INAUGURAL - DISSERTATION

zur

Erlangung der Doktorwürde

der

Naturwissenschaftlich-Mathematischen Gesamtfakultät

der Ruprecht-Karls-Universität

Heidelberg

vorgelegt von

Diplom-Evangelos Pafilis

aus: Kozani, Griechenland

Tag der mündlichen Prüfung:

DISSERTATION

submitted to the
Combined Faculties for the Natural Sciences and for Mathematics
of the Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Sciences

presented by
Evangelos Pafilis
born in: Kozani, Greece

Date of oral examination:

Web-based Named Entity Recognition and Data Integration to Accelerate Molecular Biology Research

Referees: Dr. Peer Bork
Prof. Dr. Roland Eils

Acknowledgements

First and foremost I would like to thank my supervisor Dr. Reinhard Schneider without whom the work presented in this thesis would not have been possible. Reinhard did not only give me the opportunity to work under his guidance; he provided with all the freedom and ample support to find my way in the cross roads of literature mining, data integration and web technologies. Through his visions and advices he guided me towards discovering new scientific niches and exploring novel solutions to bioinformatics challenges.

I would also like to thank my Thesis Advisory Committee members: Dr. Peer Bork, Dr. Eileen Furlong and Prof. Roland Eils for their useful input and for motivating me to pursue my work further.

Many thanks to Dr. Toby Gibson for bringing me in contact with the EMBRACE network and helping me this way to gain state of the art input in web technologies. The EMBRACE workshop has been a cornerstone.

Special thanks to Dr. Ela Hunt that introduced me to the world of data integration and never stopped providing me with support and advices.

For the Reflect project I would like to thank Dr Sean I O'Donoghue, Dr Lars Juhl Jensen, Heiko Horn, Michael Kuhn and Dr Nigel Brown not only for their support in web application developing and literature mining, but also for their enthusiasm and energy.

The OnTheFly project has also been a great collaboration for which I should thank Dr Sean Hooper and Georgios Pavlopoulos.

Venkata Satagopam has been a great colleague and is perhaps one of the few people with enough patience to support me on the EasySRS project, along with Adriano.

I would also like to thank the rest of the group members Theodoros, Bettina, Mechthilde, Andrea for all the discussion, exchange of ideas, the work we did and the fun we had together.

Dr Yan Yuan, Tobias Sack, and Michael Wahlers and the rest of the IT group for the technical support they me provided with.

Special thanks to Lindsay, Anne Marie, Carlo and Rizo whose support Was priceless. Georg, Anna, Konrad, Aidan, Niall, Pedro, Mark, Kostas, Dimitris and many more whose names I may be forgetting.

My family which are always there for me; they are always in my heart and my mind no matter how many thousands of kilometres far away they live.

Table of Contents

Summary

Finding information about a biological entity is a step tightly bound to molecular biology research. Despite ongoing efforts, this task is both tedious and time consuming, and tends to become Sisyphean as the number of entities increases. Our aim is to assist researchers by providing them with summary information about biological entities while they are browsing the web, as well as with simplified programmatic access to biological data. To materialise this aim we employ emerging web technologies offering novel web-browsing experiences and new ways of software communication

Reflect is a tool that couples biological named entity recognition with informative summaries, and can be applied to any web page, during web browsing. Invoked either via its browser extensions or via its web page, Reflect highlights gene, protein and chemical molecule names in a web page, and, dynamically, attaches to them summary information. The latter provides an overview of what is known about the entity, such as a description, the domain composition, the 3D structure and links to more detailed resources. The annotation process occurs via easy-to-use interfaces. The fast performance allows for Reflect to be an interactive companion for scientific readers/researchers, while they are surfing the internet.

OnTheFly is a web-based application that not only extends Reflect functionality to Microsoft Word, Microsoft Excel, PDF and plain text format files, but also supports the extraction of networks of known and predicted interactions about the entities recognised in a document.

A combination of Reflect and OnTheFly offers a data annotation solution for documents used by life science researchers throughout their work.

EasySRS is a set of remote methods that expose the functionality of the Sequence Retrieval System (SRS), a data integration platform used in providing access to life science information including genetic, protein, expression and pathway data. EasySRS supports simultaneous queries to all of the integrated resources. Accessed from a single point, via the web, and based on a simple, common query format, EasySRS facilitates the task of biological data collection and annotation. EasySRS has been employed to enrich the entries of a Plant Defence Mechanism database.

UniprotProfiler is a prototype application that employs EasySRS to generate graphs of knowledge based on database record cross-references. These graphs are converted into 3D diagrams of interconnected data. The 3D diagram generation occurs via Systems Biology visualisation tools that employ intuitive graphs to replace long result lists and facilitate hypothesis generation and knowledge discovery.

Zusammenfassung

Das Auffinden von biologischen Entitäten ist ein Schritt der eng an die molekularbiologische Forschung geknüpft ist. Trotz laufender Bemühungen ist diese Aufgabe sowohl aufwändig als auch zeitintensiv und wird mit steigender Anzahl an Entitäten (Einheiten) unpraktikabel . Das Ziel unserer Arbeit ist es, Forscher Zusammenfassungen von Informationen über biologische Entitäten zur Verfügung zu stellen waehrend sie das Web nutzen , sowie auch vereinfachten, programmbasierten Zugang zu biologischen Daten zu ermöglichen.

Das Programm Reflect verknüpft die Erkennung biologischer Namen von Entitäten (Einheiten) mit informativen Zusammenfassungen. Es kann auf jeder Webseite während des Browsens angewandt werden. Zudem kann es als Browser-Zusatzfunktion oder als Webseite aufgerufen werden. Es hebt Namen von Genen, Proteinen und chemischen Molekülen hervor und versieht diese dynamisch mit zusammengefassten Informationen. Letztere Funktion gibt einen Überblick über das vorhandene Wissen über eine Entitä (Eineit)t, wie etwa eine Beschreibung, die Domänenkonstellation, die dreidimensionale Struktur und Verweise zu detaillierteren Ressourcen. Der Annotationsprozess findet mittels einer leicht zu handhabenden Schnittstelle statt. Durch seine Schnelligkeit unterstützt Reflect den wissenschaftlichen Leser/Forscher beim Websurfen als interaktiver Partner.

OnTheFly ist eine web-basierte Anwendung, die nicht nur die Funktionalität von Reflect auf Dateien von Microsoft Word, Microsoft Excel, PDF und Plaintextdatein erweitert, sondern es unterstützt auch die Extraktion von Netzwerken bekannter und vorausgesagter Interaktionen erkannter Entitäten eines Dokuments.

Eine Kombination von Reflect und OnTheFly bietet somit eine Annotationsdatenlösung für Dokumente, die von Forschern der Biowissenschaften währen ihrer Arbeit genutzt werden.

EasySRS ist eine Sammlung von Remote-Methoden, welche die Funktionalität von Sequence Retrieval Ssystem (SRS) bereit stellt. Letzteres ist eine Datenintegrationsplattform, die Zugang zu Informationen der Biowissenschaften wie genetische, Protein-, Expression- und Stoffwechselwegdaten bietet. EasySRS unterstützt gleichzeitige Anfragen zu allen integrierten Ressourcen. Alle Anfragen verwenden ein gemeinsames, einfaches, web-basiertes Format. SRS sammelt und annotiert daraufhin biologische Daten.. EasySRS wurde angewandt um die Einträge der Plant Defense Mechanism database zu erweitern.

UniprotProfiler ist eine Prototypen-Anwendung, die EasySRS nutzt um basierend auf der Kreuzreferenzierung von Datenbankeinträngen Wissensgraphen zu erzeugen. Diese Graphen werden in 3D-Diagrammeder verknüpften Daten umgewandelt. Die Erzeugung der 3D-Diagramme erfolgt durch das „Systems Biology visualisation“-Werkzeugset, das lange Ergebnislisten durch intuitive Graphen ersetzt und somit die Hyphothesengenierierung und Wissensfindung ermöglicht.

Abbreviations

Ajax	Asynchronous JavaScript and XML
BLM	Biomedical Literature Mining
DAS	Distributed Annotation System
DOM	Document Object Model
FDA	Food and Drug Administration
GO	Gene Ontology
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
iHop	Information Hyperlinked over Proteins
IT	Information Technology
JAX-WS	Java API for XML Web Services
NER	Named Entity Recognition
PDB	Protein Data Bank
PDF	Portable Document Format
PDM	Plant Defence Mechanisms
REST	Representational State Transfer
SDA	Structured Digital Abstract
SMART	Simple Modular Architecture Research Tool
SOAP	Simple Object Access Protocol
SRS	Sequence Retrieval System
STITCH	Search Tool for Interactions of Chemicals
STRING	Search Tool for the Retrieval of Interacting Genes/Proteins
URL	Uniform Resource Locator
WSDL	Web Service Description Language
XML	Extensible Markup Language
SWS	SRS by Web Services

List of Figures

Introduction

Methods

Results

Discussion

1. Introduction

1.1. Prologue

Finding information relevant to a biological entity is a crucial step both for the design of biological experiments, and for the interpretation of their results. Answering questions such as:

- Which is the function if this protein?
- Is this protein a member of a known pathway?
- In which tissue is it expressed and when?
- Which domains does it comprise?
- What is its macromolecular structure?
- Is it involved in a genetic or other type of disease?

is not only the aim of ongoing biological experiments, but is also a prerequisite for their design and the analysis of their results. Gathering such pieces of contextual information is a tedious and time consuming task.

Considering that the focus of biological research is shifting from individual genes and proteins to entire biological systems (Jensen, *et al.*, 2006) this task becomes Sisyphean.

In the following thesis we present software that we developed to aid the collection of information about a biological entity. We also demonstrate how literature mining and data integration can play an active role in dealing with this task.

Additionally, we present emerging web technologies that offer novel web-browsing experiences and facilitate new ways of software communication, and demonstrate how they been applied in producing software to accelerate molecular biology research.

Finally, we discuss how a broad audience, from bench scientists to information technology experts, can benefit from such tools, and we conclude by proposing how these pieces of software can be combined with themselves and with other tools to support further life science researchers.

1.2. The Quest of Collecting Biological Information

1.2.1. A Ubiquitous Issue in Molecular Biology Research

Life science researchers use the web on a daily basis to survey the literature and to collect pieces of information pertinent to certain biological entities (Divoli, *et al.*, 2008).

1.2.1.1. Reading a News Article on the Web

Imagine a user reading a news article talking about the discovery of a new set of genes associated with a disease of his/her interest. The text contains several gene, protein, and/or chemical names he/she is not familiar with. To find more information about these entities the user has to query several sites, either generic in scope, such as Google[1] or more specific ones such as UniProt[2], EBI[3], NCBI[4], PubChem[5], and study the results.

1.2.1.2. Studying a Medline Abstract or a Full Text Article

A similar case can also emerge while reading a Medline[6] abstract or a scientific article. The more entities mentioned, the more time and effort required to collect the related information. Biologists were presented with a PubMed[7] abstract and asked to collect information about the proteins mentioned in the abstract such as domain composition, sequence and structure[8]. Not surprisingly Google, UniProt and the Sequence Retrieval System[9] (SRS) (Etzold, *et al.*, 1996) were some of the sites the biologists resorted to, only to realise how long and tedious this task can be.

[1] http://www.google.com
[2] http://www.uniprot.org
[3] European Bioinformatics Institute, http://www.ebi.ac.uk/
[4] National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov
[5] http://pubchem.ncbi.nlm.nih.gov/
[6] http://www.nlm.nih.gov/pubs/factsheets/medline.html
[7] http://www.pubmed.org
[8] During the "Exploring Protein Modular Architecture" course: February 2008, EMBL, Heidelberg, Germany
[9] http://srs.embl.de

1.2.1.3. Annotating Experimental Results

The impact of how laborious and time consuming this task can be is exemplified by the analysis of microarray experiment results. Scientists, employing mouse breast development as a model for cancer, performed microarray experiments to study gene expression in mouse mammary gland development and involution. The experiments led to the identification of 400 gene names. The researchers reported spending six months characterising only the first 100 genes by web browsing (Hunt, *et al.*, 2004).

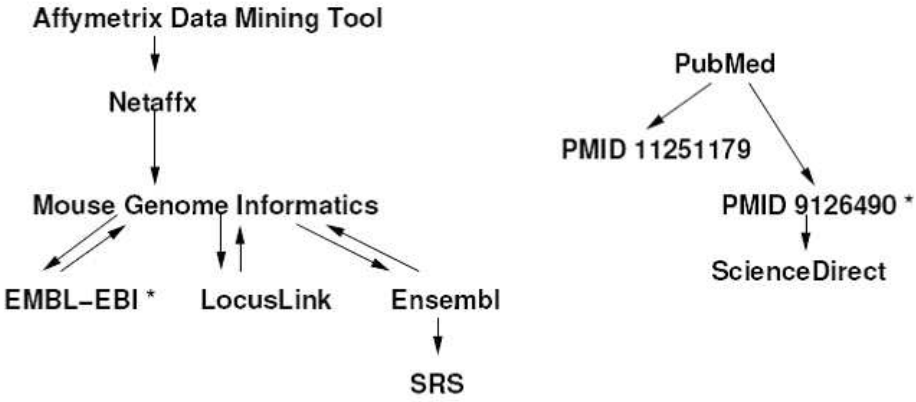


Figure I-1: Web pages visited during searching. * indicates sites that contained material that the researchers found interesting. (figure and legend from: Hunt, *et al.*, 2004)

Their practice was to start with the Affymetrix[1] database and for each probe name identified in the experiment, to find the corresponding gene. The link to the gene was followed, finally leading to PubMed articles. If the article title correlated with the research interest it was downloaded. The sequence of a typical search is shown in Figure I-1. Leading on from the Affymetrix Data Mining Tool, the researchers found useful material in EMBL, but ultimately resorted to PubMed to issue a query for the gene name listed in the Affymetrix database. PubMed produced articles that were of interest, some of which were pursued through to Science Direct[2] and ultimately printed out for subsequent study. This process was followed for all 100 genes and the researcher would like to be able to automate it (Hunt, *et al.*, 2004).

[1] http://www.affymetrix.com
[2] http://www.sciencedirect.com/

1.2.2. Programmatic Collection of Biological Information: Not an Easy Task

A bioinformatician given the task of collecting related pieces of information will have to query several sources, spend effort on unifying the results and converting them to a format that will allow him/her to perform further analysis.

This task has been characterised as disastrous and may involve techniques such as screen-scraping (extraction of data from an HTML (HyperText Markup Language) page) that has been given the title of “mediaeval torture” (Stein, 2002).

Although bioinformatics database resources can answer questions on their own data, they cannot provide information that belongs to another domain, stored in another database (Stein, 2003).

Different resources not only exist in disparate locations, but also use different systems and heterogeneous formats to store their data (Chung and Wooley, 2003; Stein, 2002).

Moreover since different scientific communities perceive certain biological concepts in a different way, the conflict persists also in the way databases are modelling real world objects (Stein, 2003).

Finally, in response to advances in biological research and technology, biological databases are subject to continuous changes in both data content and schema (Chung and Wooley, 2003). Hence data collection should be dynamic and not a one-off event.

1.2.3. Biological Information: Buried in Free Text

Either by browsing the web manually or by following a programmatic approach, users will be presented with pieces of text containing pertinent information. These texts typically mention numerous genes, proteins, complexes, small molecules or other biological concepts, which are collectively referred to as named entities. Sadly, the named entities mentioned in a text are generally not linked to the relevant resources; it is thus not as easy as it could be for a reader to find related information on a particular protein mentioned in, for example, a journal article and a new iteration of

information gathering is required (Evangelos Pafilis, Lars Juhl Jensen, Sean O'Donoghue, manuscript in preparation).

1.2.4. Acceleration Required

It is clear from what has been mentioned so far that collecting biological information consumes a considerable amount of a life sciences researcher's time; time that could have been well spent doing others tasks of biological research.

Reducing the amount of time required for literature browsing, either by providing summaries of related information about the biological entities mentioned in a piece of text, or by alleviating some of the problems in programmatic data collection, would significantly accelerate the pace of molecular biology research.

1.3. Literature Mining and Named Entity Recognition

1.3.1. Biomedical Literature Mining

Biomedical literature mining (BLM) is an umbrella term embracing the processes required to extract biomedical facts from the scientific literature. These processes range from the identification of relevant papers (information retrieval) to the integration of text analysis results with data from biological experiments. Each of them requires different levels of biological expertise, with the more demanding ones having higher discovery potential (Jensen, *et al.*, 2006).

1.3.2. Biological Named Entity Recognition

Biological named entity recognition (NER) is a building block of BLM. NER is involved with the identification of the biological entities mentioned in a piece of text. It occurs in two steps; first the words that refer to biological entities are being recognised and secondly these are given a unique identifier. Unique identification is achieved via the mapping of a biological entity name to a corresponding entry in a biological database (Jensen, *et al.*, 2006; Krallinger and Valencia, 2005). Through the use of NER a researcher can convert a piece of plain text to a piece of document linked to pertinent resources.

1.3.2.1. Approaches

Different research communities have approached the issue of biological NER according to their background. On the one hand the linguist community has tried to identify names based on the syntax of a sentence, taking into consideration the parts of speech and the syntactic roles of words. On the other hand bioinformaticians have based their efforts in the identification of entity names and their synonyms according to the information available in biological databases (Krallinger and Valencia, 2005). The latter, dictionary based systems, attempt to match words in such a way that “allows variation in how the names are written”. (Jensen, *et al.*, 2006).

Many systems try combined approaches to improve performance. However certain well performing NER methods have succeeded in doing so by removing synonyms responsible for many false positive (Jensen, *et al.*, 2006).

An important advantage of dictionary-based approaches, related to the work in this thesis, is that by including database identifiers in the synonym lists, biological entities can be recognised both by their name and their accession numbers (Jensen, *et al*., 2006).

Furthermore dictionary-based approaches tend to be memory rather than CPU intensive. On one hand this translates into increased memory requirements since in the biomedical domain large terminological resources have to be supported (Rebholz-Schuhmann, *et al*., 2008). On the other hand this means that such a system could achieve very quick responses; thus satisfying a prerequisite for developing an interactive user companion.

1.3.2.2. NER as a Stand-alone Module

NER is not only the first but also a fundamental step in the extraction of information from a BLM system (Krallinger and Valencia, 2005). The identification of biological entity names and their cross-linking to corresponding biological databases, allow NER to enrich scientific documents, and render it useful as an individual module (Jensen, *et al*., 2006).

iHop (Information Hyperlinked over Proteins) (Hoffmann and Valencia, 2004) is a good example of such a web-based tool that allows the user to browse sentences from Medline abstracts according to biomedical entities they contain (Jensen, *et al*., 2006). Their motivation is to replace the navigation of long abstract lists with an approach that “is closer to the associative organization of human memory, in which information is retrieved by connecting similar concepts”; in this case sentences extracted from Medline are cross-linked according to the genes and proteins they mention (Hoffmann and Valencia, 2004).

1.3.3. Literature Mining Tools: How Useful Are They?

Performance has reached reasonable levels for a number of basic text mining tasks. However, several questions have been raised: “Do these tools work? Are they usable, and if so, who uses them, and how? Are they cost-effective? Can they be adapted to new domains and maintained over time?” (Kevin Bretonnel Cohen, Pacific Symposium in Bionformatics 2008, call for papers[1]).

Automated recognition of biochemical entities in text, in particular, has recently improved significantly in accuracy (Hirschman *et al*., 2005). Motivation of this work is to develop simple tools that can employ a dictionary-based NER system to automatically enrich the documents that life scientists use throughout their work.

[1] http://psb.stanford.edu/psb08/cfp.html#nlp

1.4. Data Integration

1.4.1. Overview

The problem of programmatic data collection described previously would have been greatly simplified if there were a way of querying and retrieving data from all of the required sources in a simple and consistent way.

Aiming at dealing with such issues is the field of data integration, “a process that combines data from multiple, possible heterogeneous and inconsistent, data sources into a single consistent source” (Lacroix and Critchlow, 2003).

Taking into consideration the idiosyncrasy of bioinformatics data that:

- reside at different locations
- are stored in a variety of systems and formats
- can be structured in a different way even when they represent the same real world object
- use the same name to identify different things
- are always changing as new technologies appear, algorithms improve and new training sets arise

(Chung and Wooley, 2003; Stein, 2002; Stein, 2003)

renders them an ideal source of challenging enough cases on top of which new algorithms and systems could be developed.

On the biological science point of view, it is now clear that answering questions that arise in biology research is no longer possible just by querying the contents of a single database.

Each biological resource contains information about a certain subset of the biological knowledge. Although bioinformatics database resources can answer questions on their own data, they cannot provide information that belongs to another domain, stored in another database (Stein, 2003).

1.4.2. Approches

Data integration systems that have been developed in the field of bioinformatics can be divided in three categories[1] (Stein, 2003)

- Data warehouses
- View integration
- Link integration

1.4.2.1. Data warehouses

In this approach the data from multiple sources are stored locally and converted so as to comply with a common, unified, schema. The steps to create a data warehouse comprise: i. developing the global schema, ii. downloading the data from their sources, iii. converting them to the global schema and loading them in the containing-all local database ("warehouse"). The Integrated Genomic Database (Ritter, *et al.*, 1994), BioMART[2] and databases implementing the Genomics Unified Schema[3] (GUS) are examples of such systems.

1.4.2.2. View Integration

Here no fetching of the data takes place. Instead pieces of software are responsible for understanding which remote databases should be queried, compile the sub-queries required for each one, retrieve the results from each source and merge them in a single output. Examples include the Kleisli (Chung and Wong, 1999) and K2 (Tannen, *et al.*, 2003)systems.

