Chapter 1

Introduction

Protein structure affects everything around us from how enzymes work, how cells are assembled to how diseases function and spread. Biologists can use this information to cure diseases, understand how enzymes work and improve the quality of life for people all over the world. This study will highlight the important role of structural bioinformatics in solving modern day problems facing biologists. One of the main reasons for biologists under utilizing structural bioinformatics tools, is the perceived, and sometimes inherent complexity and setup of the tools. This problem can be addressed by designing more intuitive systems for biologists to obtain structural biology results. The problem is not just making the tools easy to use but also the management of the generated data. Integrating the data management and analysis tools into one, easy-to-use package would greatly assist biologists in accelerating knowledge discovery in structural bioinformatics and hence in solving pressing problems.

A modern structural biology application can be broadly divided into two basic components. The actual analysis tool which is used to generate the results and a system to manage the data generated by this application. Each of these play an integral role in the end result. If the analysis tool is based on wrong or erroneous data, the results are wrong. If the data is incorrectly managed, the analysis tool which relies on the data will give false results. Each of these roles will be discussed in the next few sections.

A good example of the role structural bioinformatics can play in solving problems, is the threat of Foot-and-Mouth Disease Virus (FMDV) to livestock all over the world. This virus can cause massive economic losses and affect people from all walks of life. Local researchers have identified some areas which would help in understanding problems such as variation in the FMDV 3C protease and 3D RNA polymerase, full proteome variation between serotypes and protein function and structure differences between various FMDV serotype capsid proteins.

The ideal solution to FMDV would be a capsid-based vaccine, but local researchers have found that there are stability differences between FMDV serotypes. Identifying the structural effects of the differences found in each serotype, could help to improve vaccine design. The capsid proteins are also important in infection and thus understanding what influence the differences have on the structure will provide vital information in understanding FMDV infection. FMDV replication speed differences have been tracked to differences in the 3C and 3D proteins. Mapping the differences to a structure and investigating the effect these differences have on function, will allow for a better understanding of virus replication and which areas of the protein are more conserved. Full proteome variation analysis will help to identify regions and features which are important to the virus. Comparing serotype-specific characteristics to proteome variation, differences between the serotypes can be mapped. The variation can then be tracked to features such as secondary structure or post translational modifications.

This is a typical example of where a group would require access to structural biology and bioinformatics tools, yet lack the resources and knowledge on how to proceed. This study aims to address this issue by providing structural bioinformatics tools that can assist the researchers in answering structural biology questions. The results can provide answers as well as guide biologists in designing experiments to verify the results from the tools.

The following sections will address the issues that biologists and structural bioinformatics programmers face with regard to the massive amount of data produced in modern high-throughput biology. Topics such as biological data management, data storage and data access will be discussed together with how it influences biologists and programmers alike. Each section is by no means an exhaustive overview of a topic but a discussion of how it applies to biologists with structural biology challenges.

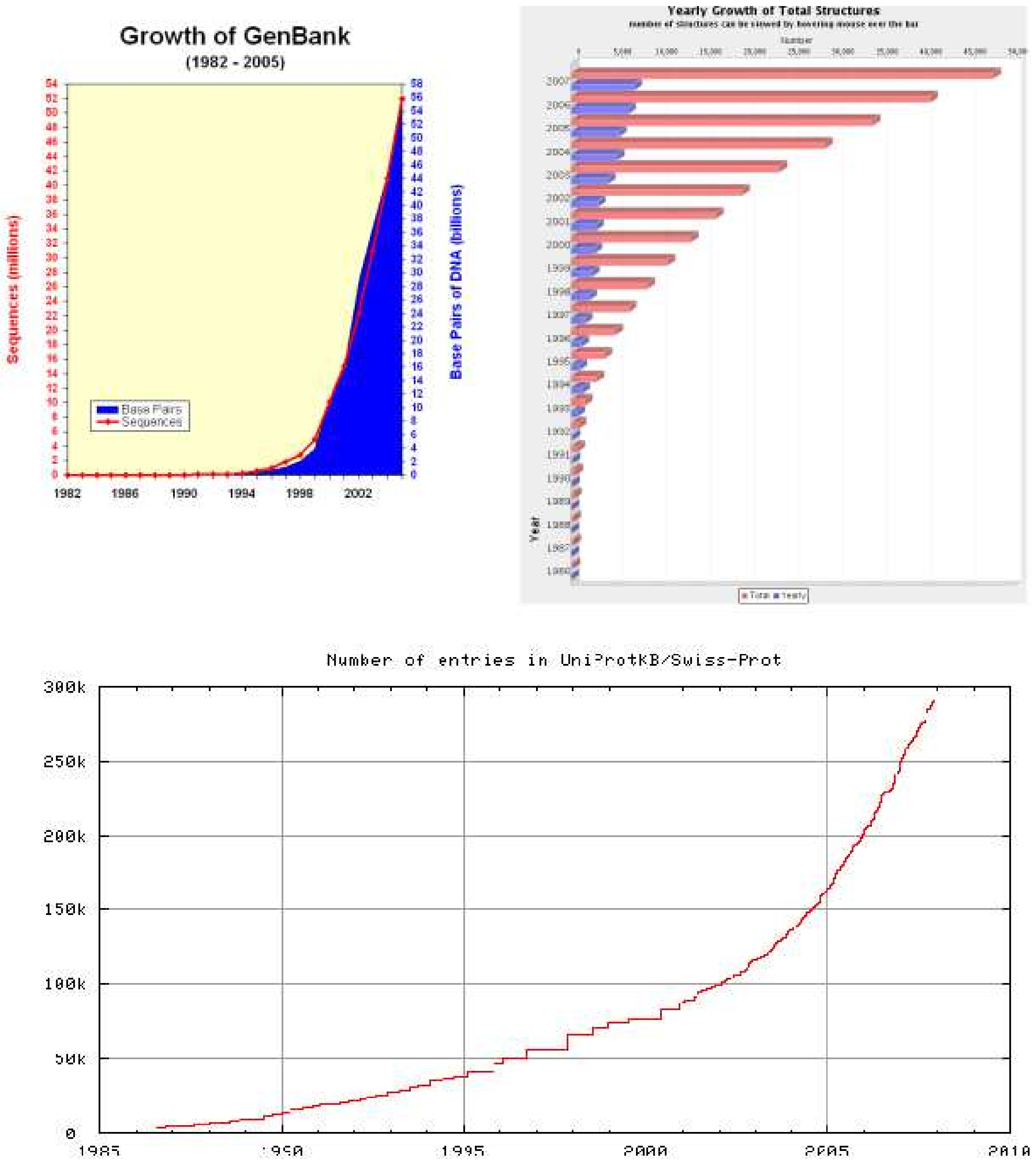

Figure 1.1: The exponential growth in data deposits as seen in GenBank, the PDB and SWISS-PROT (http://www.ncbi.nlm.nih.gov, http://www.pdb.org, http://expasy.ch).

