Tools for proteomic standards

(MIAPE documents)

Internship report

Masters in Proteomics and Bioinformatics

2006/2007

By: Xavier Robin

Supervisor: Christine Hoogland
# Table of Contents

Abstract ........................................................................................................................................................................... 2

1. Introduction .................................................................................................................................................................. 3

1.1 Definitions ................................................................................................................................................................. 3

1.1.1 Standard file formats ........................................................................................................................................... 3

1.1.2 Standards in biology ............................................................................................................................................... 3

1.1.3 Standards in proteomics ....................................................................................................................................... 3

1.1.4 Standards fragmentation ....................................................................................................................................... 4

1.1.5 Standards adoption and evolution ....................................................................................................................... 5

1.2 Evaluation of the standards .................................................................................................................................... 6

1.2.1 GelML OM ............................................................................................................................................................ 6

1.2.2 MIAPE .................................................................................................................................................................. 6

1.2.3 Interface between GelML OM and MIAPE ........................................................................................................ 7

1.2.4 Controlled vocabulary ......................................................................................................................................... 7

1.3 Objectives ............................................................................................................................................................... 7

2. Implementation ........................................................................................................................................................... 8

2.1 Requirements .......................................................................................................................................................... 8

2.2 Choice of the tools .................................................................................................................................................... 9

2.2.1 The URL scheme .................................................................................................................................................. 9

2.2.2 Templates ........................................................................................................................................................... 10

2.2.3 Forms ................................................................................................................................................................ 11

2.2.4 Authorisations .................................................................................................................................................... 11

2.2.5 Database model ................................................................................................................................................... 11

2.2.6 Database design ................................................................................................................................................ 11

2.2.7 Unicode .......................................................................................................................................................... 12

2.3 Construction of the application ............................................................................................................................. 12

2.3.1 Checklist definition and database design ......................................................................................................... 12

2.3.2 User management ............................................................................................................................................. 13

2.3.3 Controlled vocabulary ...................................................................................................................................... 13

2.3.4 Data saving ........................................................................................................................................................ 13

2.3.5 Example of a page ............................................................................................................................................. 13

2.3.6 Gel duplication .................................................................................................................................................. 16

2.3.7 Search .............................................................................................................................................................. 17

2.4 Update .................................................................................................................................................................. 17

2.5 Improvements ......................................................................................................................................................... 17

3. Conclusion ................................................................................................................................................................. 19

Acknowledgements ......................................................................................................................................................... 20

Bibliography ................................................................................................................................................................. 21

Annexes ........................................................................................................................................................................ 23

Annex 1. Database design ......................................................................................................................................... 23

Annex 2. Page screenshot ......................................................................................................................................... 27

Annex 3. MIAPE document ..................................................................................................................................... 28
Abstract

HUPO Proteomic Standard Initiative (PSI) defines standards for data representation in proteomics to facilitate data exchange comparison, and quality assessment. A set of minimal reporting requirements, called MIAPE (for Minimal Information About a Proteomics Experiment) is also provided to ensure consistency of datasets. MIAPEditor is a public repository and data entering tool for documents conforming to the MIAPE – Gel electrophoresis guidelines.

In MIAPEditor, documents are grouped by experiments and entered in a successive (but non linear) manner by registered users through a web-based interface. After publication by their author, documents can be viewed in HTML or plain text formats with no restriction.

Based on the perl Catalyst Model-View-Controller web application framework, MIAPEditor can be easily extended to handle other kinds of MIAPE documents and output formats.
1. Introduction

1.1 Definitions

1.1.1 Standard file formats

A standard file format is a format which specifications are made publicly available, so that all users can encode and read files in this format. The main goal of defining standards is to allow software to communicate with each other in order to process data produced by any tool without having to bother about file format conversions.

This is especially important to ensure transparency and interoperability and to facilitate integration and exchange from different sources, whatever the tool used. It finally ensures that scientists will be able to read and understand the document in the future.

1.1.2 Standards in biology

The field of DNA microarrays was the first domain of the biology to express the need for data reporting and encoding standards was. The huge amount of data produced by microarray experiments was mostly inaccessible to the research community, due to the lack of public repositories. Moreover, when data was published on authors’ web sites, the lack of annotation of this data made it impossible to evaluate the quality and reliability of the data.

In 2001, the first version of MIAME (Minimal Information about a Microarray Experiment) was published [1]. It was developed under the aegis of the MGED (Microarray Gene Expression Data) society and outlined the points that were the most important to ensure that the results would be interpretable and potentially verifiable by third parties. Apart from the MIAME document, MGED also developed an ontology (called MO) and a standard file format for the representation of microarray expression data (called MAGE, for MicroArray and Gene Expression), consisting of an object model (MAGE-OM), a document exchange format directly derived from the object model (MAGE-ML), and software toolkits in Perl and Java (MAGE-stk) to enable users to create MAGE-ML documents [2].

Several curated public repositories of microarray data do now exist, and their use is recommended by the MGED society [3]. The first one, ArrayExpress, has been developed at the EBI, together with its web-based submission tool, MIAMExpress, and an online visualisation tool, Expression Profiler [4]. GEO (Gene Expression Omnibus) [5], developed at the NCBI, and CIBEX (Center for Information Biology gene Expression database) [6], developed by DDBJ, are very similar to that. All these initiatives have emerged or evolved quickly after the publication of the MIAME specification document.