1.4.2.3. Link Integration

This approach is based on linking related records to each other in a similar fashion to following hyperlinks from one record to the other. NCBI/Entrez (McEntyre, 1998) is such a system (Miled, *et al.*, 2004).

[1] Depending on the document scope and the background of the researcher other classifications of data integration systems also apply. In this thesis we present a categorization that was written to address biological audience.
[2] http://www.biomart.org
[3] http://www.gusdb.org/about.php

SRS (Etzold and Argos, 1993), a variation of such system, is explained in section 1.4.3 along with its properties that influenced us on choosing this system for approaching the issue of related information collection.

1.4.2.4. Comparison

Each of these systems and approaches has their own pros and cons (Stein, 2003). Data warehouses on the one hand perform rather quickly in returning results. On the other hand though each time one of the integrated databases updates its content it will have to be downloaded again. Moreover, any change in the schema will require also an update of the scripts adapting its data to the global schema. Maintaining also such a global schema is not an easy task.

View integration systems do not need to download the data, their performance though is rather slow and the maintenance of the database specific query converters is a difficult task (Stein, 2003).

Link driven systems may be some of the most successful ones; they inherit, though, the link-associated disadvantages. Should, for example, a record become obsolete, other records pointing to it are prone on having void associations (Stein, 2003).

The emerging web technology of Web services is another variant of link-based integration. Due to their potential impact on the issue of data integration, Web services are being discussed separately in section 1.6.3.

1.4.3. The Sequence Retrieval System (SRS)

Similar to an index in the end of a book that allows readers to navigate to pages where a certain word occurs, SRS is an index of the “open-ended book of databases” that facilitates the retrieval of the records where a certain keyword exists (Figure I-2).

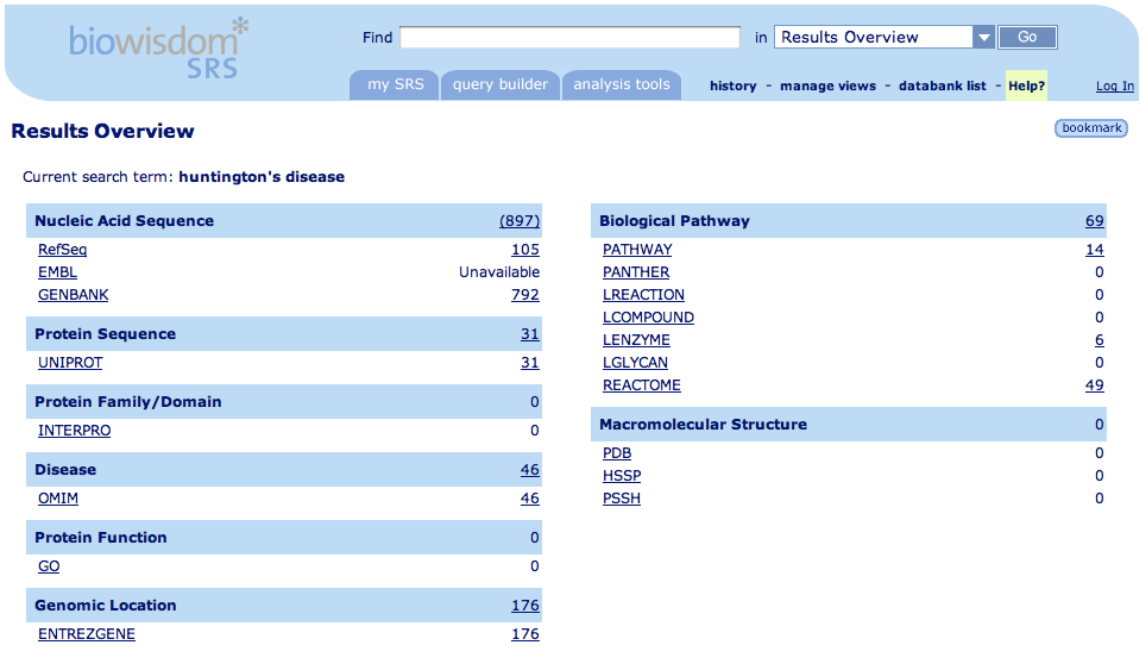


Figure I-2: Keyword-based query for “huntington's disease” on all the databases integrated by the SRS at EMBL Heidelberg (http://srs.embl.de).

Besides being an indexing system, SRS follows a data warehouse approach in the sense that the contents of the databases to be indexed are periodically downloaded locally (Hunt, *et al.*, 2004). This means that certain effort is required to maintain SRS up-to-date, it also means, though, fast performance.

SRS follows a view-integration approach in the sense that the downloaded data are left at their original format (Etzold, *et al.*, 2003). Manual work is required to map the fields of a database to a corresponding set of fields indexed by an SRS system. The existence of publicly accessible parsers for common life science databases alleviates part of this effort. Moreover, through this mapping process it is possible for SRS curators to assign common attributes to fields in different databases with the same semantics, but different names. This increases uniformity and facilitates multiple database queries (Zdobnov, *et al.*, 2002).

As mentioned already SRS is also a link-based data integration system. This is due to the SRS ability to follow explicit and implicit links among records. This feature can be used to construct an “SRS Universe”, a network of cross-linked biological database records (Figure I-3).

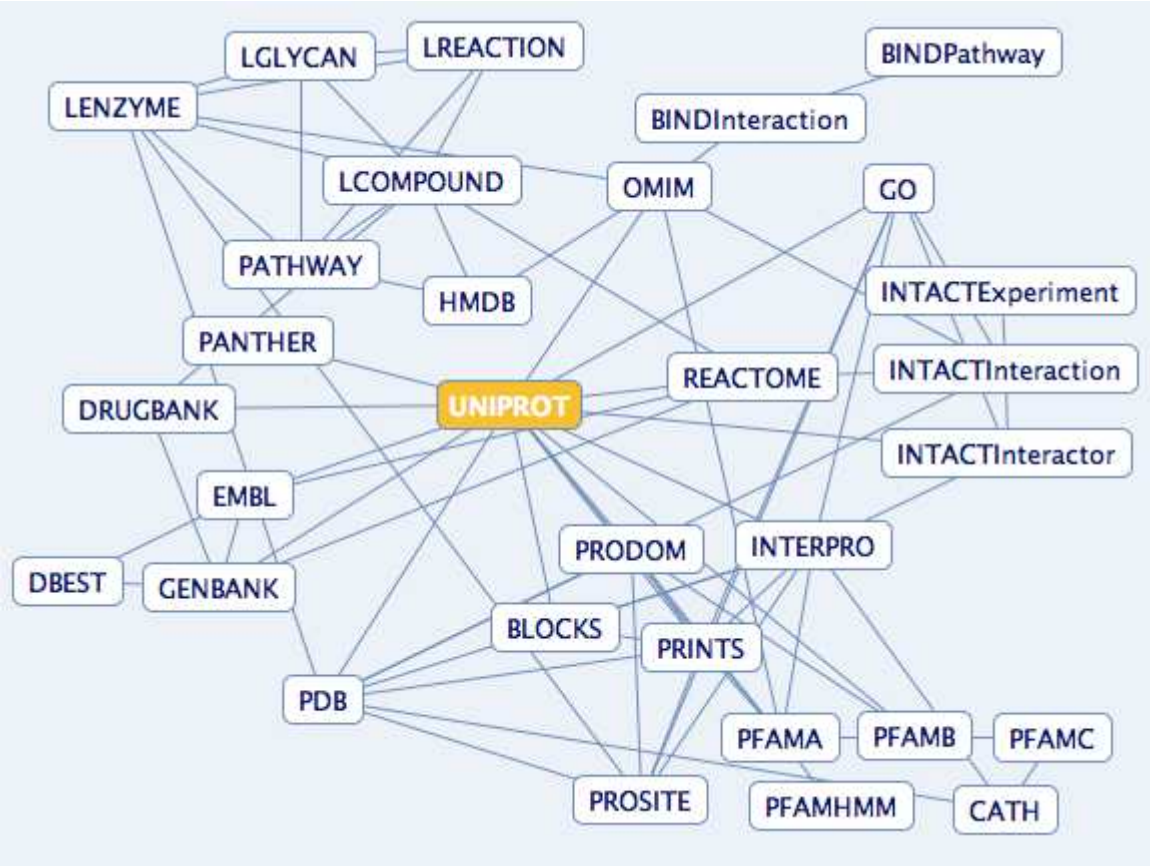


Figure I-3: A sample section of the "SRS universe" of cross-linked biological databases. The databases shown in this graph are, either directly, or indirectly, linked to UniProt and cover several life science concepts such as sequences, structures, pathways, small molecules, interactions and protein domains.

Although SRS can serve as a generic data integration system, it was originally developed for the indexing and the retrieval of biological sequences (Etzold and Argos, 1993; Etzold, *et al.*, 1996). The fact that SRS has been used in the field of life sciences for so many years, used by several academic institution and commercial entities has resulted in the capturing of information from many sources covering several concepts of life sciences. This includes genetic, protein, expression, pathway, molecular and clinical data from public and proprietary sources[1]. As supplied, SRS currently supports access to over 100 data sources. The impact of SRS on the life sciences can be conveyed from the fact that at the time of writing there are 32 publicly available SRS servers, offering, in total, access to more than 426 databases[2].

[1] http://www.biowisdom.com/navigation/srs/srs

[2] http://downloads.lionbio.co.uk/publicsrs.html This web page contains up-to-date information regarding the available public SRS servers world-wide

1.4.3.1. Programmatic Access to SRS

SRS allows simultaneous queries to all of the integrated databases from a single access point, using a simple, yet powerful, query format. To access the functionality of SRS programmatically and embed it in custom-developed application the following ways of interacting with an SRS server exist:

- via the command line. This type of communication requires that the SRS client application and the SRS server reside on the same machine (Etzold, *et al.*, 2003).
- using the SRS Objects. These include the SRS Common Object Request Broker Architecture Server (SRSCS). This solution is suited for client-server applications only on the same local area networks (LAN) (Etzold, *et al.*, 2003).
- based on the SRS wgetz web-application. A set of examples can be found at http://www.ebi.ac.uk/~srs/wiki/doku.php?id=guides:linkingtosrs. This type of access is web-based. The inclusion, nevertheless, of encoded characters and session related information in its syntax make it difficult to use.
- via the SRS web services objects[1] (SRS WSObjects™). A set of client-side components that can interact over the web with a remote SRS server in a fast and flexible way. However, they do not always fulfil the requirements imposed by the many possible usage scenarios or for access by third party applications[2].

Thus, solutions exist that provide developers with programmatic access to an SRS server. However, the requirement for an easy-to-use, web-based method, free from any firewall issues still remains.

[1] http://www.biowisdom.com/content/srs-wsobjects
[2] http://www.nettab.org/2005/docs/NETTAB2005_StainesOral.pdf

1.5. Experimental Result and Text Mining Derived Association Integration

As already presented, navigating through bioinformatics data can be accomplished based on internet-style links. However, database record cross-references are not the only type of association. Studying chromosomal position proximity, and applying clustering algorithms to sequence similarity information and expression profiles from high throughput experiments, can provide evidence on associated gene groups, potentially responsible for specific physiological processes (Kanehisa and Bork, 2003). These higher level associations can be combined further with protein-protein interaction experiment results. Overall they can provide a better understanding of cell organisation and molecular function.

STRING and STITCH are two databases that systematically collect such pieces of information by employing data integration and statistical analysis techniques.

1.5.1.1. STRING: Search Tool for the Retrieval of Interacting Genes/Proteins

The STRING database (Search Tool for the Retrieval of Interacting Genes/proteins) provides researchers with protein associations, covering a large number of organisms. Associations are not limited to direct physical binding but also include indirect (functional) ones such as co-expression (von Mering, *et al.*, 2007). Figure I-4 displays the sources used for populating the STRING database with known and predicted interactions: high throughput experimental data, mining of databases and literature, predictions based on genomic context analysis.

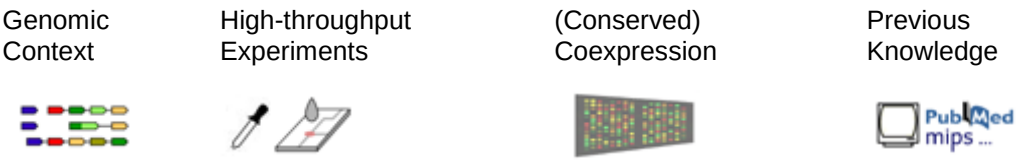


Figure I-4: The sources used for populating the STRING database with known and predicted interactions (source: http://string.embl.de)

An important feature of STRING is that the proteins it contains are derived from completely sequenced genomes and that interaction annotation is transferred among species based on evidence of orthology (von Mering, *et al.*, 2007).

1.5.1.2. STITCH: Search Tool for Interactions of Chemicals

The STITCH[1] (Search Tool for InteracTions of CHemicals) resource extends the protein association information available in STRING by integrating known and predicted chemical – protein and chemical – chemical interactions. Known interactions have been extracted from: metabolic pathways, crystal structures, binding experiments and drug–target relationships, while associations between chemicals have been predicted based on: phenotypic effects, text mining and chemical structure similarity (Kuhn, *et al.*, 2008). Based on protein – chemical association networks STITCH bridges the gap between biology and chemistry and allows for hypothesis generation regarding, e.g. the effect of a group of chemicals on certain group of proteins.

1.5.1.3. Literature Derived Associations & Related Knowledge Summaries

Both STRING and STITCH employ text mining analysis to derive associations among chemicals and proteins (either to each other or to them selves). For the NER purposes a dictionary of over 1.5 million proteins from 373 organisms and over 3 million small molecules has been used (Pafilis *et al*, in preparation). The NER module had been developed for the processing of Medline abstracts.

Moreover a user-friendly feature of STRING, STITCH is the display of summary information upon clicking a network node (Figure I-5). This feature is now programmatically accessible[2] and can be embedded by developers in their applications.

[1] http://stitch.embl.de/
[2] http://string-stitch.blogspot.com/2008/03/embedding-proteinchemical-information.html and http://string-stitch.blogspot.com/2008/02/we-have-api.html

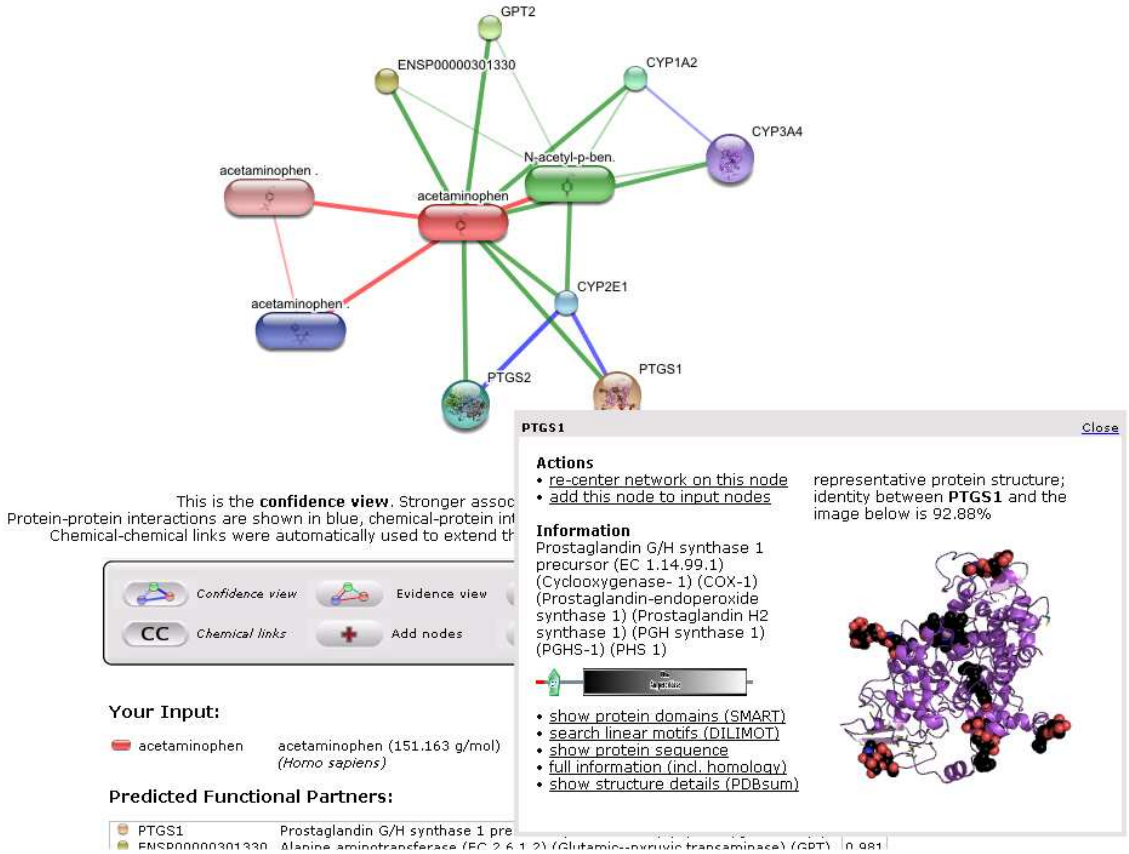


Figure I-5: Clicking on the PTGS1 node of the STITCH network displays a summary popup with a short description of the protein, an overview of is domain architecture, a 3D structure and links to related resources. This network is the result of querying STITCH for “paracetamol” and associated human proteins.

The perspective of adapting the STRING and STITCH NER machinery to documents commonly used by life sciences researchers and its coupling with informative summaries such as the previous has been a motivating energy of the work in this thesis.

1.6. Emerging Web Technologies and Bioinformatics

Technological changes and new developments in computer science and Information Technology (IT) occur even faster than in the rapidly changing domains of genomics, proteomics etc. Recently, several new technologies and trends such as Web 2.0, Service Oriented Architectures (SOA) and other Web related technologies e.g. Asynchronous JavaScript and XML (Ajax) have been introduced. Since many bioinformatics tools and biological databases are deployed through and depend on the Internet, these new technologies seem to be of considerable importance for users as well as for developers of tools (Stockinger, *et al.*, 2008).

The “driving axle” of this thesis revolves around how some of these technologies, browser extensions, Ajax and Web services, can be exploited to assist life science researchers in their quest to find relevant information.

1.6.1. Browser Extensions

Browser extensions are an emerging family of tools; tools that enable the real-time modification of web pages and their enrichment with information from other resources.

Such tools extend the capabilities of a browser by borrowing its own machinery. They are able to access the contents of a web page, modify its appearance, eg. by removing advertisements, and enrich it by collecting and displaying information from other resources (Good, *et al.*, 2006).

iHOPerator, “Userscripts for Life Sciences”, Concept Web Linker for Firefox and Cohse are only some examples of this type of technology being applied in the field of bioinformatics.

iHOPerator (Good, *et al.*, 2006) for example, a Greasemonkey[1] script for the Firefox[2] browser, enhances user's experience when visiting an iHOP gene web page (Hoffmann and Valencia, 2004) by enriching it with "tag clouds". The "tag could" comprises either Medical Subject Headings[3] (MESH) keywords assigned to the abstracts associated with the gene, or other gene names occurring in the same set of abstracts. (Good, *et al.*, 2006).

"Userscripts for Life Sciences" are a set of biology and chemistry oriented Greasemonkey scripts (Willighagen, *et al.*, 2007). ChemGS.user.js, for example, is a script that employs OSCAR3 (Corbett and Murray-Rust, 2006), a natural language processing tool, to automatically mark-up chemical names in web pages and provide users with hyper-links to their entries in PubChem[4] and a two-dimensional (2D) image of their structure (Willighagen, *et al.*, 2007).

Concept Web Linker for Firefox[5] is able to annotate behavioural, anatomical, and physiological terms, as well as chemical, disease, gene, and organism names. It modifies the font colour according to the entity type and attaches a popup with a definition of the concept and a link to related information at the Concept Web Navigator resource (Mons, *et al.*, 2008).

Cohse (Bechhofer, *et al.*, 2005) also allows users to modify web pages on-the-fly, highlight ontological terms and attach to them relevant pieces of information. Cohse can be accessed either via a Firefox extension or via its web site. The web site allows users to type-in a web address; the corresponding web page will be annotated and returned to the user.

[1] http://wiki.greasespot.net/Greasemonkey: a Firefox extension that simplifies the development of other Firefox extensions, as Greasemonkey scripts
[2] http://www.mozilla.com/en-US/firefox/
[3] http://www.nlm.nih.gov/mesh/, a controlled vocabulary produced by the National Library of Medicine and used for indexing, cataloguing, and searching for biomedical and health-related information and documents
[4] http://pubchem.ncbi.nlm.nih.gov/
[5] http://www.knewco.com/linker-plugin.php

1.6.2. AJAX: Improved User Interaction with Web-based Applications

If you have ever used an auto-complete suggestion based on what you have typed in a web form, or if you have used Google Maps[1] to locate a scientific institute or a conference venue, then you have seen in action how Ajax supports faster and better interaction with web applications.

Based on a combination of Web technologies (Yesilada, *et al.*, 2007), Ajax is a programming approach that enables a web page to request a certain piece of information from a server and update the relevant components only.

This is what makes Ajax different from conventional approaches that perform much slower since they have to update the whole web page rather than just specific elements.

In the previous examples it is only the suggestion list which is being updated based on the options supported by the server matching the letters that the user typed-in, or only the Google map area depending on the mouse behaviour.

