelated databases. For example, if the name of a gene changed, the entry in the applications would not be changed. This is the case for all three applications. The aim of this part was to fuse all tables of the applications and the wiki into one database. The gene and its main properties should be the content of the central table. All the applications should be tables in the database which are directly related to each other.

The first step is to create a database structure that fits to the organism *B. subtilis*. Next, the information has to be extracted from the existing pages via text mining. Then the data has to be correctly integrated in the database. Finally, the whole platform has to be created to display the information in a web application. *Subti*Wiki already contains a large knowledgebase but the information is cryptically hidden in the “Wiki” syntax which cannot be used for bioinformatic approaches.

As mentioned in previous chapters the applications *Subti*Pathways, *Subti*Express, *Subt*Interact were developed afterwards and separately developed. Due to that these applications were totally not related to the *Subti*Wiki database. There the same problem appeared as mentioned before. If the name of a gene changed, the content of the applications would not be changed. This is the case for all three applications. The aim of this part was to fuse the information content of all applications and the wiki and create a related database. The gene and its main properties should be the central and unique property for all applications. All the applications should refer to the same data source.

The first step is to create a database structure that fits to the organism *B. subtilis*. Next, the information has to be extracted from the existing pages via text mining approaches. Then the data has to be correctly integrated in the database. Finally, the whole web platform has to be created to display the information in a web application.

3. Material and methods

3.1 Linux

Linux is a unix-like operating system and was designed for free and open source development and distribution. The first linux operating system was published by Linus Torvalds in 1992. Over time versions based on linux appeared like ubuntu, xubuntu, debian, openSuse and many more. In this work Ubuntu (http://ubuntu.com) was used as the operating system. Ubuntu is freely available and contains a lot of preinstalled programming tools like the fundamental shell which is a command line of the era before the graphical desktop.

Table 3.1: Short list of Shell commands. The list shows some of the most frequently used commands during this work.

Command	Description
ls	Show a list of all the directories, files and elements in the current directory. A detailed list can achieved by adding the option "-la"
cd	With this command the user can change to another directory.
cp	This command enables the user to copy one element to another directory.
mv	The command moves an element or whole directory to another place.
rmdir	Remove an empty folder
rm	Remove an element
scp	Copy files to an external host
ssh	Connect to an external host
chmod	Change the access rights for users
chgrp	Change the group to which the file or directory belongs to
chown	Change the owner of a file or directory

3.1.1 Gedit

Gedit is a preinstalled text editor which was used to write the scripts. It offers simple highlighting of the corresponding programming language (C, C++, PHP, Python and many more). Furthermore, it has the typical functionalities of an editor like undo and redo, search and replace, auto indentation and many more (https://wiki.gnome.org/Apps/Gedit). Another advantage is the graphical interface which keeps the handling of the written document quite simple. The editors without a graphical interface like “vim” or “editor” are controlled via and in the shell which make it difficult to handle.

3.1.2 Terminal

The terminal is a fundamental command line used to run specific commands. It is commonly used on linux operating systems. It is an artifact of the era where the personal computer did not have a graphical interface and user was forced to tip in the commands by hand. For the inexperienced programmer the handling of the shell seems to be unhandy but it speeds up many processes like installation of software or copying files. A couple of commonly used commands are shown in the table 3.1.

3.1.3 Python

Python is a free programming language which is easy to learn. Moreover, there are many offered libraries, which contain codes that can be used. It is a language which has a defined syntax but it is not as comprehensive as the programming design in Java or C. In Python the user does not have to compile the program first, it compiles on run-time. These advantages make Python so popular and it is a good programming language to start with. The user is able to run the code and is also able to use a large pool of the Python library which contains tools to handle web sites, pictures, databases, diverse data formats, regular expressions and many more (http://www.python.org).

```
#!/usr/bin/python
# The first line is called the shebang line which is simply telling the system how to interpret the
# script.
print  “hello”                  # prints out the string hello
for i in range (1, 10):  # for loop running ten rounds
# The following indentation shows the affinity to the for loop as well as the leading colon
        print i                 # prints out number from 1 to 9
```

Figure 3.1: Introductory python script. The script prints out the word “hello” on the screen and runs a loop and prints out the variable “i” 9 times, thereby the value of the “i” is increasing by one and the number is print on the screen.

The lines starting with a “#” are comments and are not interpreted by the system, except the first line which tells the operating system which programming language should be used. In this case the system is using the python interpreter. Additionally, there are many predefined functions like “print” which is a command to print out something on screen for example “Hello” as shown above.

3.1.3.1 IDLE

IDLE is an integrated development environment for Python. It was not preinstalled on the Linux running system. The syntax highlighting is one of the features which are very helpful to write the necessary python scripts .Additionally it is totally written in Python and the Tkinter GUI toolkit. Tkinter is tool kit to handle graphical surfaces in python (https://wiki.python.org/moin/TkInter). Furthermore, the syntax check and the auto completion are very handy features to improve scripting (https://docs.python.org/2/library/idle.html).

3.1.3.2 Regular expressions in python

The package re is one of the preinstalled software source (https://docs.python.org/2/library/re.html). This package handles regular expressions. The user is able to search, to substitute or to delete patterns in a text, which is often used in the field of text mining. Regular expressions are helpful if the set of documents contains the same structured information but the language if diverging through the documents. If there is a difference in upper or lowercase or if the length of the patterns differs, the regular expression will still fit without losing information.

```
#!/usr/bin/python
import re # importing the package of the regualar expressions
text = “ATATATACCCTTTA” # the text where the pattern is
pattern = r“C{3}” # pattern we are looking for
match = re.findall(pattern, text) # search for the pattern in the text
print match # print out the result
```

Figure 3.2: The usage of regular expressions. First the package “re” has to be imported. The following script shows how one could find a pattern containing three times a “C” in the sequence of letters. In the end the match will be printed on the screen.

3.1.3.3 MySQL in python

MySQLdb (http://mysql-python.sourceforge.net/MySQLdb.html) is a python package managing database queries. This package offers an interface between the programming language python and the database language MySQL. The connection to the database can be easily established to use the usual syntax of MySQL. With this package the user can simply manipulate the database.

```
#!/usr/bin/python
import MySQLdb # import the package named MySQLdb
# connect to the database at localhost named testdb
con = MySQLdb.connect('localhost', 'testuser', 'test623', 'testdb');
cur = con.cursor()
# execute a statement to select the current version of MySQL
cur.execute("SELECT VERSION()")
# fetch the result
ver = cur.fetchone()
# print out the result
print "Database version : %s " % ver
```

Figure 3.3: The usage of the MySQLdb package. First the package has to be imported. Then the connection is established between the database server and the script. Afterwards, a query is executed to receive the information on the version of the used MySQL. Finally, the version will be displayed on the screen.

3.1.3.4 Dictionaries in python

A dictionary in python is a key-value pair. Information can be stored in a structured way (see figure 3.4). Due to the fact that the structure of a dictionary in python is nearly the same as the JavaScript object notation (JSON) it is easy to transfer one format into the other. There is also a package available which deals with operations on the JSON object in python. The name of the script is json (https://docs.python.org/2/library/json.html) which was used during this work.

```
#!/user/bin/python
import json                  # import the package
dict = {'name': 'citB', 'mw': 99.0} # dictionary in python
js = json.dumps(dict)
# {'name': 'citB', 'mw': 99.0}  the same in JavaScript
print js            # print out the JSON object
```

Figure 3.4: The usage of the json package. First the package is imported. Then the variable named “js” is the JSON object. The dictionary named “dict” is translated to a JSON object via the method “dumps”. Print it out.

3.1.4 MySQL

SQL is the abbreviation for “Structured Query Language” (https://www.mysql.de/). It is a database language which stores data in a relational database. That means that tables of the database are related to each other. In contrast to other programming language the content in the database is stored permanently. SQL is intuitive and the language is easily understandable and readable. The commands of the language are marked in red. Consequently, it is a database management system which can interactively deliver, store and delete information but it additionally controls transactions and contribute rights to the user. That means that one can control which actions can be executed on the tables and which parts can be seen to prevent vandalism or leaking.

```
1) Create a database named Test:
CREATE DATABASE Test;
2) Create a table
CREATE TABLE employee (id INT(6), first_name VARCHAR(255), last_name VARCHAR(255), age INT(3));
3) Insert something in the table to create a new entry
INSERT INTO employee (id, first_name, last_name, age) VALUES (1, 'Raphael', 'Michna', 29);
4) Update a table to change an existing entry
UPDATE employee SET first_name=”Raphael Heinrich” WHERE id=1;
5) Describe the table to see what properties are in the table
DESC employee;
6) Select something from the table
SELECT first_name, last_name FROM employee WHERE id=1;
7) Delete something from a table
DELETE FROM employee WHERE id=1;
8) Delete a whole table
DROP TABLE employee;
9) Delete the whole database
DROP DATABASE Test
```

Figure 3.5: Collection of typical SQL queries. The figure shows a manual how to change, create or delete tables and databases.

It is often used in combination with an Apache server to present the information on a web site. The combination of a linux operation system, the Apache HTTP server (https://httpd.apache.org/), MySQL and the programming language PHP are a common concept for web services. The components can be easily exchanged and offer the opportunity to build up dynamic and interactive web applications.

To visualize the relations between the tables an entity-relationship model is used. Thereby, the entity and the relations are tables filled with information (see figure 3.6).

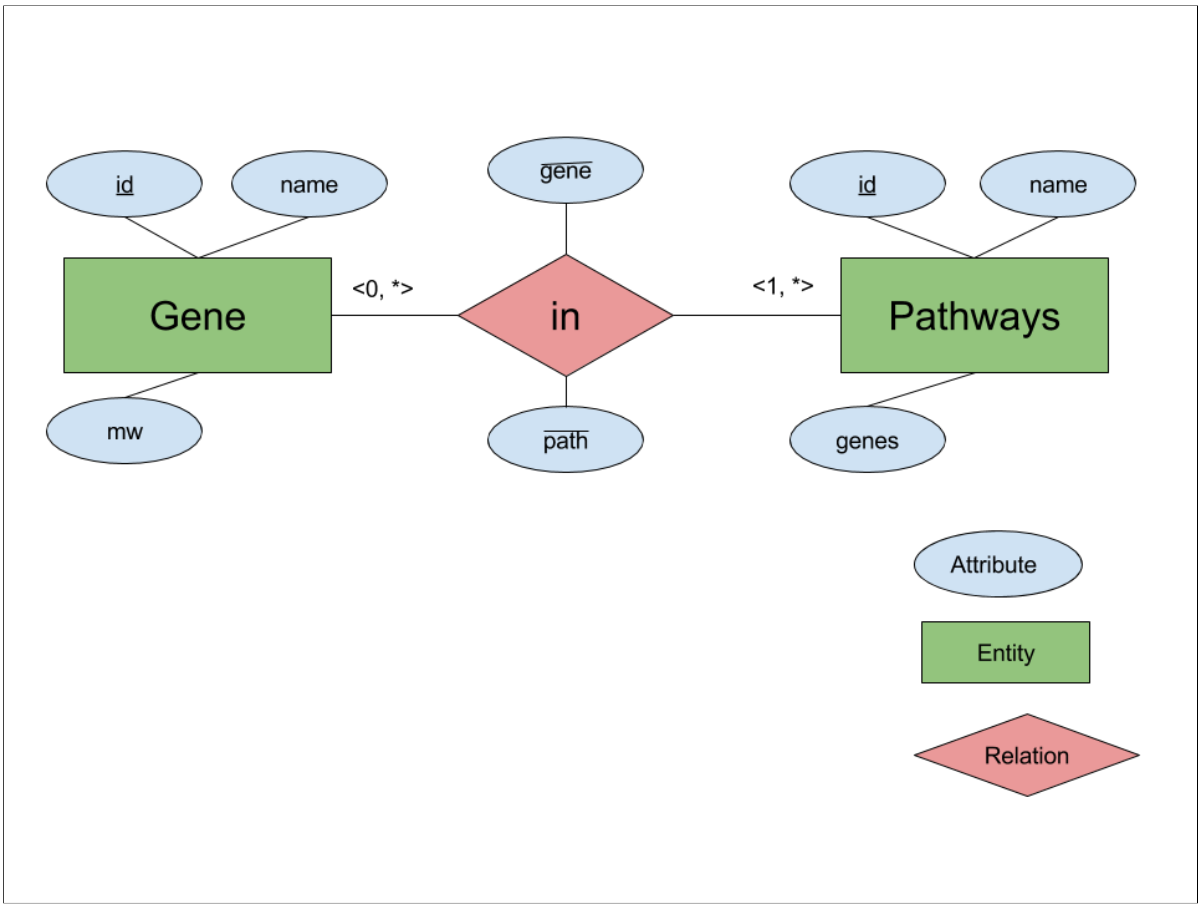
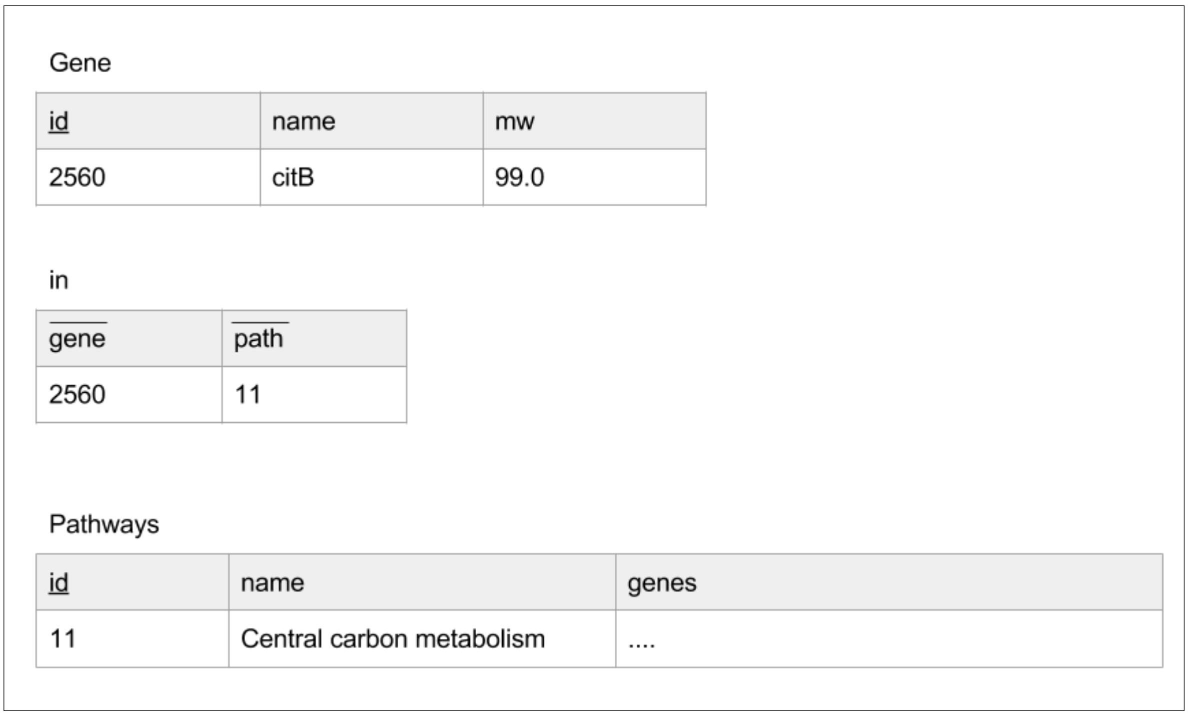


Figure 3.6: Scheme of an entity-relationship model. The scheme shows the entity-relationship model between the entity “Gene” and “Pathways”. Furthermore, a relation named “in” shows how the two tables are connected. The foreign key “path” references “id” of the entity “Pathways” and the foreign key “gene” references “id” of the entity “Gene”. The attributes with the underline are called primary key and the ones with the overline are called the foreign keys pointing on a primary key of another table.

Additionally, the entities and relations contain attributes that represent the columns in the tables (see figure 3.6). The underlined attributes of the entities represent the primary keys. The primary keys uniquely identify an entry in a table and a table can only have one primary key. The foreign keys (the attributes with the overline) mostly point to the primary keys another table. There can be more than one foreign key in a table referencing to different tables. Now, there is a real connection between the tables of two entities.



Gene

id	name	mw
2560	citB	99.0

in

$\overline{\text{gene}}$	$\overline{\text{path}}$
2560	11

Pathways

id	name	genes
11	Central carbon metabolism	....

Figure 3.7: The tables of the ER. The tables were constructed on the basis of the ER model. The columns were filled with an example using citB which is involved in the central carbon metabolism. The table “in” shows the interface between the two entities “Gene” and “Pathways”. The tables are another type of representation.

The cardinality is an additional description of the relation. The cardinality is describing how often an entity is involved in a relationship with other entities. In figure 3.6 the cardinalities are written on the lines between an entity and a relation. There are three ways how a relationship could look like. First, the relationship can be a one to one relationship, e.g. each gene is in exactly one pathway. Second, the relationship can be a one to many, e.g. each gene is involved in more than one pathway. Finally, the relationship can be a many to many, e.g. many genes are involved in more than one pathway. In the ER model the syntax “<0, *>” can be read as “A gene cannot be involved or more than once in a pathway” (<minimum, maximum>). With this scheme in mind, the tables for the database can be easily constructed. The following example shows how the framework for the tables is now implemented with MySQL statements (see figure 3.8).

```
CREATE TABLE Gene (    id int(5) NOT NULL AUTO_INCREMENT,
                                name VARCHAR(255),
                                mw FLOAT(8,2),
                                PRIMARY KEY (id)
                           );  # The statement to create the "Gene" table
CREATE TABLE Pathway ( id int(4) NOT NULL AUTO_INCREMENT,
                                name VARCHAR (255),
                                genes BLOB,
                                PRIMARY KEY (id)
);  # The statement to create the "Pathway" table
CREATE TABLE in (gene int(5) REFERENCES Gene(id) ,
                       pathway int(4) REFERENCES Pathway (id)
); # The statement to create the "in" table with the references to "Gene" and "Pathway" tables
```

Figure 3.8: The implementation of the entity-relationship model. The figure shows how the three tables named “Gene”, “Pathway” and “in” are created. Furthermore, in the tables ”Pathway” and “Gene” the “id” columns become the primary key. Now, the pathways and genes can be uniquely identified by an index. The “in” table contains two columns including “gene” the foreign key referencing on the “Gene.id” and “pathway” the foreign key referencing on the “Pathway.id”

3.2 Hypertext markup language and cascading style sheet

The Hypertext markup language (HTML) is a markup language which can be parsed by the browser. Due to the variance of browsers like firefox, opera or chrome the interpretation can differ among them. Additionally, the user is requesting the page via media devices like desktop computer, smart phones, tablets etc. Therefore a cascading style sheet is defined where the programmer can determine how to react on different devices. The styling can be inserted in the web page directly but mostly it is externally stored in a separate file. Now, one can design the page by manipulating elements. The elements are called tags (e.g. <html>, <title>). They have an opening and a closing tag like <html> ...</html>. The tags can also have ids or class to specify them (e.g. <div id=”test”>). With this information the designer can create step by step a web page with the proper shape. The example below shows how the div tag is styled with red background color, a white font color and the font size is a bit increased.

```
<html>
<head>
        <title>Start</title>
        <style>
        #test{
                background-color: red;
                color: white;
                font-size: 1.2em;
        }
        </style>
</head>
<body>
        <div id="test" >Hello, welcome to <i>Subti</i>Wiki</div>
</body>
</html>
```

Figure 3.9: A simple website. The HTML file represents the source code of simple website. The significant tags define areas used in HTML. In the style tag the design of the website can be defined. The div tag with the id "test" will be presented in the browser with a red background and a white font color. Furthermore the font size will be a bit increased. The style is generally stored in an external CSS file.

3.3 JavaScript

JavaScript is a free available programming language to create dynamic websites. It is used for dynamic elements on the page like popup messages. It is not related to the programming language Java. The script can be directly integrated in the HTML or it is mostly separately stored in an external file. If the code is integrated in the HTML file the lines are included in so-called script "tags". There are already many libraries available to create dynamic networks e.g. the d3 library (http://d3js.org/).

This scripting language is executed on the client side. The server is simply providing the data and JavaScript is dynamically manipulating the data to fit to the website. Therefore, there is no burden for the external host (server) by creating the website.

```
<html>
<head>
        <title>Test</title>
        <script type="text/javascript">
        function ask(){
                alert("What is your name?");
        }
        </script>
</head>
<body>
        <div id="test" onclick="ask();">Hello</div>
</body>
</html>
```

Figure 3.10: The integration of JavaScript. The script tag allows the programmer to include a script which will be executed interactively in the browser. The div tag with the id “test” can be clicked. The click event starts a script which will display a sentence on the screen. Generally the scripts are stored in external JavaScript files.