1.1. Biological Data Management

Data production in modern biological sciences is growing at an exponential rate. This is due to high throughput methods (structure as well as sequence-based) and genome sequencing projects. Data banks such as GenBank (Benson *et al.*, 2006), the Protein Data Bank (PDB, Berman *et al.*, 2000) and Swiss-Prot (Gasteiger *et al.*, 2003) have all shown exponential growth in the last few years (Fig. 1.1). This exponential growth in data production has resulted in enormous datasets that need to be stored, curated and managed. Larger databases have overcome the problem of data management to a certain extent by forcing data depositors to conform to a certain format when depositing data. This allows for a more automated approach to data management. Some data banks have even gone further and are employing people to verify and cross check data before it is deposited. A good example of this is Swiss-Prot, which is a database dedicated to manual curation and storage of protein sequences. Before a protein sequence is accepted into Swiss-Prot, a human will verify the function and description of the protein by looking at various papers about the protein and comparing the data. If the function and description are deemed to be correct, it is included in Swiss-Prot. This type of data management is highly labour intensive and takes a long time for each protein sequence to be verified. Swiss-Prot also hosts another section of protein sequences called TrEMBL. TrEMBL is a computer translated version of cDNA sequences found in the EMBL database and thus contains very little annotation and may be of variable quality or hypothetical.

Not only is the storage of these datasets a problem but also the presentation of the data to the user in an effective way. The large data banks have improved during the last few years by presenting users with easy to use web-based interfaces to search the data. This allows the users to easily find and access the data located in a specific database. Larger service providers such as the PDB and SWISS-PROT, have taken it one step further by incorporating data from other sources as well when a user views a record. The PDB for example links out to Pubmed, Pubchem and to protein fold details at SCOP (Conte *et al.*, 2000) and CATH (Pearl *et al.*, 2005), while Swiss-Prot provides links to EMBL, PIR, UniGene, ModBase, InterPro and Pfam among others.

1.2. Data Storage

All data banks/databases rely on the storage and linking of data. Small amounts of data are easy to store and process with the processing power available today, but certain datasets are just too large e.g. the GenBank dataset in May 2008 was 66 GB and that of the PDB 6.5 GB of compressed text files (approximately 27 GB uncompressed). These large datasets require an efficient and fast way of storing and retrieving data. A good example is the Macromolecular Structure Database (MSD, Boutselakis *et al.*, 2003) from the European Bioinformatics Institute (EBI). Their approach was to parse out all the data from the PDB, correct it as far as possible using external analytical chemistry tools, enhance the data by extracting cross links between different data types and then storing it in a custom relational database. This has the drawback of increasing the dataset size when compared to the PDB dataset (27 GB uncompressed vs. 300 GB uncompressed for MSD). However the added advantage is that the relevance of the data is increased and by storing it in a relational database, it also increases the speed and efficiency by which the dataset can be queried by users.

There are two main types of general data storage: flat-file based or storage in a relational database such as Oracle or MySQL. Both have advantages and disadvantages (Table 1.1). Flat-files are defined as data being stored in a single file on disk with fields separated by delimiters. Relational databases are defined as databases which define relations between data sets using the Structured Query Language (SQL) to perform operations on the data, using a database management system.

SQL is a computer language that was designed to facilitate the management and retrieval of data as well as database access control and schema management. SQL has been standardized by American National Standards Institute (ANSI, http://www.ansi.org) and the International Organization for Standardization (ISO, http://www.iso.org). This was done to enable applications to be moved between different database systems without major code rewrites.

Another major problem in storing data is redundancy. A good example of this is Gen-Bank. There is an enormous number of sequences which only differ by one or two bases

Table 1.1: Comparison between Flat-file data storage and Relational database data storage (Doyle, 2001).

	Relational database	Flat-file
Advantages	- Data entered only once - Files/tables are linked - Can handle complex search criteria	- Fast for storage of static information - Access speed limited by disk speed - Can be stored on shared file system
Disadvantages	- Usually hosted on one file server - Security need to be considered carefully - Direct users need additional training	- Difficult to search - Difficult to change/update data - No relations between different files

or amino acids. These sequences are usually Expressed Sequence Tags (ESTs) which were deposited. All of these ESTs are distributed with the full version of GenBank. The non-redundant version is distributed without these "duplicates". The PDB has a similar policy with regard to crystal structures. One protein sequence may have a few different conformations/structures depending on the crystallization conditions and ligands present. Some databases remove this redundancy to create a smaller, more manageable dataset, yet these redundant sequences contain a wealth of data that can also be utilized. Thus once again, there is a trade off between storing a smaller non-redundant dataset versus storing a larger, redundant dataset.

1.3. Data Models

All of the databases/data banks discussed in the previous sections store data in some way or another. Some of these systems such as MSD use a data model to store the data. A data model is a description of the organization of, and relationships between, data in a manner that reflects the information structure. This model is also usually used as a database structure.

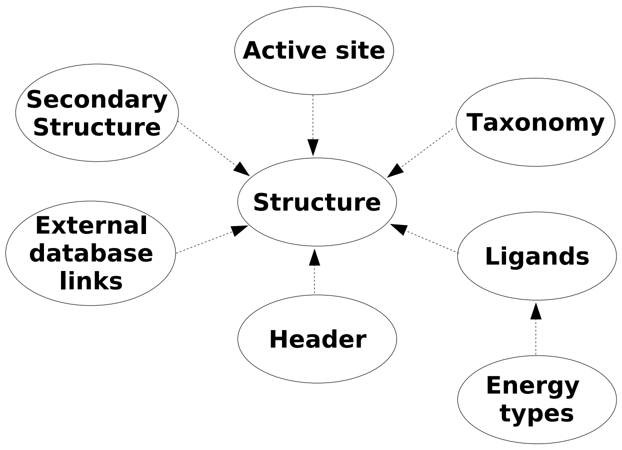

Figure 1.2: A high level overview of the data model of MSD (http://www.ebi.ac.uk/msd-srv/docs/dbdoc/). The main Structure entity is enhanced by linking it to other data types.