Besides performance, Ajax's ability to pose specific requests to a server is of particular importance when developing architectures that enable seamless access to bioinformatics tools and data since it can provide the underlying machinery on top of which such a platform could be built.

If the web browser extensions described in the previous section are able to dynamically update and modify the contents of a web page, to a great extend, they can do so because they employ Ajax "under the hood".

Should a browser extension be developed to invoke, in an Ajax-fashion, a server-side component to apply NER in the contents of a web page, that could be the means of recognising the named entities in a web page, and linking them to biological database information.

[1] http://maps.google.com/

1.6.3. Web Services: Emerging Data Integration Technology

As mentioned already in the Data Integration section, providing access to bioinformatics data available at different locations is a challenge that has to be met to answer biological questions. An emerging technology that could serve bioinformatics in this respect is the one referred to as "Web service".

The term Web service was originally coined as a specific World Wide Web Consortium (W3C) standard[1], however, more recently it has been used to refer to *any method of programmatic access over the underlying technologies of the Web* (Stockinger, *et al.*, 2008).

A Web service can be seen as "a published interface to a type of data or computation" (Stein, 2002) that developers can employ to access data stored in a remote location, or invoke a programmatic analysis over the web.

The applicability of Web services in the field of bioinformatics is clear: they could be used for the development of client applications able, not only to query and collect data from many sources, but also to invoke bioinformatics pipelines to process those pieces of data. This would be achieved without any local software installation or running heavy-load processes.

From the data providers point-of-view Web services could serve as the means of disseminating information to third party applications; complementing this way the serving of data to researchers via human-readable web pages.

In the context of this thesis we would like to employ Web services technologies to simplify the web-based access to an SRS data integration system and thus facilitate the collection of pertinent knowledge over the web.

A number of alternative approaches in providing and using Web services are emerging (technology review based on Stockinger, *et al.*, 2008):

[1] http://www.w3.org/2002/ws/

1.6.3.1. W3C Web Services (SOAP-based web services)

W3C Web Services[1] are based on three W3C XML Schema (eXtensible Markup Language Schema) that attempt to provide a comprehensive computer-readable description of the entire process of discovering a service, identifying its interface and functionality, and consuming its data. SOAP (originally called Simple Object Access Protocol)[1] acts as a messaging protocol, enabling the encoding and decoding of messages; WSDL (Web Service Description Language) [1] defines the public-facing interface to the web service and UDDI (Universal Description, Discovery and Integration) describes how services may be registered in directories so that potential users can find them. In contrast to SOAP and WSDL, UDDI is not widely accepted and therefore not commonly used (Stockinger, *et al.*, 2008).

1.6.3.2. REST Services

REST (Representational State Transfer) (Fielding and Taylor, 2002) is a high-level architectural term that is used to describe a more *laissez faire* approach to web-based interprocess communication. The approach here is to establish a set of principles that provide guidance on how to make best use of the web's existing technologies, rather than on defining additional protocols. In a RESTful service, the required operation and its parameters are encoded as standard HTTP (HyperText Transfer Protocol) GET or POST requests (i.e. in its simplest form, as part of the URL (Uniform Resource Locator)) and results are returned by the server in whatever format it likes (though 'best practice' suggests this should be an XML document for ease of parsing by the client).

Simply put, REST is based on a basic HTTP request–response exchange pattern where requests are expressed via pure HTTP commands without any additional protocol or standard on top.

The main advantage of the RESTful approach is that it requires very little language support beyond the ability to generate or decode a HTTP stream.

[1] http://www.w3.org/2002/ws/

In bioinformatics, the Distributed Annotation System (DAS) (Dowell, *et al.*, 2001) can be seen as one successful example of using REST principles. DAS is used for exchanging biological sequence and annotation data. DAS specifies URL templates for retrieving sequence-based resources and a set of XML schemas for the data itself. DAS separates out the underlying sequence data (‘reference objects’, which can be proteins, DNA, or more recently, structures) from annotations on that data. Service providers therefore do not need to serve the sequence data itself but can add annotations to existing sequences. Finally, DAS provides a registry which (among other things) describes the service provider, the DAS commands that the server understands, and the type of data it provides (Prlic, *et al.*, 2007).

2. Methods

Reflect, OnTheFly, EasySRS and UniprotProfiler are all pieces of software aiming, altogether, to simplify the access of life science researchers to biological database pieces of information.

From a software architecture point of view they drift away from the monolithic application paradigm and employ, or constitute, components deployed over the web. Both Reflect and OnTheFly are multi-layered web applications, while EasySRS services are a server side component and UniprotProfiler one of their clients.

2.1. Reflect

2.1.1. Architecture Overview

The core component of Reflect is a consolidated dictionary that links names and synonyms to source data entries. Two types of user interfaces were created: a standard web page and plug-ins for Firefox and Internet Explorer. The user interfaces were constructed using HTML, JavaScript, XML-based User Interface Language[1] (XUL), and Document Object Model[2] (DOM) events (Pafilis *et al*, in preparation).

Communication between the user interfaces and the server is via XMLHttpRequest[3] objects. HTML documents are sent to the server, which uses regular expressions to distinguish text from HTML markup. The text of each document is first parsed for organism names, then parsed again for protein names associated with the organisms found in the first parse, or with a default organism set by the user. Recognized gene, protein, or small molecule names are substituted with JavaScript calls to create a summary popup via overLib[4] (Figure M-4). Substitutions occur only at the free text portion of an HTML page. The text of existing hyperlinks is excluded from the annotation. The document is returned to the browser with previous HTML tags and their

[1] http://www.mozilla.org/projects/xul/xul.html
[2] http://www.w3.org/TR/DOM-Level-2-Events/
[3] http://www.w3.org/TR/XMLHttpRequest/
[4] http://www.bosrup.com/web/overlib/

attributes unaffected, hence preserving the original document format. (Pafilis *et al*, in preparation).

The basic components of Reflect are explained in greater detail in the following paragraphs, along with the security setup, necessary for releasing Reflect to the public.

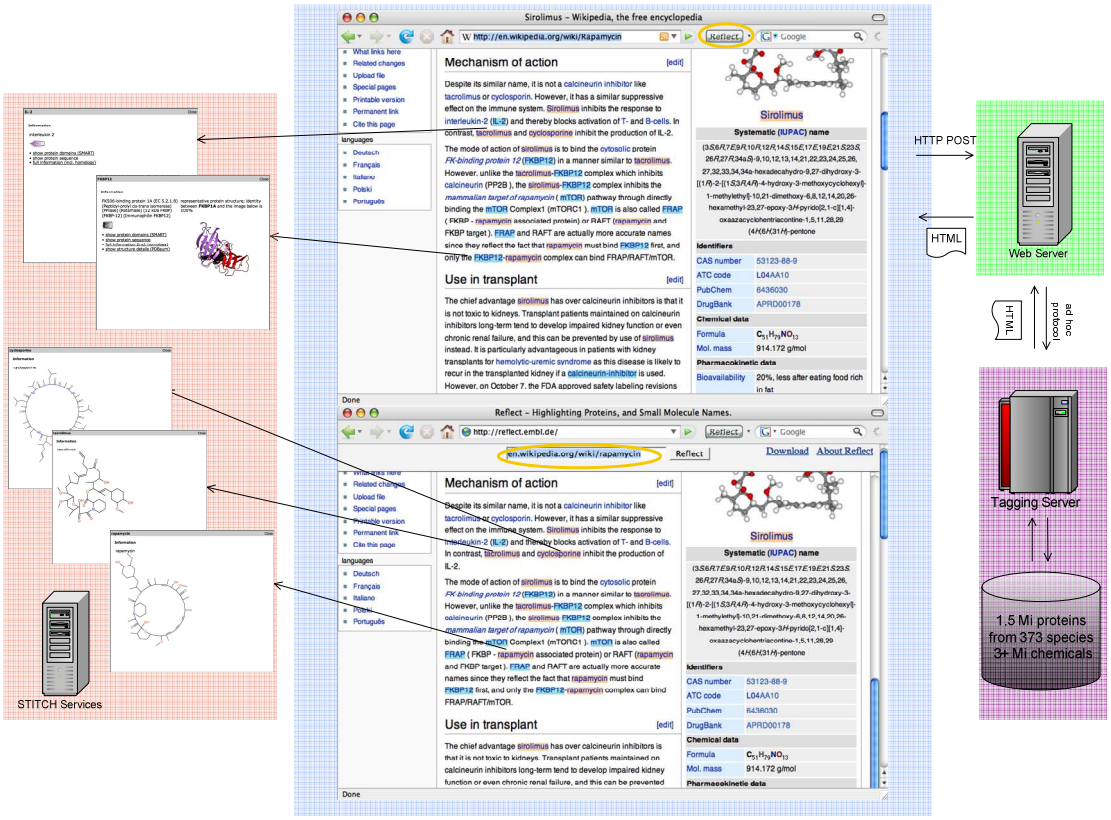


Figure M-1: Reflect's architecture. The reflected version of the Wikipedia rapamycin entry (blue box); the user may annotate this page either via using the browser extension (blue box, top) or via typing in the URL in the Reflect web page (blue box, bottom). Protein names are highlighted in blue, while chemical names in orange. In green background: the web server, and in purple: the tagging server maintaining the Reflect consolidated dictionary in memory. Shown in the orange box are the summary popups served by STITCH upon clicking on an annotated entity name; protein popups in the upper part and chemical popups in the lower part.

2.1.2. The Web Server

The Reflect web server (Figure M-1: green box), an Apache Tomcat[1] instance, acts as the gateway to the back-end tagging server. When invoked from the browser extensions it forwards the page contents directly to the back-end tagging server. When invoked from the web site it additionally fetches the user-specified web site contents. After the annotation finishes, the web server is responsible for delivering the updated page contents to the user. An ad-hoc protocol has been developed for the communication between the web server and the tagging server.

2.1.3. The Tagging Server

The tagging server (Figure M-1: purple box), originates from the Perl module used in populating the STRING (von Mering, *et al.*, 2007) and the STITCH (Kuhn, *et al.*, 2008) databases with literature-derived associations. For Reflect the tagger was adapted to handle HTML documents and converted to a daemon. Based on regular expressions, the tagging server annotates only the free-text portions of an HTML page and leaves clauses, such as links, scripts and pre-formatted pieces of text, unaffected. Figure M-2 describes the pre-processing steps that defines which parts of the text should be annotated or not.

The named entity recognition is dictionary based. This dictionary is the core component of Reflect that links names and synonyms to source data entries. It currently contains over 1.5 million proteins from 373 organisms and over 3 million small molecules (Figure M-3). The entire 13 GB dictionary is kept in RAM to enable fast tagging.

[1] http://tomcat.apache.org/

```
    # Flags the parts of an HTML page that should not be tagged.
    ######
    ## flags any opening/closing tag
    $incomingMessage =~
s/(<\/?\w+((\s+\w+(\s*=\s*(?:".*?"|'.*?'|[^\'\">\s]+))?)+\s*|\s*)\/?>
\s*)/ _CC_START_ $1 _CC_END_ /gis;

    ## flags any <script> contents
    $incomingMessage =~ s/ _CC_START_ (<script[ >].*? <\/script
*?>\s*) _CC_END_ /my $s = $1; $s =~ s# _CC_(START|END)_ ##g; "
_CC_START_ ".$s." _CC_END_ "/egis;

    ## flags any <a> contents
    $incomingMessage =~ s/ _CC_START_ (<a [^>]*?href.*? <\/a *?>\s*)
_CC_END_ /my $s = $1; $s =~ s# _CC_(START|END)_ ##g; " _CC_START_
".$s." _CC_END_ "/egis;

    ## flags any <pre> contents
    $incomingMessage =~ s/ _CC_START_ (<pre[ >].*? <\/pre *?>\s*)
_CC_END_ /my $s = $1; $s =~ s# _CC_(START|END)_ ##g; " _CC_START_
".$s." _CC_END_ "/egis;

    ## flags any <head> contents
    $incomingMessage =~ s/ _CC_START_ (<head[ >].*? <\/head *?>\s*)
_CC_END_ /my $s = $1; $s =~ s# _CC_(START|END)_ ##g; " _CC_START_
".$s." _CC_END_ "/egis;
```

Figure M-2: Perl code used to discriminate the free text of an HTML page that should be annotated by Reflect. This algorithm works by adding a flag at the beginning and at the end of a piece of an HTML code that should be tagged. The order of steps is significant and should not be changed. Initially the “protection” flags are added around all of the HTML tags; subsequently any protection tags found within the content of (a). a <script> (b). an anchor (<a>) tag, (c). a preformatted (<pre>) and (d). a <head> tag are removed. This way the name matching machinery can sequentially read the HTML page contents and understand if the current word it is processing is located within a piece of text that should be tagged or not.

```
9606    ENSP00000233946 IL1RT1  Ensembl_Uniprot/SWISSPROT_GN
9606    ENSP00000226730 IL2     Ensembl_Uniprot/SWISSPROT_GN HUGO
9606    ENSP00000226730 IL2_HUMAN       Uniprot/SWISSPROT
9606    ENSP00000296870 IL3     Ensembl_Uniprot/SWISSPROT_GN HUGO
9606    ENSP00000349365 IL30    Ensembl_HUGO_Previous_Symbols
-1      CID000000401    cycloserin      DTP/NCI_xPharm
-1      CID000002909    cyclosporin     xPharm
-1      CID005311131    cypermethrin    xPharm
-1      CID005748293    cyprodime       ChemIDplus_xPharm
```

Figure M-3: Sample of the Reflect dictionary; a tab separated file comprising: the species NCBI taxonomy id for proteins, or a concept id for other entity types (e.g. “-1“ for chemicals), the primary key of the biochemical entity in its representative database. ENSEMBL proteins identifiers are being used for human proteins, PubChem identifiers for chemicals, an entity alias, and the resource from which the alias was retrieved, e.g. UniProt, HUGO, ChemIDplus and xPharm.

The orthographic variant generation process contributes significantly to such a memory requirement. Upon start-up the initial entity names are processed and the dictionary is expanded so as to include name variants simulating the different ways in which the same name may be written. For example, the in-memory dictionary should not only include a name such as IL2 but also the IL-2 and IL 2 variants, where a hyphen or a space is added between the letters and the number. These perturbations are required to increase the name recognition ability of the system.

Additionally a dictionary of organism names is being used for the prediction of the species to which a piece of text is referring to.

Finally, a list of aliases contains names that should be ignored. These include words likely to increase the number of false positives annotations, such as very common English words.

2.1.4. Informative Summaries

The last Reflect component, the informative summary for an annotated element (Figure M-1: orange box), is served dynamically via an HTTP GET request to the STITCH Summary Service[1]. The mechanism is simple: using JavaScript (Figure M-4) the returned HTML is placed inside an inline frame (iframe) and displayed in an overLib pop-up. The STITCH Summary Service has been implemented by Michael Kuhn (EMBL).

```
<a lang="JavaScript"
   onclick="overlib(protein_option_text(
                    '9606.ENSP00000226730','','',50),
                    STICKY,CAPTION,'IL2',
                    DRAGCAP,CLOSECLICK,HAUTO,OFFSETY,0);"
   onmouseout="nd();">
   <font style="background-color: rgb(153, 221, 221);">IL-2</font>
</a>
```

Figure M-4: The JavaScript code that replaces the name IL-2. Red font: the overLib parameters defining the popup position and behaviour. Green font: two JavaScript methods invocations: protein_option_text() that generates the iframe call and overlib() the function to display the popup. Blue font the composite primary of the human protein corresponding to IL2. The key comprises the NCBI species taxonomy identifier, followed by the organism selected reference database, in this example ENSEMBL[2] for the Homo sapiens species.

[1] http://string-stitch.blogspot.com/2008/03/embedding-proteinchemical-information.html
[2] http://www.ensembl.org

2.1.5. Security Concerns

The Reflect web site allows users to enter the address of a web page to be annotated. This implicitly requires the Reflect server to fetch of the page that will then be processed.

Providing users with such an option opens up the possibility of the Reflect server being misused as a proxy in attacking public servers, accessing sites with malicious content, and/or accessing EMBL internal documents or subscribed material (Figure M-5). To prevent such cases a series of technologies have been used to secure the Reflect server.

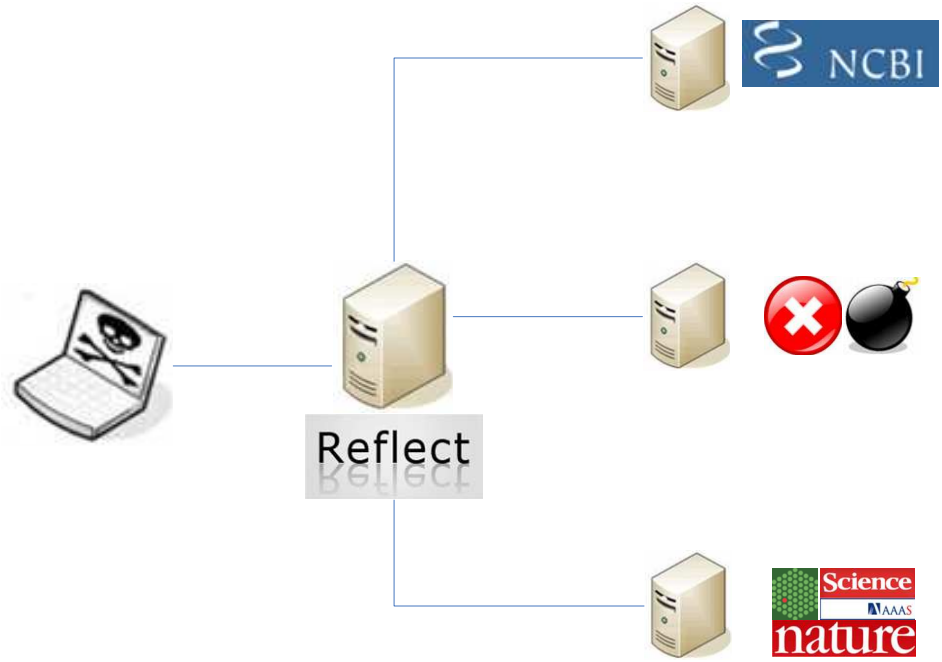


Figure M-5: Possible threats for http://reflect.ws. A malicious user may exploit Reflect's web page fetching functionality and attempt to attack public sites such as NCBI, to access illicit material, and/or to access EMBL internal documents or subscribed material.

First, a URL content filtering service[1], similar to those used in schools to protect students, has been employed to restrict the web pages that the Reflect server may download. The filtering service has been provided by Websense Inc[2].

[1] http://en.wikipedia.org/wiki/Content_filtering
[2] http://www.websense.com/global/en/

Second, certain web resources commonly used by life science researchers, such as PubMed, impose restrictions on how frequently they can be accessed from a certain internet location. Their reasoning for doing so is to prevent abuse of their contents. The risk in this case is that the Reflect web site could be employed by third party software either to circumvent the frequency restrictions, or even to attack these resources. Fail2Ban[1] and Apache DOS-evasive[2] are pieces of software that monitor the web server usage and prevent such cases by blocking the connection with the malicious software.

Last, specific network setup was required to prevent the Reflect web page from accessing internal EMBL web pages and subscribed material. The setup was performed from the EMBL support group in collaboration with the EMBL library.

[1] http://www.fail2ban.org
[2] http://www.zdziarski.com/projects/mod_evasive/

2.2. OnTheFly

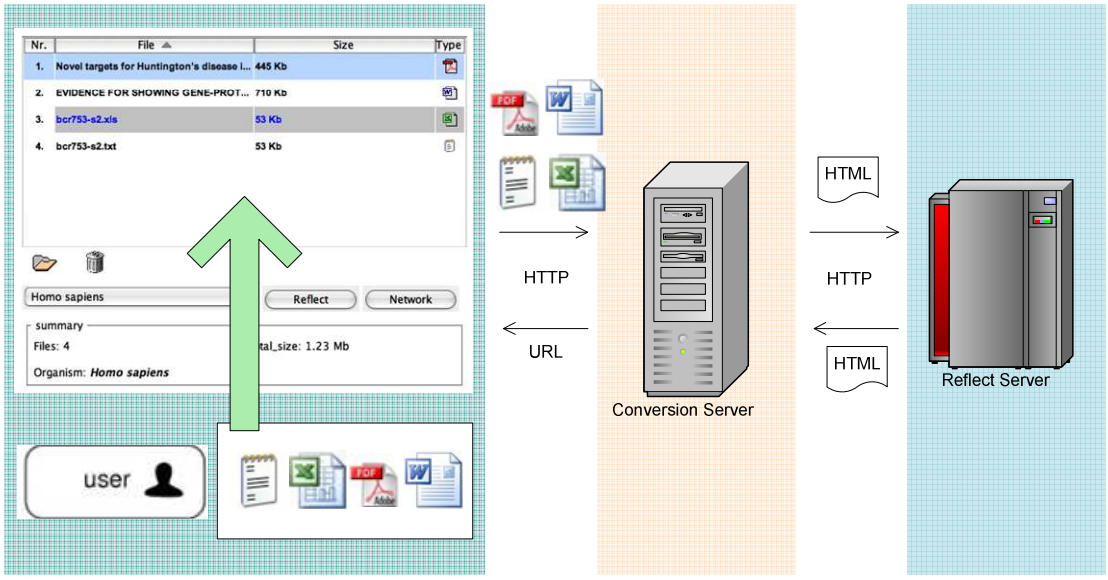


Figure M-6: OnTheFly's architecture and functionality. A user can drag and drop a file in the OnTheFly applet (green box). By clicking on the "Reflect" button (green box), the selected files will be sent first to the conversion server (orange box) that will convert them into HTML pages, which will then be sent for tagging to the Reflect server (blue box). The reflected HTML will be stored for a very short period of time on the conversion server and a URL pointing to it will be returned to the user's browser. The organism selection drop-down list (green box) enables users to define a species protein dictionary to be used by default. Finally, users may also click on the "Network" button (green box) to extract the STITCH derived networks of associations of the document entities.