3.3.1 JSON

JavaScript Object Notation is a format defining an object and it is a common way to store information. Due to the big data problem JSON is a comfortable way to store and send large sets of data through the web. Additionally, the combination of JSON storage and Javascript applications is a popular way for interactive web performance.

As one can see in the example above (see figure 3.11) the JSON object is structured in key-value pairs ({“key”: “value”}). Nested JSON objects give the opportunity to send many objects compressed in a single dataset.

3.3.2 AJAX

AJAX is the abbreviation for “Asynchronous JavaScript and XML” which is a technique for web development to increase pace of a web application. AJAX runs on the client side and it is retrieves data from the server in the background without interfering with the static page. Ajax is not simply a technology, but it is more like a collection of technologies. HTML and CSS in combination mark up and style the delivered information. JavaScript can easily access the document object model to dynamically display the presented information and the user is able to directly interact with the content. JavaScript provides a method for exchanging data asynchronously between the client and the server to prevent reloading the whole page again.

```
# a simple example of a JSON object
{"name":"citB", "locus":"BSU18000", "function": “TCA cycle”}
# a nested example of a JSON object
{genes :          [{"name":"citB", "locus":"BSU18000", "function": “TCA cycle”},
                  {"name":"eno", "locus":"BSU33900", "function": “glycolysis”}], ...
}
```

Figure 3.11: The types of JavaScript object notation. The first example shows the simple example of JSON object containing information in key-value pairs. It contains information on the name, locus and function of a gene. The second example shows a nested JSON which contains a list of many genes. The complexity of the object depends on the data which is sent through the web.

4. Results

The main points of this work were to create a proper database model for the biological object *B. subtilis*, to extract the information from the old version of *Subti*Wiki, to partly replace the mediawiki engine by creating a new web engine (see appendix 7.1) and to alter the existing applications to fit to the new database model. The database model was changed to prepare it for bioinformatic approaches and simplify the data maintenance. The resulting database is not classical "Wiki" anymore but the users are still able to edit and to participate. The existing applications were rewritten to perform uniformly with the same source of data

4.1 Construction of the database

The first step was to define the main structure for the database to provide the necessary information in the proper way. The entity-relationship model visualizes the relations between the tables of the database (see appendix 7.2.1). The main problem in the older version was that the gene pages of *Subti*Wiki where just connected via hard links and so by update of the name or another property the applications have to be modified by hand and for the user for example it was not possible to search for gene/ protein with a proper range. As a result, the centralization and the indexing of the genes were the main goals of the work. Due to that the information has to be extracted from the current version of *Subti*Wiki and inserted in the newly constructed database named "*subti*bank" (see table 4.1).

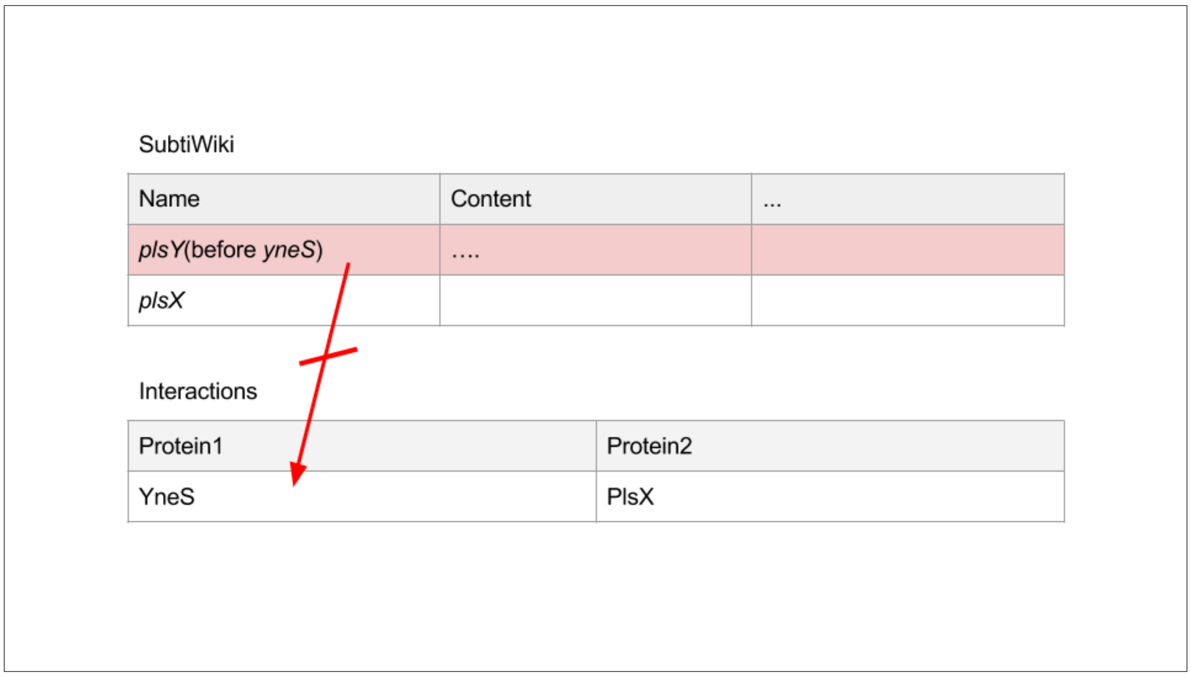
Table 4.1: Tables of *subti*bank. The left column contains the name of the newly constructed table and the right column contains the description.

Name	Function
Gene	It is the central table containing the main information on each gene. Several columns describe the function, molecular weight, iso-electric point, product etc. Each gene has a unique index number to prevent the possibility of confusion.
Metabolites	This table is a first approach to integrate the metabolites in the same manner as the genes/ proteins. The table contains unique index number, the name and the PubChem number (https://pubchem.ncbi.nlm.nih.gov/) to have access to the structure and other properties of the metabolites.
Browser	This table contains the information on the positioning of the gene in the genome. Therefore the transcription start and stop as well as the strain are included. Additonally the index number of the "Gene" table is

	referencing to the corresponding genetic coordinates relative to the origin of replication.
Downshift	This table is describing the position of a transcriptional down shift. This means the transcription is interrupted. An additional column is shows on which strand it is happening.
Upshift	It is the opposite of the “Downshift” table. The coordinates show the start of transcript.
Pathways	This table has three columns containing the unique index number, the name as the topic and a column named “json” which contains the coordinates of the markers that should be projected on the interactive maps.
in_path	This is a joined table between “Pathways” and “Gene” (“Metabolites”) which is used to facilitate the search of genes/ metabolites in the genetic pathways. An additional column named “type” describes if the object the user is searching for whether a gene or metabolite.
Interact	This is the table for the protein-protein interaction data. The columns contain the index number of two different genes and a “Pubmed”-number which confirmed the interaction by a published reference.
Proteomics	The first reference contains again the index number of the “Gene” table referencing to the gene/ protein of interest. Then there are 16 columns which represent 16 different conditions. The total amount of protein molecules per cell was measured and the values are stored in these columns.
Transcriptomics	The first reference contains again the index number of the “Gene” table referencing to the gene/ protein of interest. Then there are 104 columns which represent 104 different conditions. The gene expression levels were measured and the values are stored in these columns.
T_con	This table is related to the table “Transcriptomics”. The rows in this table describe the conditions which were tested and measured in the “Transcriptomics”. Moreover, a “Pubmed”-number is added to have a published reference for confirmation.
P_con	This table is related to the table “Proteomics”. The rows in this table describe the conditions which were tested and measured in the “Proteomics”. Moreover, a “Pubmed”-number is added to have a published reference for confirmation.
User	The aim of this table is only to register the amount of edits a user fulfills. Therefore, the *Subti*Wiki nick name is saved as well as the number of edits.
History	The table was implemented to have a closer look on the edits that were done previously. The whole gene object is saved in a JSON object. That means there is a JSON object of the new version, the old version and a backup version. The backup was implemented to have a safe version the whole system. Additionally, the user, the time and the frequency of edits

	are registered in the table.
Page	This table is registering how many times a page is clicked and in the end how many clicks in total are attempted.
Renaming	If the function of an unknown gene was discovered mostly the name changes due the associated functionality. In this case, the table saves the index number of gene, the new name, the old name and a “Pubmed”-number to confirm the change by a publication. Furthermore, the the user and the time are registered to control this renaming action.

The central table in the newly constructed database was “Gene” (see appendix 7.2.2) which contains defined columns to store information on name, synonyms, function, product, molecular weight and other properties. Additionally, each gene was indexed to obtain the uniqueness of the gene. Now, the gene cannot be confused any longer due to renaming (see figure 4.1).



SubtiWiki

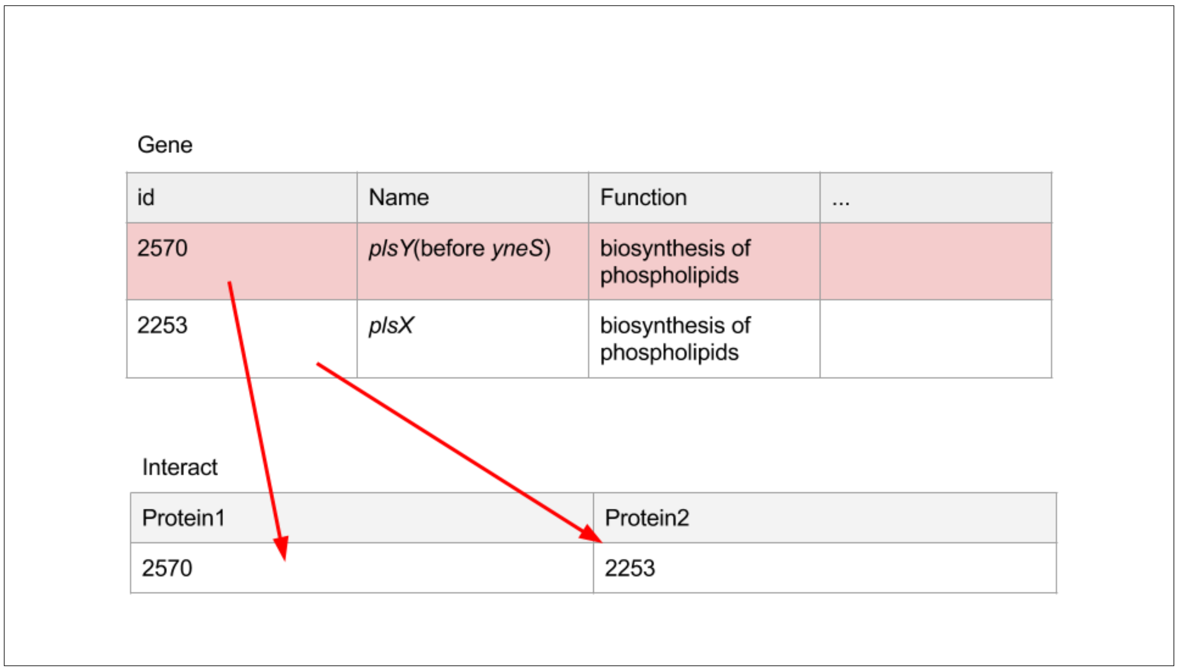
Name	Content	...
plsY(before *yneS*)	….	
plsX		

Interactions

Protein1	Protein2
YneS	PlsX

Figure 4.1: Confusion of names. The example shows to snippets of the tables “SubtiWiki” and the table “Interactions”. In “SubtiWiki” the name of *yneS* was changed, but “Interactions” still contains the entry with the old name. The interaction partners of *plsY* would not be found in the interaction table.

In the older version, the tables of *Subti*Wiki and the other applications were totally separated which would lead to different names present in the tables (see figure 4.2). As a result, the modification of the gene name should not lead to confusions between the different applications.


Gene

id	Name	Function	...
2570	*plsY*(before *yneS*)	biosynthesis of phospholipids	
2253	*plsX*	biosynthesis of phospholipids	

Interact

Protein1	Protein2
2570	2253

Figure 4.2: Indexing of genes. In the new structure of *Subti*Wiki the gene obtain a unique index and in all applications only the indexes were used to prevent confusions. The tables are not complete for further information (see the appendix 7.2.2).

In the next step the tables for the protein-protein interactions, expression and pathways had to be constructed which do not contain the name of the genes or a synonym any more but the index of the “Gene” table to guarantee uniqueness. The table for the protein-protein interactions called “Interact” was extended by the column “pubmed” to directly provide a reference to confirm the interaction. Following, a python scripts looked for the index number in the “Gene” table and replaced the names in the “Interact” table with the corresponding index. Sometimes the program could not find the corresponding name in the “Gene” table due to the varying names and therefore, a few entries had to be done manually.

*Subti*Pathways also obtains another database table. In the older version each pathway was stored in a single table, leading to a large accumulation of tables in the database. The structure was changed in a way that all 50 pathways are now stored in a single table named “Pathways”. All the markers are stored in the JSON format which is sent to the client-side. The JSON object contains the indexes of genes/proteins and the indexes of metabolites. To facilitate the search of genes/ proteins or metabolites a table “in_path” was constructed to look for the places where the desired protein or metabolite appears in the metabolism.

*Subti*Express had the right structure and so the tables were just renamed to "Transcriptomics", "Proteomics" providing the expression values of the genes/proteins and "T_con", "P_con" providing the description of the conditions.

With this a related database was constructed and had to be filled with the relevant information. Therefore, a python script runs through the existing content of *Subti*Wiki. That procedure will be described in the next chapter.

4.2 Extracting data from *Subti*Wiki

To receive the information from *Subti*Wiki the whole database was dumped to have a similar database on the local server. In figure 4.3 the shell commands are presented to backup and to restore a database. The content from *Subti*Wiki was dumped (see figure 4.3) to work on the content on the local system.

```
# Backup and restore a database


mysqldump -u root -p subtiwiki > subtiwiki.sql # backup a database
mysql -u root -p subtiwiki < subtiwiki.sql # restore a database
```

Figure 4.3: Backup and restore of a database. The shell commands show how to dump a whole database and how to restore it again. The comments are marked in red with a leading sharp.

The old database of *Subti*Wiki contains the relevant information on the genes, categories, regulons and many more. The most relevant information which should be newly organized in a defined structure was the gene page. This content should be extracted and then organized in a defined database structure. Therefore, python scripts (see appendix 7.3) were used to load the gene page from the database, read the text, extract the information and insert it in "*subti*bank". First the content was loaded and then saved as an external text file. Then the algorithm searches for the relevant information in the single text files representing one single gene. The regular expressions were used to find the right positions of the pattern. Some properties of the gene were first inserted in separate columns like the molecular weight, isoelectric point, function and many more (see table 4.2). Now, it is much easier to auto-generate lists and to compare the properties like for example the molecular weight or the isoelectric point. Before the new structure, the sheets with this content were manually curated. Still the sheets on categories and regulons are manually curated but this will be part of the discussion.

Table 4.2: The content of the gene *citB*. The sheet shows the columns of the newly generated database and its content for the gene *citB*. The column "json" stores the content which is written in the newly created syntax. The other columns are further properties of the gene.

Column name	**Content**
id	2560
name	citB
locus	BSU18000
func	TCA cycle
mw	99.0
pI	4.0
Essential	no
product	trigger enzyme: aconitate hydratase (aconitase)
dna	ATGGCAAACGAGCAAAAAACTGCAGCAAAAGACG...
aminos	ANEQKTAAKDVFQARKTFTTNGKTYHYYSLKALEDS...
json	==[SW|Categories] containing this gene/protein== [SW|carbon core metabolism], [SW|trigger enzyme], [SW|RNA binding regulators], [SW|most abundant proteins] ==Gene== ...
ec	E.C.4.2.1.3
uniprot	P09339
subtilist	BG10478
path	citB_1926680_1929409_1
syn	-

Fortunately, the information on *Subti*Wiki was well structured which facilitated the extraction of the data. The old content was simply rewritten to fit to the newly designed syntax (see appendix 7.6). It should be similar to the markup language of the mediawiki engine. Additionally, a JSON object of all properties shown in table 4.2 was created to fill the columns of the "History"-table. As a predefined setting all three columns (new, old, backup) were filled with the same JSON object.

Additionally, the protein-protein interactions had to be changed. In the old interaction tables many protein names did not change over time because the columns of the different databases were related. Therefore, a script had to look for the right name and especially for the right index to replace the name in the column with the corresponding index. Additionally, it erases the double annotated interactions and extracted the Pubmed number from the corresponding gene page if there was any annotated.

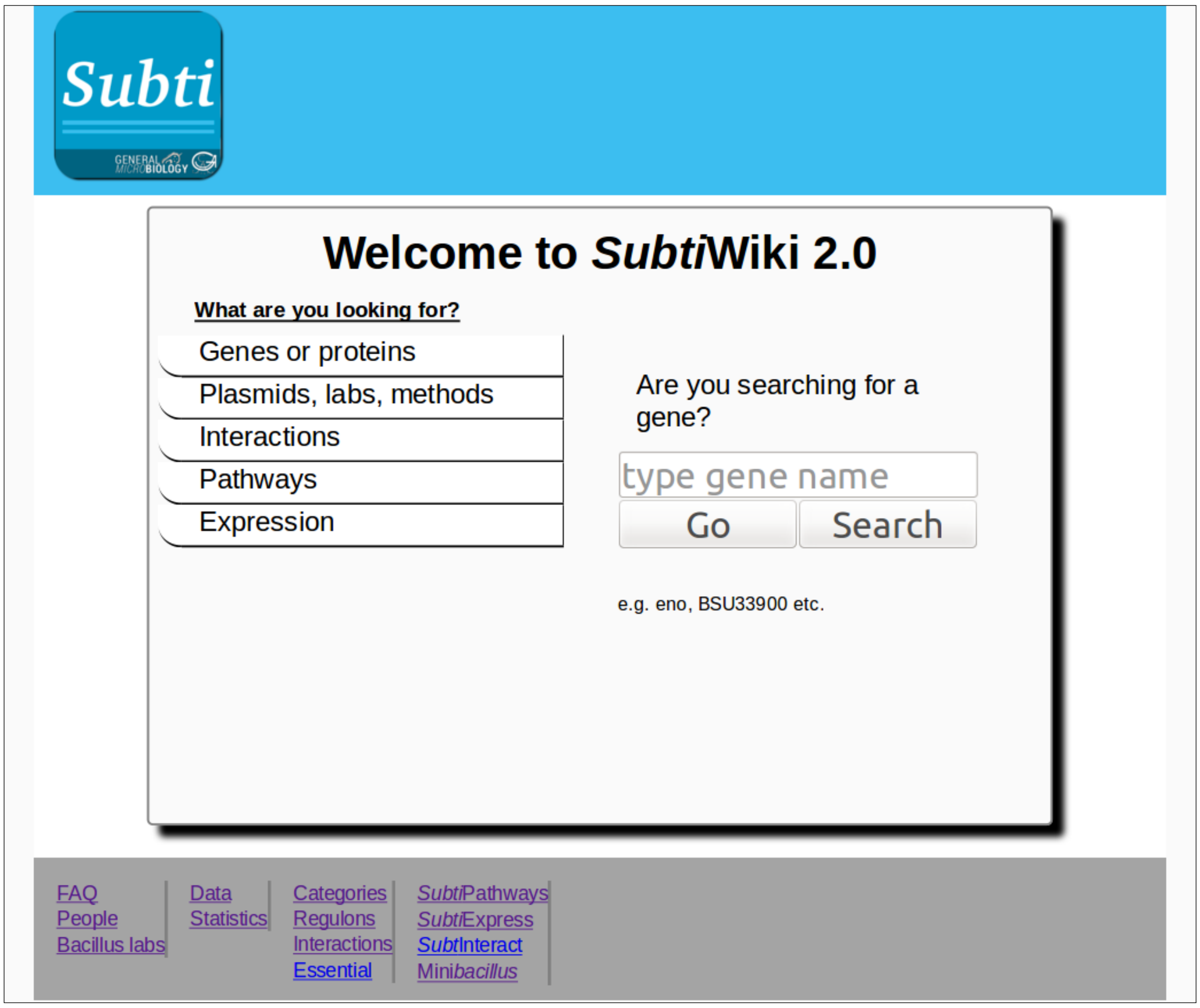


Figure 4.4: The start page of *Subti*Wiki. In the left bar the user can choose a topic and then type in the keyword. For a usual search the user can click on the "Go" button and for an extended search on the "Search" button. In the footer some general links are provided like contact, labs and FAQ.