The data model used in MSD is based on the hierarchical structure of proteins and works in a top down manner. A structure serves as the main data object and other types of data such as active sites, ligands and taxonomy are added (Fig. 1.2). A structure entity is divided into many different sections (Fig. 1.3). Through a series of cross-links these different entities contain all the data about a structure. The MSD data model allows for various cross-links and external references to be incorporated into the model thus adding value to the pure structure data.

The Functional Genomics Experiment (FuGE) is an attempt to facilitate data standard convergence between the different high-throughput techniques used in biology (Jones *et al.*, 2006; Jones *et al.*, 2007). FuGE provides a foundation for the description of complete laboratory workflows and provides mechanisms for developing new data formats and for the integration of data between techniques. FuGE was designed so that different facets of a "'omics" experiment can be captured and stored. This includes data such as protocols, sample sources and results. Providing a common platform to store common data types would allow for data to be shared among different groups. This would, for example, allow a microarray study using MicroArray Gene Expression object (MAGE) data model to share a basic set of information with someone doing a study using the Proteomics Standards Initiative (PSI) data models. The FuGE model also allows for rich

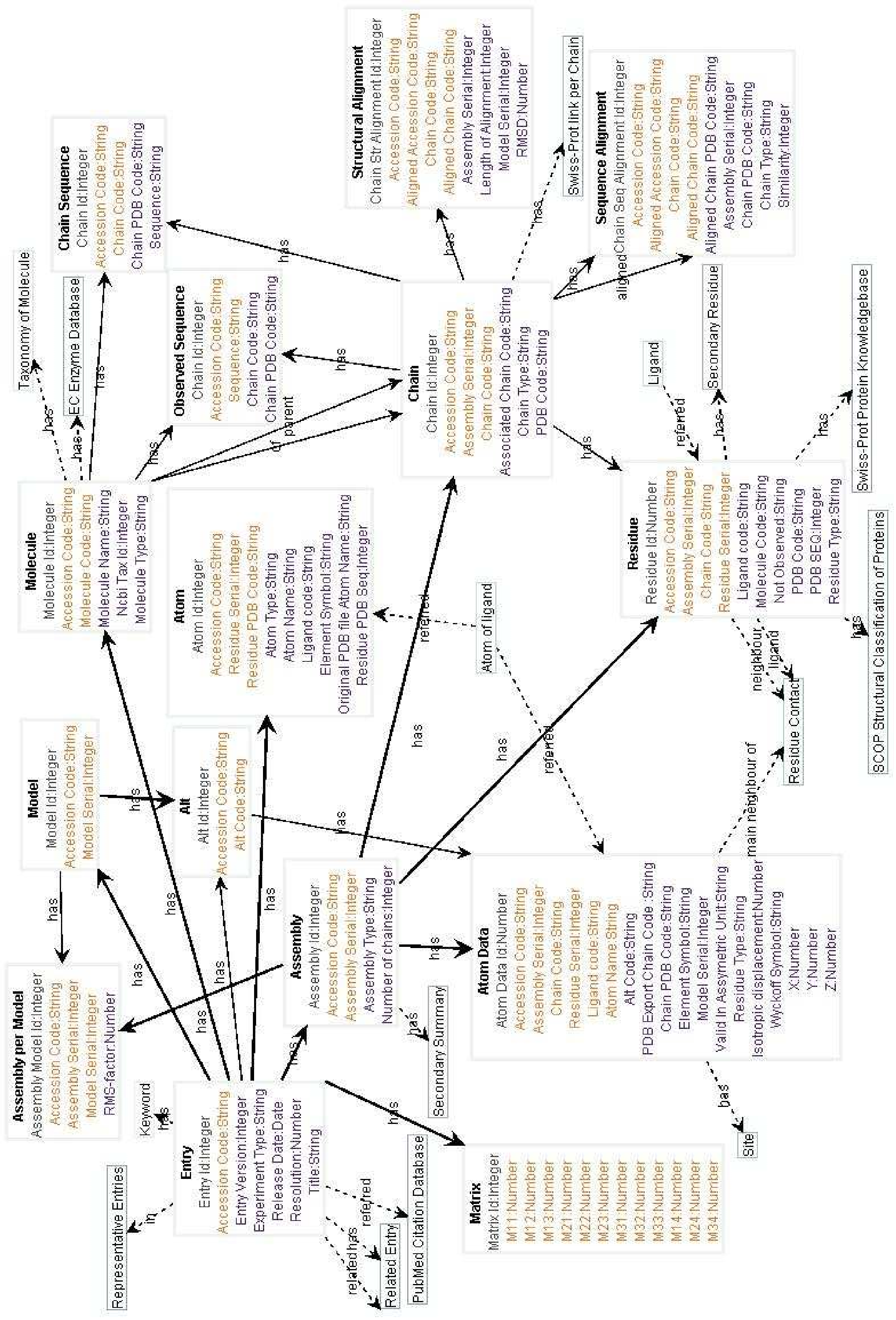

Figure 1.3: The different entities belonging to the Structure entity in MSD (http://www.ebi.ac.uk/msd-srv/docs/dbdoc). This data model is a representation of the data as well as a diagram of the actual database structure.

Table 1.2: The two categories of FuGE with the packages in each category (Jones *et al.*, 2007).

FuGE	Common	Audit
		Description
		Measurement
		Ontology
		Protocol
		Reference
	Bio	ConceptualMolecule
		Data
		Investigation
		Material

annotation of samples and because of the underlying standard model, it will allow data sharing between samples and methods.

FuGE has 10 different packages contained in two categories: `Common` and `Bio` (Jones *et al.*, 2007). The `FuGE.Common` class consists of `Audit`, `Description`, `Measurement`, `Ontology`, `Protocol` and `Reference` (Fig. 1.4). `Audit` provides security settings, `Measurement` provides slots for values and units, `Ontology` provides for external referencing vocabularies, `Protocol` provides a model for procedures and workflows and `Reference` provides links to external database references. `Description` allows free text annotations and descriptions for all objects and inherits directly from `Describable`. All objects in FuGE can be represented under the `Common` category. `FuGE.Common` has two base classes: `Describable` and `Identifiable`. All FuGE objects belong to either one of these classes.