OnTheFly comprises a client – server architecture (Figure M-6). The front end is an Applet written in Java 1.5. It can be accessed directly either from its web page or its Java desktop application. The Applet technology was chosen as it enriches web pages with file drag-and-drop functionality and thus maximizes ease of use.

The server side components comprise a set of document converters along with the modules required to invoke Reflect.

OnTheFly currently supports Microsoft Word, Excel, Adobe PDF and raw text formats. The converters employed are: Verypdf[1] that transforms PDF to HTML documents and Ultra-PPT-To-HTML-Converter[2] which converts Microsoft Office documents into HTML pages (employing the machinery of the former). The document converters were selected based on their ability to maintain the format of the original document. An *ad-hoc* Java module has been implemented for the conversion of plain text to HTML. Any data exchange was based on the HTTP protocol, which on the server side is facilitated by an Apache Tomcat.

Evangelos Pafilis was responsible for the components that invoke Reflect and for the arrangement of the overall architecture communication.

[1] http://www.verypdf.com
[2] http://www.ultrashareware.com/Ultra-PPT-To-HTML-Converter.htm

2.3. Easy SRS Services and UniprotProfiler

EasySRS is a Java 5 server side application that employs the Java API for XML Web Services[1] (JAX-WS) framework and an Apache Tomcat server to SOAP messages over HTTP. Underlying EasySRS exist SRS WSObjects™ [2] (Web Services objects), a set of client-side components that interact with a remote SRS server, also via web services, to expose that server's functionality.

Despite the complexity of the WSObjects data types, EasySRS has been *ab initio* developed as a set of simple operations returning easy-to-handle, data types such as list of strings. A Web Service Definition Language (WSDL) detailed description of the operations is available at http://srs.embl.de/easy.

The choice of implementation technologies and the method design aimed at simplifying the development process for a variety of programming languages. For this reason the EMBRACE Network of Excellence[3] (European Model for Bioinformatics Research and Community Education) Web services development guideline[4] has been followed. EMBRACE is a research community that aims to support the interoperability ("cross-talk") between bioinformatics tools and services. Their recommendations derive from the Web Services Interoperability Organization[5] (WS-I) guidelines which have a similar aim for tools and services not restricted in scope.

UniprotProfiler is a Java 1.5 desktop application that employs ("consumes") the EasySRS services to perform UniProt related queries. Communication to the EasySRS services is also based on the JAX-WS libraries. UniprotProfiler uses the Arena3D (Pavlopoulos *et al.*, (A) in preparation) libraries to generate a 3D visualisation of the query results (Figure R-10).

[1] https://jax-ws.dev.java.net
[2] http://www.biowisdom.com/content/srs-wsobjects
[3] http://www.embracegrid.info
[4] http://www.embracegrid.info/page.php?page=tech_documents
[5] http://www.ws-i.org

3. Results

To assist researchers in the collection of relevant knowledge we provided users with the automated annotation of biological entities mentioned in a piece of text, and with web-services-based access to centralised biological resources. The former has been used to deliver summaries about the recognised entities in easy and intuitive ways, while the latter has been employed to enrich the entries of the Plant Defence Mechanism database and to generate Systems Biology knowledge graphs of associations.

3.1. Document Annotation: Automated Enrichment of Biochemical Entity Names

Life science researchers use the web even on a daily basis to survey the literature and to collect pieces of information pertinent to certain biological entities (Divoli, *et al.*, 2008).

A typical iteration of this manual process could be summarized as follows; i, search the literature for related articles, ii, read through the text, ii, detect bioentities of interest, and iii, query in one or more databases. This task is rather tedious and time consuming task on its own and tends to become Sisyphean as the number of entities increases e.g. by following up on an article's references (Pavlopoulos *et al.*, (B) in preparation).

Text mining tools aim at assisting with this task and have reached reasonable performance. However, several questions have been raised: "Do these tools work? Are they usable, and if so, who uses them, and how? Are they cost-effective? Can they be adapted to new domains and maintained over time?" (Kevin Bretonnel Cohen, Pacific Symposium for Bioinformatics 2008, call for papers).

Currently a trend in bioinformatics is to offer user-friendly tools in an intuitive and integrated environment that exploits familiar interaction metaphors. This aims to protect users from the technological complexities, hiding them behind these metaphors (drag-and-drop, cut-and-paste, etc), without trivialising the problems in question and limiting the kind of functionality available. Ultimately the goal is to provide interfaces that “just work”, so that users do not spend unnecessary effort with auxiliary tools but concentrate on their research (Stockinger, *et al.*, 2008).

The goal of the work presented here is to develop simple tools that can employ text mining and data annotation techniques to automatically enrich documents life scientists use throughout their work.

3.2. Reflect: Automated Annotation of Biochemical Entities in Web Pages

3.2.1. Overview

Reflect is a system that facilitates the annotation of biochemical entities in web pages. It not only recognises and highlights gene, protein and small molecule names, but also maps them to their corresponding identifiers in structured databases, and attaches to them summaries of related knowledge.

The fact that Reflect is able to process web pages practically means it can annotate an unlimited number of documents. The functionality of Reflect, i.e. recognition of the entity name, annotation of the text and the summaries it attaches, is demonstrated for a PubMed abstract (Figure R-1) and the full text version (Figure R-2) of a recent article on Huntigton's disease, along with a press release (Figure R-3) communicating the article's findings.

Purpose of the experiments described in this article was to study the effect of Food and Drug Administration (FDA) approved drugs in the process of autophagy, a mechanism that allows cells to degrade aggregate-prone proteins and, hence, affect the impact of diseases such as Huntington's chorea (Williams, *et al.*, 2008).

Reflect was able to recognise and annotate not only such FDA drug names but also the proteins mentioned as members of the novel autophagy-regulating pathways proposed.

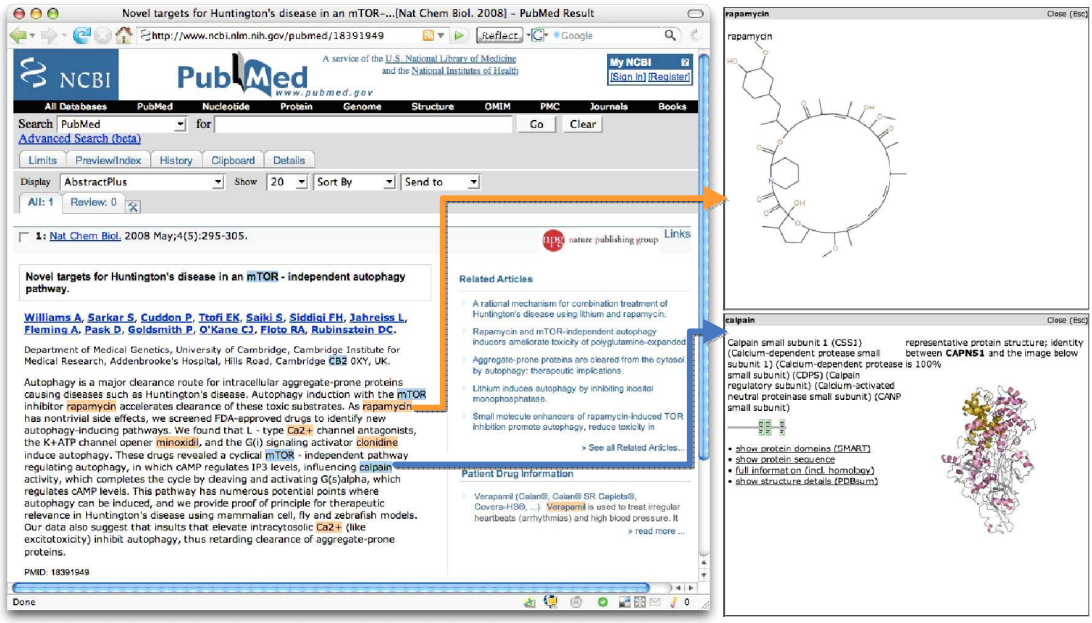


Figure R-1: Result of using Reflect on the web page of a PubMed abstract on Huntington's disease. All recognized genes and proteins are highlighted in blue, small molecules in orange. Clicking on highlighted terms opens a hyperlinked summary popup (Pafilis *et al.,* in preparation).

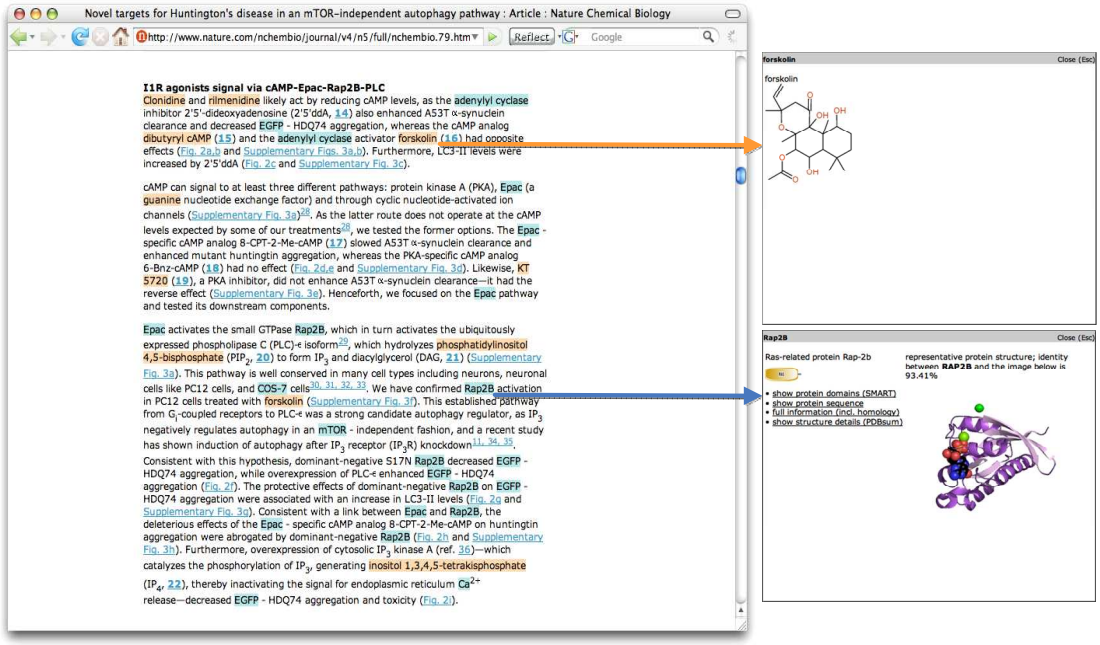


Figure R-2: An example of Reflect being used to annotate full text articles. The full text HTML version of the abstract annotated in Figure R-1 is shown (left side). By clicking on the recognised protein and chemical entity names, a user can easily retrieve all the pieces of information supported by the Reflect system (right side).

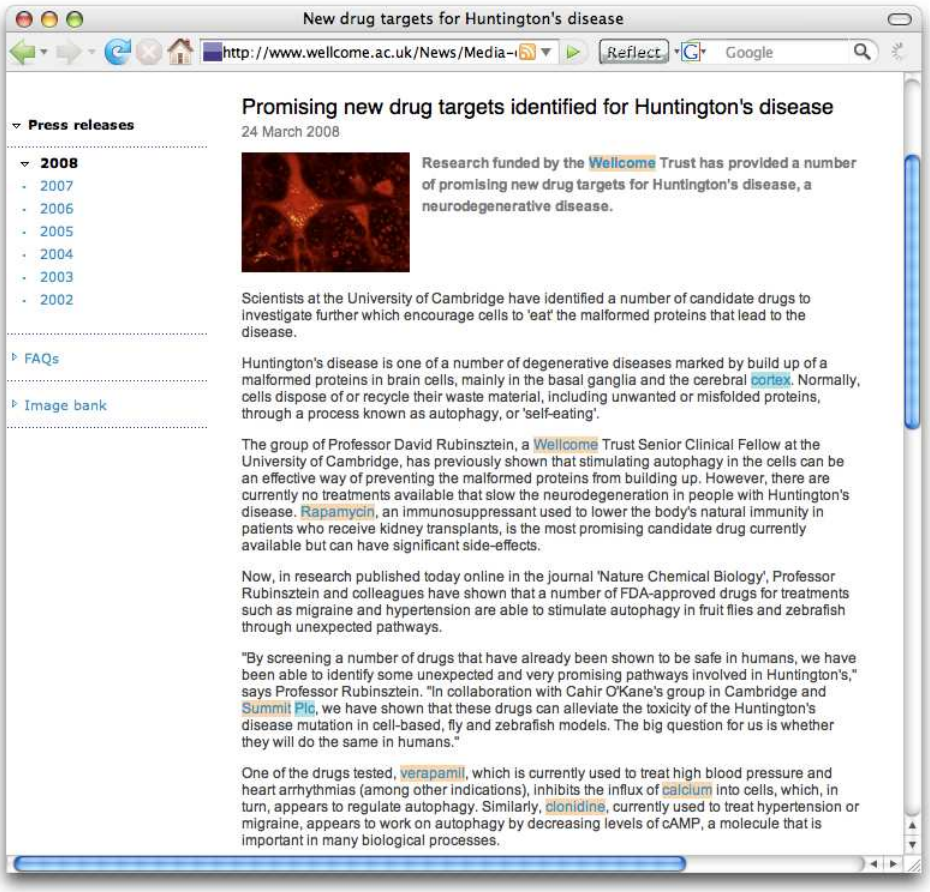


Figure R-3: the annotated version of a Wellcome Trust press release[1] communicating the discoveries described in the article shown in Figures R-1 and R-2.

3.2.2. Functionality

While annotated pages retain their original format and functionality, gene and protein names are highlighted in blue and small chemical molecule names in orange.

Clicking on any of the annotated terms displays a summary pop-up. For each protein, the informative summary provides users with a short description, the domain composition, if possible a representative Protein Data Bank (PDB) structure, and links for further navigation to related resources, such as the sequence. For a chemical, the summary consists of the structure, linked to PubChem[2] (Figures R-1, R-2 and R-3).

[1] http://www.wellcome.ac.uk/News/Media-office/Press-releases/2008/WTD039292.htm
[2] http://pubchem.ncbi.nlm.nih.gov/

The domain composition is derived from the Simple Modular Architecture Research Tool (SMART) database (Letunic, *et al.*, 2006), the 3D structure from the PDBsum (Laskowski, 2001) resource, the sequence and the chemical structure from the STITCH database.

To improve users' experience the pop-up window can be dragged around the screen and closes as soon as the user clicks outside of it.

3.2.2.1. Reflection Errors

Figure R-3 shows that erroneous annotations do occur. "Wellcome", "cortex", "Summit" and "Plc" are false positives. Since the page annotation is dictionary based, the fact that everyday English words are being used to identify biological entities (Pearson, 2001) this type of errors are expected to occur. Moreover similar errors may occur when Reflect is being used to annotate documents from a different context than life sciences.

3.2.3. Performance

The processing time for a typical five page article is about half a second. However, the total annotation time depends on how quickly the text can be communicated to the server. Overall, it takes up to a second (transfer and processing time) for a typical five page article over a regular broadband connection. This allows Reflect to be a practical, interactive companion, for researchers.

3.2.4. Interacting with Reflect

Reflect has been developed to operate in a seamless and transparent fashion making web page annotation as easy as possible. To materialise this aim users are provided with two types of interfaces: a browser extension and a web page. The browsers currently supported are Firefox and Internet Explorer.

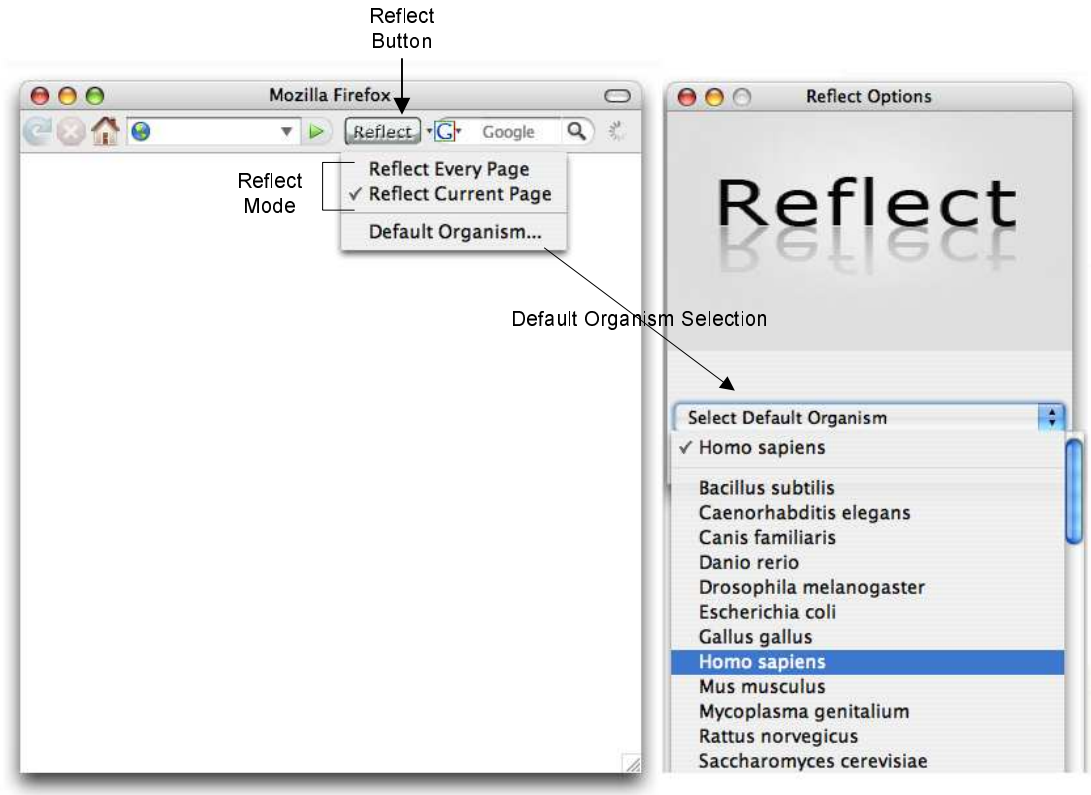


Figure R-4: The Reflect Firefox extension interface and preferences menu. To annotate a page all a user has to do is to click on the Reflect button, when Reflect is set to the "Reflect Current Page". If Reflect has been set to "Reflect Every Page" every new page will be annotated upon loading. To improve name recognition Reflect allows users to set the organism whose proteome dictionary will be used by default (left side).

Installing the browser extension is a simple clicking procedure. Using it is equally easy (Figure R-4). The Reflect extension supports two modes of operation: a user can either annotate the current page by clicking the Reflect button, or have every new web page the users visits reflected (Firefox only). In the latter case, the Reflect button behaves as a switch that enables and disables the tagging process (Figure R-4).

As an alternative, users not familiar with, or without the privileges to install browser extensions, can access Reflect via its web site: http://reflect.ws (Figure R-5). All a user has to do is to copy and paste a URL into the input field. The target web page will be retrieved, tagged and returned to the user. http://reflect.ws supports most modern browsers and offers recursive tagging of the links in the reflected pages.

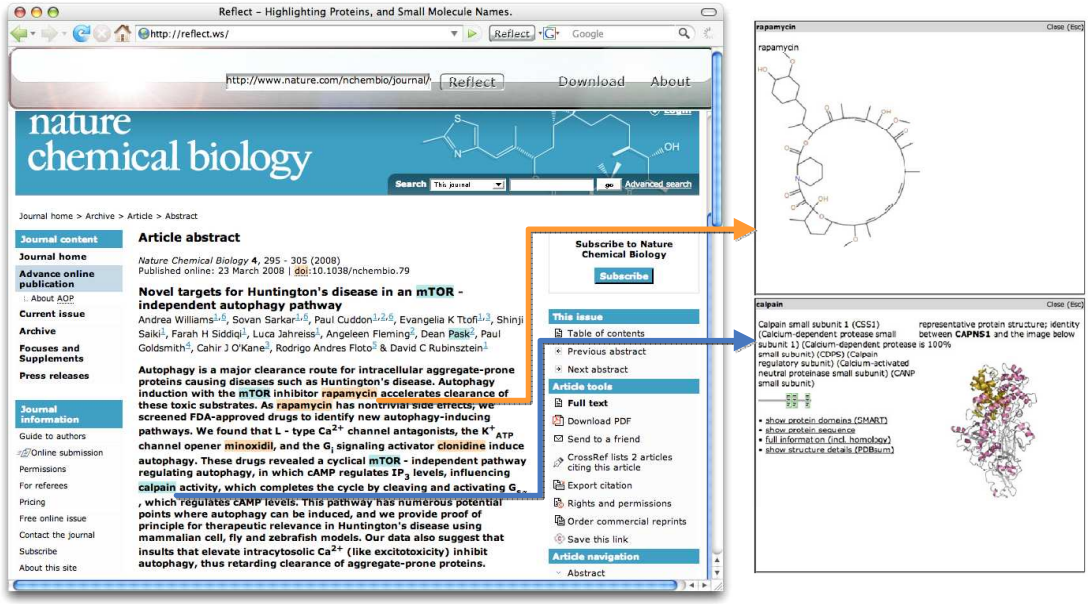


Figure R-5: The abstract of Williams, *et al.*, 2008 annotated using the Reflect web application. Reflection occurred after the user copied and pasted the URL of the article abstract of the Nature Chemical Biology web site. While the abstract can be reflected by the web application, this is not allowed for the full text. http://reflect.ws can process publicly accessible web pages only.