4.3 Main page

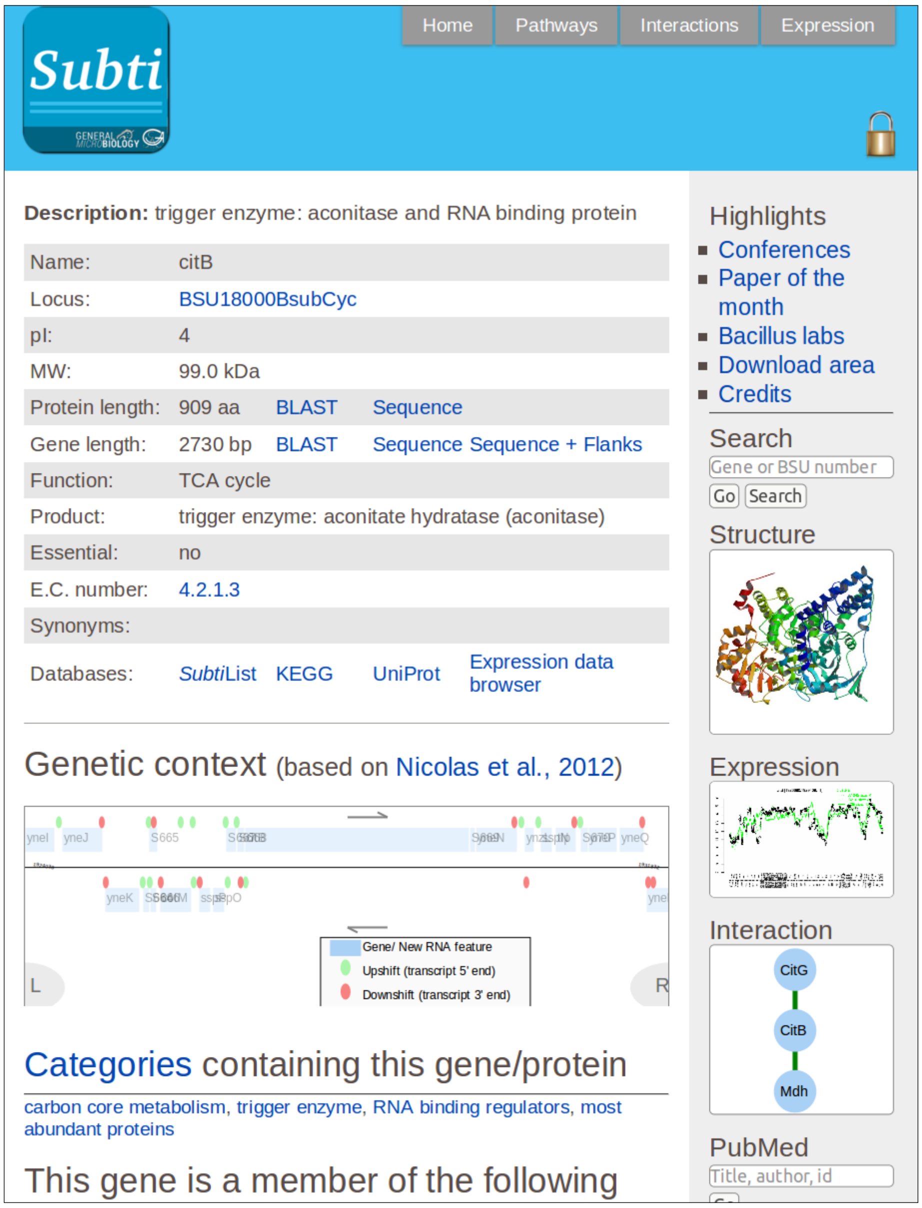
To allow the search on specific topic, the main page was newly constructed (http://www.subtiwiki.uni-goettingen.de). In the old version the main page rarely used and it was inconvenient to get to applications. On the main page the user is now able to choose between the different applications (see figure 4.4) and the user can directly start searching for a gene in a pathway or protein-protein interactions which was not possible in the previous version. Additionally, the page was adapted to different media devices to guarantee the proper presentation on each device. In the footer of the page the user can find main information on contact, statistics, data and examples of the different applications.

4.4 ***Subti*Wiki 2.0**

The previous version of the gene pages was not interactive enough. The genetic was a static image and the raw text of the wiki engine makes it difficult to compare the genes. Furthermore, the information cannot be used in other applications as in *Subti*Pathways where the info windows were filled with information on the gene. Therefore, the design and the engine had to be changed to connect the applications with the gene pages.

At the top of each page (see figure 4.5) links to the applications are provided. The general description of the gene like the main properties molecular weight, isoelectric point, function and main database entries are provided in a table underneath the header. This table also enables the user to directly blast the gene or protein sequence. At the right site of the page the links to the main events, paper of the month, credits, download area and other highlights are accessible. The link to the download area provides access to tables which contain the information on the gene, or on the interactions. The tables are auto generated and manually curated like in the previous version. Below the highlights the search box for the genes and extended search for content is provided. In the same column the structure of the protein is presented if there is any annotated one. Additionally, the expression picture (Nicolas et al., 2012) and the interactions are integrated. Below the interaction pattern there is a Pubmed (http://www.ncbi.nlm.nih.gov/pubmed) search box to accelerate the search for Pubmed specific content.

The slogan of *Subti*Wiki is still one gene one page, but now there are shown more possibilities which are provided by interactive features like the genetic context which is below the general description table (see figure 4.5). This is a first improvement of the new *Subti*Wiki database. The genetic context shows the genetic arrangement of the corresponding gene and the adjacent genes. It is covering an area of about 7000 base pairs. This feature is constructed at the client-side by JavaScript using SVG (Scalable Vector Graphics) elements (see appendix 7.4.1). The surrounding genes can be clicked to guide the user to the desired gene. Additionally, the user can move up and downstream the genome by clicking the elements at right and left bottom corner of the SVG element. Another feature is the interactive protein-protein interaction pattern using a SVG element. The user can find it on the right side of the page under the expression profile. The elements in the SVG are also clickable to guide the user to the corresponding gene. If the line between the two nodes is green there is a reference annotated to the interaction. The user can also click the line to access the paper. The interactive features are programmed in JavaScript (see appendix 7.4).



Description: trigger enzyme: aconitase and RNA binding protein

Name:	citB		
Locus:	BSU18000BsubCyc		
pI:	4		
MW:	99.0 kDa		
Protein length:	909 aa	BLAST	Sequence
Gene length:	2730 bp	BLAST	Sequence Sequence + Flanks
Function:	TCA cycle		
Product:	trigger enzyme: aconitate hydratase (aconitase)		
Essential:	no		
E.C. number:	4.2.1.3		
Synonyms:			
Databases:	*Subti*List KEGG	UniProt	Expression data browser

Genetic context (based on Nicolas et al., 2012)

Categories containing this gene/protein

carbon core metabolism, trigger enzyme, RNA binding regulators, most abundant proteins

This gene is a member of the following

Highlights
- Conferences
- Paper of the month
- Bacillus labs
- Download area
- Credits

Search

Gene or BSU number

Go Search

Structure

Expression

Interaction

PubMed

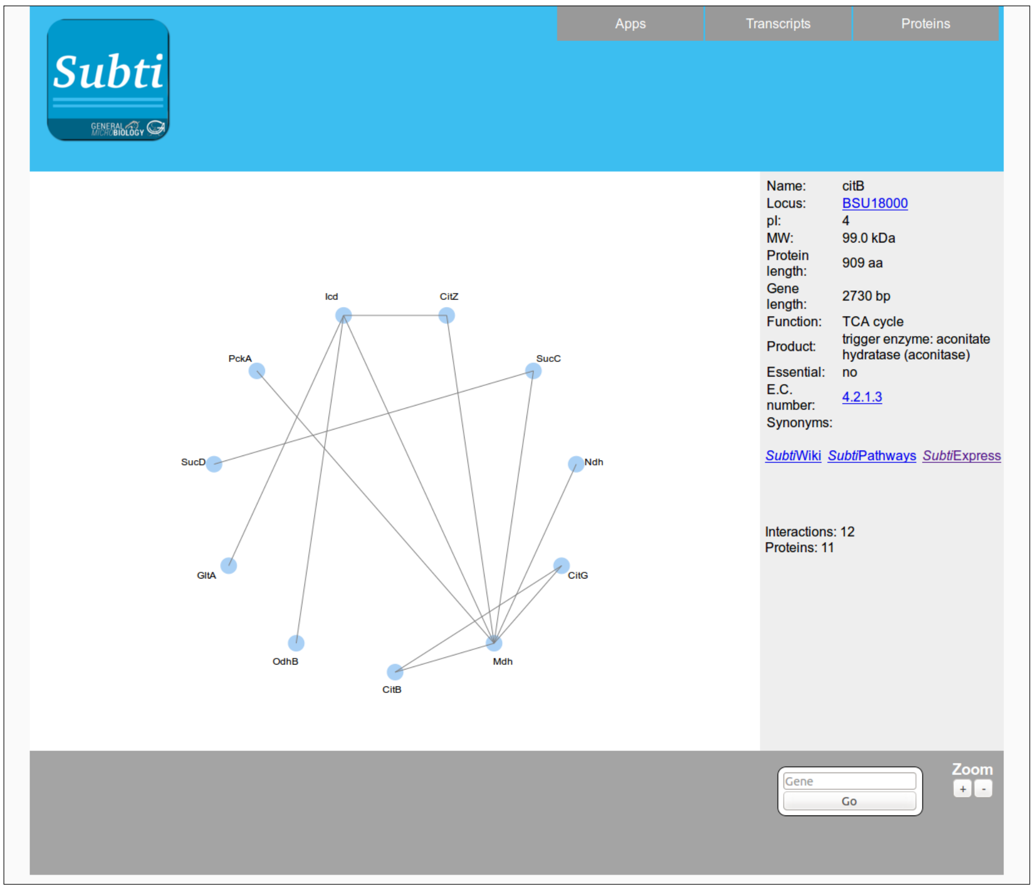
Title, author, id

Figure 4.5: The *Subti*Wiki page of *citB*. At the top the user will find the links to further applications like expression, interaction and pathways. Below the table with general information is presented. Furthermore, the user can find information on the genetic, the structure, the expression levels and interaction profile (http://subtiwiki.uni-goettingen.de/bank/index.php?gene=citB).

Under the general description there is more information on the gene. The content of this part of the page is stored in the "json" column of the "Gene" table described in the previous chapter. For this purpose, a new syntax had to be invented to allow the easy editing of the gene page (see appendix 7.6). But the content of the page is still structured but not as restricted in editing as the other properties of the “Gene”. It contains information on categories, regulons, protein, expression and regulation, biological materials and references. At the bottom of the page the user will find contact to the responsible people, contact to the creators, further applications and again database entries related to this gene.

4.5 Translating the json content

The php script (see appendix 7.5.2) receives information from the “Gene” table. With the data the php script has to create a HTML page. Therefore, the script has to search for regular expressions in the text and replace the pattern with another set of symbols. The new symbol should be readable by the browser that it can be constructed, e.g. the string “==Categories==” is translated to “<h2>Categories</h2>” (see appendix 7.6).



Name:	citB
Locus:	BSU18000
pI:	4
MW:	99.0 kDa
Protein length:	909 aa
Gene length:	2730 bp
Function:	TCA cycle
Product:	trigger enzyme: aconitate hydratase (aconitase)
Essential:	no
E.C. number:	4.2.1.3
Synonyms:	

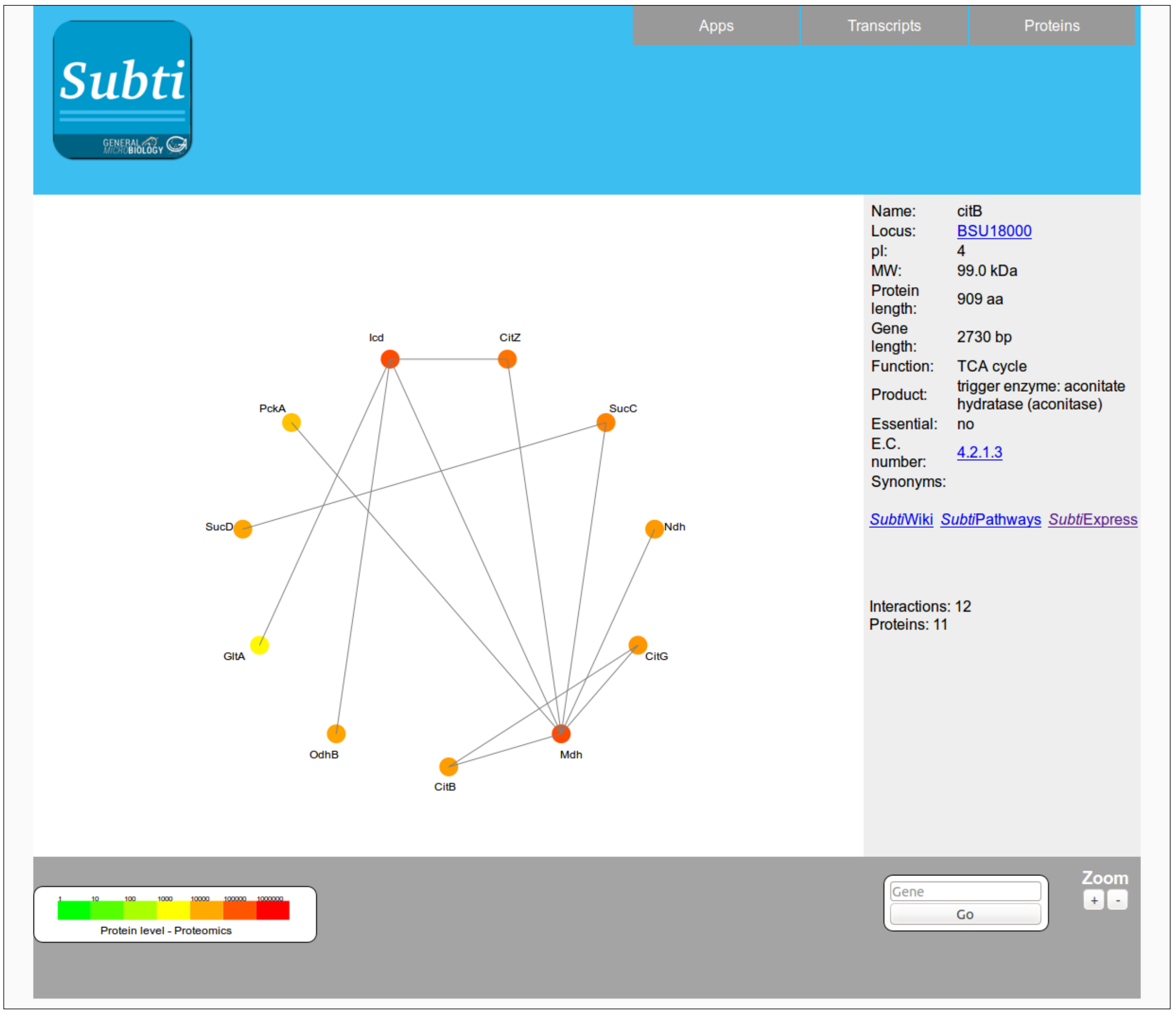
*Subti*Wiki *Subti*Pathways *Subti*Express

Interactions: 12
Proteins: 11

Figure 4.6: Simple interaction pattern of CitB. The scheme shows the interaction profile of CitB in the third zoom level. It is an interactive SVG that can be clicked. At the right the user finds a short description of the protein as well as further links to the other applications.

4.6 Interactions with “Omics”-data

This application was constructed to visualize protein-protein interactions. The previous version of *Subt*Interact was not so convenient due to an external JavaScript library (http://d3js.org/) which was too complex for the needed purposes. Therefore, an own library (see appendix 7.4.2) was created to construct an interactive interaction pattern using scalable vector graphics. The elements in the dynamic picture are again clickable. By moving over the items the user receives information of the gene in the right column of the page. This information is a shorter version of the provided description of the corresponding *Subti*Wiki page. By clicking on the node the user will be guided to the interaction pattern of the gene clicked (see figure 4.6).



Name:	citB
Locus:	BSU18000
pI:	4
MW:	99.0 kDa
Protein length:	909 aa
Gene length:	2730 bp
Function:	TCA cycle
Product:	trigger enzyme: aconitate hydratase (aconitase)
Essential:	no
E.C. number:	4.2.1.3
Synonyms:	

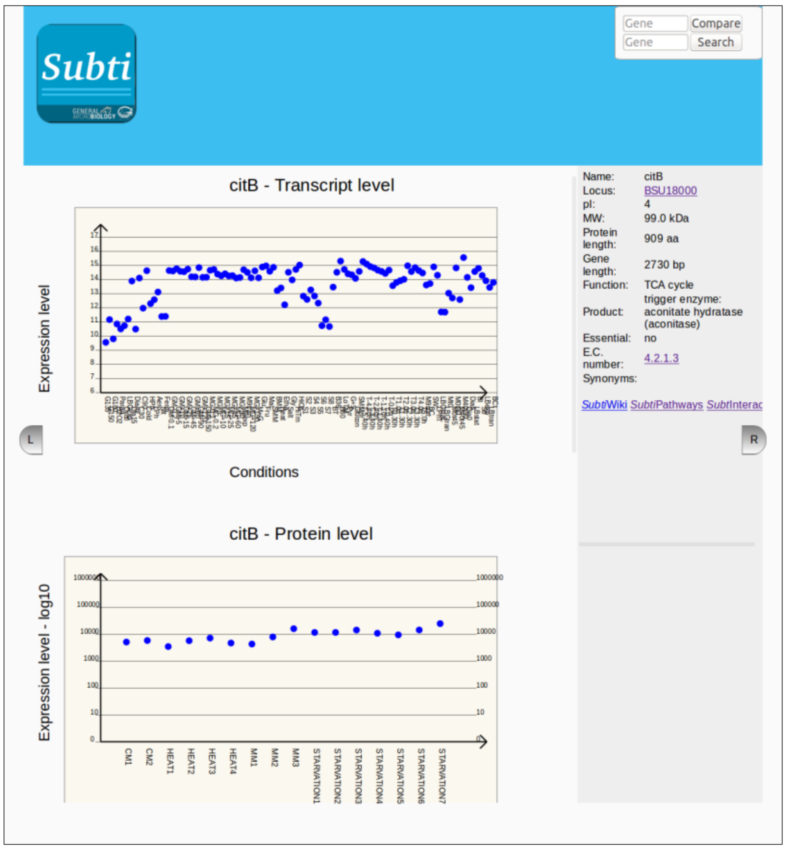
*Subti*Wiki *Subti*Pathways *Subti*Express

Interactions: 12
Proteins: 11

Figure 4.7: The interaction profile of CitB with colored nodes. On top of the interaction profile proteomic data was added to see how much of a gene is available. The nodes are now color-coded and the legend shows how much of the protein is available (logarithmic scale).

Moreover, one can zoom between the layers of interactions by clicking the plus and minus provided on each page. The first zoom level shows the directly interacting proteins, the second layer shows the adjacent proteins of the direct interaction partners and so on.

Additionally, one of the aims was to integrate the transcriptomic and proteomic data with the interaction pattern. The user can choose a condition in the upper navigation bar. The color-coded transcriptional level or the quantitative amount of protein is further described in a legend in the left-bottom corner (see figure 4.7).