Each of these classes are further separated to provide adequate methods to store protocols and samples. The `Identifiable` base class provides a unique identifier for each object in the system and `Identifiable` inherits from `Describable`. This provides each object in the FuGE system with a unique identifier which is linked to a free text description and security settings. `Identifiable` also provides a logical point from which to extend the FuGE system. `FuGE.Bio` contains `ConceptualMolecule`, `Data`, `Investigation` and `Material`. The `ConceptualMolecule` category provides classes for the storage of DNA, RNA and amino acid sequences but only in a limited way. In theory this can be extended to other molecules. `Data` provides a way to link to multidimensional experimental data using the subclass `ExternalData`. `Investigation` allows for overall experimental design

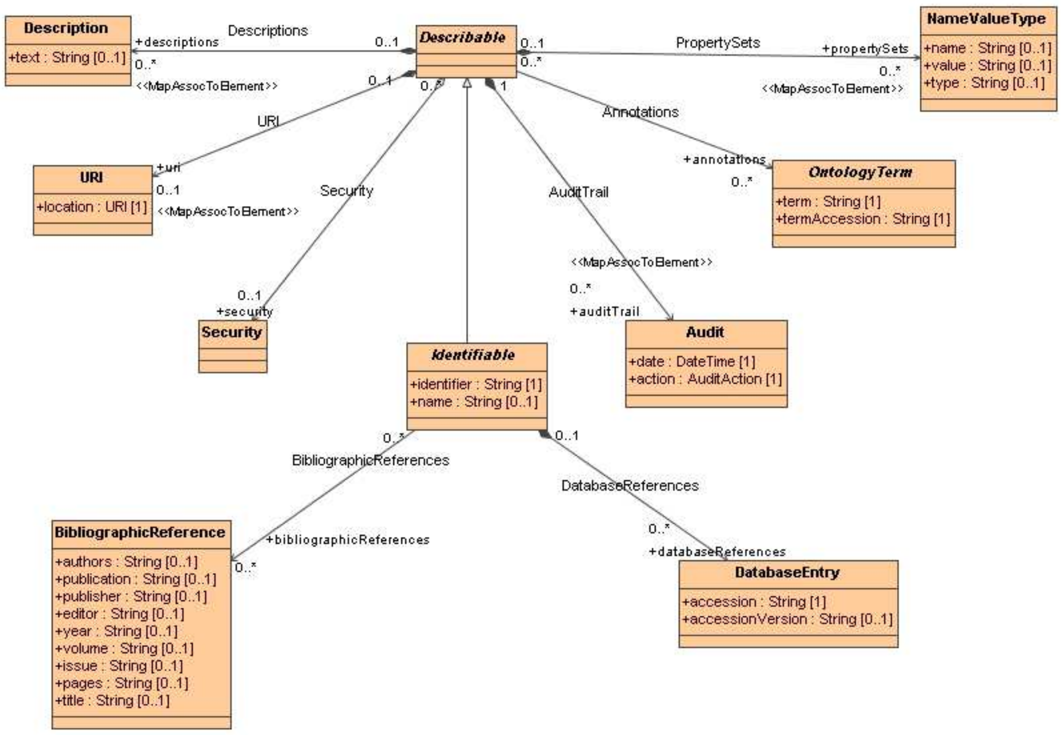

Figure 1.4: The classes of `FuGE.Common` (http://fuge.sourceforge.net). These classes allow for the storage of basic information about each sample.

storage as well as storage of experimental variables. `Material` caters for sample source identification using a controlled vocabulary.

Both MSD and FuGE are successful in storing specific data but most users use a range of data types. Whereas MSD stores all the structural data, it does not cater for storing analysis results nor does it store the methods used. FuGE stores laboratory procedures and protocols but it does not store extensive specific data such as sequences or structures (only basic storage is supported). This basic storage allows for a model that is very compatible between systems and also makes it easy to expand for a specific system. An ideal functional genomics system would store protocols, data and results in a data model compatible with models such as FuGE and MSD. The Functional Genomics Information Management System (FunGIMS) utilizes a data model which stores the most important parts of both FuGE and MSD in one data model without losing the integrity of each separate model yet provides an interface to both. Some parts of FuGE were not used as they represent experimental protocols and conditions and FunGIMS only caters for data storage and analysis.

1.4. Information Management Systems

While major databases host public data, laboratories often need to host their own data in a specialized way. Systems that host data in this way are usually referred to as a Laboratory Information Management System (LIMS). The main characteristic of a traditional LIMS is that it manages data and tracks samples through the system.

The last few years saw an explosion of LIMS, all specialized for dedicated tasks. For example a LIMS simply called LIMS was developed for tracking high throughput genetic sequencing and candidate mutant screening (Voegele *et al.*, 2007), CLIMS (Crystallography IMS) to organize the large amounts of data generated by crystallization experiments (Fulton *et al.*, 2004). PARPs was developed for managing liquid chromatography tandem mas spectrometry and the associated protein identification and data management (Droit *et al.*, 2007), PACLIMS for managing eukaryotic genome-wide mutational screens and the functional annotation thereof (Donofrio *et al.*, 2005), a 2-D gel electrophoresis LIMS was developed to deal with large-scale proteomic studies (Morisawa *et al.*, 2006) and MACSIMS for dealing with data mining from multiple sequence alignments (Thompson *et al.*, 2006). T.I.M.S is an example of a very specific LIMS designed for tracking genotyping data flow and analysis in a laboratory (Monnier *et al.*, 2005).

LIMS users are usually facilities or users who generate relatively large quantities of data in efforts such as large-scale sequencing or high throughput crystallographic studies. The large amount of data needs to be stored efficiently and analyzed in a consistent and effective way. This is one of the major advantages of LIMS but when it comes to detailed data analysis, it can also be a disadvantage. The trade off between being able to store and do basic analysis on a large amount of data and being able to do detailed analysis on a small set of data is one of the drawbacks of LIMS. Some systems like CLIMS can store a large amount of data but does not allow the user to do a detailed analysis of the structure. Other systems such as T.I.M.S. provides a very specific service for a subset

Table 1.3: Comparison between the technologies in currently available LIMS.

LIMS	**Main feature**	**Language**
LIMS	Automated high throughput mutation scanning	MySQL + Java
CLIMS	Crystallization procedure management	MySQL + Java Rich client
PARPs	Liquid Chromatography data management and analysis	Oracle + Perl
PACLIMS	Managing high throughput sequencing data and protocols	PostgreSQL + Perl
MACSIMS	Protein family alignment and data extraction	ANSI C
TIMS	Sample management and parsing of TaqMan data	Visual Basic

of data. LIMS can greatly enhance throughput in a lab as they allow for centralized storage and standardized analysis protocols. All data are treated and interpreted in the same way, providing a big advantage when doing analysis. It also allows users access to centralized analysis tools. All LIMS need to store data in some way. Most LIMS rely on the proven technology of relational databases with additional data stored as flat-files (Table 1.3).