3.2.5. Automated Organism Recognition and Disambiguation

To perform NER in a given piece of text it is important to use the dictionary of the organism(s) the text refers to. The text tagger assigns organisms to the pieces of text based on the existence of species names and/or representative keywords such as yeast, mouse and human.

Reflect not only inherits this organism recognition feature, but also extends it based on user interaction. The Firefox extension allows users to select a species to be used by default, and include its proteome dictionary in the recognition process (Figure R-4).

To resolve ambiguous protein names, i.e. names shared among species, in the current version, Reflect, will map the entity name to the protein belonging to the default organism.

This default organism selection feature is not included in the Reflect web page; *Homo sapiens* is used as the default species instead.

3.3. **OnTheFly: Automated Annotation of Microsoft Office, PDF and Plain Text Documents**

3.3.1.1. Researchers Use More Documents Than HTML Pages

Reflect is capable of handling HTML documents. This renders possible the processing of an unlimited number of documents. Yet, many times researchers survey the literature in Portable Document Format (PDF), rather than HTML, files, use text processors, such as Microsoft Word to author their documents, and importantly: use plain text or spreadsheets to store their experimental results.

Some of these types of documents can be processed by Reflect with an intermediate step; that of saving the file as an HTML page before viewing it with the web browser.

Such a process, however, is rather tedious, drifts apart from the “simple tool” philosophy, and does not apply to PDF documents (unless extra effort and money is spent on specialised software).

OnTheFly (Pavlopoulos *et al.*, (B) in preparation) couples the Reflect machinery with automated document conversion and extends the document annotation to PDF, Microsoft Excel, Microsoft Word and plain text files.

3.3.1.2. Interacting with OnTheFly

OnTheFly is freely available at: http://onthefly.embl.de. It can be accessed either directly from this web page, or via the client desktop-application. In both cases, a user drags and drops a document file to the application (Figure R-6). The text is subsequently extracted and converted to an HTML page, maintaining its format. This temporary HTML page is subsequently processed by Reflect. The results are intuitively presented as a richly annotated and interactive HTML version of the original article. All this is achieved with a simple drag-and-drop motion (Pavlopoulos *et al.*, (B) in preparation).

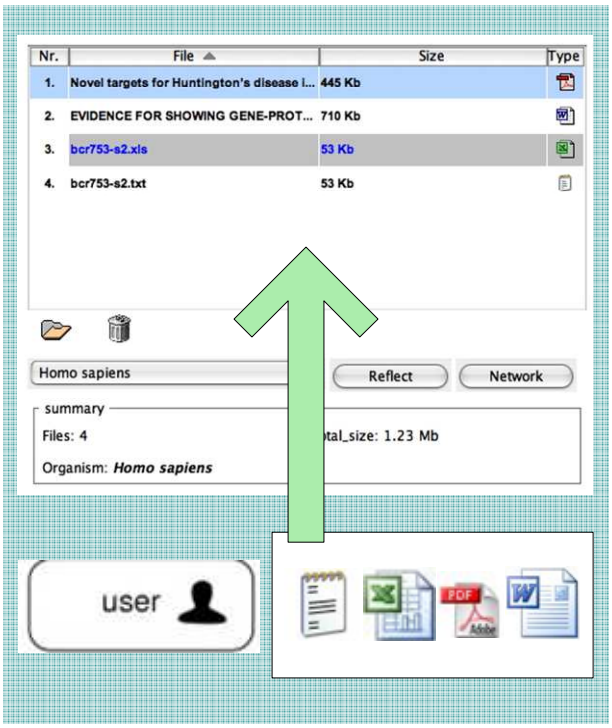


Figure R-6: The OnTheFly interface. File loading can occur via a simple drug and drop gesture. Clicking on “Reflect” will return an annotated HTML representation of the selected documents. Clicking on “Network” will generate a graph of known and predicted associations for the biochemical entities contained in the selected documents (according to the STITCH database). Similarly to the Reflect Firefox extension, users may select the annotation default organism from the drop down menu on the middle left.

3.3.1.3. Interaction Network Extraction

OnTheFly does not only inherit the functionality of Reflect (Pafilis *et al*, in preparation), ie named entity recognition and attachment of related knowledge summaries. It can also offer a network representation of the biochemical entities mentioned in a document. In such a network, biochemical entities are represented as nodes connected by edges based on their predicted interactions in STITCH (Kuhn, *et al*., 2008). The network may well include additional biochemical entities based on genomic context, experimental result and other types of evidence. Network generation is not restricted to a single document, it can also be applied to a collection of documents (Pavlopoulos *et al*., (B) in preparation). Such a network constitutes an important starting point for discovery; it enables information from the literature, the biological databases and in-house data to be brought together.

The network generation machinery is borrowed from the STITCH database and has been developed by Michael Kuhn.

3.3.2. Biological Applications

3.3.2.1. Annotating a Full Text Article as PDF

To demonstrate the functionality of OnTheFly a full text article on protein-protein interaction prediction (Pitre, *et al*., 2006), stored locally as a PDF file, has been annotated. Figure R-7 displays the annotated HTML version of the article along with part of the network of associations of the biochemical entities contained in it. For the annotation *Saccharomyces cerevisiae* was used as the default organism. The overall annotation time was between 15 to 20 seconds.

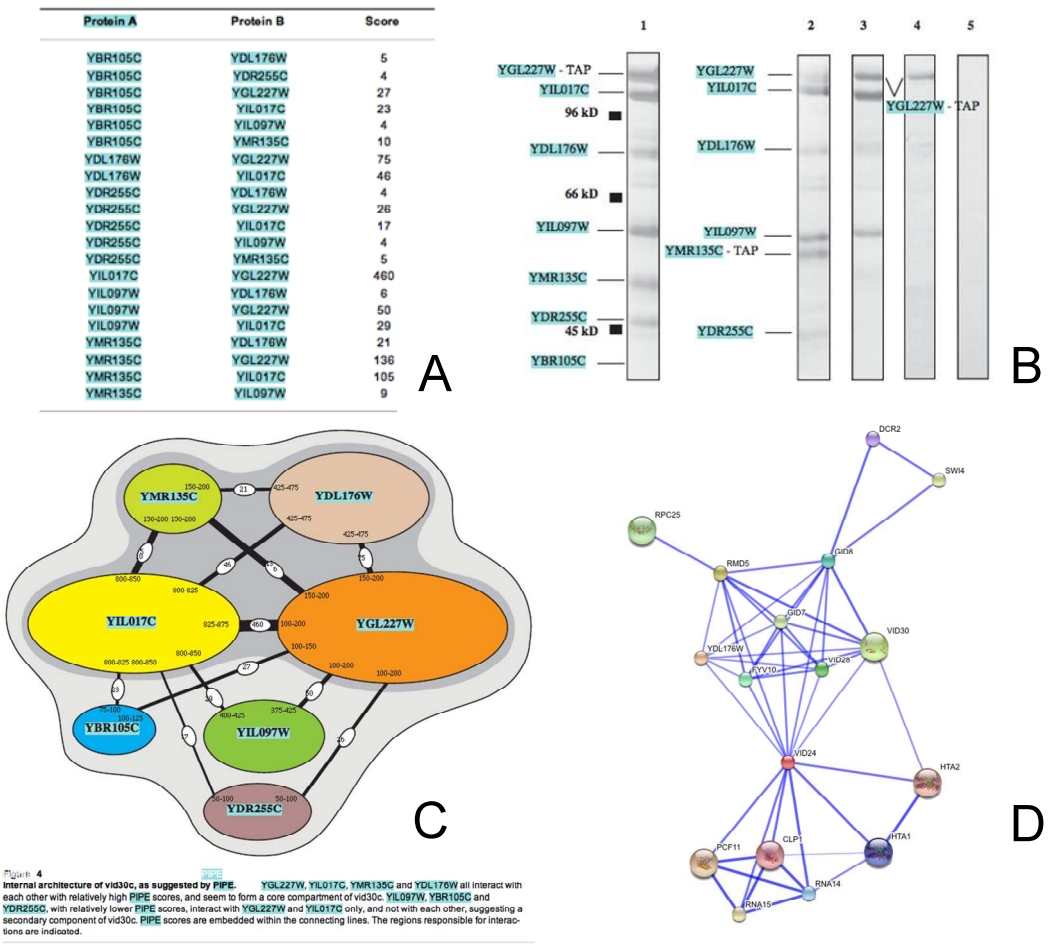
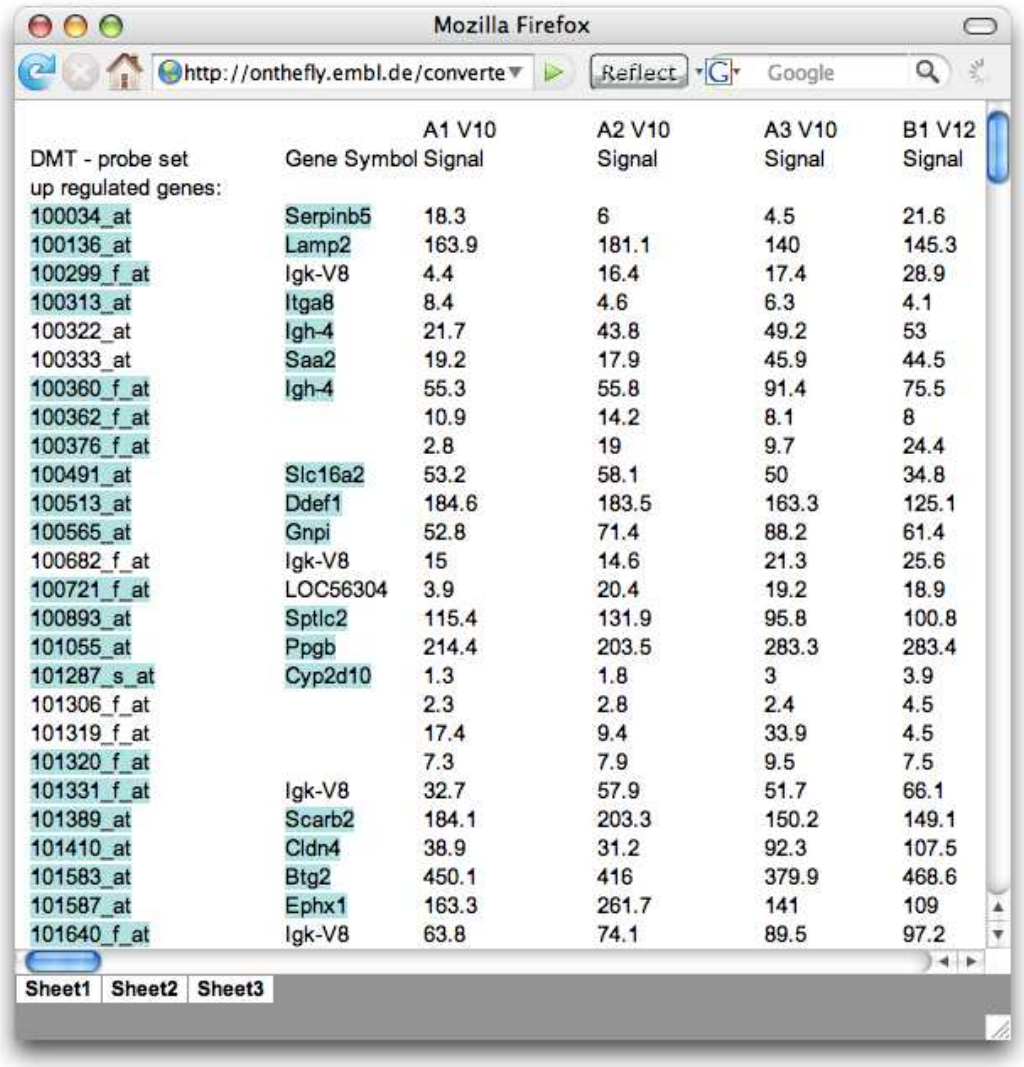


Figure R-7: Parts of the annotated version of the PDF full text article (Pitre, *et al.*, 2006): A result table (A), an experimental result figure (B), a result figure (C). Shown in D is the network of associated entities, according to the STITCH database, for the proteins shown in part C. The systematic way in assigning names[1] to the *Saccharomyces cerevisiae* genes contributes in the high number of recognised entities.

[1] http://www.yeastgenome.org/help/yeastGeneNomenclature.shtml

3.3.2.2. Annotating Experimental Results

Online articles increasingly tend to present the bulk of their data as supplementary material. These files are often in PDF, spreadsheet or raw text formats, and follow no conventions as to the structuring of their content. OnTheFly is therefore ideal for parsing large tabular data such as supplementary tables (Pavlopoulos e*t al.*, (B) in preparation). OnTheFly uses Reflect's named entity recognition which is a dictionary based one. Thus it inherits the advantage of being able to recognize identifiers (Jensen, *et al.*, 2006) and bridge them to other databases, greatly reducing the need for manual cross-referencing.



DMT - probe set up regulated genes:	Gene Symbol	A1 V10 Signal	A2 V10 Signal	A3 V10 Signal	B1 V12 Signal
100034_at	Serpinb5	18.3	6	4.5	21.6
100136_at	Lamp2	163.9	181.1	140	145.3
100299_f_at	Igk-V8	4.4	16.4	17.4	28.9
100313_at	Itga8	8.4	4.6	6.3	4.1
100322_at	Igh-4	21.7	43.8	49.2	53
100333_at	Saa2	19.2	17.9	45.9	44.5
100360_f_at	Igh-4	55.3	55.8	91.4	75.5
100362_f_at		10.9	14.2	8.1	8
100376_f_at		2.8	19	9.7	24.4
100491_at	Slc16a2	53.2	58.1	50	34.8
100513_at	Ddef1	184.6	183.5	163.3	125.1
100565_at	Gnpi	52.8	71.4	88.2	61.4
100682_f_at	Igk-V8	15	14.6	21.3	25.6
100721_f_at	LOC56304	3.9	20.4	19.2	18.9
100893_at	Sptlc2	115.4	131.9	95.8	100.8
101055_at	Ppgb	214.4	203.5	283.3	283.4
101287_s_at	Cyp2d10	1.3	1.8	3	3.9
101306_f_at		2.3	2.8	2.4	4.5
101319_f_at		17.4	9.4	33.9	4.5
101320_f_at		7.3	7.9	9.5	7.5
101331_f_at	Igk-V8	32.7	57.9	51.7	66.1
101389_at	Scarb2	184.1	203.3	150.2	149.1
101410_at	Cldn4	38.9	31.2	92.3	107.5
101583_at	Btg2	450.1	416	379.9	468.6
101587_at	Ephx1	163.3	261.7	141	109
101640_f_at	Igk-V8	63.8	74.1	89.5	97.2

Figure R-8: The annotated version of an Excel spreadsheet with experimental results from an oligonucleotide microarray experiment in mouse (Stein, *et al.*, 2004).

Figure R-8 shows the annotated version of an Excel spreadsheet with experimental results from an oligonucleotide microarray experiment studying the involution of the mouse mammary gland (Stein, *et al.*, 2004). OnTheFly can be used for the recognition of, in this case, Affymetrix[1] microarray probeset identifiers. For the annotation *Mus musculus* has been used as the default organism. Certain names and identifiers in Figure R-8 have not been recognised. In the Discussion of Reflect we elaborate on possible approaches to improve the NER and avoid such issues.

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

3.4. Simplifying Search and Retrieval Operations on Biological Resources

Answering questions that arise in biological research is no longer possible just by querying the contents of a single database. Bioinformatics database resources can answer questions on their own data, they cannot, however, provide information that belongs to another domain, stored in another database (Stein, 2003).

Data integration systems aim at providing researchers with access to all of the information they need in a consistent format (Lacroix and Critchlow, 2003).

SRS brings together the information from many sources covering several concepts of life sciences. This includes genetic, protein, expression, pathway, and clinical data[1]. Based on indexing technologies, SRS allows simultaneous queries to all of the integrated databases from a single access point, using a simple, yet powerful, query format.

As seen in the Introduction (section 1.4.3.1) several solutions exist to provide developers with programmatic access to an SRS server. However, the requirement for an easy-to-use, web-based method, free from any firewall restrictions still exists.

The work we present aims to provide exactly such a solution able to facilitate the collection of pertinent knowledge.

In the following paragraphs we describe the set of operations that can be performed, elaborate on how they can be combined to collect biologically relevant pieces of information and demonstrate how they have been used for the annotation of biological data.

[1] http://www.biowisdom.com/navigation/srs/srs

3.5. EasySRS: Simplified SRS Web Services

EasySRS comprises a set of simple operations that support multiple database search, retrieval and cross-referencing queries; from a unique access point, using simple query syntax.

3.5.1. Components

To provide developers with an intuitive interface EasySRS services have been separated in two components: Retriever and Informer.

3.5.1.1. Retriever

A web service that allows users to perform search, retrieval and cross-linking operations on an SRS data integration system. This includes the execution of queries that may span multiple databases, the partial or full retrieval of database records, and the serving of record-cross-references.

3.5.1.2. Informer

Provides researchers with information about an SRS biological data integration system. That includes the biological databases supported, the fields that can be used to query a database and the views of the records in database that can be retrieved.

3.5.2. Availability and Supported Databases

The Web Services Definition Language (WSDL) files describing the available operations can be found at: http://srs.embl.de/easy/.

The publicly available EasySRS instance is connected to the SRS server in EMBL Heidelberg (http://srs.embl.de). This SRS instance currently indexes the information of more than 100 life sciences databases. The latter cover, among others, nucleic and amino acid sequences, proteins structures, domains and families, and biological pathways.

Due to a strong interest in the overlap between biology and chemistry, databases such as the Human Metabolome Database[1] (HMDB), Drugbank[1], and PubChem[2] have also been integrated.

[1] http://www.hmdb.ca

3.5.3. Supported Operations

The EasySRS services provide users with web-based remote access to the SRS functionality. They facilitate not only search and retrieval operations, but also traversal of the “SRS universe” that can be generated via the SRS linking capabilities.

3.5.3.1. Search Queries

Queries in SRS, hence also in EasySRS, follow a simple pattern of selecting the database of interest and defining the field to be queried and the query term.

getMatchingEntries is the main search method for querying the SRS system. It queries the given "fields" in a user defined set of "databases" for the occurrence of the given "terms". The fields must exist in all of the databases being queried. Between fields and terms there is one-to-one correspondence. The field-term query pairs are combined by a boolean operator. Allowed operator values are "AND", "OR" and "BUTNOT". Wild characters, such as “*” (matches any character) and the boolean operator “|” (“OR”) in the given terms are allowed.

The following queries[3] demonstrate the range of the search operation available:

- retriever.getMatchingEntries("UNIPROT_SWISSPROT","id","PAX6_*", "AND");
  returns all the proteins of the PAX6 family in the SwissProt[4] database (all the “UNIPROT_SWISSPROT” records whose “id” matches the “PAX6_*” pattern).
- retriever.getMatchingEntries("UNIPROT_SWISSPROT","id","PAX6_*|PAX7_*","AND");
  returns all the proteins of the PAX6 and the PAX7 family in the SwissProt database (all the “UNIPROT_SWISSPROT” records whose “id” matches the “PAX6_*” or “PAX7_*” pattern).

[1] http://www.drugbank.ca
[2] http://pubchem.ncbi.nlm.nih.gov
[3] Based on examples from http://www.ebi.ac.uk/~srs/wiki/doku.php
[4] http://expasy.org/sprot/

- retriever.getMatchingEntries("UNIPROT_SWISSPROT","id,desc","PAX6_*|PAX7_*,fragment","BUTNOT");
  returns all the proteins of the PAX6 and the PAX7 family in the SwissProt database, which are not described as fragments (all the "UNIPROT_SWISSPROT" records whose "id" matches the "PAX6_*" or "PAX7_*" pattern and do not contain the term "fragment" in their description field ("desc").

More information on the SRS query language can be found at: http://www.ebi.ac.uk/~srs/wiki/doku.php?id=guides:srsquerylanguage.

3.5.3.2. Retrieval Queries

The data retrieval query capabilities range from the retrieval of a complete entry to the retrieval of certain parts of record, such as the sequence or the description of biological entry, to the retrieval of specific fields.

getSequence, *getEntryView* and *getSpecificFields* are methods that could be used for the retrieval of data indexed by the SRS system.

- retriever.getSequence("UNIPROT_SWISSPROT","acc","P47599");
  returns the sequence of the specified protein.
- retriever.getEntryView("UNIPROT_SWISSPROT","acc","P47599","DescriptionClass");
  returns a description of the specified protein.
- retriever.getEntryView("UNIPROT_SWISSPROT","acc","P47599","CompleteEntry");
  returns the complete protein record.
- retriever.getSpecificFields("UNIPROT_SWISSPROT","acc","P47599","Gene,Synonyms,SeqLength");
  returns the gene name, the synonyms and the length of the specified protein.

The *getEntryView* requires that an entry *view* parameter is also provided (eg. *DescriptionClass* and *CompleteEntry* in the second and third query). A *view* in SRS refers to a predefined set of record fields, useful for serving parts of the record for a specific purpose. e.g. to create an overview about certain proteins the *DescriptionClass* view would suffice.