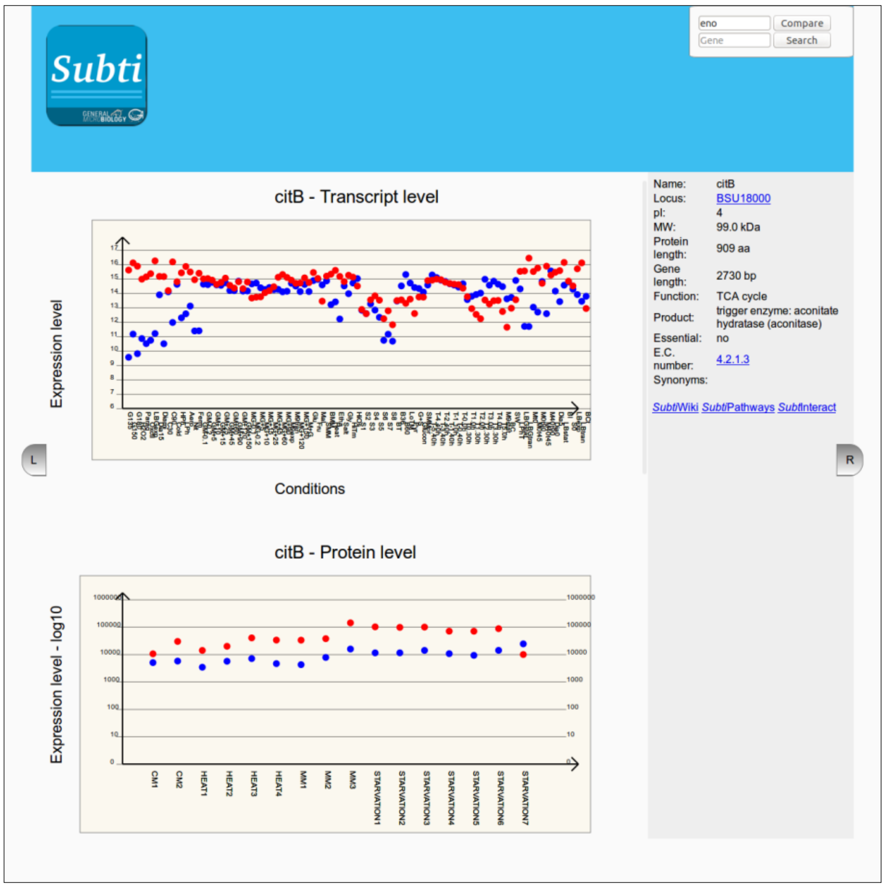
Name:	citB
Locus:	BSU18000
pI:	4
MW:	99.0 kDa
Protein length:	909 aa
Gene length:	2730 bp
Function:	TCA cycle
Product:	trigger enzyme: aconitate hydratase (aconitase)
Essential:	no
E.C. number:	4.2.1.3
Synonyms:	

*Subti*Wiki *Subti*Pathways *Subt*Interac

Figure 4.8: The expression profile of *citB*. The upper chart of the image shows the gene expression of citB under 104 conditions (Nicolas et al., 2012). Underneath the quantitative amount of protein is presented under 16 different conditions (in a logarithmic scale). In the right column a short description of the corresponding gene is presented. Under this a short description of the condition will appear if the user clicks on a dot.

4.7 Expression levels

The integration of the large amount of data provided by transcriptomic and proteomic approaches is a competitive challenge in molecular science. *Subti*Express (Nicolas et al., 2012) was a first approach to visualize the expression of a gene under specific conditions. Therefore, we generated a JavaScript library (see appendix 7.4.3) to create an interactive graph using SVG elements (see figure 4.8).



Name:	citB
Locus:	BSU18000
pI:	4
MW:	99.0 kDa
Protein length:	909 aa
Gene length:	2730 bp
Function:	TCA cycle
Product:	trigger enzyme: aconitate hydratase (aconitase)
Essential:	no
E.C. number:	4.2.1.3
Synonyms:	

SubtiWiki SubtiPathways SubtiInteract

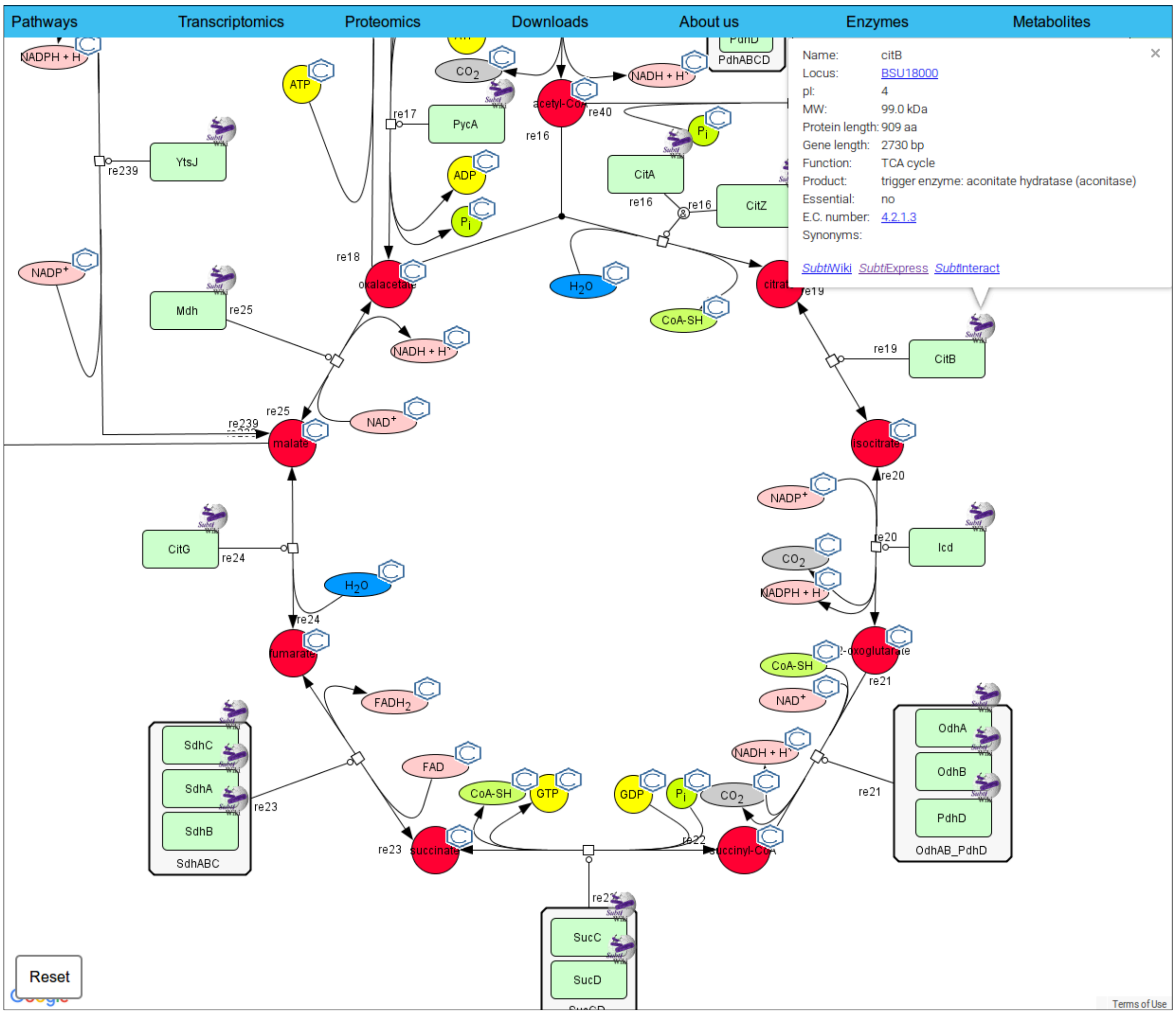
Figure 4.9: Comparison of expression profiles. The blue dots represent the expression profile of the gene/ protein *citB* and the red ones the expression profile of *eno*.

The graph contains the expression values of the desired gene. By moving over a dot in the graph the user will receive a short description of the corresponding condition and the exact value of the dot. By clicking on the dot the detailed description of the single condition will appear in the right bottom corner of the sidebar. By the way, the user will find the shortened description table of the gene also in the sidebar. A new feature is the comparison of two expression profiles provided (see figure 4.9). For this purpose, the user can use the top form field to insert a second gene and run the program to see a second set of red-colored dots.

Moreover, the page offers the information on the genomic arrangement in the surrounding of the gene by a static picture at the bottom of the page.

4.8 Pathways with "Omics"-data

*Subti*Pathways had to be rewritten due to an update of Google Maps API from version two to version three. This was an occasion to tune the system a bit and to make the application more dynamic. The JSON object of the table "Pathways" delivers the position information of each marker existing in the picture. Info windows on the markers provide a shortened description as known in the other applications (see figure 4.10). In the top navigation bar the user can access other pathways, download the XML file of this pathway or choose the metabolite or protein in a drop down menu.

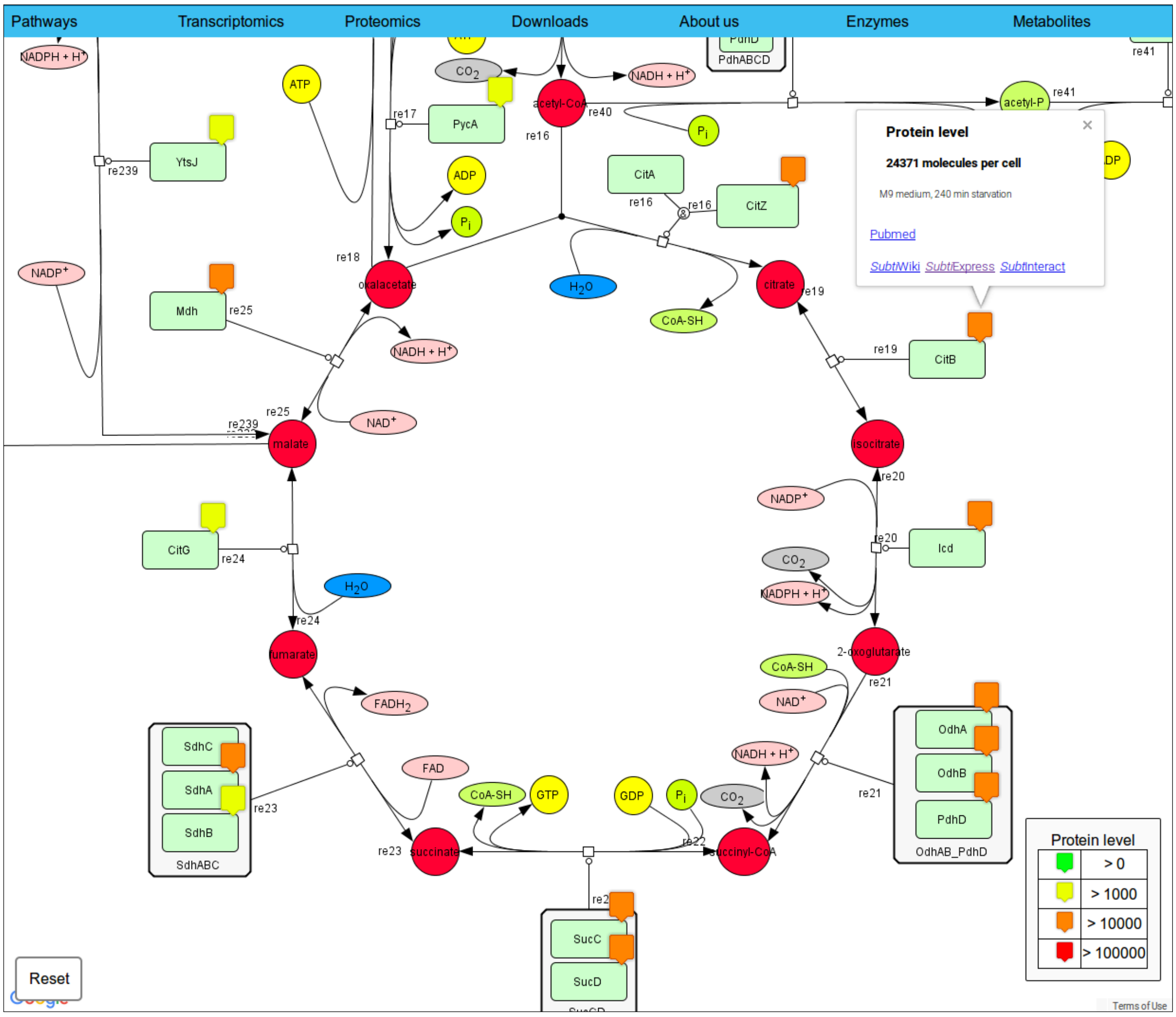


Name: citB
Locus: BSU18000
pI: 4
MW: 99.0 kDa
Protein length: 909 aa
Gene length: 2730 bp
Function: TCA cycle
Product: trigger enzyme: aconitate hydratase (aconitase)
Essential: no
E.C. number: 4.2.1.3
Synonyms:

*Subti*Wiki *Subti*Express *Subti*nteract

Figure 4.10: Central carbon metabolism of *B. subtilis.* The markers are placed on the pathway and the info window of CitB is opened up. The Info window contains the short description of the corresponding gene/ protein (http://subtiwiki.uni-goettingen.de/apps/pathway.php?pathway=11).

As in *Subt*Interact the proteomic and transcriptomic data was integrated and therefore two additional tabs in the navigation bar were included called “Transcriptomics” amd “Proteomics” (see figure 4.11). In the tabs the user can choose a condition of measured proteomic or transcriptomic data to project color-coded marker on the map. A small legend at the right bottom corner explains the color-coded marker. A legend on the right side decodes the color of the marker and by clicking the “Reset” button the user can return to the general mode.



Protein level

24371 molecules per cell

M9 medium, 240 min starvation

Pubmed

SubtiWiki SubtiExpress SubtInteract

Protein level

🟩	> 0
🟨	> 1000
🟧	> 10000
🟥	> 100000

Figure 4.11: Color-coded marker on the pathway. The legend explains the color code of the marker and by clicking on the marker the user receives the amount of protein, the description of the condition, a link to publications as well as links to the other applications of *Subti*Wiki.

4.9 Statistics

An additional feature is that a counter was integrated for the gene pages to register the number of accesses. By entering a gene page the counter in the table “Gene” is increased by one.

Also the edits are registered to control recent changes done by the pages. Therefore a php script named Statistics.php was implemented to show the attempts and edits in total as well as the most edited or visited pages.

4.10 Security issues

One important issue was to guarantee secure editing. Therefore, the mediawiki cookies were used to allow access to the editing interface of the gene pages. Due to the cookies it was not necessary to create new accounts for all already registered users in *Subti*Wiki. The user will only recognize a new interface but the newly created syntax is nearly the same.

4.11 Redirect from old wiki pages

Due to the fact that many users bookmarked a certain page of interest, it was necessary to redirect the user to the desired gene page of the newly constructed web interface. Reasonably, a mediawiki extension called "Bank" was created to redirect the user to the proper directory (see appendix 7.5.2). Next, the text of each page was replaced by a short command to force the mediawiki engine to move to the new gene page. The newly instantiated syntax "{{#bank: citB}}" starts a program which guides the user to the gene page with the title of *citB*. With this extension the user can also search for the gene in the old version.

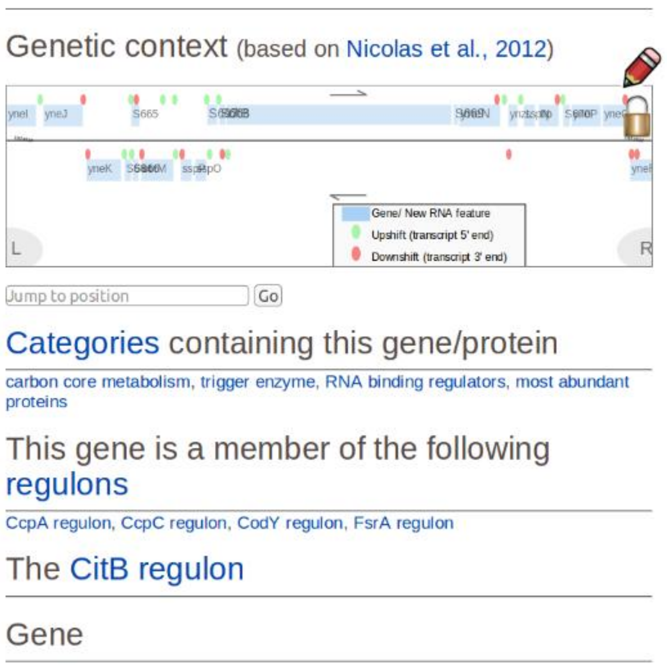
5. Discussion

In the discussion we will take a closer look on the advantages and disadvantages of the newly created infrastructure of the database and the currently available applications. One might ask why the new version of *Subti*Wiki is necessary for the community. Indeed, there are not so many database structures available focusing on biological objects and especially on single organisms like in *B. subtilis*. The mediawiki engine is a nice tool to store data in plain text but if the content becomes more complex (like metabolic pathways, protein-protein interactions etc.) it is inconvenient to run calculations or to extract the necessary information from the raw text. Therefore, further applications have to be created. *Subti*Pathways was created to visualize the complex network of the metabolism and the regulation (Lammers et al., 2010). We will take a look how other databases handle this problem. Additionally, the structure of the database was limited and it was not so easy to run comparisons on genes concerning molecular weight or other properties. Before *Subti*Wiki 2.0, external scripts scan the text to find the weight of the gene and insert it in an external file. Sometimes the program could not find the weight due to typing errors leading to a gap of information. As the information on genes and properties continuously changes, the scripts have to scan the text frequently. This behavior was frequently repeated over time instead of inserting it directly into tailored columns and working on this data. The structured query language offers the opportunity to compare the genes easily without running an external script which extracts the information of each gene page.

5.1 Improvements in comparison to the old version

Because of the central table “Gene” in the newly constructed database the maintenance is much easier. The change of a name does not interfere with the name in the other three applications where the gene appears. Now, the genes are just numbers and do not depend on the varying names during the process of renaming. Additionally, the information in each application is always the same as on the gene page. The indexing of the genes also prevents the mismatch of a gene in an application and the tables are now related to each other. Possibly, there are genes which have the same synonym but the algorithm will first search for the name and then extend the search on the synonyms. Sometimes a gene has synonym that is the name of another gene. This can lead to confusion of the names but the manual curation of the pages can produce such rare events. Due to the redirect on the *Subti*Wiki pages the user is able to search in the old version of *Subti*Wiki and the user will still be guided to the right directory. Still there are some rare pages where the redirect tag of the “Bank” extension was not properly integrated but they will be detected in the future. The wiki is a nice tool to store information that only contains raw text but a gene and its properties can be used for analyzing challenges. *Bsub*Cyc is storing the main properties of a gene in the same way. The position on the genome, the sequence and many more are stored in defined columns to make these properties available for all other applications. For example, the sequence of a gene and its position are then used in the genomic browser to display the genomic arrangement. On the other hand *Ecoli*Wiki provides this information as a link to *Eco*Cyc but there is also an interactive browser integrated in *Ecoli*Wiki (http://browser.porteco.org/fgb2/gbrowse/MG1655/) as an additional application. *Ecoli*Wiki is comparable to the old version of *Subti*Wiki where the information is stored as pure text and links are provided to enter external applications. On the gene pages of *Ecoli*Wiki there is no interactivity only static object like images.

There are parts of the *Subti*Wiki page containing information which is not directly describing the gene or its properties. This kind of data should be maintained in the traditional wiki way because there the community members could contribute with information on biological materials (mutants, vectors, plasmids), protein (family, domain, catalysis), expression (feedbacks) and regulation (termination, activation). This part of the page can be easily edited by the registered *Subti*Wiki user.



Genetic context (based on Nicolas et al., 2012)

Categories containing this gene/protein

carbon core metabolism, trigger enzyme, RNA binding regulators, most abundant proteins

This gene is a member of the following regulons

CcpA regulon, CcpC regulon, CodY regulon, FsrA regulon

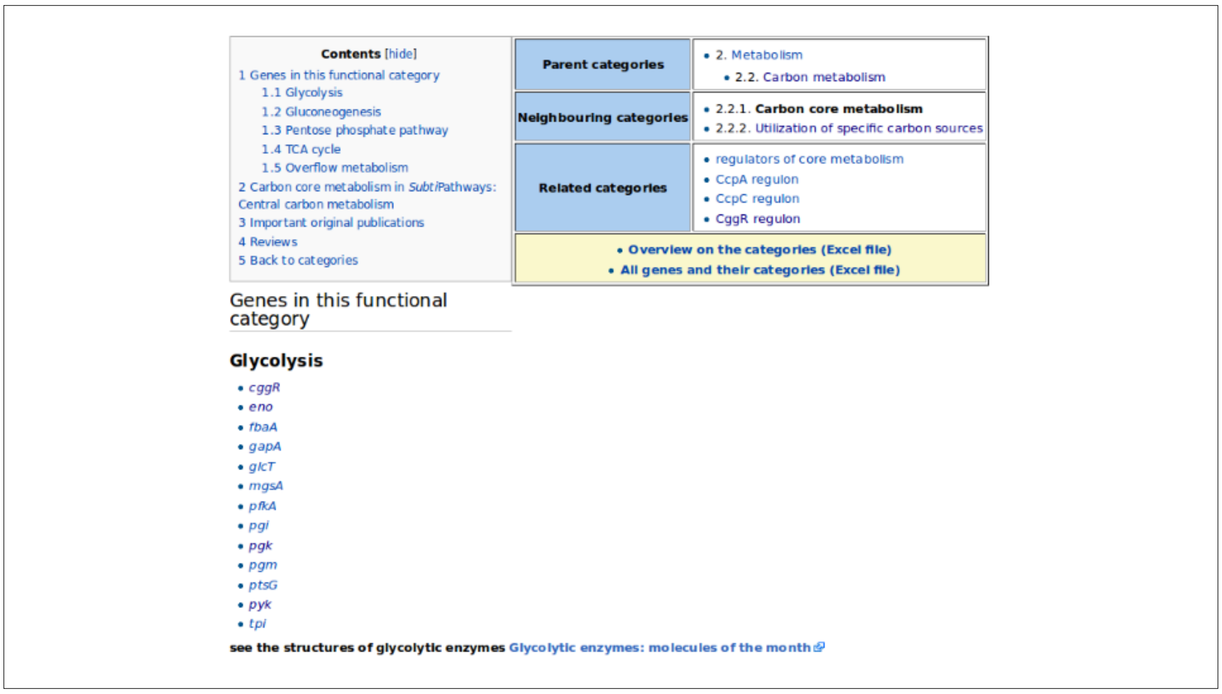
The CitB regulon

Gene

Figure 5.1: The flexible part of a gene page. The picture shows the part of the page which can be easily changed by a registered user. For this purpose, the user has to log in and then follow the specific editing syntax to change something on the page.