One of the biggest advantages of LIMS is the ability to organize data. This is in sharp contrast to classical biology where results were written on paper in laboratory books and data stored on various CDs and DVDs. LIMS provides a way to store and search through data in an organized and systematic manner, thus increasing efficiency. The organization, analysis and data storage abilities of a LIMS will be illustrated in chapters 3-5 when various structural problems such as the capsid proteins in FMDV are investigated. Web-based systems provide an advantage to novice users venturing into structural bioinformatics, as web interfaces are experienced as a familiar environment, and preclude the need for the installation of local software, which sometimes has complicated dependencies. Available web-based systems for structural bioinformatics vary in terms of the level of analysis functionality available and the level of knowledge required for use. The Spice DAS client is an example of a system for viewing and performing basic exploration of a protein structure, starting with a PDB ID (Prlic *et al.*, 2005). Spice also provides a DAS-based annotation of especially structural properties of the protein being viewed. Web helper-based applications such as Cn3D also allow extensive visualization of protein features and structural alignments, together with the preparation of protein structure figures for reports and publication (Hogue, 1997). STRAP provides a Java web-start application to perform extensive multiple alignments and superimposition of protein structures, together with protein structure views and sequence-based analysis of structural features (Gille and Frömmel, 2001). Various other structural tools are also available depending on the needs of the user.

1.5. Common Structural Analysis Needs of Biologists

The Holy Grail of structural biology is the ability to predict the three dimensional structure of a protein given only the amino acid sequence. Although it sounds relatively easy, the solution to this problem is one of the most sought after in science. As protein structure is inherently linked to protein function, knowing the structure of a protein allows one to derive the function of that protein and change it. Once a structure can be predicted accurately from sequence, it allows the researcher to do *in silico* mutations and obtain a reliable result in a short space of time. This will not replace the need for experimental work, but provide assistance to guide experiments better. It will eliminate various problems encountered with proteins not expressing or not crystallizing. Protein structure can also help guide a researcher in designing more efficient and accurate experiments to address biological problems.

Biologists are familiar with working with DNA sequences or proteins *in vitro*. Often during this process, very little time is spent thinking about the protein in three dimensions. When keeping a three dimensional picture in mind, it gives a new perspective on the problem. If the protein structure is known or well studied it allows for much easier data retrieval, but when working on an unknown structure, the task of getting information about a protein can become rather daunting. By adding protein structural knowledge, they can guide or enhance experiments e.g. using a protein structure to identify possible sites for mutagenesis studies. However, accessing the protein data can sometimes be problematic.

Discussions with biologists have identified a few main problems which often prohibit them from utilizing protein analysis tools. Two main problems were cited, that of accessibility to programs and a lack of knowledge of new programs/databases. More and more programs are being released by authors on the Internet and thus the problem of accessibility will lessen with time. Biologists are generally comfortable with using a few general purpose programs or servers such as Excel, Word, NCBI Blast, the Genbank server and maybe one or two other specific programs or servers. Due to the nature of modern biology, these are the programs they use on a regular basis and they are not thus exposed to other servers and databases. Most of these programs are either web-based or are preinstalled on their computers, thereby leaving biologists with very little interaction regarding program installation and setup. This is in contrast to most open source structural programs which the user has to install by themselves. These mostly run on UNIX-based systems and requires a basic knowledge of the operating system and the program's syntax. Although these problems can be resolved relatively easily, they are seen as a major barrier to the more widespread usage of protein structure programs. In some cases this can be attributed to the perceived complexity of UNIX-type systems. Some authors of programs have realized this and started releasing their programs for the Windows and Apple Macintosh operating systems as well. Although this is a step in the right direction, it still does not solve the problem of setting up the program and the analysis. The ideal solution would be to have a system administrator who is capable in both Windows and UNIX environments, and who will assist the users with setting up these programs. Unfortunately, the responsibility usually falls on the researcher to install and manage the programs.

Another factor mentioned was the lack of knowledge of available databases or programs. This problem is two-fold. Firstly biologists should strive to read beyond their own field of interest, and not be hesitant to search for programs or servers. Secondly some programs or databases are simply not published in well known journals. The Nucleic Acids Research journal tries to address both of these problems with a yearly, open access issue of all the known, biological databases and servers but this does not cover any structural bioinformatics programs. A good approach, however, would be to have someone with a strong interest in structural bioinformatics, keep abreast of developments in the field. Even something as simple as subscribing to journal alerts, would be helpful. The best approach would be to have a person such as a postdoctoral student or technical staff member dedicated to looking for new programs, making them available and providing support for these programs. Such a person should ideally be aware of the different types of projects in a group, have a good biology background and be capable of installing and managing the application server as well as run the programs for users. He or she could also develop a website which allows for easy access to all of these tools to local researchers. Most biologists would just need an introduction to the program and a few basic guidelines to get started and continue on their own.

The problem with regards to program use and knowledge lies not only with the users thereof but also with the programmers. A program that has a good user interface with clearly defined functions and a good explanation of each step, is as valuable to the user as the person guiding them. The onus is on programmers to provide documentation, examples and a good interface for users, but unfortunately this is lacking in many programs. Another problem, which can be traced to the point-and-click method used in Windows, is the lack of understanding of file formats and the amount of information contained in a file. Many biologists are hesitant to explore inside files. A good example of this is a PDB file of a protein. Most users will simply load the protein in a visualization program and ignore the valuable information contained in the file header and comments. This problem can only be addressed by making users aware of the extra information and making them comfortable with exploring text files.

Biologists have an array of needs that can be resolved by using structural biology programs. When an unknown protein sequence is identified, most biologists just do a BLAST search in an effort to identify it. Although this usually yields results, there are many more tools that can be used to gain knowledge about a protein. By simply using the sequence, a biologist can identify whether the protein is a membrane protein or not, protein function may be derived from certain sequence patterns contained in the sequence and in some cases even cellular localization can be determined. This can be taken a few levels higher to a three dimensional view of the protein. From a similar protein structure, details such as active site residue conformations, residue interactions and sometimes even function, can be derived. If a similar protein structure is found, a homology model can be built which can guide the biologist in identifying active sites, important secondary structures and guiding site-directed mutagenesis experiments to confirm function. Analysis of the protein structure can also help in identifying surface areas involved in protein-protein interactions and identify flexible areas in proteins. These types of data can all be combined to give a far better understanding of the protein and the way it functions. It can also serve as a starting point for the researcher to investigate function or structure in more detail using molecular biology.

Some of the functionalities mentioned, are available on web servers around the world. Unfortunately, a lack of knowledge often prevented biologists from exploring the full range of programs available. This was one of the motivations behind this project, to provide services to local biologists in a centralized and locally available solution. If these services and programs can be provided and maintained locally, it would benefit researchers greatly.