3.5.3.3. Cross-linking Queries

getLinkedEntries is the method that allows users to follow the indexed database record cross-references. This method provides the means for navigating through the "universe" of linked life sciences databases.

- retriever. getLinkedEntries ("UNIPROT_SWISSPROT","acc","P47599"," PATHWAY")
- retriever. getLinkedEntries ("UNIPROT_SWISSPROT","acc","P47599"," INTERPRO")
- retriever. getLinkedEntries ("UNIPROT_SWISSPROT","acc","P47599"," KEGGGENES_NA")

These queries respectively return the KEGG[1] pathway, the domain composition (according to INTERPRO[2]), and the genes according to KEGG, related to the specified protein.

[1] Kyoto Encyclopedia of Genes and Genomes: http://www.kegg.com/
[2] http://www.ebi.ac.uk/interpro/

3.5.4. Biological Applications

3.5.4.1. Combining EasySRS Queries to Enrich Biological Data

Data become more valuable in the context of other data (Zdobnov, *et al.*, 2002). The previously mentioned methods can be combined so as to enrich biological data annotation. Table R-1 describes a series of queries that can be employed to collect:

- functional, subcellular localisation and biological process information (according to the Gene Ontology[1] (GO) annotation)
- pathway, and
- structural context information.

It is of particular importance that highly cross-linked data sets become a domain knowledge base (Zdobnov, *et al.*, 2002). This makes possible queries such as “give me all the proteins that participate in the same pathways as a given protein” (Table R-1), or “give me all the proteins and small chemical molecules that co-crystallize with my protein”.

[1] http://www.geneontology.org/

Operation	Input Parameters				Biological Aspect
Biological function, subcellular localization and biological process information					
getLinkedEntries	database: uniprot	field:acc	term: p42858	linkToDatabase:go	retrieves the function, the subcellular localization and the biological process associated with this protein
getEntryView	database: go	field:id	term:GO: 0003714	View:DescriptionClass	returns a short summary about the specified GO term
Pathway information					
getLinkedEntries	database: uniprot	field:acc	term: p42858	linkToDatabase :pathway	retrieves the pathways according to the KEGG database in which this protein is a member
getEntryView	database: pathway	field:id	term: hsa05040	view:DescriptionClass	returns a short summary about the specified KEGG pathway
getLinkedEntries	database: pathway	field:id	term: hsa05040	linkToDatabase: uniprot	returns the proteins which are members of this KEGG pathway
getLinkedEntries	database: uniprot	field:acc	term: p42858	linkToDatabase: panther	retrieves the pathways according to the PANTHER[1] database in which this protein is a member. (PANTHER database contains information only about human pathways)
getEntryView	database: panther	field:id	term: 14744	view:DescriptionClass	return a short summary about the specified PANTHER pathway
getLinkedEntries	database: panther	field:id	term: 14744	linkToDatabase: uniprot	returns the proteins related to this PANTHER pathway (applicable only to human proteins and pathways)
Structural Information					
getLinkedEntries	database: uniprot	field:acc	term: p04637	linkToDatabase:pdb	returns the structures related to this protein
getEntryView	database: pdb	field:id	term:1H26	view:DescriptionClass	returns a short summary about the specified PDB structure
getLinkedEntries	database: pdb	field:id	term:1H26	linkToDatabase: uniprot	returns the proteins known to be related to this with this structure. This could provide information about proteins that interact to form a larger complex

Table R-1: Combining EasySRS retrieval and linking queries to collect pieces of information regarding the biological context of biochemical entity.

[1] http://www.pantherdb.org/pathway/

3.5.4.2. Case Study: Data Warehousing for the Plant Defence Mechanism Database

To facilitate the study of the molecular mechanisms involved in plant defence against pathogens, the Plant Defence Mechanisms, (PDM), database has been developed (Barbosa-Silva, *et al*., 2007). The sequences contained in the database comprise seed sequences that were recognised via Amplified Fragment Length Polymorphism[1] (AFLP) experiments, along with putative orthologs obtained through the Seed Linkage clustering approach, based on the bidirectional best-hit strategy (Barbosa-Silva, *et al*., 2008).

To improve the annotation of the data deposited in the PDM database, the information from a diverse set of databases has been integrated. These pieces of information were obtained by invoking the EasySRS services library and include:

a. the total number of entries deposited in the following databases: UniProt, RefSeq[2], UniRef[3] (100, 90 and 50), and PIR[4] for each plant represented in the PDM database (Figure R-9 B).

b. the annotations in the UNIPROT, GO, PFamA[5], INTERPRO[6], and Prosite[7] databases, which can be related to the PDM sequences using their UNIPROT identifier as query (Figure R-9 C).

3.5.4.3. SRS.php

Although writing a client to use the EasySRS web services is a simple issue, the recurring use of the services in the PHP (Hypertext Preprocessor) programming language, lead to the development of a library to simplify the access to the EasySRS functionality even further (Barbosa-Silva, *et al*., 2007) (Figure R-9 A, C).

[1] http://en.wikipedia.org/wiki/Amplified_fragment_length_polymorphism
[2] http://www.ncbi.nlm.nih.gov/RefSeq/
[3] http://www.ebi.ac.uk/uniref/
[4] Protein Information Resource, http://pir.georgetown.edu/
[5] Protein Family database, http://pfam.sanger.ac.uk/
[6] http://www.ebi.ac.uk/interpro/
[7] http://expasy.org/prosite/

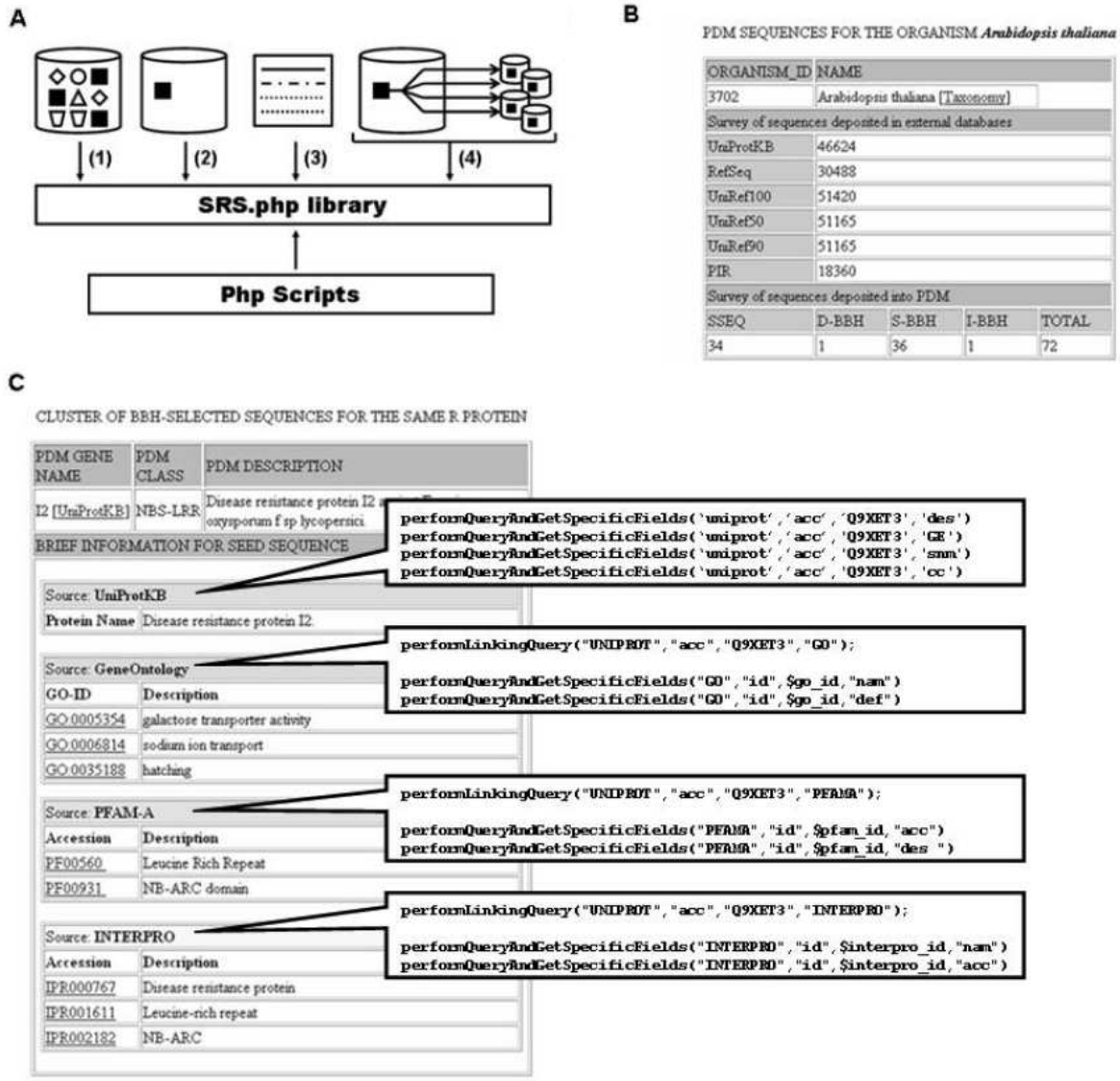


Figure R-9: **A**. Using the library SRS.php, the user can (1) count the number of matches (small black squares) in an SRS database (DB1), (2) retrieve specific records (black square), and (3) retrieve specific attributes of each deposited record (different lines). Additionally (4), the user can link the record (small black square) to the universe of SRS databases. **B**. Number of records deposited for Arabidopsis thaliana in different databases. **C**. Retrieved annotations for gene I2 (UniProt accession: Q9XET3) of *Lycopersicon esculentum* in Plant Defense Mechanisms (PDM) from different databases (UNIPROT, GO, PFam-A, Interpro); the screenshots represent examples of the SRS.php information retrieval for PDM records (Barbosa-Silva, *et al.*, 2007).

3.6. UniprotProfiler and Novel Visualisation Approaches to Support Knowledge Discovery

3.6.1.1. UniprotProfiler and Visualizations Using Arena3D

UniprotProfiler, a Java desktop application, exploits the EasySRS services to generate protein centric graphs of biomedical knowledge. UniprotProfiler allows for keyword-based queries against UniProt; the retrieval of pertinent information for the matching records, and, most importantly, the linking to associated biomedical entities, according to the user preference, such as genes, chemicals and pathways (Figure R-10).

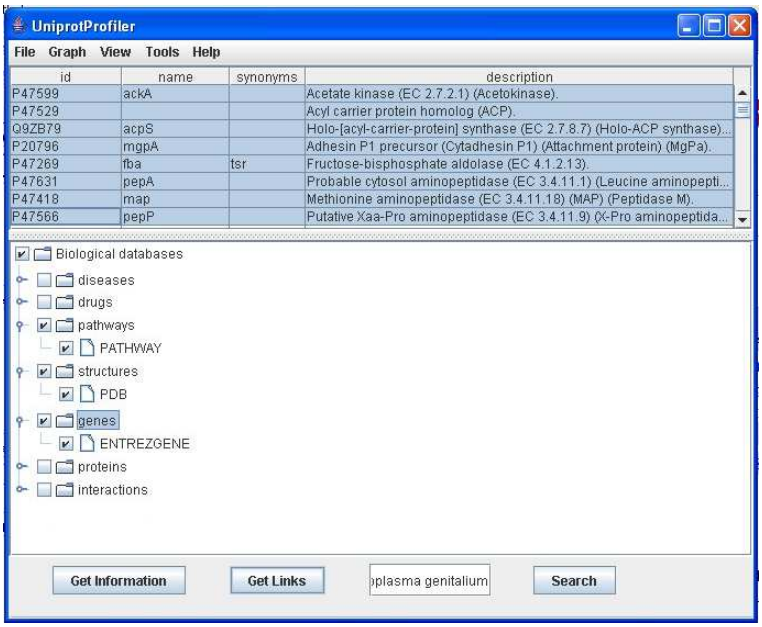
3.6.1.2. Whole Proteome Analysis

UniprotProfiler has been integrated with Arena3D (Pavlopoulos *et al*., (A) in preparation) and used for whole proteome analysis. Figure R-10 shows the search results for the *Mycoplasma genitalium* proteins in UniProt and a multi-layer 3D representation of the related genes (from ENTREZGENE, purple), structures (from PDB, yellow), and pathways (from KEGG, green). Additionally, Figure R-11 is a schematic representation of the EasySRS queries used to generate such a graph of associated entities.

The fact that Arena3D allows clustering algorithms to be applied on each layer, an extra level of association is introduced. This brings together entities deemed similar according to an appropriate criterion, such as sequence similarity for genes, proteins and/or Tanimoto 2D similarity for chemicals.

The clustering algorithms can be applied on a layer include: k-means, affinity propagation, neighbor joining, tree clustering, or UPGMA (unweighted pair-group method with arithmetic mean) (Pavlopoulos *et al*., (A) in preparation)

Researchers can easily explore such association graphs based on Arena3D's visualisation features, such as panning, zooming and rotating. This would assist them in their quest of inferring new knowledge by indirectly associating entities not linked otherwise.



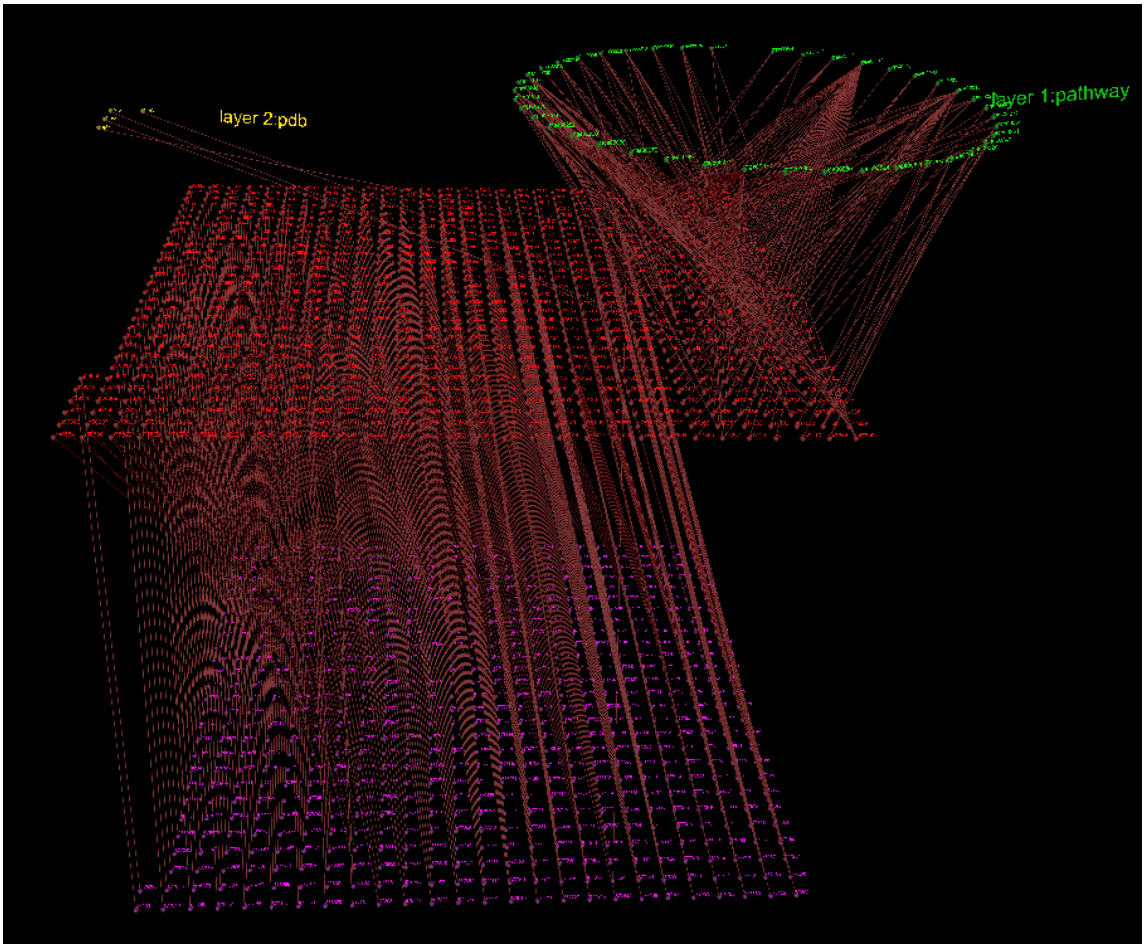


Figure R-10: Top: Search results for *Mycoplasma genitalium* proteins in UniProt; shown with a brief description. bottom: A multi-layer 3D representation of genes (purple), structures (yellow), and pathways (green) related to the Mycoplasma genitalium proteins (red).

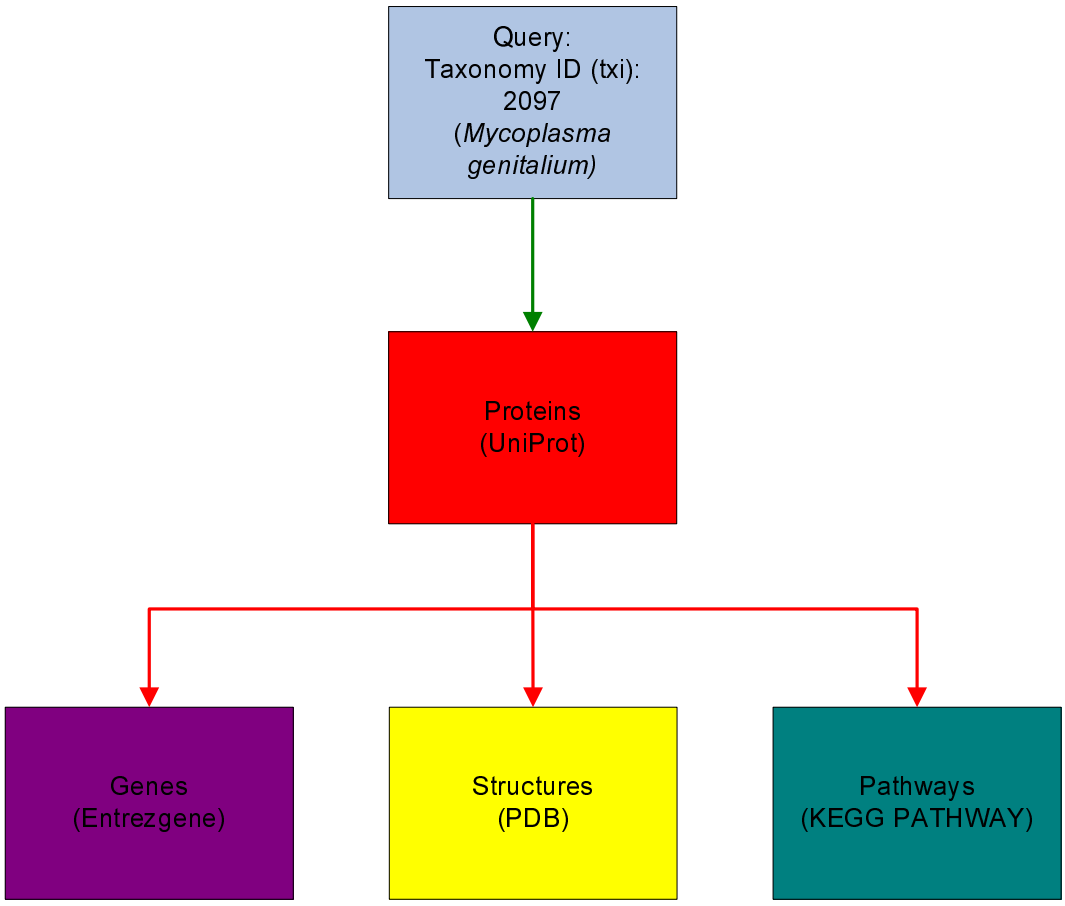


Figure R-11: a schematic representation of the EasySRS queries used to generate the cross-reference-based graph of associations. The green arrow displays the query used to retrieve all the *Mycoplasma genitalium* proteins. The red arrows represent cross-linking queries to associate the retrieved proteins with genes, structures and pathways.

4. Discussion

Finding information about a biological entity is a step that is tightly bound to molecular biology research. This task consumes a considerable amount of time and is repeated whenever a researcher examines experimental results, studies the literature, or even follows the news on his/her favourite scientific topic. A significant amount of effort is required not only for researchers browsing the web to collect the pieces of information they require, but also for scientists following a programmatic approach.

Literature mining and data integration have contributed significantly to this task, the former to extract facts about biomedical entities, and the latter to enable cross-domain queries.

However, there are still open issues on how close they are to the researchers and how easy they are to use. The pieces of software presented in this thesis have tried to address these issues by simplifying the access of life science researchers to pieces of information related to biochemical entities.

4.1. Reflect

4.1.1. Motivation

Reflect derives its momentum from the same driving force that made Tim Berners Lee (TBL) invent the World Wide Web. In TBL's case the motivation originated from the requirement to cross-link the documents generated by the different research groups at the European Organization for Nuclear Research (CERN) (Berners-Lee, 2000).

In Reflect the motivation is to link the document a life scientist is currently viewing in his/her browser with information about the biochemical entities it contains that are available in existing biological databases. Reflect moves a step further by allowing linking to occur in an automated fashion through the use of named entity recognition tools. Web based communication and browser extensions are some of the means to achieve this end.