The applications also obtain a new surface on the client side to fit to the provided structure and the format of the data (JSON format). The engine behind the protein-protein interactions was changed to allow the projection of a color on the nodes. The colors on the nodes were derived from transcriptomic and proteomic measurements. Now, the overlay of the secondary information enables the user to see if the gene is expressed. With this we integrated the information on transcriptomic and proteomic data.

*Subti*Wiki was implemented to create an easily editable web site. The idea behind a wiki was to make entries quite fast and without "any" restriction. Therefore, the surface equals the one from the old version of *Subti*Wiki to allow the fast editing but with some restrictions. The main properties that include the product, function, molecular weight and all the other well defined properties of the gene can be only altered by the administrator. The additional content under the interactive genetic context can be easily changed by a registered user in *Subti*Wiki (see figure 5.1).



Contents [hide]

1 Genes in this functional category
- 1.1 Glycolysis
- 1.2 Gluconeogenesis
- 1.3 Pentose phosphate pathway
- 1.4 TCA cycle
- 1.5 Overflow metabolism

2 Carbon core metabolism in *Subti*Pathways: Central carbon metabolism
3 Important original publications
4 Reviews
5 Back to categories

Parent categories	• 2. Metabolism • 2.2. Carbon metabolism
Neighbouring categories	• 2.2.1. **Carbon core metabolism** • 2.2.2. Utilization of specific carbon sources
Related categories	• regulators of core metabolism • CcpA regulon • CcpC regulon • CggR regulon
• Overview on the categories (Excel file) **• All genes and their categories (Excel file)**	

Genes in this functional category

Glycolysis

- *cggR*
- *eno*
- *fbaA*
- *gapA*
- *glcT*
- *mgsA*
- *pfkA*
- *pgi*
- *pgk*
- *pgm*
- *ptsG*
- *pyk*
- *tpi*

see the structures of glycolytic enzymes Glycolytic enzymes: molecules of the month

Figure 5.2: A category page of the old wiki. The picture shows an old category page including a list of genes/ proteins. The page contains raw text including hard links to the gene pages which are involved in this category.

Furthermore, the data of the "wiki" should be accessible for everyone. Therefore, a "Download area" is provided to download excel files containing all desired information. This area facilitates the maintenance of the data. We do not have files which have to be modified separately after each modification in the database. Most of the files are auto-generated extracting the data from the database. So the files are always up-to-date. The sheets which contain information on regulons and categories are not yet integrated in the existing database. The categories still exist on the old wiki engine where the page is a collection of links (see figure 5.2) and they are not organized in a database. By editing the page, the administrator has to manually alter the additional excel file. This data cannot be auto-generated yet. Therefore, the entity-relationship model has to be adapted and the relevant information has to be inserted in the tables (see figure 5.3). *Bsub*Cyc is providing the same service. There it is called "Smart tables". Here the user can easily download diverse tables covering all possible topics like pathways, genes, regulation, transcripts and many more.

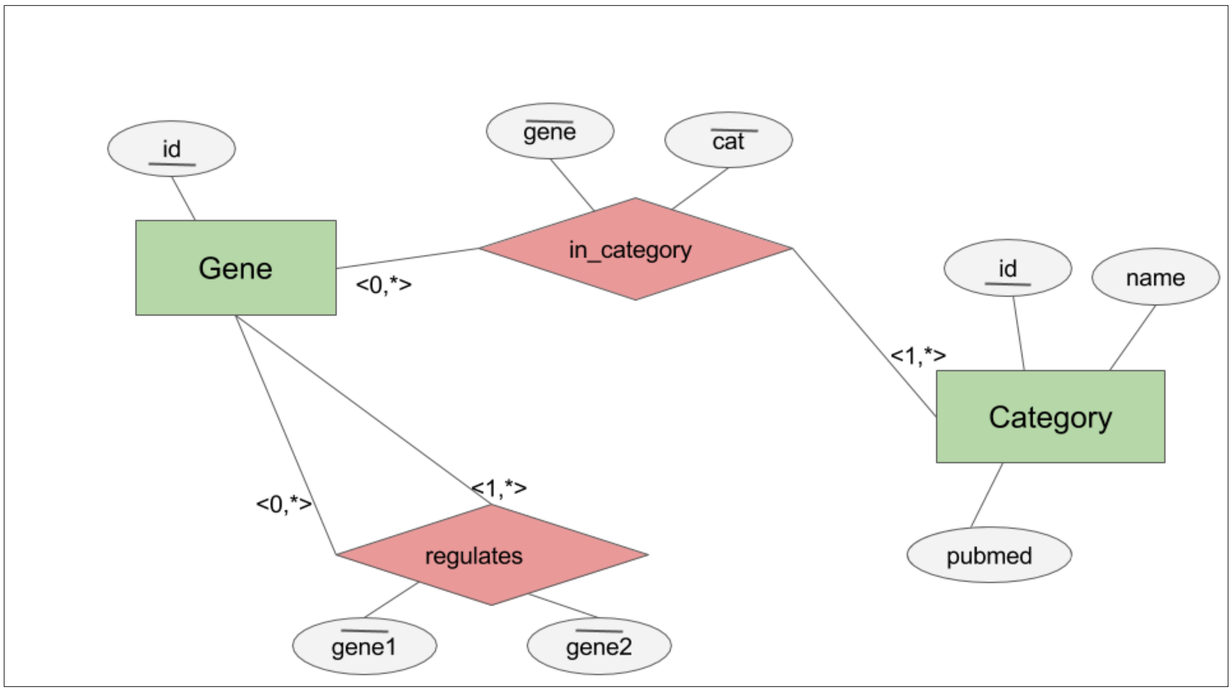
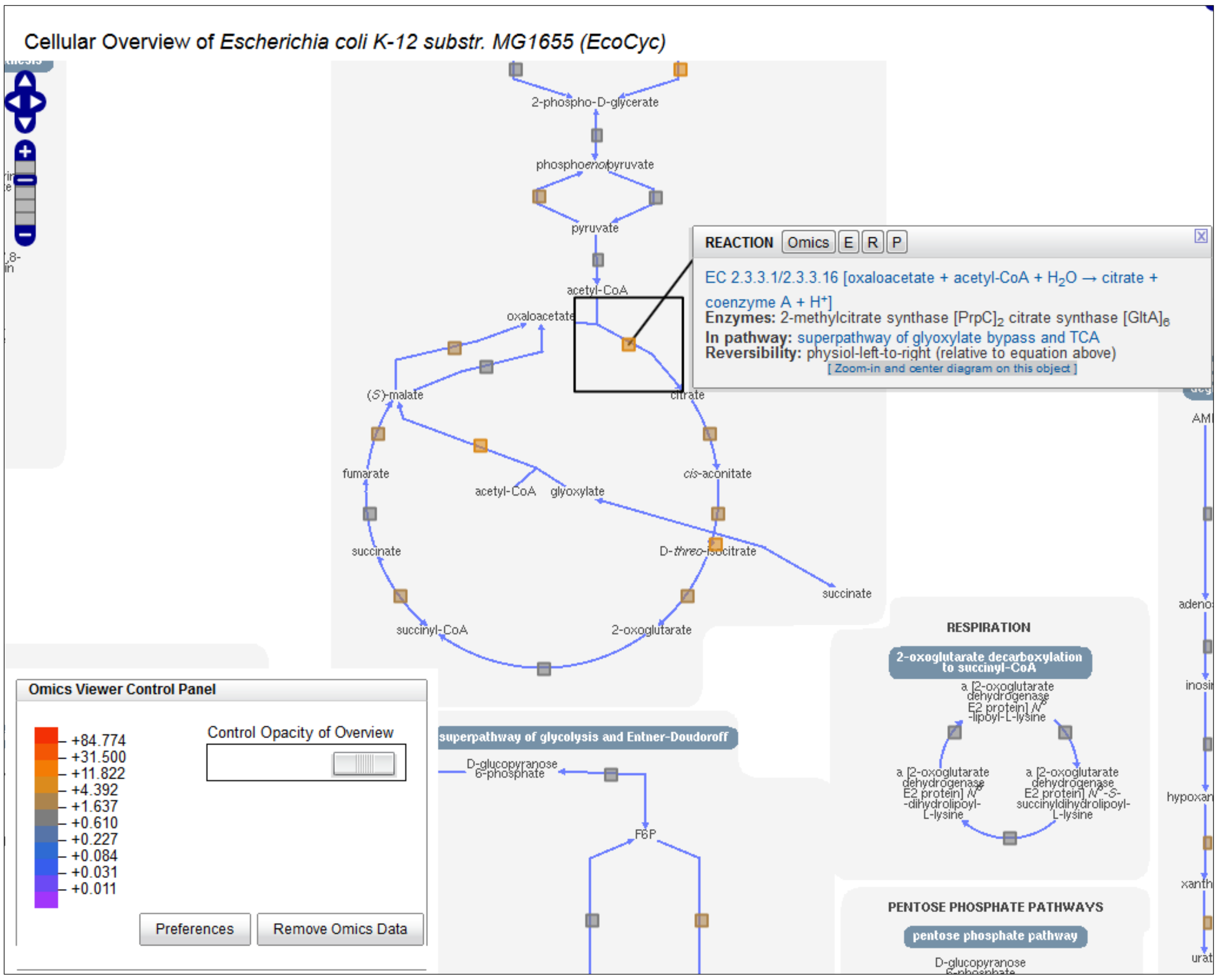


Figure 5.3: Snapshot of the proposed entity-relationship model. The figure shows the proposed relation between the "Gene" entity and the regulons and the categories. The picture is not the full entity-relationship model. The foreign keys with the overline as well as the primary keys with underline are marked in the extract of the entity-relationship model.

5.2 *Subti*Wiki compared to other databases

In the beginning *Subti*Wiki was just covering gene annotation but over time new applications were developed to visualize upcoming data. *Subti*Wiki is a frequently often database primarily providing information on the gene. The applications *Subti*Pathways, *Sub*Interact and *Subti*Express extended the knowledge on the interplay between the genes/proteins. In comparison to *Subti*Wiki, *Bsub*Cyc mainly focus on the catalytic reaction, location on the genome, regulation, function and localization. There is no specific information on biological materials (plasmids, vectors etc) or labs working on a specific topic. *Bsub*Cyc is just a database of a large collection of databases named BioCyc. Within the database this genes can be internally compared to other organisms in BioCyc. This is a functionality which *Subti*Wiki cannot offer. On the other side *Ecoli*Wiki is covering a different category of information. This database is also collecting information on genes but the focus is more on mutant strains. There are many annotated mutants for a gene.


Cellular Overview of *Escherichia coli K-12 substr. MG1655 (EcoCyc)*

REACTION

EC 2.3.3.1/2.3.3.16 [oxaloacetate + acetyl-CoA + H_2O → citrate + coenzyme A + H^+]
Enzymes: 2-methylcitrate synthase $[PrpC]_2$ citrate synthase $[GltA]_6$
In pathway: superpathway of glyoxylate bypass and TCA
Reversibility: physiol-left-to-right (relative to equation above)
[Zoom-in and center diagram on this object]

Omics Viewer Control Panel

🟥	+84.774
🟧	+31.500
🟧	+11.822
🟫	+4.392
🟫	+1.637
⬛	+0.610
🟦	+0.227
🟦	+0.084
🟦	+0.031
🟪	+0.011

Control Opacity of Overview

Omics
E
R
P
2-phospho-D-glycerate
phosphoenolpyruvate
pyruvate
acetyl-CoA
oxaloacetate
(S)-malate
citrate
fumarate
acetyl-CoA
glyoxylate
cis-aconitate
succinate
D-threo-isocitrate
succinate
succinyl-CoA
2-oxoglutarate
RESPIRATION
2-oxoglutarate decarboxylation to succinyl-CoA
a [2-oxoglutarate dehydrogenase E2 protein] N6-lipoyl-L-lysine
a [2-oxoglutarate dehydrogenase E2 protein] N6-dihydrolipoyl-L-lysine
a [2-oxoglutarate dehydrogenase E2 protein] N6-S-succinyldihydrolipoyl-L-lysine
superpathway of glycolysis and Entner-Doudoroff
D-glucopyranose 6-phosphate
F6P
PENTOSE PHOSPHATE PATHWAYS
pentose phosphate pathway
D-glucopyranose 6-phosphate
AMP
adeno
inosi
hypoxan
xanth
urat
Preferences
Remove Omics Data

Figure 5.4: The TCA cycle in *Eco*Cyc with "omics" data. The reactions in the TCA cycle contain different colors to express the different expression levels. On clicking on a reaction the user will open a window containing more information on the reaction.

If we compare *Subt*Interact with *Bsub*Cyc or *Ecoli*Wiki none of them is providing interaction profiles. This data based on two hybrid or SPINE experiments. In *Subt*Interact we overlayed the interaction profile additionally with expression data. But in the other databases there are no applications for protein-protein interactions.

There are many websites providing information on metabolic pathways. *Bsub*Cyc and KEGG provide interactive pathways as in *Subti*Pathways. In *Subti*Pathways we integrated "omics" data on top of the metabolic and regulatory pathways. The KEGG database also offers the opportunity to overlay a network with "omics" data to produce a second layer of information. This enables the researcher to observe abundance of protein amounts under different conditions. *Eco*Cyc also offers the opportunity to discover metabolic pathways. The pathways are divided into specific topics but there is also the possibility to take a look on the whole metabolic pathway. Furthermore, *Eco*Cyc provides the opportunity to overlay a metabolic pathway with "omics" data (see figure 5.4) as we introduced in *Subti*Pathways. The user can set test data on the map to see a colored overlay. There are so many tools and possibilities that it becomes overwhelming. The user can upload a dataset, import them from other databases, highlight reactions, enzymes, compounds and pathways in the network etc. Moreover, the metabolic pathways can be interactively explored clicking on the reactions shown on the picture. By clicking on the node, the user will open up a popup window containing explicit information of the reaction and a link to the *Eco*Cyc page. The KEGG database is also focusing on the reactions whereas *Subti*Pathways covers the compounds and enzymes separately. *Eco*Cyc is a very powerful database covering many scopes of *E. coli* as regulation, pathways, genomic arrangement. Furthermore, it combines different data sources as the metabolic networks are enriched with "omics" data. *Eco*Cyc bases on a classical database storing the information in a strictly organized and structured way.

*Subti*pathways integrates transcriptomic and proteomic data. For this purpose, color-coded markers are placed on the map to visualize the expression of a gene or the amount of a protein. There the user can take a look how the protein/transcript amounts behave in the background of regulatory or metabolic background. We are not looking on the reactions as in *Eco*Cyc. In *Subti*Pathways the whole pathway is divided into 50 smaller ones. Each pathway is covering a specific topic but the pathways the pathways are totally separated from each other. Therefore, there is no general overview of the whole metabolism as in *Eco*Cyc.

The large problem with *Subti*Pathways is that the markers on the static metabolic picture are not related to each other. Due to that there are no comprehensive analyses possible. For example in a mathematically defined graph the user is able to search for the shortest path from one metabolite to the other or to delete a gene to see if there is an alternative path. In the context of synthetic biology and the new challenges in minimization of genomes, the shortest path algorithms could predict new deletion candidates. This could be just a hint and has to be proven in the lab. This new insights could help to optimize metabolic rates which would be interesting for the synthetic biology. Therefore, the E.C. number could be a first attempt to enable the analysis on metabolic and regulatory pathways. The E.C. number is a unified number describing the enzymatic activity of a protein and it is used for all organisms. Therefore, the reactions could be integrated in the database structure to allow comprehensive analysis on the already existing pathway data (see figure 5.5).

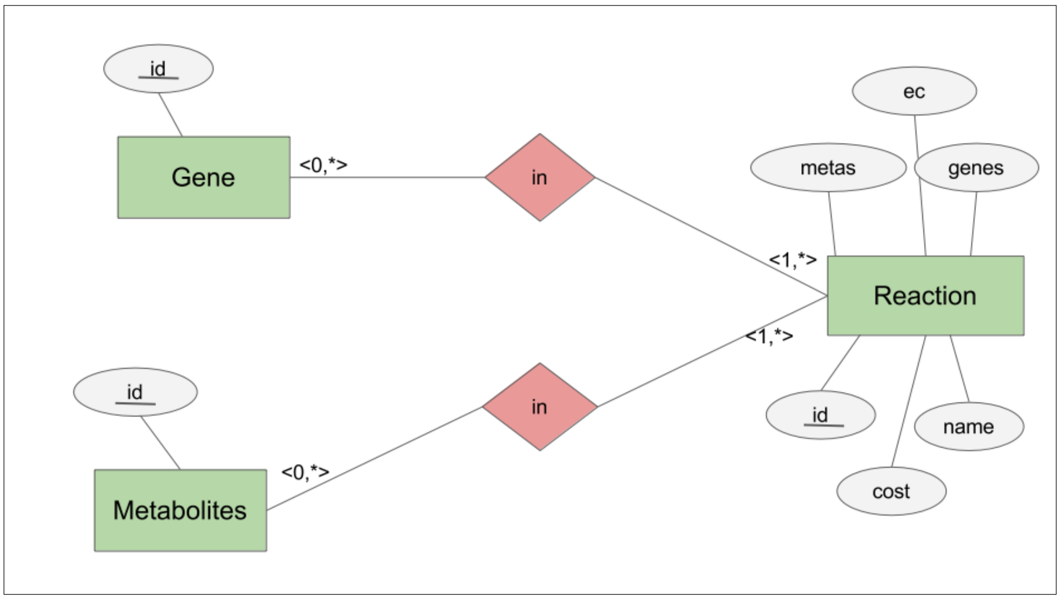


Figure 5.5: The extension of the database with reactions. The database could be extended by the reactions to allow the comprehensive analysis of metabolic and regulatory networks. Now, the “Reaction” table contains the metabolites and genes as well as the E.C. number and the cost for the reaction. The entity-relationship is a snapshot of the real database structure.

The first step will be to create a simple model for the reactions. The cost of a reaction has to be defined. For this purpose, the usage of ATP and other energy sources have to be simplified. This is the background of the engine but furthermore a visualization tool has to be developed to allow the researcher the exploration of the pathway with focus on the reactions. If there is only interest on a specific path like from a single metabolite to another one, there is no need to also display the surrounding metabolites, proteins and other molecules. The specific analysis of a path and the introduction of further data make it possible to crosslink different information which is describing the processes in the organism *B. subtilis*. With evolving methods in the field of “Omics” it becomes more and more important to adapt the bioinformatic analyzing tools to the approaching methods and especially to the richness of data. A big challenge is to integrate and to interpret the large amount of data. Due to the fine methods results in molecular biology only differ slightly and it is not so obvious to identify the difference in large datasets. Therefore, the analyzing tools have to be adapted to needs of the researcher.