1.6. The Functional Genomics Information Management System (FunGIMS)

The overall FunGIMS project was conceived when researchers from the Forestry and Agricultural Biotechnology Institute at the University of Pretoria, approached the Bioinformatics and Computational Biology Unit to provide them with bioinformatics support services related to the *Eucalyptus* genome sequencing project. They required a system which would allow them to store their sequences, annotate the data and do various types of analysis on the sequences, all in a local environment. From these requirements, FunGIMS was expanded to include different types of data such as protein structure and small molecule data.

The philosophy behind FunGIMS was based on allowing researchers access to various tools and data sources in an easy to use environment with extensive data management capabilities. Various problems related to data sources and tool access were identified by the researchers. These problems included the slow bandwidth in South Africa, the high costs associated with Internet use and the problem of storing data. The problem of data storage surfaced as one of the primary concerns. Researchers were used to sharing computers and thus stored data on CDs, laboratory books and memory sticks. This resulted in data being distributed in various places and formats. It also posed a problem to supervisors when they needed access to the data of students. A central repository where students can store and analyze data while still allowing supervisors access, would solve this problem to a large extent. The ability to store data was one of the primary factors considered during the design of FunGIMS. To prevent duplication of designing a way to store the data, it was decided to use FuGE as a starting point. As FunGIMS and FuGE had a similar goal with regard to storing data, it would be a great benefit to use this standardized way of storing data. It would also allow researchers the ability to share data between FunGIMS and FuGE compliant systems.

The slow and expensive bandwidth also affected the design of FunGIMS by forcing local repositories of all the major databases to be installed and used. Local repositories of all the major databases were set up. This allowed very fast, local access to these databases which allows for extensive integration between the different databases. All the services would also be hosted locally, thus providing fast access to data and results for the researchers. By keeping all the databases in one, central location, it made administration and updating of the databases far easier. A system administrator could automate the downloading of the database updates and keep all the databases up to date.

Another major goal of FunGIMS was integration between data types. Usually a database only provides one type of data with a few links to related data. Ideally, a system would provide a user with relevant links to other types of data e.g. when looking at a cDNA sequence, the system would provide the user with links to the protein sequence, protein structure (if present), literature references and possible small molecule interactions. This would allow the researcher to get an overall view of the specific product, instead of just looking at the details of a specific length of sequence. Integration between public and private data is also provided but only in the sense that public data is integrated with private data. Thus a user with private data can makes links to and see public data, use private and public data but still prevent access to and integration with the private data.

The overall scientific goal of FunGIMS is to provide the user with a set of tools and access to a large amount of data in one convenient place. The idea is not to replace the use of each individual tool but to provide the user with results which can serve as a starting point. For some biologists this will provide enough information to allow them to continue down a specific route. Others may want to pursue a specific topic in more detail. The separate modules cater for the main types of functional genomics data used. Each module helps the user to do analysis relevant to that topic and tries to provide links to other data types in FunGIMS. Currently FunGIMS consists of Sequence, Structure, Genomic and Small molecule modules and will in the future include modules for Microarray, Genotype and Literature data. Each of these modules are specialized to deal with a different type of data. All the modules overlap with each other to some extent, but each still provides unique functions for the specific data type e.g. proteins have a sequence that is mostly dealt with in the Sequence module whereas the structural aspects are dealt with in the Structure module. Integration between the different datatypes in each module is of vital importance. A good example is that of a user interested in a specific protein and its function. The Structural module will provide access to structural data on the protein, but at the same time it will provide links to DNA sequences, genome locations, genotype data, microarray results and related literature (where available). This will allow the user to see under which conditions the protein is up or down-regulated, which SNPs have been identified in the cDNA sequence and where the DNA coding for the protein is located on the genome. FunGIMS aim to provide an environment in which a user can access different types of data that are all linked by a common element (in this case, a protein). This type of data integration is fast becoming the future of all databases and provides a far more complete overview of a specific protein.

Each of the modules was approached from the view of the researchers, what they would want to accomplish, which tools they would use and how they would use such a module. This prevented modules from being designed according to developers rather than to assist the researcher. During the design process, researchers were consulted on commonly used tools, the way in which they used the tools and ways in which they wanted data to be presented. Usability was also kept in mind to make the tools easy to use. To facilitate easy use of the system, it was decided to focus on a web-based system, rather than a standalone system. This presents the user with a familiar environment (web browser) and allows for minimal hardware and software installations. While benefiting the user, such a system will also benefit the administrators as they need to install and maintain only one server, instead of a number of computers at various workstations.

FunGIMS supplies a variety of these services but this specific study focuses on protein and protein structure-related services. The Structural module of FunGIMS aims to provide the users with three different types of tools: Explorative, Analysis and Modelling tools. Explorative tools allow the user to explore known protein structures and their features, Analysis provides a selection of general tools to allow the user to analyze a sequence or patterns found in a sequence and Modelling allows the user to build homology models and generate scripts for various molecular dynamics programs. The scripts are intended as a stepping stone to encourage user-driven investigation.

The Analysis section will provide tools such as Prosite (de Castro *et al.*, 2006) and Hidden Markov Model searches against Pfam (Finn *et al.*, 2006). Prosite is a tool used to find motifs in a sequence which may aid in identification of the protein. A motif can be defined as an element or short stretch of amino acids that is linked to a specific functional or structural protein feature such as glycosylation or protein specificity. When referring to a motif that identifies protein specificity, the amino acid sequence of the motif must be unique to that specific activity. Some motifs are very short and inaccurate. A good example of these include glycosylation sites which are often only one or two residues in length and may thus occur at random on a protein sequence. Prosite uses regular expressions to search the motifs against a sequence. A regular expression is a way to match text patterns to strings and find the matches. These text patterns may include wildcards which allows for any specific character to be found at a position as well as specific combinations of characters.

A way to improve the accuracy of motifs is to use Hidden Markov Models (HMM). The Hmmer tool used in the Analysis section is a good example of HMM use. A HMM is a probabilistic model which takes into account the residues before and after the motif as well as the order of the residues in a motif. In the calculation it may incorporate a set amount of residues before or after the current position and this is referred to as the order of the HMM. Thus a 5th order HMM would consider five residues before and after the current position during a calculation as well as the order of the residues in the pattern. This implies that the pattern and position of residues in a protein sequence can be used to identify it or to generate HMM's that can be used to search for other proteins containing the same pattern. Using HMMs a model of a protein family can be built. HMM's are discussed in detail in Bystroff and Krogh, 2008 This allows programs such as Hmmer to accurately identify the family to which a protein belongs. In the Analysis module, Hmmer is used to search a sequence against the Pfam database. Pfam is a database built up of domain HMMs of every known protein family. It uses manually curated alignments of protein families to generate HMMs of the areas that can be used to identify each family. The more members in the family, the more accurate the domain HMM in Pfam.