4.1.2. Behavior and Functionality

The fact that it takes one second to annotate a typical five page document (over a regular internet connection) allows Reflect to be used as an interactive companion, which is practical for the researchers. Pages such as Medline abstracts and news articles can be annotated in about a second.

The summary popup adds another layer of usefulness since it offers a quick overview of related knowledge instead of a plain link to the corresponding database entry. The simple user interface hides the complexity of annotation machinery away from the user.

These reasons justify the positive feedback that Reflect has received so far from researchers, most, but not all of whom, are biologists[1].

While the benefit for the biologist is obvious, for the non-biologist Reflect was even more attractive. Not only due to the fact that it highlights entities which they could not perceive as proteins or chemicals, but also because Reflect points to related entries in databases that they are not aware of.

This issue gains greater importance as molecular biology becomes more of an interdisciplinary science and scientists from related fields become active collaborators.

4.1.3. Implementation

Reflect is a multi-component Web-based application. Its implementation requires the “orchestration” of several pieces of software both on the client and on the server side.

To maximize user friendliness the emerging technology of browser extensions has been employed to create a web-surfing “companion”. The browsers extended, Internet Explorer (IE) and Firefox (FF), have been chosen based on either their popularity (IE), or their support for more than one operating system (FF).

[1] At the time of writing Reflect is among the semi-finalists of the Elsevier Grand Challenge: Knowledge Enhancement in the Life Sciences (http://www.elseviergrandchallenge.com/).

Using HTML, XUL (XML User Interface Language) and JavaScript allowed the implementation of a user interface as powerful as a normal desktop application, facilitating the capturing of user events and the corresponding update of the graphical interface.

Moreover Ajax (Asynchronous JavaScript and XML) has proven a rather powerful technology for the exchange of data with the Reflect server based on the familiar protocol.

Since HTML, JavaScript and HTTP have traditionally been used in the field of bioinformatics, developing browser extensions to satisfy the requirements of molecular biology research is something that the community could easily support.

Additionally these technologies facilitated the reuse of client side components in developing the http://reflect.ws web site. Providing users with this alternative interface increased Reflect's outreach quite considerably since it provides:

- an easy way to demonstrate the functionality of the system
- an alternative for those users that do not want or are not allowed to install browser extensions
- a cross-browser and cross-platform solution

On the downside, http://reflect.ws can process only pages deployed in the web with no access restrictions. For example, this means that files on the researcher's local file system, or subscribed material in his/her library cannot be accessed.

From a technical point of view the fact that http://reflect.ws opens all the security issues mentioned in the Methods sections, means that a considerable amount of time, effort and money have to be spend to secure a system offering such functionality.

4.1.3.1. Web Server

The web server that resides between the front-end components and the tagging server currently performs certain “house keeping tasks” such as communicating the pieces of text to be annotated, and, in the case of a web page, injecting JavaScript libraries and fetching the requested web page.

Should the Reflection requests rise significantly such a middle layer could drive their distribution to more than one tagging server, parallelizing the procedure; hence, increasing the robustness and scalability of the system.

4.1.3.2. Tagging Server

Using a centralized server to perform the text annotation is the main reason that such a task can be performed so quickly. The dictionaries are already loaded in the server memory in hash tables. Looking up words in the hash tables follows a constant-time O(1) complexity on average, regardless of the number of items in the table[1]. Moreover the size of life science dictionaries cannot be handled by an average desktop machine unless they are restricted in coverage.

Keeping the dictionaries on the server side not only removes the burden of maintaining them up-to-date from the users, but it also allows them to be shared among the users. This establishes the infrastructure on top of which a community-curated dictionary can be implemented.

Both the terminology updates at the server level, based on the new pieces of information available in the biological databases, and the community based editing can provide solutions to keep the dictionaries as up-to-date as possible.

4.1.3.3. Informative Summaries

Adding the summary popups in a web page is a combination of replacing the entity names with JavaScript calls, injecting the required JavaScript libraries, and serving the popup content from a server only on request, i.e. after the user has clicked on an entity name.

[1] http://en.wikipedia.org/wiki/Hash_table

No matter how complicated this may sound, this mechanism is a neat and simple solution to always deliver to the user the latest version of the popup no matter which version of Reflect he/she is using.

Unless a cached version already exists, the users' browser will always request the latest version of the JavaScript libraries from the server. Currently the popup inline frame points to the STITCH Summary Service[1]; however, by just switching a URL the popup could be populated from another server. Of course nothing prevents the use of more than one inline frames to collect information related to a biochemical entity from many different resources.

4.1.4. Current Limitations: Named Entity Recognition Performance

Reflect is a fast and easy companion that assists the researchers while they are browsing not only through the literature but also the web. Several questions remain to be answered though. How well is Reflect performing in terms of the biochemical entities it annotates? Are there known cases where the performance is low? If yes, what could be done to improve it?

Reflect's named entity recognition is dictionary based. This implies that the quality of the annotation will mirror the extent to which the dictionary has been curated so as to reduce not only the number of false positives (erroneously annotated gene, protein or chemical names) but also the false negatives, names that have been missed because, for example, they are not yet contained in the dictionary.

At the time of writing Reflect has still to be tested against a test set of biological text corpora, specific for the assessment of text mining systems. An upgrade of the named entity recognition module is already taking place and involves both the addition of aliases and the curation of the dictionary. It will subsequently be followed by an assessment against the BioCreAtIvE I corpus[2].

[1] http://string-stitch.blogspot.com/2008/03/embedding-proteinchemical-information.html
[2] http://biocreative.sourceforge.net/

4.1.5. Reflect and Related Tools

The opportunities opening up in the field of bioinformatics from the advances in biomedical literature mining, the flourish of biological databases, and the web technologies that became available, resulted in intense and active development of tools and services similar to Reflect.

The Concept Web Linker for Firefox and the ChemGS.user.js[1] are also browser extensions able to dynamically enrich the contents of a web page and improve its appearance. To achieve this type of functionality a sequence of steps has to take place.

First the parts of an HTML page that contain plain text have to be identified and differentiated from other components that an HTML document may contain, such as scripts and tag attributes. Traversing the DOM tree and reading the text portion of the appropriate DOM elements is one approach (Concept Web Linker, ChemGM.user.js), while another is to apply rules, e.g. regular expressions, to whole of the HTML page that will define the free-text clauses (Reflect).

Once the pieces of text to be annotated have been identified they should be processed so that entities of biomedical relevance can be located. The named entity recognition (NER) may take place either on the client side (ChemGM.user.js) or in a remote server. In the former case the machinery will be restricted to the capabilities of the local system, while in the latter case, communication between the server and each client has to be estabished (Reflect, Concept Web Linker).

As described in the Reflect tagging server discussion, high performance systems are able to satisfy the resource requirements of life science dictionaries. WhatIzIt (Rebholz-Schuhmann, *et al.*, 2008), is another service that processes plain text and links the terms it recognizes to their corresponding databases. WhatIzIt also employs a central server that holds the terminologies in memory and keeps the dictionaries up-to-date.

[1] Part of the “Userscripts for Life Sciences” set of GreaseMonkey scripts

A distinguishing feature among the tools is the entity types that they support. ChemGM.user.js identifies any mention of chemical entities in the text. Reflect recognizes gene, protein and chemical names, Concept Web Linker is able to annotate behavioural, anatomical, and physiological terms, as well as chemical, disease, gene, and organism names.

The annotation added also differs among the tools. ChemGM.user.js, for example, highlights the chemical names and attaches to them hyperlinks to their entries in PubChem and to a 2D drawing of their structure (Willighagen, *et al.*, 2007). Concept Web Linker modifies the font colour according to the entity type and attaches a popup with merely a definition of the concept and a link to related information at the Concept Web Navigator resource (Mons, *et al.*, 2008). Reflect, as shown in the Results, attaches the most informative and graphical based knowledge summary.

Inserting hyperlinks, pieces of JavaScript and/or style sheets are all valid approaches for updating the HTML page contents and attaching the annotation.

Although these tools come as browser extensions (either self-contained: Reflect, Concept Web Linker), or as scripts that require an intermediate engine (“Userscripts for life sciences” require the Greasemonkey Firefox extension to function), once installed, they share the same privileges as any other piece of software on the user's computer. For security purposes users should install extensions from trusted sources only (Willighagen, *et al.*, 2007).

For the users that either are not allowed, or do not want to install browser extensions, web pages with similar functionality exist. Reflect, Cohse and the Concept Web Linker allow users to browse the web, or at least selected resources such as UniProt and PubMed. Such an approach, besides the no-need-to-install advantage, provides the user with pages that can be viewed with any web browser from any typical desktop machine.

4.1.6. Future Directions

Reflect is a user-friendly application that assists life science researchers in their everyday web and literature surveys. Reflect could evolve in many directions to increase its practical value even further. These may range from the simple addition of more aliases in the dictionary to the establishment of a platform that will allow a community of researchers to share their expertise and tailor the system to their specific requirements.

4.1.6.1. Enriched Dictionaries and Informative Summaries

More aliases and entity types could easily be added in the Reflect dictionary, since it is based on a simple, tab-separated file (Figure M-3). New entity types such as diseases, pathways and organisms will be added in the annotation pipeline. The terminologies will be derived from public resources such as OMIM[1] and MESH, KEGG and GO, and NCBI Taxonomy[2] respectively.

Ideally the addition of an entity type should be accompanied with a graphical, representation or relevant knowledge, e.g. a pathway, a taxonomy diagram and/or a organism picture. That would not only maintain the user-friendliness of Reflect, but also enrich the available summary information.

The current summary popup should be enriched further with information such as adding subcellular localization prediction for the proteins and interaction partners for both the proteins and the chemicals. Moreover when a protein has the same name in more than one species, the user should be able to select which protein to view. Should a chemical and a protein share a common name, the user should be able to browser this information in separate tabs. Figure D-1 shows the next Reflect summary release that incorporates these features.

The new Reflect summary popup is the work of Heiko Horn, as part of his diploma studies.

[1] http://www.ncbi.nlm.nih.gov/omim/
[2] http://www.ncbi.nlm.nih.gov/Taxonomy/

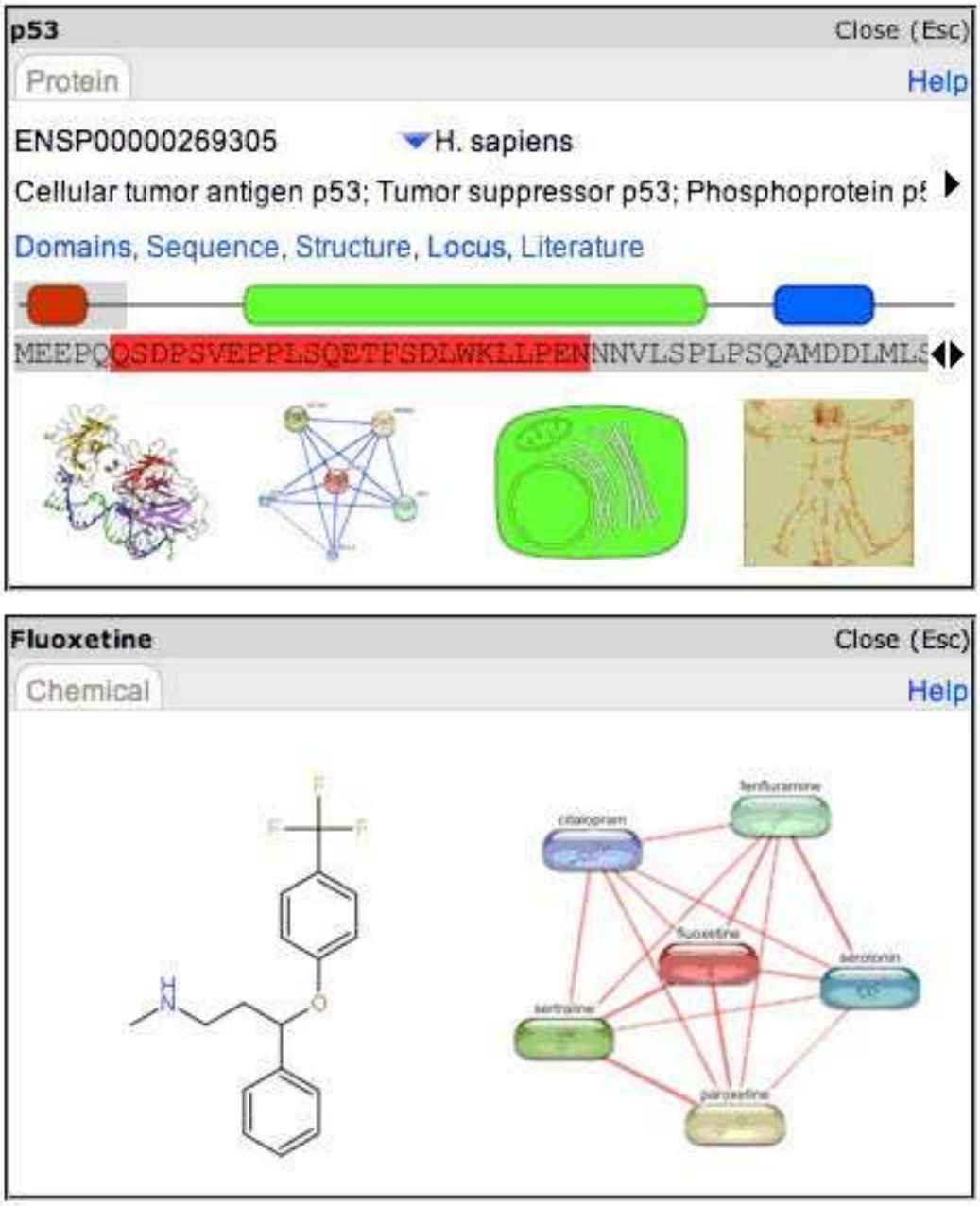


Figure D-1: The next version of the Reflect summary popup: the summaries returned for the protein “p53” and for the chemical “fluoxetine” are shown. The new features include: subcellular localization prediction for the proteins and interaction partners and links to related literature for both the proteins and the chemicals. The sequence and domain architecture viewer is interactive and highlights corresponding regions. The drop-down list in the protein species name indicates that another protein with the same name in another organism exists and can be selected.

4.1.6.2. Browser Support

Supporting more browsers is another step forward. Currently the Firefox browser has better support than the Internet Explorer browser. Achieving the same level of functionality in both browser extensions is one of the next things to do. Extending Reflect to even more browsers is currently out of the scope. The lack of common agreement in the way browsers interpret JavaScript and HTML and of a common platform for plug-in development results in a significant amount of effort being required for the implementation of an extension for each.

4.1.6.3. Community-based Use of Reflect

The fact that the Reflect dictionaries reside on the server-side, as well as the modules that populate the pop-ups, means that they are shared among users. This is particularly important for supporting the transition of Reflect from the authority (server) / consumer (users) model towards a community based one, such as the one followed by Wikipedia[1].

Such a switch will not only enable the collaborative editing of the information in the summary popups, but also help curate the dictionary. The advantages of such an approach will range from the simple correction of a description field or a subcellular localization prediction, to removing erroneous aliases and adding missing synonyms. The extensions are planned for a next release.

Several Reflect instances could then be adopted by different research communities assisting the population of specific databases holding pieces of manually curated knowledge.

[1] http://en.wikipedia.org

4.1.6.4. Reflect As a Named Entity Recognition Platform

Since NER is a building block of biomedical literature mining and Reflect performs such a task, the annotation of web pages could be exploited further, always to the advantage of the researcher. Storing and analyzing co-reference information among proteins and chemicals could populate a database of associations. Such a task could initially apply to pieces of information collected from the "reflection" of selected web resources (e.g. OMIM database records) and proceed in a monitored way. This would guarantee the quality of the associations gathered and prevent the aggregation of erroneous annotations. These associations could then be:

- presented back to the users, such as "this proteins has been mentioned with this set of chemicals"
- used for web document clustering. In this case a list of related web pages could be proposed to the user
- to build profiles reflecting the researcher interest in terms of biochemical entities and commonly used biological resources (combined with the browser extension capabilities if storing data on the user side)

4.1.6.5. Using Reflect as an Authoring Tool

So far Reflect has been presented as a tool to annotate existing HTML pages. As shown, OnTheFly extends the Reflect machinery to Microsoft Office Word, PDF and Excel files. Still remaining though, is the requirement for a Reflect Microsoft Office extension. Such a piece of software would greatly assist the annotation of life researchers' documents during the authoring process.

Although the exact way a user will interact with such a tool is still unclear, the fact that Reflect can be invoked programmatically paves the way for further research and development. Will, for example such an authoring companion follow an auto-complete suggestion approach?

ProTag is an existing Microsoft Office extension that employs the LiMB mark-up and the ProThesaurus biological name services to recognize proteins names and/or identifiers in the text and convert them to Smart Tags[1] (Szugat, *et al*., 2005).

An objective well worth achieving is the conversion of a scientific document into a list of the entities it mentions. This way the compilation of a structured digital abstract (SDA) (Seringhaus and Gerstein, 2007) would be greatly assisted. To compensate for the named entity recognitions errors and omissions the user should be provided with review and correction capabilities. The SDAs could then be exploited further by text mining tools in knowledge acquisition (Gerstein, *et al*., 2007).

[1] http://office.microsoft.com/en-us/word/HP030833041033.aspx

4.2. OnTheFly

4.2.1. Motivation

Unleashing the full potential of combining named entity recognition with information stored in biological databases, and making it available via friendly user interfaces, requires the support of more document types rather than just HTML pages.

Files in formats like PDF, Microsoft Office Word and Excel, as well as plain text, are commonly used by life science researchers to study the literature, store their own pieces of text and/or save their experimental results.

OnTheFly (Pavlopoulos *et al.*, (B) in preparation) enables the conversion of such types of documents into HTML and subsequently employs the Reflect annotation machinery to present them back as enriched and interactive HTML documents.

In the following paragraphs we elaborate on whether OnTheFly achieves what it promises, on its user friendliness, and on the extra features it offers in comparison to Reflect. Finally we present directions in which OnTheFly could evolve to satisfy further the requirements of life science researchers.

4.2.2. Performance

OnTheFly is a system whose performance could be assessed in a number of ways, such as the quality of the document conversion, the time required to annotate a document and the accuracy of the annotation.

Our aim was to establish a working prototype that would offer users adequately preserved documents, fast conversion, and association network generation. At the level of a prototype web-based application OnTheFly has been successful in satisfying these criteria. As seen in the Results figure R-7 OnTheFly is able to maintain the format of the converted documents, including column separation, tables and figures.

Freely available and/or open source converters performed poorly, in contrast to certain proprietary ones. The document converter web search and assessment was conducted by Georgios Pavlopoulos (EMBL).

The time taken to process the full text article shown in Figure R-7 was between 15 to 20 seconds. While this amount of time is considerable, it is a fair price to pay for a 15 page long article, that includes images and tables and is 1 MB in size. Additionally, this is an overall time measurement, which also includes the document communication to the server; a network-speed dependent step in the process.

Using HTML as the intermediate file format proved the *lingua franca* in rendering possible the annotation of all the supported document types.

OnTheFly employs the Reflect tagging machinery to annotate the converted HTML document. This means that the performance of the named entity recognition will be that of Reflect.

4.2.3. Behaviour and Functionality

Similar to Reflect, OnTheFly employs a familiar desktop action, a drag and drop, to simplify the document loading. . Both the desktop and the web-based interfaces offer this piece of functionality, hence they justify the choice of the JApplet as the front-end implementation technology.

After loading the files, the user can retrieve the tagged version of the document, or extract a network representation of the biochemical entities in a document. In this network, the entities are represented as nodes connected by edges based on their known and predicted interactions according to the STITCH data resource (Kuhn, *et al.*, 2008).

The whole process occurs without requiring the installation of specialised pieces of software and does not put any processing load on the client.

4.2.4. Knowledge Network Generation

Through its ability to generate networks of related knowledge based on known and predicted associations OnTheFly serves as an excellent entry point into fields, which the reader is unfamiliar with, or for reviewers who want a quick summary of previous work and background information.

Importantly, the network generation is not restricted to one document. OnTheFly allows networks to be created based on the biochemical entities contained in multiple documents. This feature can prove rather useful even in the most naive of scenarios: the analysis of two separate experimental datasets and the visual search for overlaps or connections between them, based on the known and predicted association information.

In this way, OnTheFly becomes an attractive tool, not only for computer-based scientific readers, but also for data annotators and analysts that want to bring together information from the literature, the biological databases and their in-house data.

4.2.5. High Level Comparison with Reflect

The functionality of Reflect and OnTheFly may be similar, but their scope differs. The Reflect browser extension, acts as an interactive companion following the user while he/she is browsing the web. OnTheFly on the other hand resembles more to an analysis tools. Users already have a file they want to annotate and subsequently study the entities contained in it.

In other words Reflect can assist users by reducing the time taken to find related information about the entities in a given web page, while OnTheFly can save time by serving as a data annotation and analysis starting point.

The Reflect Microsoft Office extension proposed in Reflect's future directions would certainly blur the gap between the two for the benefit of the researchers.

4.2.6. Future Directions

The next steps for OnTheFly range from simple "house keeping tasks" such as assessing the scalability and moving to a bigger server if necessary, to simplifying tedious procedures such as the collection of references from an article, and the initiation of bioinformatics analyses and workflows, using the biochemical entities mentioned in the document as a starting point.

The server currently hosting OnTheFly has been adequate for development purposes and can handle requests coming from within the EMBL. However due to the ease of use and the practical value of the tool, switching to a more powerful system would be required if the tool becomes widely used.