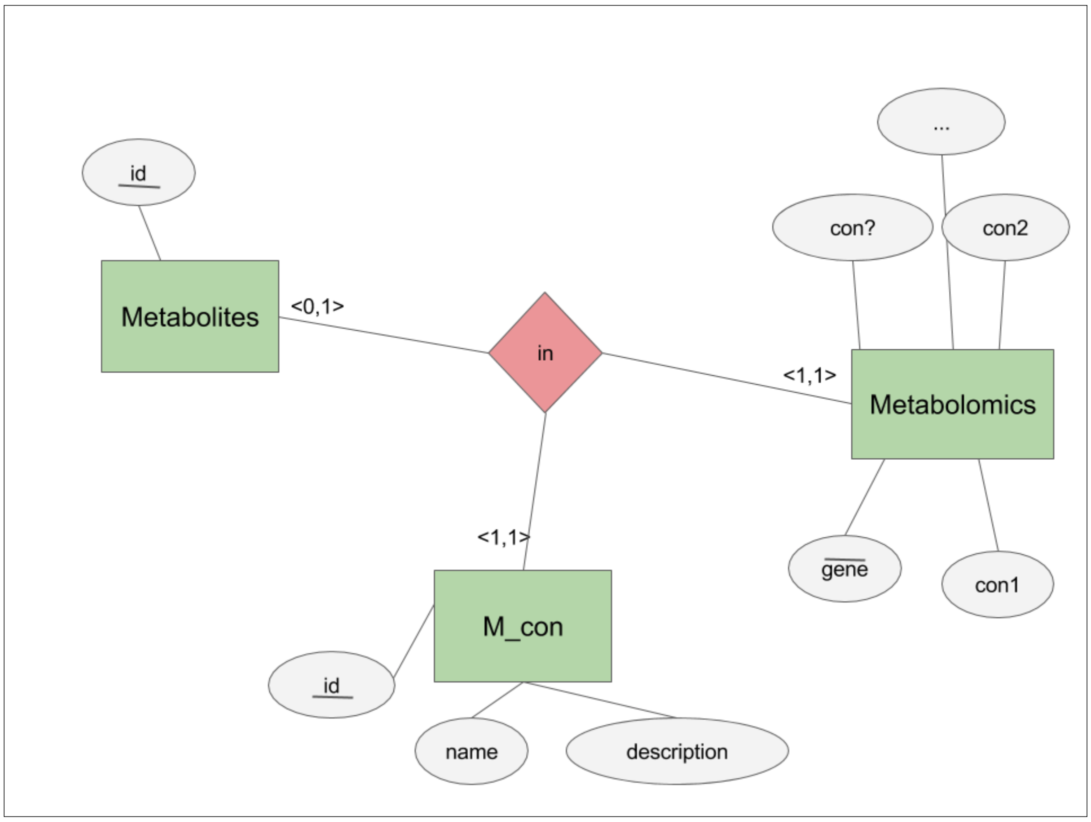
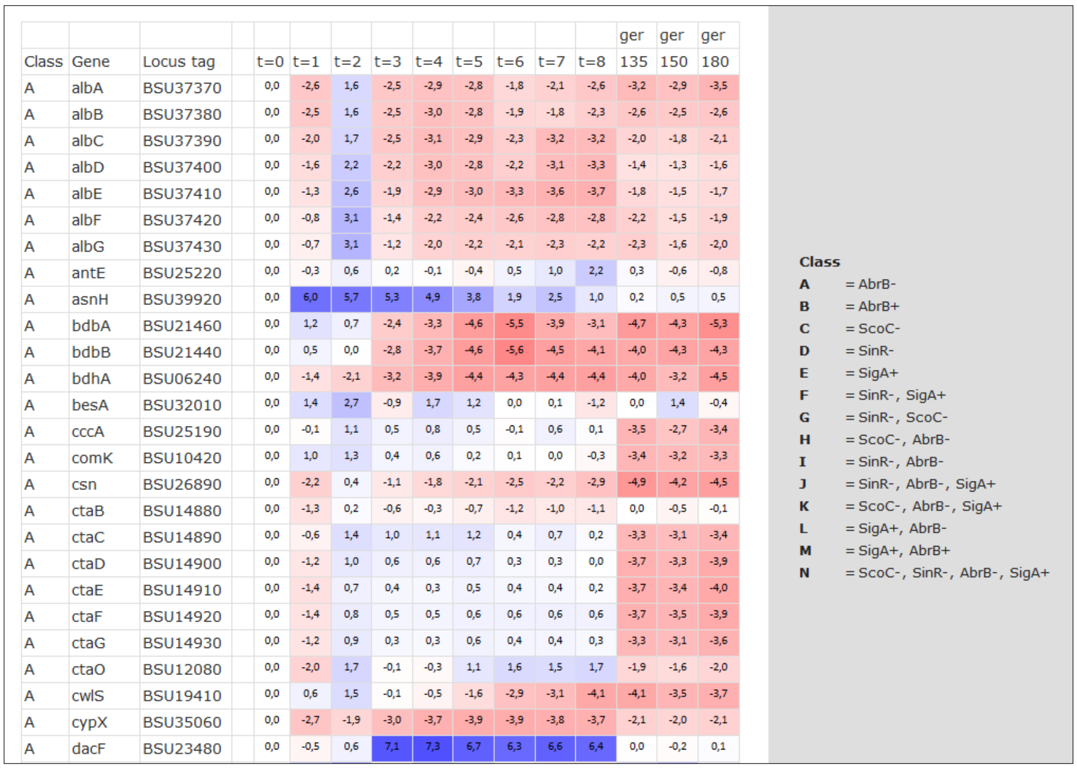


Figure 5.6: Including metabolomics data. The database can be extended by metabolomic data to integrate them in the pathways and further calculations. “M_con” is a table containing information on the conditions used in the measurements. Whereas the “Metabolomics” table contains the measured data points and the attribute “gene” refers to the attribute “id” of the “Metabolites” table.

As well as the proteomic and the transcriptomic data, the metabolomic data can be integrated in the pathways to interactively observe metabolic fluxes (see figure 5.6). The metabolome is the next part of the puzzle to be integrated in the analysis of the organism. In the cell the metabolites are highly interacting compounds (Tang, 2011). The main parts of a pathway are annotated (transcripts, proteins, metabolites) to have a general overview and the main focus is not only on the genes/ proteins anymore.



											ger	ger	ger
Class	Gene	Locus tag	t=0	t=1	t=2	t=3	t=4	t=5	t=6	t=7	t=8	135	150	180
A	albA	BSU37370	0,0	-2,6	1,6	-2,5	-2,9	-2,8	-1,8	-2,1	-2,6	-3,2	-2,9	-3,5
A	albB	BSU37380	0,0	-2,5	1,6	-2,5	-3,0	-2,8	-1,9	-1,8	-2,3	-2,6	-2,5	-2,6
A	albC	BSU37390	0,0	-2,0	1,7	-2,5	-3,1	-2,9	-2,3	-3,2	-3,2	-2,0	-1,8	-2,1
A	albD	BSU37400	0,0	-1,6	2,2	-2,2	-3,0	-2,8	-2,2	-3,1	-3,3	-1,4	-1,3	-1,6
A	albE	BSU37410	0,0	-1,3	2,6	-1,9	-2,9	-3,0	-3,3	-3,6	-3,7	-1,8	-1,5	-1,7
A	albF	BSU37420	0,0	-0,8	3,1	-1,4	-2,2	-2,4	-2,6	-2,8	-2,8	-2,2	-1,5	-1,9
A	albG	BSU37430	0,0	-0,7	3,1	-1,2	-2,0	-2,2	-2,1	-2,3	-2,2	-2,3	-1,6	-2,0
A	antE	BSU25220	0,0	-0,3	0,6	0,2	-0,1	-0,4	0,5	1,0	2,2	0,3	-0,6	-0,8
A	asnH	BSU39920	0,0	6,0	5,7	5,3	4,9	3,8	1,9	2,5	1,0	0,2	0,5	0,5
A	bdbA	BSU21460	0,0	1,2	0,7	-2,4	-3,3	-4,6	-5,5	-3,9	-3,1	-4,7	-4,3	-5,3
A	bdbB	BSU21440	0,0	0,5	0,0	-2,8	-3,7	-4,6	-5,6	-4,5	-4,1	-4,0	-4,3	-4,3
A	bdhA	BSU06240	0,0	-1,4	-2,1	-3,2	-3,9	-4,4	-4,3	-4,4	-4,4	-4,0	-3,2	-4,5
A	besA	BSU32010	0,0	1,4	2,7	-0,9	1,7	1,2	0,0	0,1	-1,2	0,0	1,4	-0,4
A	cccA	BSU25190	0,0	-0,1	1,1	0,5	0,8	0,5	-0,1	0,6	0,1	-3,5	-2,7	-3,4
A	comK	BSU10420	0,0	1,0	1,3	0,4	0,6	0,2	0,1	0,0	-0,3	-3,4	-3,2	-3,3
A	csn	BSU26890	0,0	-2,2	0,4	-1,1	-1,8	-2,1	-2,5	-2,2	-2,9	-4,9	-4,2	-4,5
A	ctaB	BSU14880	0,0	-1,3	0,2	-0,6	-0,3	-0,7	-1,2	-1,0	-1,1	0,0	-0,5	-0,1
A	ctaC	BSU14890	0,0	-0,6	1,4	1,0	1,1	1,2	0,4	0,7	0,2	-3,3	-3,1	-3,4
A	ctaD	BSU14900	0,0	-1,2	1,0	0,6	0,6	0,7	0,3	0,3	0,0	-3,7	-3,3	-3,9
A	ctaE	BSU14910	0,0	-1,4	0,7	0,4	0,3	0,5	0,4	0,4	0,2	-3,7	-3,4	-4,0
A	ctaF	BSU14920	0,0	-1,4	0,8	0,5	0,5	0,6	0,6	0,6	0,6	-3,7	-3,5	-3,9
A	ctaG	BSU14930	0,0	-1,2	0,9	0,3	0,3	0,6	0,4	0,4	0,3	-3,3	-3,1	-3,6
A	ctaO	BSU12080	0,0	-2,0	1,7	-0,1	-0,3	1,1	1,6	1,5	1,7	-1,9	-1,6	-2,0
A	cwlS	BSU19410	0,0	0,6	1,5	-0,1	-0,5	-1,6	-2,9	-3,1	-4,1	-4,1	-3,5	-3,7
A	cypX	BSU35060	0,0	-2,7	-1,9	-3,0	-3,7	-3,9	-3,9	-3,8	-3,7	-2,1	-2,0	-2,1
A	dacF	BSU23480	0,0	-0,5	0,6	7,1	7,3	6,7	6,3	6,6	6,4	0,0	-0,2	0,1

Class

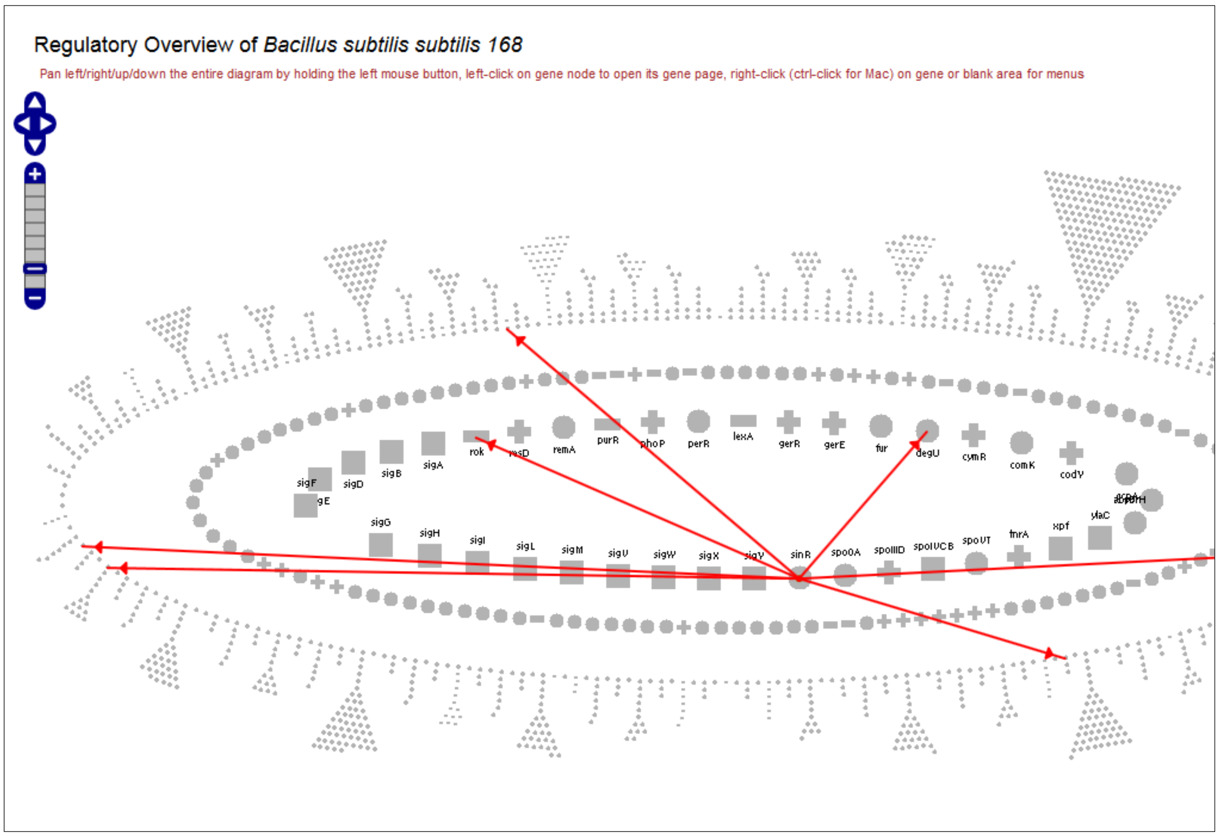
A	= AbrB-
B	= AbrB+
C	= ScoC-
D	= SinR-
E	= SigA+
F	= SinR-, SigA+
G	= SinR-, ScoC-
H	= ScoC-, AbrB-
I	= SinR-, AbrB-
J	= SinR-, AbrB-, SigA+
K	= ScoC-, AbrB-, SigA+
L	= SigA+, AbrB-
M	= SigA+, AbrB+
N	= ScoC-, SinR-, AbrB-, SigA+

Figure 5.7: An extract of the heatmaps provided on SporeWeb. The figure shows an extract of the heatmap. The colors of the cells in the table shows the gene expression at defined time points. The first column shows how the gene is regulated.

*Subti*Express was implemented to visualize the gene expression level and protein amount under many different conditions. Therefore, an interactive graph was implemented to visualize the fluctuation. In *Bsub*Cyc there is no comparable application showing only the expression levels. There this data was combined with other topics like the metabolic pathways. Sporeweb (Eijlander et al., 2014) is another database which focuses on the regulatory networks during sporulation in *B. subtilis*. On this platform the gene expression levels are shown in heatmaps (see figure 5.7). This is a fancy way to visualize the expression for more than one gene. The newly implemented feature in *Subti*Express allows displaying a second expression profile on the diagram (see figure result expression). Moreover, the heatmaps contain information on the regulator. Nevertheless, the platform is specialized on the sporulation.

Finally, I want to take a closer look on the regulation of genes. In *Subti*Wiki there is a special page covering the regulons. Moreover, regulons are annotated on the gene page if there is potential regulator known. Nevertheless, there is no interactive application visualizing this complex network of regulations. The information is manually curated and organized in a database. In contrast to *Subti*Wiki, *Bsub*Cyc is providing an interactive tool named “Regulatory Overview” (see figure 5.8). The main regulators are placed in the centre. The user can interactively discover the regulatory network.

Again, *Bsub*Cyc offers the possibility to overlay the network with “omics” data.

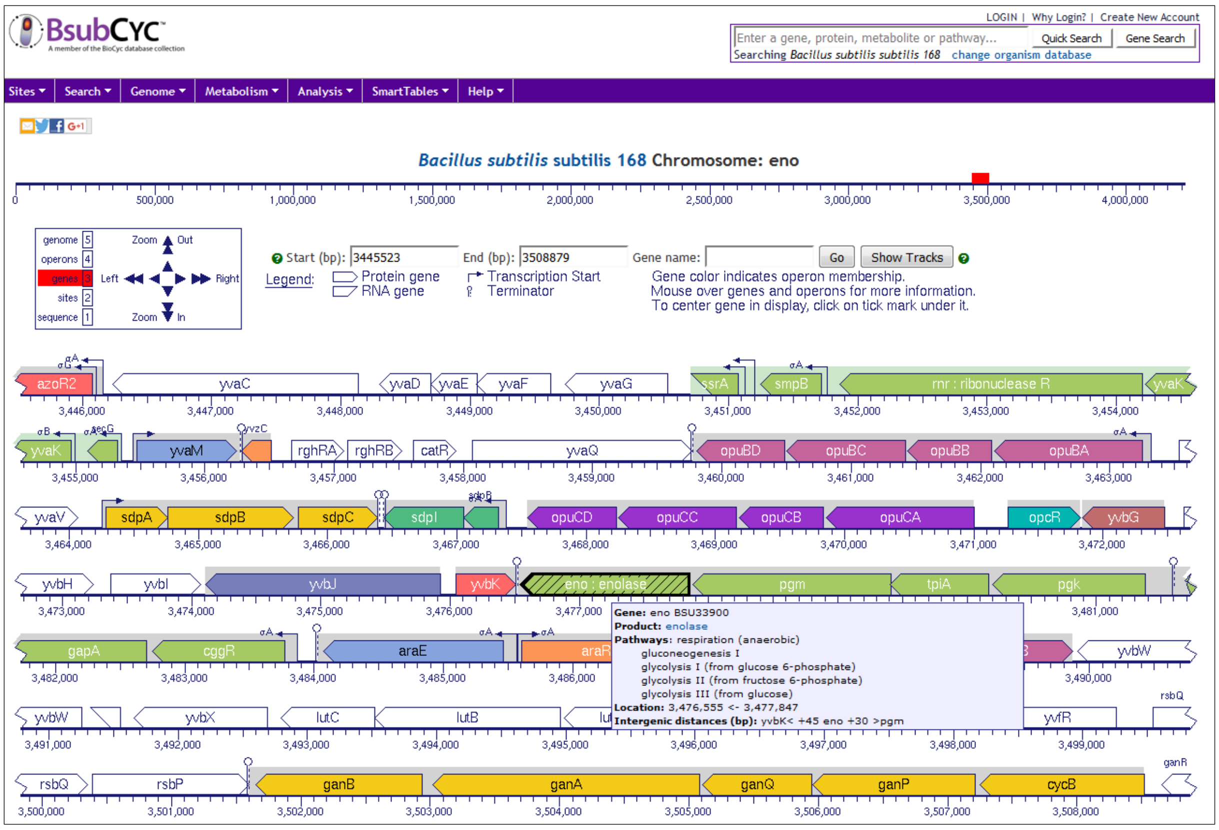

Regulatory Overview of *Bacillus subtilis subtilis 168*

Pan left/right/up/down the entire diagram by holding the left mouse button, left-click on gene node to open its gene page, right-click (ctrl-click for Mac) on gene or blank area for menus

Figure 5.8: Regulation application in *Bsub*Cyc. The figure shows the general regulation network of Bacillus subtilis. The red arrows are pointing to the genes which are regulated by SinR.

*Bsub*Cyc also provides an interactive genome browser (see figure 5.9) the properties can be inserted in the popup windows where the user can interactively explore the genome arrangement. *Subti*Wiki also offers the possibility to show the main properties in the other applications. SubtiWiki does not provie a separate genome browser as an application. The genome browser is integrated on each gene page.

As a result, BioCyc is focussing on the understanding of regulatory and metabolic networks as well as integration of high-throughput not only in *Bacillus subtilis* but also in other organisms. *Subti*Wiki starts to move to this direction but the focus is still on single gene annotation.



Searching *Bacillus subtilis subtilis 168* change organism database

***Bacillus subtilis* subtilis 168** **Chromosome: eno**

Start (bp): 3445523 End (bp): 3508879 Gene name:

Legend: Protein gene, RNA gene, Transcription Start, Terminator

Gene color indicates operon membership.
Mouse over genes and operons for more information.
To center gene in display, click on tick mark under it.

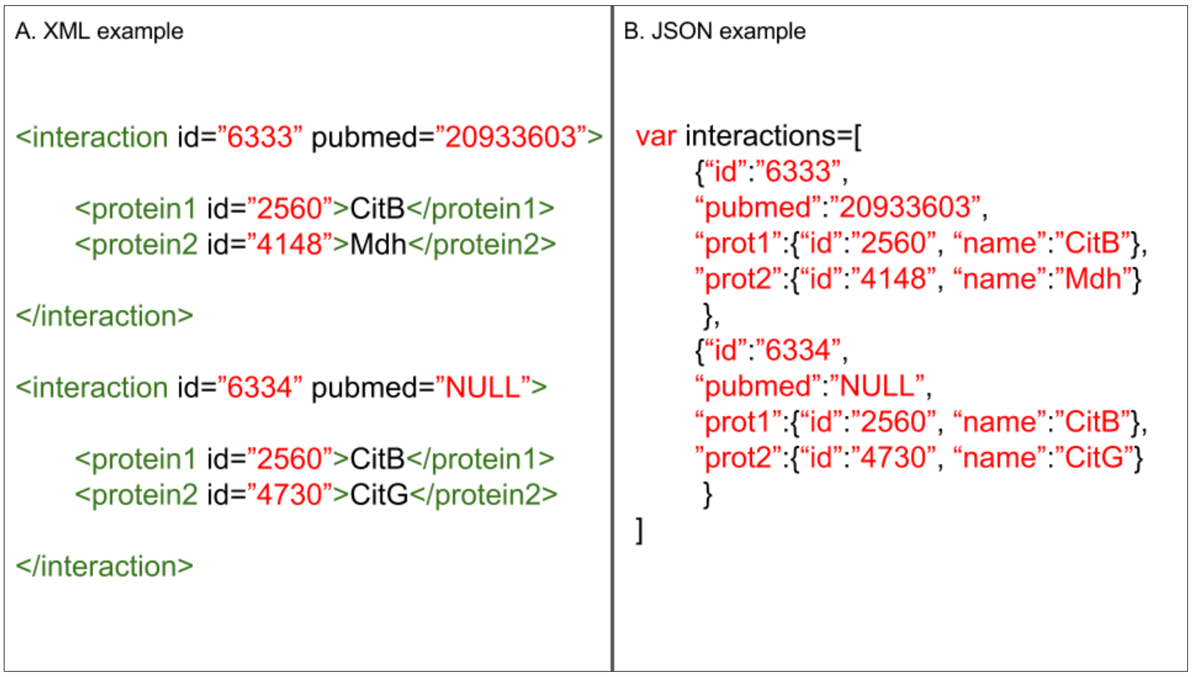
Gene: eno BSU33900
Product: enolase
Pathways: respiration (anaerobic)
gluconeogenesis I
glycolysis I (from glucose 6-phosphate)
glycolysis II (from fructose 6-phosphate)
glycolysis III (from glucose)
Location: 3,476,555 <- 3,477,847
Intergenic distances (bp): yvbK< +45 eno +30 >pgm

Figure 5.9: The genome browser of *Bsub*Cyc. The picture shows the genomic arrangement of *eno* and a popup window showing further information on *eno*. By moving over a gene the user can interactively explore the genes. A popup window opens and shows information on name, product, pathways and location.