The Tmhmm (Sonnhammer *et al.*, 1998) and S-tmhmm (Viklund and Elofsson, 2004) tools are also incorporated into the Analysis section. Tmhmm use HMMs to classify whether a protein has membrane crossing α-helices. These are recognized using HMMs based on length and hydrophobicity. A standard transmembrane helix is usually 20 residues long as this is the minimal length needed to cross a membrane while the residues are in a helical conformation. S-tmhmm uses HMMs to identify the orientation of a transmembrane helix in the membrane. It will give each residue a probability of whether it is on the inside in the cytosol or whether it faces the outside of a membrane.

Also included in the Analysis section are tools such as PROCHECK (Laskowski *et al.*, 1993) and the WHAT IF model check (Vriend, 1990). These tools use statistical data derived from the PDB to evaluate various parameters in a protein structure or model. These include parameters such as bond lengths, bond angles, planarity of atoms and packing environments of amino acids. Each of the tools will compare the results from the submitted structure to the statistical values and then judge it as either being within or outside acceptable limits. When analyzing models this is very useful as it can identify areas which were badly modelled.

Most proteins are made up of various secondary structural elements. To identify these elements, the DSSP program (Define Secondary Structures of Proteins) measures all the angles between the atoms in a protein and classify every residue as either being in a loop, β-strand or α-helix.

The Modelling section includes tools related to homology modelling and molecular dynamics simulations. Homology modelling a method whereby a structure of an unknown protein is built based on the structure of a related or homologous protein. Protein structure is much more conserved than protein sequence and this is the basis of homology modelling. Modelling programs usually take at least two parameters, a known structure and an alignment between the sequence for which the model is to be built and the sequence of the known structure. The coordinates for every region that aligns is then copied to the new target structure. Where regions don't align or where gaps or deletions are present, the program will try to build the structure based on statistical averages in combination with forcefields. After a basic model has been built, the program needs to refine the model. There are various steps and methods but the most well know is the satisfaction of spatial restraints. This method will adjust all the interactions between atoms to satisfy known restraints such as bond lengths and bond angles. An extension to this method is the modelling of the amino acid side chains. Because the side chains can rotate and have more rotational degrees of freedom, it is a more complex task to model. One of the approaches is to use a library of observed side chain conformations and model each side chain based on those conformations. This is fairly quick but does not always take the surrounding environment into account and thus some programs optimize the side chain conformation to include environmental conditions. Loop modelling presents another challenge as they are very flexible and usually lack a template. Most programs will either use a library of observed loops to try and model a loop section or, if the loop is short enough, will try *ab initio* modelling of the loop. Because of the loop flexibility, both these approaches have their drawbacks. Loop libraries do not contain all the known conformations of a loop as *ab initio* modelling of loops is in its infancy.

As structure is so conserved, this general approach is valid for the most proteins. General homology modelling theory holds that when there is 30%-50% sequence similarity, the backbone of the protein is correct, when the similarity is between 50-%70%, the side chains are also correct, and anything above 75% similarity will result in side chain specific contacts, or sometimes atoms, to be correct. Anything below 25%-30% is considered to be in the 'twilight zone'. To build models in this range requires a lot of extra knowledge about the protein which cannot be gained from structure and alignment alone.

The field of molecular dynamics encompasses the movement of proteins as simulated by an algorithm. These simulations provide valuable information regarding the interactions between amino acids in a protein. It can also be used as a guide in designing experiments to investigate the importance of amino acids in protein movement and interactions. It must be kept in mind that molecular simulations still have some limitations and it must be used as a tool to facilitate and guide experimental work. In order to to improve the simulations, much better models of the interactions between atoms and residues needs to be built. Tools to generate molecular dynamics scripts are also included in the Modelling section. Molecular dynamics is the application of Newton's Laws of Motion to a set of atoms over time to predict how they will move. With proteins the matter becomes more complex as certain atoms are bound to one another and undergoes short and long range interactions. Various algorithms and programs have been implemented to deal with these elements. The general terms in a molecular dynamics forcefield include energetic terms for the following: bond length, bond angle, dihedral angles, long range interactions, hydrogen bond interactions and Van Der Waals interactions. Each program treats these terms differently and assigns different values to each term based on empirical or calculated data. When starting a simulation, the program will try to perform an energy minimization on the protein. This is a technique whereby the algorithm tries to obtain the minimum energy for a protein by adjusting all the physical factors such as bond length, side chain orientations and atom-atom interactions. The Modelling section provides the user with a choice of programs for dynamics as well as modelling. For dynamics only scripts are provided as running simulations are very resource intensive and are not feasible on a web server. Some simulations may run for weeks at a time and thus take up valuable resources.

All of the tools mentioned will provide the user with extra information about the protein.

These programs will produce data for the user and thus FunGIMS provides data storage and act as an interface to the data. In addition to this, it also provides group-linked user management. This feature was requested by local biologists who wanted to consolidate data storage yet retain individual and group control over the data. Such a system would allow the users to store certain subsets of data on the server and retrieve it for later analysis. It also allows a separation between private and public data, which is important as some individuals may be working on projects that are of commercial value. FunGIMS allows these users to keep their data private, yet incorporate and enrich their data with public data from various sources. FunGIMS also allows for private and public data to be kept apart in that private data cannot be accessed by users who do not have the necessary access rights.

By providing the user with exploration, analysis and modelling tools in one central location, together with allowing for storage of the results, it facilitates knowledge discovery.

1.7. Application to Foot-and-Mouth Disease Virus

Foot-and-Mouth Disease Virus (FMDV) is a highly contagious disease found in cloven hoofed animals and a range of other hosts. Infections can cause large economic losses as well as a decrease in animal productivity. Local researchers at the Agricultural Research Council (ARC) have been working on a vaccine design against FMDV but have encountered numerous problems. Most of the vaccine design work is based on the capsid proteins of the virus as these are the main proteins exposed to the humoral immune system of the animal although the cellular immune system also plays a role. Due to the different serotypes found in FMDV, it is difficult to make a general vaccine. Current vaccine efforts are serotype-specific, sometimes even subtype-specific. Sequence analysis showed that there are a few sequence differences between the capsids of the various serotypes. The capsid plays a vital role in virus stability and entry into the cell and thus any capsid sequence variation might have an effect on virus spreading. Some of the problems during vaccine design were found to be related to structural aspects of the virus capsid proteins and the researchers had no means of using experiments to identify the differences in the structure. Since the researchers had no real experience in dealing with protein structure in a three dimensional environment, they required assistance and advice to use structural bioinformatics to solve urgent problems. Analysis or simulation programs were run, based on the advice given and interpretation of the results on the basis of the protein structure were provided to them. This collaboration is of vital importance as it helps them to direct experiments and interpret the results they see in the laboratory. The results have helped them to understand how variation in the capsid protein sequences affect the structure of the capsid and its effect on virus capsid stability.