A tedious authoring step is the extraction of references from an article. Since article are also available as PDF files, it would be worth investigating if OnTheFly can assist researchers with such a task, by returning, for example, a library of references upon drag and drop (Sean Hooper, personal communication).

Bringing bioinformatics analyses and workflows closer to the document is another important perspective. This would include the generation of a bioinformatics analysis report for the biochemical entities mentioned in a set of documents. Tasks may range from the simple retrieval of the FASTA sequence of proteins, or their structures, to multiple sequence alignments and homology search reports.

From a software architecture point of view, it would be worth to investigate how Reflect and OnTheFly could share modules to improve the functionality of both. Currently, OnTheFly employs two services to function. The front-end uses the document conversion service to generate an HTML version of the document, and the OnTheFly server uses the Reflect tagging machinery to process this document.

Nothing prevents the document conversion service to be invoked by other applications. It would be possible for example for Reflect to offer users the possibility to process a PDF file as soon as it has been opened in the browser. In a similar fashion the Reflect web page could invoke the document conversion service when the users have typed in a URL pointing to a Microsoft Word, Excel, PDF or plain text file.

4.3. EasySRS Services and UniprotProfiler

4.3.1. EasySRS Services

4.3.1.1. Motivation

The aim of this thesis is to assist the dissemination of biological information to life science researchers. Easy-to-use tools like Reflect and OnTheFly demonstrate how pieces of biological knowledge can be delivered to a broad spectrum of users; users with any level of biological background and/or information technology (IT) expertise.

However, it is also within our aim to simplify the access to pieces of biological information for a more specific set of users; users with biology and information technology skills that follow a programmatic approach.

EasySRS employs web services technologies to provide developers with web-based search and retrieval operations to the Sequence Retrieval System (SRS). By exposing an SRS server, EasySRS services enable simple, yet powerful queries to several biological databases containing among others, genetic, protein, pathway, metabolic and chemical data.

The pilot projects that drove this development were: a. the annotation of a plant defence mechanism database (PDM), and b. the generation of knowledge graphs based on biological record cross-referencing information.

In the following paragraphs we elaborate on the experience gained from developing the EasySRS services. In particular we discuss the pieces of functionality served by the EasySRS services. We comment on our choice to follow community proposed standards and procedures, and we outline the next steps in extending these pieces of software for the interest of life science researchers.

4.3.1.2. Behaviour and Functionality

The EasySRS services are separated in two components the Retriever and the Informer. The Retriever is the main component whose operations can be used to search and retrieve biological records, as well as record cross-references. The Informer, on the other hand, acts as a meta-service and provides developers with information regarding the SRS system. The pieces of meta-information can be used to automatically formulate queries, e.g. “retrieving all the possible cross-referenced entries for a given database record”. The reason for this separation was to clarify the different concerns and make the services easier to comprehend.

EasySRS queries are based on the SRS query language and inherit its power and simplicity. As seen in the Result section, a wide range, from simple to advanced, queries are supported. All operations follow the simple “database, query field, query term” format and the more advanced queries additionally include the use of wildcard characters and boolean operators.

The data retrieval operations support several degrees of granularity, from certain fields or views of a record to the complete entry. Finally, EasySRS services also support cross-linking queries, a feature that also allows SRS to be used for cross-reference based navigation of the life science resources.

Compared to the rest of the methods of accessing SRS presented in the introduction EasySRS services can be invoked from a remote location, over the web, without any firewall issues.

Additionally, compared to the SRS WSObjects^{TM} technology, EasySRS significantly reduces the amount of SRS expertise required to query the system. Although, a certain learning curve is still required, the information available in an SRS web application or by the Informer service provides developers with adequate support.

4.3.1.3. Implementation

Working with the EasySRS services not only included the development of the server side components but also client side ones. The choice of implementation technologies was always based on the criterion of supporting easy client development. For this reason the EMBRACE suggestions and guidelines, deriving from the Web Services Interoperability Organization (WS-I) recommendations have been followed.

These standards do support interoperability and developing code against the recommended technologies (WS-I compliant WSDL and SOAP) has been as simple as even a drag and drop, e.g. in the case of working with the Java programming language. However, , due to poor library support interoperability issues occurred when clients were coded in languages used less frequently in the software industry, such as Perl and Python.

This is an important element to consider when developing web services for the field of bioinformatics where a variety of programming languages are being used. This conclusion derives not only from our experience with EasySRS services but is also one of the conclusions reached in a workshop we participated in, discussing issues in the design, implementation and/or deployment of web services in the context of sequence analysis (Stockinger, *et al*., 2008).

Simultaneously supporting services architectures such as REST, that do not require specialised libraries on the client-side, is one approach to overcoming this issue. Another is to look for the most up-to-date libraries, e.g. the Perl XML Compile[1] library solved the problems of previous libraries in accessing SOAP messages (Jan Christian Bryne, personal communication).

[1] http://perl.overmeer.net/xml-compile/

4.3.1.4. EasySRS and Related Services

The fact that SRS is a data integration system commonly used in life sciences, and the lack of an easy-to-use set of web services to query such a system, have paved the way for other projects, similar to EasySRS, which were developed at the same time: The SRSBiology API[1] developed by BioWisdom Ltd [2] and "SRS by Web Services" (SWS) (Romano and Marra, 2008).

The SRSBiology API, like EasySRS, is based on the SRS WSObjects™. SRS Biology API aims at reducing the query writing complexity by supporting queries against concepts such as "gene" or "protein" instead of database names. XML files are being used to map the concepts to the databases they correspond to. EasySRS services do not follow this approach; the responsibility of defining which databases will represent which real world objects is transferred to the client side.

For example in UniprotProfiler users may select the concepts to which the retrieved proteins will be linked-to via a selection-tree (Figure R-10). The mapping between the tree concepts and the corresponding databases is maintained in the UniprotProfiler configuration file.

SWS employs the SRS wgetz web-application rather than the SRS WSObjects™ to query an SRS server. An important feature of the SWS systems is that it maintains information on the current status of the publicly available SRS servers and is able to distribute queries to alternative servers when a server is non-responding or unreachable. EasySRS currently does not support such functionality. EasySRS can connect to only one SRS server.

4.3.1.5. Future Directions

The EasySRS services can evolve in several directions to assist the dissemination of biological pieces of information to developers. Next steps could include the improvement of the EasySRS services them selves, as well as their combination with other available services.

[1] http://pybios.molgen.mpg.de/EMICD/Deliverables/Deliverable4.6.pdf
[2] http://www.biowisdom.com/

Following the REST paradigm and enabling software communication based on the HTTP protocol would be an important step. This would be beneficial not only for bioinformaticians working with languages for which the SOAP libraries are inadequate but also those already familiar with HTTP-based style of communication.

To increase the robustness of the system it would be advisable to incorporate functionality similar to that of SWS to enable EasySRS to connect to alternative SRS servers in case of a network or other disruption.

A more ambitious aim is to expose, via a common set of methods, the EasySRS services together with web services providing access to other data integration platforms. This would enable developers to seamlessly query the overall available resources, although each system would integrate information from certain databases only (Labarga A., Pafilis E, in preparation).

Finally, the EasySRS services could be combined with named entity recognitions services such as Reflect. This would enable biological records to be served annotated and enriched rather than in their original plain format.

4.3.2. UniprotProfiler

UniprotProfiler aims at bringing together the information available in an SRS platform with Arena3D, a visualisation software rendering biological networks in the three dimensional space.

To facilitate the development of a working prototype UniProt was chosen as the focal point. UniprotProfiler has been successful in achieving its basic aims: to query UniProt, retrieve brief summaries, link the protein entries to database records correnspondingto other real world concepts, to generate 3D visualisations of the results.

While UniprotProfiler can exist as an independent tool it became apparent that tighter interaction with Arena3D is required. UniprotProfiler as is, can be used, for example, to collect information about biochemical entries related to all the proteins of a given organism, however that would result in a single 3D graph generation.

To maximize the benefit in combining the interactiveness of Arena3D with the information that can be disseminated via the EasySRS services the interoperability between Arena3D and UniprotProfiler should be fine-grained. Only by merging the two it would be possible to modularise the search, retrieve and link functionality and render it available to any of the entities selected by the user in an Arena 3D graph.

5. Conclusions

The work presented in this thesis revolved around the issue of accelerating and simplifying the delivery of biological information to life science researchers. Overall, the pieces of software developed were aiming at a broad audience, from bench scientists to IT experts.

Simplicity was one the driving principles; either in the form of easy-to-use tools that hide complex procedures behind familiar user actions, such as a click or a drag and drop, or in the form of a simple and intuitive programming interface that facilitates third party software development.

Modifying and re-using existing components such as data integration and text mining systems was another main principle. The aim in this case was to expose such pieces of software in a way that would extend their scope and enrich their capabilities.

Emerging web technologies provided us with the means of achieving this aim.

The Reflect service has shown that NER can be coupled with informative summaries and that text can be integrated with information in biological databases. Web browser extensions can enrich web pages based on the information from multiple resources, in this case based on the results of NER and the attachment of dynamic summary popups.

Moreover the easy-to-use interface and the at-a-glance summary made Reflect attractive not only to biologists but to a broader audience comprising researchers from other scientific fields, teachers, students, simple users with an interest in biology.

The fact that browser extensions employ HTML, JavaScript and HTTP-based communication, technologies traditionally used in the field of bioinformatics, developing browser extensions to satisfy the requirements of molecular biology research is something that the community could easily not only support but also benefit from.

The OnTheFly service extended Reflect's functionality to other types of documents used by life sciences researchers through out their work. Additionally, OnTheFly serves as an analysis starting point since it supports the generation of networks of known and predicted associations for the biochemical entities mentioned in one or more documents. These documents may range from full text articles, to spreadsheets, to plain text result files. Hence, OnTheFly greatly supports the integration of literature with information in biological databases and with in-house data. On a more practical level deploying the OnTheFly document conversion via a web server removed the need to install and execute software locally.

EasySRS provides developers with search, retrieve and link functionality for a number of life science resources integrated in an SRS server. EasySRS's simple interface combined with the easy, yet powerful, SRS query syntax, render it attractive in developing modules that integrate bioinformatics tools with the information available in biological databases.

Web services on the one hand simplified software development and overcame access issues when using firewalls. However, the Web services following the World Wide Web Consortium (W3C) standards did not provide equal support for all the programming languages. This is particularly important for the field of bioinformatics where a variety of programming languages are being used. Supporting at the same time both W3C Web services and architectures that do not require specialised libraries (e.g. REST) is one approach in overcoming this issue. Another way is the constant search for the most up-to-date libraries.

UniprotProfiler exemplified how services such as EasySRS can be coupled with visualisation software to generate large scale association graphs supporting whole proteome analysis.

The components developed pave the way for enabling seamless access to bioinformatics tools and data, via a simple front-end interface, applicable to the biological entities mentioned in a piece of text.

OnTheFly is already employing Reflect to annotate documents. Reflect may borrow OnTheFly's conversion functionality to process PDF, Excel, Word and plain text files during web browsing, as well as the network generation feature.

EasySRS may not only employ Reflect to deliver enriched biological database records, but also to automatically generate links and improve the network of cross-linked bioinformatics resources.

Reflect, OnTheFly and EasySRS clients could use network visualisation software such as Arena3D to provide users with graphs displaying data from many sources rather than long list of results.

Finally, the underlying web-based communication could be employed to invoke other services applicable to the entities recognised in a piece of text, for example a sequence analysis pipeline. This way third party bioinformatics tools and resources could also be integrated.

6. References

Barbosa-Silva, A., Pafilis, E., Ortega, J.M. and Schneider, R. (2007) Development of SRS.php, a Simple Object Access Protocol-based library for data acquisition from integrated biological databases, *Genet Mol Res*, 6, 1142-1150.

Barbosa-Silva, A., Satagopam, V.P., Schneider, R. and Ortega, J.M. (2008) Clustering of cognate proteins among distinct proteomes derived from multiple links to a single seed sequence, *BMC bioinformatics*, 9, 141.

Bechhofer, S.K., Stevens, R.D. and Lord, P.W. (2005) Ontology driven dynamic linking of biology resources, *Pacific Symposium on Biocomputing*, 79-90.

Berners-Lee, T. (2000) *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. Collins.

Chung, S. and Wooley, J. (2003) Challenges Faced in the Integration of Biological Information In Lacroix Z and T., C. (eds), *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann.

Chung, S.Y. and Wong, L. (1999) Kleisli: a new tool for data integration in biology, *Trends in biotechnology*, 17, 351-355.

Corbett, P. and Murray-Rust, P. (2006) High-Throughput Identification of Chemistry in Life Science Texts. In, *Computational Life Sciences II*. 107-118.

Divoli, A., Hearst, M.A. and Wooldridge, M.A. (2008) Evidence for showing gene/protein name suggestions in bioscience literature search interfaces, *Pacific Symposium on Biocomputing*, 568-579.

Dowell, R.D., Jokerst, R.M., Day, A., Eddy, S.R. and Stein, L. (2001) The distributed annotation system, *BMC bioinformatics*, 2, 7.

Etzold, T. and Argos, P. (1993) SRS--an indexing and retrieval tool for flat file data libraries, *Comput Appl Biosci*, 9, 49-57.

Etzold, T., Harris, H. and S., B. (2003) SRS: An Integration Platform for Databanks and Analysis Tools in Bioinformatics. In Lacroix Z and T., C. (eds), *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann.

Etzold, T., Ulyanov, A. and Argos, P. (1996) SRS: information retrieval system for molecular biology data banks, *Methods in enzymology*, 266, 114-128.

Fielding, R. and Taylor, R. (2002) Principled design of the modern web architecture, *ACM Transactions on Internet Technology (TOIT)*, 2, 115 - 150.

Gerstein, M., Seringhaus, M. and Fields, S. (2007) Structured digital abstract makes text mining easy, *Nature*, 447, 142.

Good, B.M., Kawas, E.A., Kuo, B.Y. and Wilkinson, M.D. (2006) iHOPerator: user-scripting a personalized bioinformatics Web, starting with the iHOP website, *BMC bioinformatics*, 7, 534.

Hirschman, L., Colosimo, M., Morgan, A. and Yeh, A. (2005) Overview of BioCreAtIvE task 1B: normalized gene lists, *BMC bioinformatics*, 6 Suppl 1, S11.

Hoffmann, R. and Valencia, A. (2004) A gene network for navigating the literature, *Nature genetics*, 36, 664.

Hunt, E., Pafilis, E., Tulloch, I. and Wilson, J. (2004) Index-Driven XML Data Integration to Support Functional Genomics. In Rahm, E. (ed), *Proceeding of the International Workshop on Data Integration in Life Sciences, DILS'04, Lecture Notes in Computer Science*. Springer Verlag, 95-109.

Jensen, L.J., Saric, J. and Bork, P. (2006) Literature mining for the biologist: from information retrieval to biological discovery, *Nature reviews*, 7, 119-129.

Kanehisa, M. and Bork, P. (2003) Bioinformatics in the post-sequence era, *Nature genetics*, 33 Suppl, 305-310.

Krallinger, M. and Valencia, A. (2005) Text-mining and information-retrieval services for molecular biology, *Genome biology*, 6, 224.

Kuhn, M., von Mering, C., Campillos, M., Jensen, L.J. and Bork, P. (2008) STITCH: interaction networks of chemicals and proteins, *Nucleic acids research*, 36, D684-688.

Lacroix, Z. and Critchlow, T. (2003) Introduction. In Lacroix Z and T., C. (eds), *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann.

Lacroix, Z. and Critchlow, T. (2003) Introduction, Glossary. In Lacroix Z and T., C. (eds), *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann.

Laskowski, R.A. (2001) PDBsum: summaries and analyses of PDB structures, *Nucleic acids research*, 29, 221-222.

Letunic, I., Copley, R.R., Pils, B., Pinkert, S., Schultz, J. and Bork, P. (2006) SMART 5: domains in the context of genomes and networks, *Nucleic acids research*, 34, D257-260.

McEntyre, J. (1998) Linking up with Entrez, *Trends Genet*, 14, 39-40.

Miled, Z., Li, N., Liu, Y., He, Y., Lynch, E. and Bukhres, O. (2004) On the Integration of a Large Number of Life Science Web Databases. In Rahm, E. (ed), *Proceeding of the International Workshop on Data Integrationin Life Sciences, DILS'04, Lecture Notes in Computer Science*. Springer Verlag, 172-186.

Mons, B., Ashburner, M., Chichester, C., van Mulligen, E., Weeber, M., den Dunnen, J., van Ommen, G.J., Musen, M., Cockerill, M., Hermjakob, H., Mons, A., Packer, A., Pacheco, R., Lewis, S., Berkeley, A., Melton, W., Barris, N., Wales, J., Meijssen, G., Moeller, E., Roes, P.J., Borner, K. and Bairoch, A. (2008) Calling on a million minds for community annotation in WikiProteins, *Genome biology*, 9, R89.

Pafilis, E., Horn, H. O'Donoghue, SI., Jensen, LJ., Kuhn, M., Brown, NP., Schneider R. Reflect: Automated Annotation of Biochemical Terms, in preparation.

Pavlopoulos, G., O'Donoghue, SI., Satagopam, VP., Soldatos, T., Pafilis, E., Schneider, R. Arena 3D: Visualization of Biological Networks in 3D, manuscript (A) in preparation.

Pavlopoulos, G., Pafilis, E., Hooper, S., Schneider, R. Document-based text parsing and data retrieval with OnTheFly, manuscript (B) in preparation.

Pearson, H. (2001) Biology's name game, *Nature*, 411, 631-632.

Pitre, S., Dehne, F., Chan, A., Cheetham, J., Duong, A., Emili, A., Gebbia, M., Greenblatt, J., Jessulat, M., Krogan, N., Luo, X. and Golshani, A. (2006) PIPE: a protein-protein interaction prediction engine based on the re-occurring short polypeptide sequences between known interacting protein pairs, *BMC bioinformatics*, 7, 365.

Prlic, A., Down, T.A., Kulesha, E., Finn, R.D., Kahari, A. and Hubbard, T.J. (2007) Integrating sequence and structural biology with DAS, *BMC bioinformatics*, 8, 333.

Rebholz-Schuhmann, D., Arregui, M., Gaudan, S., Kirsch, H. and Jimeno, A. (2008) Text processing through Web services: calling Whatizit, *Bioinformatics (Oxford, England)*, 24, 296-298.

Ritter, O., Kocab, P., Senger, M., Wolf, D. and Suhai, S. (1994) Prototype implementation of the integrated genomic database, *Computers and biomedical research, an international journal*, 27, 97-115.

Romano, P. and Marra, D. (2008) SWS: accessing SRS sites contents through Web Services, *BMC bioinformatics*, 9 Suppl 2, S15.

Seringhaus, M.R. and Gerstein, M.B. (2007) Publishing perishing? Towards tomorrow's information architecture, *BMC bioinformatics*, 8, 17.

Stein, L. (2002) Creating a bioinformatics nation, *Nature*, 417, 119-120.

Stein, L.D. (2003) Integrating biological databases, *Nature reviews*, 4, 337-345.

Stein, T., Morris, J.S., Davies, C.R., Weber-Hall, S.J., Duffy, M.A., Heath, V.J., Bell, A.K., Ferrier, R.K., Sandilands, G.P. and Gusterson, B.A. (2004) Involution of the mouse mammary gland is associated with an immune cascade and an acute-phase response, involving LBP, CD14 and STAT3, *Breast Cancer Res*, 6, R75-91.

Stockinger, H., Attwood, T., Chohan, S.N., Cote, R., Cudre-Mauroux, P., Falquet, L., Fernandes, P., Finn, R.D., Hupponen, T., Korpelainen, E., Labarga, A., Laugraud, A., Lima, T., Pafilis, E., Pagni, M., Pettifer, S., Phan, I. and Rahman, N. (2008) Experience using web services for biological sequence analysis, *Briefings in bioinformatics*.

Szugat, M., Guttler, D., Fundel, K., Sohler, F. and Zimmer, R. (2005) Web servicing the biological office, *Bioinformatics (Oxford, England)*, 21 Suppl 2, ii268-269.

Tannen, V., Davidson, S. and Harker, S. (2003) The Information Integration System K2. In Lacroix Z and T., C. (eds), *Bioinformatics: Managing Scientific Data*. Morgan Kaufmann.

von Mering, C., Jensen, L.J., Kuhn, M., Chaffron, S., Doerks, T., Kruger, B., Snel, B. and Bork, P. (2007) STRING 7--recent developments in the integration and prediction of protein interactions, *Nucleic acids research*, 35, D358-362.

Williams, A., Sarkar, S., Cuddon, P., Ttofi, E.K., Saiki, S., Siddiqi, F.H., Jahreiss, L., Fleming, A., Pask, D., Goldsmith, P., O'Kane, C.J., Floto, R.A. and Rubinsztein, D.C. (2008) Novel targets for Huntington's disease in an mTOR-independent autophagy pathway, *Nature chemical biology*, 4, 295-305.

Willighagen, E.L., O'Boyle, N.M., Gopalakrishnan, H., Jiao, D., Guha, R., Steinbeck, C. and Wild, D.J. (2007) Userscripts for the life sciences, *BMC bioinformatics*, 8, 487.

Yesilada, Y., Bechhofer, S. and Horan, B. (2007) COHSE: Dynamic Linking of Web Resources. 10 -11.

Zdobnov, E.M., Lopez, R., Apweiler, R. and Etzold, T. (2002) The EBI SRS server--recent developments, *Bioinformatics (Oxford, England)*, 18, 368-373.