5.3 Is JSON the right storage format

In this part we will discuss the usage of JSON as a transport and storage format. JSON is the format that contains the information and is delivered to the client. Moreover, the each edit was delivered as JSON to the client. This format is uniquely used over all applications. Due to the usage of JavaScript on the client-side the JSON object can be directly used in JavaScript and it does not have to be decrypted. The simple structure of this object is well-defined and the properties are well deducible. Furthermore, the object can be easily stored in the database as pure text or the properties can be divided and then inserted in the columns of the database. The programming language PHP is providing many libraries to handle JSON objects. It can decode JSON objects to translate it into arrays and furthermore it can create JSON objects from arrays.

XML is also a commonly used data format to transfer data through the internet. XML was the first type used in the AJAX concept. In this concept not the page is reloaded but simply the requested parts are refreshed to increase the pace of the web site. On the client-side, JavaScript is then deducing the parts from the XML using a library but the elements cannot be directly created. Many applications in biology are using the XML format to store information.



A. XML example

```
<interaction id="6333" pubmed="20933603">

     <protein1 id="2560">CitB</protein1>
     <protein2 id="4148">Mdh</protein2>

</interaction>

<interaction id="6334" pubmed="NULL">

     <protein1 id="2560">CitB</protein1>
     <protein2 id="4730">CitG</protein2>

</interaction>
```

B. JSON example

```
var interactions=[
     {"id":"6333",
     "pubmed":"20933603",
     "prot1":{"id":"2560", "name":"CitB"},
     "prot2":{"id":"4148", "name":"Mdh"}
      },
     {"id":"6334",
     "pubmed":"NULL",
     "prot1":{"id":"2560", "name":"CitB"},
     "prot2":{"id":"4730", "name":"CitG"}
      }
]
```

Figure 5.10: The comparision of XML and JSON. A) This is an example of the XML format. In green the tags are described. Each interaction has an id and pubmed number and it consists of two child nodes – the proteins which interact. Due to the tags the XML can become quite large by increasing data. **B)** In contrast the same content as in the XML is structured in JSON. This format can be directly used in JavaScript applications and the tags are not present anymore. Here the interactions consist of a list with JavaScript objects.

CellDesigner is an application to create metabolic pathways (http://www.celldesigner.org). The information of the pathways is stored in an XML file – the biologists call it SBML (System Biology Markup Language) – but it is simply a unified syntax for the tags. This was the first approach to create a general markup language for biological objects (Hucka et al., 2003). The pathways in *Subti*Pathways are created by this application but the XML file was only used to identify the positions of proteins and metabolites. The content of the XML file is much higher, defining the reaction and the relation of the items. *Subti*React is a project to structure the reactions. The XML files can be a resource for the reactions, without annotating manually all the reactions again. In the end the reactions should be again in JSON format. The semi-structured format of JSON and XML are a great property to transfer large datasets through the web. In this case, JSON is the preferred data type due to JavaScript which is used in all applications (see figure 5.10). Furthermore, the dictionaries in python are similar to the JSON format and easily transferable. In the text-mining part we worked with python and therefore the JSON format was a good solution to easily store and use the created output.

5.4 How to optimize the current database

If one takes a closer look on the database structure, there are still some points to improve. For some relations and entities no foreign keys or primary keys were defined to increase the speed of lookup. For example, if the “Browser” entity does not have a reference to the “Gene” entity. There the “Browser” id should be the foreign key and point to the “Gene” id. This is something that should be improved and all the closely related entities should be marked with appropriate keys. In theory this would be better but in practice this rarely used. Moreover, the tables which are closely related to “Gene” should be automatically updated if the “Gene” table gets updated. For example, if we delete a gene from the “Gene” then all the information of the gene should be deleted from the other tables. This is something where foreign keys are needed to react on the changes in the main table. In SQL there is a command - named “ON UPDATE/DELETE CASCADE” - which can be added to the table and which would exactly fulfill these tasks. The central aim of *Subti*Wiki was to annotate genes of *B. subtilis*, but by increasing data the needs of researchers change. Therefore, the structure of scripts and also the database became more complicated. As mentioned there are already plans to extend the database by reactions, metabolomic data, categories and regulation. Finally, a database structure would be constructed which could be generally used for prokaryotic organisms.

5.5 How to optimize the current scripts

With extending the database storage, also scripts have to be implemented to store and to deliver the new data. For this purpose, the available scripts have to be adapted. For example, the *Subti*Pathways scripts have to be adapted to the metabolomic data which would be available in the database. This data has to be integrated in the pathways. First, the php scripts have to extract and format the data and send it back to the client. On the client side JavaScript has to integrate the data interactively on the map.

Additionally, many scripts contain repetitions and unnecessary comments as well as many functions that are also present in other files. Over time some functions were not needed anymore but there are still in the script without any function. The first step would be to get rid of all multiple used functions, put them in an additional file and make them globally available. To get rid of all the unused script would increase the pace of loading a page.

Finally, the HTML and CSS files are also a source of repetitions. The styling of a page could be more generalized and collected in a single file to reduce the memory usage a bit. But this is just fine-tuning due to the fact that this files are quite small.

5.6 Perspectives of *Subti*Wiki

*Subti*Wiki was initiated to collect information on single genes. Over time, the information on *B. subtilis* has increased and there was a need for visualization to receive an overview on a subject group (like pathways). Transcriptomic and proteomic data became an interesting topic for cellular processes in *B. subtilis*. The quantitative knowledge on mRNA/protein and metabolite abundance leads to new cellular insights at defined time points. Yet unknown genes could be categorized by the gene expression levels (Nicolas et al., 2012).Using these profiles many genes were identified which are highly expressed in sporulation. The proteomic data also have the potential to reveal unknown functionalities of genes. The metabolomic data which is not integrated yet could be used to enrich the knowledge on metabolic as well as regulatory networks. Now, the snapshots of compound abundance could lead to further models concerning cellular behavior and processes. *Subti*Wiki was prepared for this era by adapting the database to large datasets. By overlaying different types of information (for example, pathways and expression) new models could be predicted which have to be proven experimentally. *Subti*Wiki has to connect the different fields of the cell to create a general overview of the bacterium.

As mentioned in previous chapters many tools and analyzing methods are available but they could not be used in the old version of *Subti*Wiki. For this purpose, the content in *Subti*Wiki 2.0 was reorganized. Furthermore, there are still parts missing (regulation, reaction, category) which are not yet integrated in the database. *Bsub*Cyc has a fancy tool to visualize the regulation network and the metabolic network. These networks can be enriched by expression values. The same was done for *Subti*Wiki. The pathways and the interactions were enriched with proteomic and transcriptomic data. Additionally, *Bsub*Cyc offers the possibility to integrate metabolomic data which gives a more general overview on the metabolic pathway. These datasets are not yet integrated into *Subti*Wiki which could be done in future work.

*Subti*Wiki is the fundamental source for the work on the mini*bacillus* project. For this purpose, *Subti*Wiki could be extended by an application to study the consequences of a gene deletion. Sometimes these are not so obvious if a gene is involved in many cellular processes. But if the user is able to have a summary of all consequences it could be easy to identify new deletion candidates.

Furthermore, the specific analysis of metabolic pathways could lead to a better modeling of minimal pathways and identifying the necessary branches and paths. Computer networks are also constructed to find the shortest path from one computer to the other (https://en.wikipedia.org/wiki/Dijkstra's_algorithm). This theory could also be implemented for metabolic pathways. For this purpose, the database structure as well as the visualization and the analyzing tools have to be adjusted to the algorithm.

The database framework could also be used for other prokaryotic organisms. The engine of

*Subti*Wiki could be a precursor for other biological communities trying to integrate more complex data and visualization tools. It is a package which can be easily used and offers the functionality to work on pathways, interactions and expression. On the one hand the datasets were successfully integrated but on the other hand new tools have to be added which analytically work on networks and interactions. The main goal of *Subti*Wiki is that the researcher working on a topic is able to find the desired information easily and to use the application intuitively.

All in all, *Subti*Wiki became a powerful tool covering gene annotation, metabolic and regulatory pathways, interaction profiles and expression levels.

6. References

Abram F. (2014) Systems-based approaches to unravel multi-species microbial community functioning. Comput Struct Biotechnol J. 13:24-32.

Baba T., Ara T., Hasegawa M., Takai Y., Okumura Y., et al (2006) Construction of Escherichia coli K-12 in-frame, single-gene knockout mutants: the Keio collection. Mol Syst Biol 2:2006.0008.

Benson D. A. , Cavanaugh M., Clark K., Karsch-Mizrachi I., Lipman D. J., et al (2013) GenBank. Nucleic Acids Res. 41:D36-D42.

Berman H. M. (2008) The Protein Data Bank: a historical perspective. Acta Crystallogr A. 64:88-95.

Boutros M., Ahringer J. (2008) The art and design of genetic screens: RNA interference. Nat Rev Genet 9:554–566.

Brooksbank C., Bergman M.T., Apweiler R., Birney E., Thornton J. (2014) The European Bioinformatics Institute's data resources 2014. Nucleic Acids Res. 42:D18-D25.

Christen B., Abeliuk E., Collier J. M., Kalogeraki V. S., Passarelli B., et al (2011) The essential genome of a bacterium. Mol Syst Biol 7:528.

Cohen K. B., Hunter L. (2008) Getting started in text mining. PLoS Comput Biol. 4:e20.

Commichau F. M., Rothe F. M., Herzberg C., Wagner E., Hellwig D., et al (2009) Novel activities of glycolytic enzymes in Bacillus subtilis: interactions with essential proteins involved in mRNA processing. Mol Cell Proteomics. 8:1350-1360.

de Berardinis V., Vallenet D., Castelli V., Besnard M., Pinet A., et al (2008) A complete collection of single-gene deletion mutants of Acinetobacter baylyi ADP1. Mol Syst Biol 4:174.

Eijlander R. T., de Jong A. , Krawczyk A. O., Holsappel S., Kuipers O. P. (2014) SporeWeb: an interactive journey through the complete sporulation cycle of Bacillus subtilis. Nucleic Acids Res. 42:D685-D691.

Fernández S., Ayora S., Alonso J. C. (2000) Bacillus subtilis homologous recombination: genes and products. Res Microbiol. 151: 481-486.

Finn R. D., Bateman A., Clements J., Coggill P., Eberhardt R. Y., et al (2014) Pfam: the protein families database. Nucleic Acids Res. 42:D222-D230.

French C. T., Lao P., Loraine A. E., Matthews B. T., Yu H., Dybvig K. (2008) Large-scale transposon mutagenesis of Mycoplasma pulmonis. Mol Microbiol 69:67–76.

Hu J. C., Aramayo R., Bolser D., Conway T., Elsik C. G., et al (2008) The emerging world of wikis. Science. 320:1289-1290.

Hucka M., Finney A., Sauro H. M., Bolouri H., Doyle J. C., et al (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models.

Bioinformatics. 19:524-531.

Juhas M., Reuß D. R., Zhu B., Commichau F. M. (2014) Bacillus subtilis and Escherichia coli essential genes and minimal cell factories after one decade of genome engineering. Microbiology. 160:2341-2351.

Juhas M., Eberl L., Glass J. I. (2011) Essence of life: essential genes of minimal genomes. Trends Cell Biol 21:562–568.

Kanehisa M., Goto S., Sato Y., Kawashima M., Furumichi M., Tanabe M. (2014) Data, information, knowledge and principle: back to metabolism in KEGG. Nucleic Acids Res. 42:D199-D205.

Karp P. D., Weaver D., Paley S., Fulcher C., Kubo A., et al (2014) The EcoCyc Database. Ecosal Plus. 2014.

Kato J., Hashimoto M. (2007) Construction of consecutive deletions of the Escherichia coli chromosome. Mol Syst Biol 3:132.

Kobayashi K., Ehrlich S., Albertini A., Amati G., Andersen K. K., Arnaud M., et al. (2003) Essential Bacillus subtilis genes. Proc Natl Acad Sci U S A 100:4678–4683.

Kodama Y., Shumway M., Leinonen R. (2012) The Sequence Read Archive: explosive growth of sequencing data. Nucleic Acids Res. 40:D54-D56.

Kunst F., Ogasawara N., Moszer I., Albertini A. M., Alloni G., et al (1997) The complete genome sequence of the gram-positive bacterium Bacillus subtilis. Nature. 390:249-256.

Lammers C. R., Flórez L. A., Schmeisky A. G., Roppel S. F., Mäder U., et al (2010) Connecting parts with processes: SubtiWiki and SubtiPathways integrate gene and pathway annotation for Bacillus subtilis. Microbiology. 156:849-859.

Langridge G. C., Phan M. D., Turner D. J., Perkins T. T., Parts L., et al (2009) Simultaneous assay of every Salmonella typhi gene using one million transposon mutants. Genome Res 19:2308–2316.

Maaβ S., Wachlin G., Bernhardt J., Eymann C., Fromion V., et al (2014) Highly precise quantification of protein molecules per cell during stress and starvation responses in Bacillus subtilis. Mol Cell Proteomics. 13:2260-2276.

Maass S., Sievers S., Zühlke D., Kuzinski J., Sappa P. K., et al (2011) Efficient, global-scale quantification of absolute protein amounts by integration of targeted mass spectrometry and two-dimensional gel-based proteomics. Anal Chem. 83:2677-2684.

Mäder U., Schmeisky A. G., Flórez L. A., Stülke J. (2012) SubtiWiki--a comprehensive community resource for the model organism Bacillus subtilis. Nucleic Acids Res. 40:D1278-D1287.

Magrane M., Consortium U. (2011) UniProt Knowledgebase: a hub of integrated protein data. Database (Oxford). 2011:bar00.

McCutcheon J. P., Moran N. A. (2010) Functional convergence in reduced genomes of bacterial symbionts spanning 200 My of evolution. Genome Biol Evol 2:708–718.

McIntosh B. K., Renfro D. P., Knapp G. S., Lairikyengbam C. R., Liles N. M., et al (2012) EcoliWiki: a wiki-based community resource for Escherichia coli . Nucleic Acids Res. 40:D1270-D1277.

Metzker M. L. (2010) Sequencing technologies - the next generation. Nat Rev Genet. 11:31-46

Moszer I., Jones L. M., Moreira S., Fabry C., Danchin A. (2002) SubtiList: the reference database for the Bacillus subtilis genome. Nucleic Acids Res. 30:62-65.

Mount D. W. (2007) Using the Basic Local Alignment Search Tool (BLAST). CSH Protoc. 2007:pdb.top17.

Moya A., Gil R., Latorre A., Peretó J., Pilar Garcillán-Barcia M., de la Cruz F. (2009) Toward minimal bacterial cells: evolution vs. design. FEMS Microbiol Rev 33:225–235.

Muntel J. , Fromion V. , Goelzer A., Maaβ S., Mäder U., et al (2014) Comprehensive absolute quantification of the cytosolic proteome of Bacillus subtilis by data independent, parallel fragmentation in liquid chromatography/mass spectrometry (LC/MS(E)). Mol Cell Proteomics. 13:1008-1019.

Nicolas P., Mäder U., Dervyn E., Rochat T., Leduc A., et al (2012). Condition-dependent transcriptome reveals high-level regulatory architecture in Bacillus subtilis. Science. 335:1103-1106.

Pearson W. R., Lipman D. J. (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci U S A. 85(8):2444-2448.

Rzhetsky A., Seringhaus M., Gerstein M. B. (2009) Getting started in text mining: part two. PLoS Comput Biol. 5:e1000411.

Schallmey M., Singh A., Ward O. P. (2004) Developments in the use of Bacillus species for industrial production. Can J Microbiol. 50:1-17.

Stannek L., Egelkamp R., Gunka K., Commichau F. M. (2014) Monitoring intraspecies competition in a bacterial cell population by cocultivation of fluorescently labelled strains. J Vis Exp. 83:e51196.

Stephens Z. D., Lee S. Y., Faghri F., Campbell R. H., Zhai C., et al (2015) Big Data: Astronomical or Genomical? PLoS Biol. 13:e1002195.

Tang J. (2011) Microbial metabolomics. Curr Genomics. 12:391-403

Voelker U., Voelker A., Haldenwang W. G. (1996) The yeast two-hybrid system detects interactions between Bacillus subtilis sigmaB regulators. J Bacteriol . 178:7020-7023.

Yu G., Wang X. C., Tian W. H., Shi J. C., Wang B., et al (2015) Genomic Diversity and Evolution of Bacillus subtilis. Biomed Environ Sci. 28:620-625.

7. Appendix

7.1 Infrastructure

Table 7.1: The files and their directories. The following table shows the relevant files with a short description.