The goal of FunGIMS is to provide tools for researchers in this kind of situation, to allow them to do research in an unfamiliar field while minimizing the technical difficulties hindering them. Most of the tools in the Structural module of FunGIMS, were specifically chosen to assist the researchers in conducting the most common structural bioinformatics and analysis on the proteins of the different virus strains. The functionality of the Structural module in FunGIMS was used together with other tools to aid in the investigation of three aspects of FMDV. Each problem additionally illustrates a specific feature/s of FunGIMS and its application to a specific problem. The three problems are:

- Annotation of the FMDV proteome. The FMDV genome is small and codes for fourteen proteins on a precursor polypeptide. The motif-finding tools in FunGIMS were used to find protein motifs in the proteome and compare the distribution of these motifs on 9 serotypes.
- Variation in FMDV 3C protease and 3D RNA polymerase. These two enzymes are important in the replication of FMDV and are usually highly conserved. The homology modelling tools in FunGIMS were used to build models of various SAT serotypes. The variation found in various subtypes of each of the three SAT serotypes was then compared and mapped to the protein structure to locate variation hot spots and to identify potential surface interaction areas.
- FMDV capsid stability and variation analysis. The FMDV capsid is vital to the virus as it protects the virus from the environment and assists in cell entry. It is also the main focus of vaccine design and thus understanding the interaction and differences between the various capsid proteins is highly important. The homology modelling

and molecular dynamics tools in FunGIMS were used to build models of the capsid proteins of various SAT2 subtypes. These models were used to map variation in the capsid and, in conjunction with molecular dynamics simulations, investigate the stability of the serotype capsids at differing pH values.

The three aspects investigated help to show the variety of problems that FunGIMS can be applied to and the way in which it helps to facilitate knowledge discovery in each case. A more detailed introduction about each FMDV topic is given at the start of the relevant chapter.

Problem Statement

Foot-and-Mouth Disease Virus (FMDV) is highly contagious virus infecting cloven-hoofed animals. A few key problems were identified by local researchers, all relating to structural aspects of the virus capsid proteins but they had no structural biology experience. A system called FunGIMS was designed, which attempts to help address these problems specifically in the investigation of FMDV and also to provide other researchers with an introductory environment for structural biology investigations, leading them towards the later use of more advanced tools. FunGIMS is a Functional Genomics and Information Management System. It provides an easy to use, web-based interface to perform a variety of analysis on various different data types. This project focused on providing easy access to structural data as well as intuitive and easy-to-use interfaces to the most commonly used structural bioinformatics tools.

The complexity and setup of structural biology tools have long been a barrier for biologists who want to make use of these tools. Most structural biology tools usually run on a UNIX type operating system. The vast majority of these tools has been validated extensively in literature and by their respective authors. Each program has a different syntax and method of operating, which may be frustrating to the normal biologist. By providing access to these tools via the web and by using a simple form-type input, most of the syntax and related problems are dealt with. An ideal solution would be to provide most of the tools and data via a web interface, which is a familiar environment for most users and which will help and guide users to perform independent structural biology work. Although the system makes it easier for the biologist to use the tools, the onus is still on the user to understand the function of each tool and how to interpret the results. The responsibility of tool setup and installation will be that of an experienced person such as a system administrator thereby allowing the biologist to focus on science. The system was also designed to facilitate the addition of new tools.

The integration and ease of use of the Structural module in FunGIMS is illustrated in a series of investigations performed on FMDV. The first problem is the way in which variation differs between FMDV serotypes with regard to their full proteomes. Insights into variation can help in identifying areas prone to accumulating variation. The second problem relates the variation found in two of the most conserved proteins in FMDV, 3C protease and 3D RNA polymerase. Variation hotspots in these proteins help to identify areas where interactions with other proteins occur and can help to pinpoint areas vital to enzymatic function. The third problem involves the stability of the FMDV capsid proteins under different pH levels and the way in which variability in the VP1-3 proteins affects stability. Stability of the capsid is vital for virus distribution as well as infection.

Although the tools were used on three FMDV cases, they are generically applicable to most proteins and problems related to protein structure. An integrated system such as FunGIMS, will provide access to a variety of tools as well as allow easy application of these tools to various problems related to protein structure.

Specific Aims

The aim of this project is the development of a Structural module in the FunGIMS system and its application specific problems in FMDV. The system allows a user to perform protein structural analysis in an environment with a minimal need for local client-side computing resources. The aims of this project is balanced between providing useful interfaces and tools for the user and programming a robust, extensible environment for protein analysis which can be applied to FMDV.

In Chapter 2 the development and design methodology of FunGIMS and the Structural module will be discussed. The aim was to design a system that is easy to use, easily expandable and allows the user to store and analyze data. The problem of tool incorporation into the module will also be discussed. Tools were incorporated into the system in a modular manner.

Chapters 3-5 each deals with an investigation of a specific aspect of FMDV, illustrating the role that FunGIMS was able to play in a specific problem/area of interest identified by local researchers in the study of Foot-and-Mouth Disease Virus (FMDV).

Chapter 3 describes the use of protein sequence-based tools in the Structural module of FunGIMS to annotate and identify similar patterns and functions in the FMDV proteome. This was applied to various FMDV serotypes to characterize the different proteomes and the functional relationship between them.

Chapter 4 uses homology modelling to characterize the variation seen in the highly conserved 3C protease and 3D RNA dependant RNA polymerase proteins of FMDV. The aim is to identify hotspots in the enzymes which are more or less prone to variation and which may be linked to functional and structural differences between the South African Territories (SAT) FMDV serotypes.

Chapter 5 investigates the functional and structural effect of mutations in the capsid proteins of FMDV. Capsid proteins are used in FMDV vaccine design and thus a thorough understanding of the changes found in these proteins is necessary. The homology modelling and molecular dynamics functionality of the Structural module of FunGIMS was used to investigate the effect of the various mutations on virus capsid and pH stability.