Directory	Description
./index.php	This is the start page of *Subti*Wiki. There the user can choose her/ his application and run the request.
./search.php	The script offers the opportunity to search for keywords on the gene pages.
./query.php	The script contains a list of excel files which can be downloaded interactively.
./README	Contains a short description of the whole system.
./apps/	A directory containing all the additional applications like *Subti*Pathways, *Subti*Express, *Subt*Interact
./bank/	A directory containing the scripts for the new web platform.
./php/	The directory contains all the php scripts to deliver all the information on the gene, security, editing and history etc.
./wiki/	The directory containing the old wiki engine which will not be described any further.
./static/	The directory contains pictures and css files.
./apps/expression.php	The script generates the expression profiles previously known as *Subti*Express
./apps/expression/	The directory contains additonal script and a css file to generate the expression profile.
./apps/expression/data.php	A script to deliver data of the transcript or protein levels
./apps/expression/expression.css	A style sheet to specify the construction of *./apps/expression.php*
./apps/interact.php	The script generates the interaction profiles previously known as *Subt*Interact.
./apps/interact/	Simply contains a single styling sheet
./apps/interact/style.css	An additional styling sheet to specify the shape of *./apps/interact.php*
./apps/pathway.php	The script generates the metabolic and regulatory pathways previously known *Subti*Pathways.
./apps/pathway/	The directory contains scripts on editing, generating and

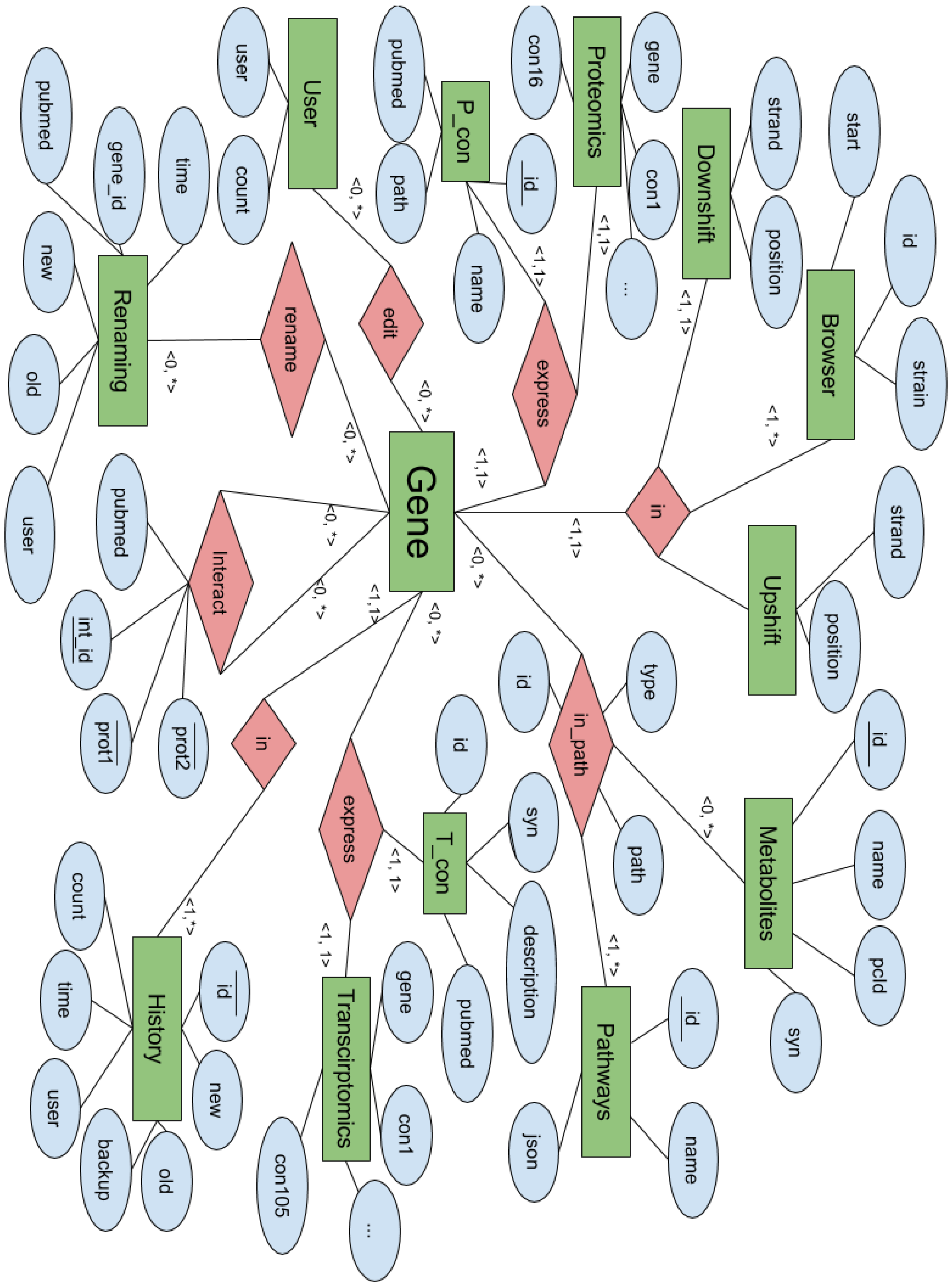
	delivering data of pathways as well as many pictures.
./apps/pathway/action.php	
./apps/pathway/admin.php	It is the main administrative page to generate, delete or edit pathways. It is just accessible with administrative rights.
./apps/pathway/data.php	The script deliver information on single genes/metabolites but also information on the expression profiles which is used to project a colour-coded marker on the pathway.
./apps/pathway/edit.php	It is a script to edit a single pathway. Here the user can assign unknown molecules to the existing ones or generate totally new ones.
./apps/pathway/saveE.php	The script saves the single edit.
./apps/pathway/save.php	The script saves the newly generated pathway.
./apps/pathway/search.php	The script is the surface of a search tool to determine in which a gene or a metabolite is involved.
./apps/pathway/PNG	This directory contains the images of the markers which are interactively projected on the pathways
./apps/pathway/tiles	For the interactive pathways the folder stores the tiles to generate a zoomable interface.
./apps/pathway/zip	The directory contains all XML files which can be downloaded and modified by CellDesigner (http://www.celldesigner.org/).
./apps/action.php	This is a script which receives a variable named “a” and the content of “a” decides which kind of information should be delivered.
./php/	The directory contains all the necessary php scripts that deliver the necessary information for biological content.
./php/Administration.php	This is a general script providing interface to add new genes, interactions, regulations and metabolites.
./php/Browser.php	This file is a script which delivers the relevant data for the genetic context in *Subti*Wiki.
./php/Connection.py	This is a php script with aliased suffix. The script is the main entry for all other script which tries to connect to the database and retrieve relevant information.
./php/Edit.php	The script is the main script to ensure the right edit. The kind of editing is done here.
. /php/Expression.php	This delivers the information for *Subti*Express so speaking the transcriptomic and proteomic data of a gene/protein.
./php/Gene.pl	The php script is aliased and contains an object describing the “Gene” and its properties. The object contains function which delivers the relevant information.
./php/genes.pl	A script that can deliver all metabolites and all genes. The

	created lists are used in the search boxes to propose names by the recent input.
./php/History.php	The script is web platform showing the recent and the previous version of a specific gene. Additionally, it shows which registered user recently changed the page and when the page was changed.
./php/Interact.pl	The script contains an object outputting a JSON object for the interact application.
./php/Lists.php	This is the script which generates the excel sheets which can be downloaded in the “Download area”.
./php/Login.py	The script checks if the user is logged in and which rights the user should have to edit the pages. If the user is not logged in, it is restricted to edit the page.
./php/Metabolite.pl	The script contains an object describing a metabolite.
./php/Pathway.pl	The script contains an object describing a pathway.
./php/Preview.php	The script creates a preview in the search.php to give a first overview on the single gene page.
./php/Pubmed.php	This script extracts the abstracts of the NCBI database and the short description is shown on the lower part of a gene page in the green boxes. The boxes reference to the Pubmed entry.
./php/Query.php	This script is delivering the informatin for the excel sheets.
./php/Recent.php	A script to displaying the recent changes of gene pages. Only the administrator is able to observe the recent changes.
./php/Renaming.php	In some cases the name of gene changes and therefore the script can register the change.
./php/Rollback.php	The script enables the administrator to rollback the recent version or into the backup version of the gene page.
./php/SECURITY.py	The script is a short check if the user has the right to access a file or not. Otherwise the user will be redirected to another place.
./php/Statistics.php	The script is a web platform which provides statistical data on edits, clicks and most prominent user in the database.
./php/Submission.php	The edit of a page has to be controlled. The regular edit of a page encounters the change in the “Gene” table as well as the “History” table. An object called “Submission” is created to do the updates in the specific tables.
./php/Submit.php	The script controls the action and which type should be edited. The types that can be changed are interaction, regulation, metabolite, browser, gene or a single page.
./js/	The directory contains the script written in JavaScript.
./js/capitalize.js	The script extends the String object by transferring the first

	letter of a word to upper case.
./js/chart.js	The script creates the interactive charts presented in *Subti*Express.
./js/default.js	With this script the interactive pathways are initiated.
./js/gene.js	The script creates the interactive interaction profile on a gene page and moreover the 3 D structure is shown on the page if a structure exists.
./js/genomeBr.js	The script creates the interactive genetic context which displayed on the screen.
./js/gradient.js	The script creates an gradient scale that is used to show the expression value of the color-coded nodes shown in *Subt*Interact if the SVG is overlayed with transcriptomic or proteomic data.
./js/interactome.js	The script creates the interactive interaction profiles shown in *Subt*Interact.
./js/legend.js	The script creates a legend for the pathways shown in *Subti*Pathways if the the pathways are overlayed transcriptomic or proteomic data.

7.2 MySQL

7.2.1 Entity-relationship model (without attributes of "Gene" entity)



7.2.2 Creation statements

```
 CREATE TABLE `Browser` (
  `id` int(6) NOT NULL,
  `start` int(9) DEFAULT NULL,
  `stop` int(9) DEFAULT NULL,
  `strain` int(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `Downshift` (
  `strand` int(1) DEFAULT NULL,
  `position` int(8) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

 CREATE TABLE `Upshift` (
  `strand` int(1) DEFAULT NULL,
  `position` int(8) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

 CREATE TABLE `Gene` (
  `id` int(6) NOT NULL AUTO_INCREMENT,
  `name` varchar(15) NOT NULL,
  `locus` varchar(40) DEFAULT NULL,
  `func` blob,
  `product` blob,
  `syn` tinyblob,
  `json` blob,
  `mw` float DEFAULT NULL,
  `pI` float DEFAULT NULL,
  `description` text,
  `essential` varchar(10) DEFAULT NULL,
  `uniprot` varchar(20) DEFAULT NULL,
  `ec` varchar(30) DEFAULT NULL,
  `subtilist` varchar(30) DEFAULT NULL,
  `path` varchar(30) DEFAULT NULL,
  `dna` blob,
  `aminos` blob,
  `count` int(10) DEFAULT '0',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6023 DEFAULT CHARSET=latin1

CREATE TABLE `History` (
  `id` int(6) DEFAULT NULL,
  `new` blob,
  `old` blob,
  `backup` blob,
  `user` varchar(255) DEFAULT NULL,
  `time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  `count` int(10) DEFAULT '0',
  KEY `id` (`id`),
  CONSTRAINT `History_ibfk_1` FOREIGN KEY (`id`) REFERENCES `Gene` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1

 CREATE TABLE `Interact` (
  `int_id` int(6) NOT NULL AUTO_INCREMENT,
  `prot1` int(6) NOT NULL,
  `prot2` int(6) NOT NULL,
  `pubmed` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`int_id`),
  KEY `prot1` (`prot1`),
  KEY `prot2` (`prot2`),
  CONSTRAINT `Interact_ibfk_1` FOREIGN KEY (`prot1`) REFERENCES `Gene` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `Interact_ibfk_2` FOREIGN KEY (`prot2`) REFERENCES `Gene` (`id`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=7790 DEFAULT CHARSET=latin1

CREATE TABLE `Metabolites` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  `pcId` int(8) DEFAULT NULL,
  `syn` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=300 DEFAULT CHARSET=latin1

CREATE TABLE `P_con` (
  `id` int(6) NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `description` blob NOT NULL,

  `pubmed` int(10) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=latin1

 CREATE TABLE `Pathways` (
  `id` int(3) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) DEFAULT NULL,
  `json` blob,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=50 DEFAULT CHARSET=utf8

CREATE TABLE `Proteomics` (
  `gene` int(6) NOT NULL,
  `con1` int(6) DEFAULT '0',
  `con2` int(6) DEFAULT '0',
  `con3` int(6) DEFAULT '0',
  `con4` int(6) DEFAULT '0',
  `con5` int(6) DEFAULT '0',
  `con6` int(6) DEFAULT '0',
  `con7` int(6) DEFAULT '0',
  `con8` int(6) DEFAULT '0',
  `con9` int(6) DEFAULT '0',
  `con10` int(6) DEFAULT '0',
  `con11` int(6) DEFAULT '0',
  `con12` int(6) DEFAULT '0',
  `con13` int(6) DEFAULT '0',
  `con14` int(6) DEFAULT '0',
  `con15` int(6) DEFAULT '0',
  `con16` int(6) DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `Renaming` (
  `gene_id` int(5) DEFAULT NULL,
  `new` varchar(20) DEFAULT NULL,
  `old` varchar(20) DEFAULT NULL,
  `user` varchar(255) DEFAULT NULL,
  `time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,

  `pubmed` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `T_con` (
  `id` int(11) DEFAULT NULL,
  `real_name` varchar(15) DEFAULT NULL,
  `syn` varchar(50) DEFAULT NULL,
  `description` longtext,
  `pubmed` int(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `Transcriptomics` (
  `gene` int(6) NOT NULL,
  `con1` decimal(5,3) DEFAULT NULL,
  `con2` decimal(5,3) DEFAULT NULL,
  `con3` decimal(5,3) DEFAULT NULL,
  `con4` decimal(5,3) DEFAULT NULL,
  `con5` decimal(5,3) DEFAULT NULL,
  `con6` decimal(5,3) DEFAULT NULL,
  `con7` decimal(5,3) DEFAULT NULL,
  `con8` decimal(5,3) DEFAULT NULL,
  `con9` decimal(5,3) DEFAULT NULL,
  `con10` decimal(5,3) DEFAULT NULL,
  `con11` decimal(5,3) DEFAULT NULL,
  `con12` decimal(5,3) DEFAULT NULL,
  `con13` decimal(5,3) DEFAULT NULL,
  `con14` decimal(5,3) DEFAULT NULL,
  `con15` decimal(5,3) DEFAULT NULL,
  `con16` decimal(5,3) DEFAULT NULL,
  `con17` decimal(5,3) DEFAULT NULL,
  `con18` decimal(5,3) DEFAULT NULL,
  `con19` decimal(5,3) DEFAULT NULL,
  `con20` decimal(5,3) DEFAULT NULL,
  `con21` decimal(5,3) DEFAULT NULL,
  `con22` decimal(5,3) DEFAULT NULL,
  `con23` decimal(5,3) DEFAULT NULL,
  `con24` decimal(5,3) DEFAULT NULL,
  `con25` decimal(5,3) DEFAULT NULL,
  `con26` decimal(5,3) DEFAULT NULL,
  `con27` decimal(5,3) DEFAULT NULL,
  `con28` decimal(5,3) DEFAULT NULL,
  `con29` decimal(5,3) DEFAULT NULL,
  `con30` decimal(5,3) DEFAULT NULL,
  `con31` decimal(5,3) DEFAULT NULL,
  `con32` decimal(5,3) DEFAULT NULL,
  `con33` decimal(5,3) DEFAULT NULL,
  `con34` decimal(5,3) DEFAULT NULL,
  `con35` decimal(5,3) DEFAULT NULL,
  `con36` decimal(5,3) DEFAULT NULL,
  `con37` decimal(5,3) DEFAULT NULL,
  `con38` decimal(5,3) DEFAULT NULL,
  `con39` decimal(5,3) DEFAULT NULL,
  `con40` decimal(5,3) DEFAULT NULL,
  `con41` decimal(5,3) DEFAULT NULL,
  `con42` decimal(5,3) DEFAULT NULL,
  `con43` decimal(5,3) DEFAULT NULL,
  `con44` decimal(5,3) DEFAULT NULL,
  `con45` decimal(5,3) DEFAULT NULL,
  `con46` decimal(5,3) DEFAULT NULL,
  `con47` decimal(5,3) DEFAULT NULL,
  `con48` decimal(5,3) DEFAULT NULL,
  `con49` decimal(5,3) DEFAULT NULL,
  `con50` decimal(5,3) DEFAULT NULL,
  `con51` decimal(5,3) DEFAULT NULL,
  `con52` decimal(5,3) DEFAULT NULL,
  `con53` decimal(5,3) DEFAULT NULL,
  `con54` decimal(5,3) DEFAULT NULL,
  `con55` decimal(5,3) DEFAULT NULL,
  `con56` decimal(5,3) DEFAULT NULL,
  `con57` decimal(5,3) DEFAULT NULL,
  `con58` decimal(5,3) DEFAULT NULL,
  `con59` decimal(5,3) DEFAULT NULL,
  `con60` decimal(5,3) DEFAULT NULL,
  `con61` decimal(5,3) DEFAULT NULL,
  `con62` decimal(5,3) DEFAULT NULL,
  `con63` decimal(5,3) DEFAULT NULL,
  `con64` decimal(5,3) DEFAULT NULL,
  `con65` decimal(5,3) DEFAULT NULL,
  `con66` decimal(5,3) DEFAULT NULL,
  `con67` decimal(5,3) DEFAULT NULL,
  `con68` decimal(5,3) DEFAULT NULL,
  `con69` decimal(5,3) DEFAULT NULL,
  `con70` decimal(5,3) DEFAULT NULL,
  `con71` decimal(5,3) DEFAULT NULL,
  `con72` decimal(5,3) DEFAULT NULL,
  `con73` decimal(5,3) DEFAULT NULL,
  `con74` decimal(5,3) DEFAULT NULL,
  `con75` decimal(5,3) DEFAULT NULL,
  `con76` decimal(5,3) DEFAULT NULL,
  `con77` decimal(5,3) DEFAULT NULL,
  `con78` decimal(5,3) DEFAULT NULL,
  `con79` decimal(5,3) DEFAULT NULL,
  `con80` decimal(5,3) DEFAULT NULL,
  `con81` decimal(5,3) DEFAULT NULL,
  `con82` decimal(5,3) DEFAULT NULL,
  `con83` decimal(5,3) DEFAULT NULL,
  `con84` decimal(5,3) DEFAULT NULL,
  `con85` decimal(5,3) DEFAULT NULL,
  `con86` decimal(5,3) DEFAULT NULL,
  `con87` decimal(5,3) DEFAULT NULL,
  `con88` decimal(5,3) DEFAULT NULL,
  `con89` decimal(5,3) DEFAULT NULL,
  `con90` decimal(5,3) DEFAULT NULL,
  `con91` decimal(5,3) DEFAULT NULL,
  `con92` decimal(5,3) DEFAULT NULL,
  `con93` decimal(5,3) DEFAULT NULL,
  `con94` decimal(5,3) DEFAULT NULL,
  `con95` decimal(5,3) DEFAULT NULL,
  `con96` decimal(5,3) DEFAULT NULL,
  `con97` decimal(5,3) DEFAULT NULL,
  `con98` decimal(5,3) DEFAULT NULL,
  `con99` decimal(5,3) DEFAULT NULL,
  `con100` decimal(5,3) DEFAULT NULL,
  `con101` decimal(5,3) DEFAULT NULL,
  `con102` decimal(5,3) DEFAULT NULL,
  `con103` decimal(5,3) DEFAULT NULL,
  `con104` decimal(5,3) DEFAULT NULL,
  `con105` decimal(5,3) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

 CREATE TABLE `User` (
  `user` varchar(255) DEFAULT NULL,
  `count` int(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `in_path` (
  `id` int(5) DEFAULT NULL,
  `path` int(3) DEFAULT NULL,
  `type` varchar(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

7.3 Python code

http://subtiwiki.uni-goettingen.de/static/python.txt

7.4 JavaScript code

7.4.1 genomeBr.js

http://subtiwiki.uni-goettingen.de/js/genomeBr.js

7.4.2 interactome.js

http://subtiwiki.uni-goettingen.de/js/interactome.js

7.4.3 chart.js

http://subtiwiki.uni-goettingen.de/js/chart.js

7.4.4 legend.js

http://subtiwiki.uni-goettingen.de/js/legend.js

7.4.5 request.js

http://subtiwiki.uni-goettingen.de/js/request.js

7.5 PHP code

7.5.1 Gene.pl

http://subtiwiki.uni-goettingen.de/static/Gene.txt

7.5.2 Bank extension

http://subtiwiki.uni-goettingen.de/static/Bank.txt

7.6 New Markup language

Editing Syntax	Translation to HTML	Description
== **Header** ==	<h2>Header</h2>	A title of a category
***Subtitle**	<h3>Subtitle</h3>	A subititle of a category
****line**	<li>**line** </li>	• line
[[**gene**]]	<a href="index.php?gene=**gene**" target="_blank">**gene**</a>	Link to a gene page
[[**gene**|**synonym**]]	<a href="index.php?gene=**gene**" target="_blank">**synonym**</a>	Link to a gene page but the synonym will be visible
[**SW**|**topic**]	<a href="../wiki/index.php?title=**topic**" target="_blank">**topic**</a>	A link to the old version of *Subti*Wiki

[**SW**|**topic**|**synonym**]	<a href="../wiki/index.php?title=**topic**" target="_blank">**synonym**</a>	A link to the old version of *Subti*Wiki but using a synonym
[**Pubmed**|**id**]	<a href="http://www.ncbi.nlm.nih.gov/pubmed/**id**" target="_blank">**PubMed**</a>	A link to a PubMed entry
[**PDB**|**id**]	<a href="http://www.rcsb.org/pdb/explore/explore.do?pdbId=**id**" target="_blank">**id**</a>	A link to the PDB database
''**italic**''	<i>**italic**</i>	The font is written in italic
'**bold**'	<strong>**bold**</strong>	The font is written in bold letters

Lebenslauf

Persönliche Daten

Vorname	Raphael Heinrich
Name	Michna
Geburtsdatum:	12.6.1986 in Paderborn
Adresse	Goßlerstr. 68a, 37075 Göttingen

Wissenschaftlicher Werdegang

Sommer 2006	Abitur am Gymnasium Schloß Neuhaus -Paderborn
Wintersemester 2007/08	Beginn des Biologiestudiums in Göttingen
Sommersemester 2010	Bachelorarbeit *“Entwicklung von Software zur Erstellung interaktiver Diagramme des Stoffwechselweges von Bacillus subtilis”*
Wintersemester 2010/11	Beginn des Masterstudiums “*Microbiology and Biochemistry*”
Wintersemester 2012/13	Masterarbeit *“SubtiExpress – A SubtiWiki application to visualize gene expression under stress conditions and during different growth phases”*
Dezember 2012 – November 2015	Promotionsarbeit im GGNB-Programm “*Microbiology and Biochemistry*”