Now, nearly every high-throughput technology is in the process of standardizing their data, taking inspiration from the work of MGED. They basically all share the same three axes MGED used first: a document specifying the minimal information that must be reported together with the experimental data; controlled vocabularies to describe data and protocols; and finally standard file formats to encode the data (and metadata) [7].

1.1.3 Standards in proteomics

Like other “omics” fields, proteomics experiments produce large amounts of data. In proteomics, standards are produced by the Proteomics Standards Initiative (PSI), a working group of the Human Proteome Organisation (HUPO). PSI was started on 28-29 April 2002 in the HUPO meeting in Washington. Its official goals are to develop annotation standards for proteomics [8].

All the standards developed by PSI come in the form of three major elements (corresponding to the three axes MGED previously used). First, a MIAPE document, specifying the minimal metadata (i.e. the data about the data) elements that must be reported to allow a scientific scrutiny of a proteomic publication; it does not dictate experimenters how to perform the experiments, which in a dynamic field like proteomics is better performed by the mean of peer review, and aims to be a balance between sufficiency (they should “allow a reader to understand and critically evaluate the interpretation and conclusions, and to support their experimental corroboration”) and practicability (meaning that “achieving 'MIAPE compliance' should not be so burdensome..."
as to prohibit the widespread use of the guidelines”) [9]. Secondly, a controlled vocabulary to ensure a consistent data representation, allowing automatic analysis and comparison and searches of proteomic datasets. Finally, a data exchange format for efficient electronic transmission of the data [10].

Concrete applications that require the provision of a rich experimental description include (but are not limited to): “the discovery (for example by database search) of data sets generated by specific techniques, the sharing of successfully-employed methods, the assessment of data and analyses in the light of the methods deployed, the informed corroboration of results through the performance of parallel or orthogonal studies, and finally the promotion of intercompatibility between public repositories and software tools (i.e. encouraging the development of resources built on the assumption that specific (meta)data will be present in all descriptions that proclaim themselves to be ‘MIAPE compliant’)”. These applications, together with the practicability principle mentioned previously, are taken into account to guide the specific decisions on the (meta)data that should be required in each MIAPE module [9]. They would greatly benefit from the availability of MIAPE-compliant data sets.

Once defined, the standards need to continue to evolve to adapt to the changing requirements of the proteomics field. On the other hand, standards must be stable enough to allow involved parties to implement them appropriately. This should be achieved by the PSI by maintaining stable XML schemas (for about two years) and dynamic controlled vocabularies (updated every two weeks) [10]. MIAPE guidelines should not be released without a period of at least one year between successive versions of any MIAPE module [11].

Several journals, like Molecular and Cellular Proteomics (for protein identification [12], [13], [14] and gels [15]) or Proteomics (for MS-based experiments, analysis of 2-D PAGE, protein identification and characterization [16]), did not wait for the PSI to have stable standards and are already requiring authors to follow their own internal guidelines when reporting their data. This leads to inconsistency between the guidelines, thus preventing efficient automatic analysis of the datasets published in different journals.

PSI is divided into several working groups that each focuses on a specific subdomain of the proteomics. Table 1 shows the different groups, their projects and status.

<table>
<thead>
<tr>
<th>Working group</th>
<th>Reporting Guidelines (MIAPE)</th>
<th>Data exchange format</th>
<th>Controlled vocabulary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Molecular interactions</td>
<td>Published (MIMIx)</td>
<td>MIF (published, [17])</td>
<td>PSI-MI (published)</td>
</tr>
<tr>
<td>Mass spectrometry</td>
<td>Mass spectrometry (published)</td>
<td>mzData (published)</td>
<td>MS CV (draft)</td>
</tr>
<tr>
<td></td>
<td>MS Informatics (published)</td>
<td>analysisXML (draft)</td>
<td></td>
</tr>
<tr>
<td>Protein modifications</td>
<td>No</td>
<td>No</td>
<td>MOD</td>
</tr>
<tr>
<td>Gel electrophoresis</td>
<td>Gel electrophoresis (published)</td>
<td>GelML (draft)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gel image informatics (draft)</td>
<td>GelInfoML (draft)</td>
<td></td>
</tr>
<tr>
<td>Sample processing</td>
<td>Chromatography, Capillary electrophoresis and Sample processing (drafts)</td>
<td>spML (draft)</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Status of the different HUPO PSI working groups.

1.1.4 Standards fragmentation

Due to the lack of coordination between the different initiatives developing standards, and even between the working groups of the PSI, nearly all the standards produced so far are mutually incompatible. Each standardization organisation and working group tends to duplicate —in a necessarily incompatible way—common components that have already been developed by other groups. That means that it is not possible to encode a whole proteomics workflow in a single file, including, for example, the collection and preparation of a sample, its separation on a gel, and its subsequent analysis by mass spectrometry. Each part need to be stored in an independent file and handled separately, in a different format, making it particularly difficult to capture the links between the different parts of the workflow.

The RSBI (Reporting Structure for Biological Investigation) working group is beginning to find synergies between the different communities working with microarrays (environmental genomics, nutrigenomics and toxicogenomics). HUPO PSI and the MSI (Metabolomic Standardization Initiative) are also having a close look at it [18]. This contributed to the development of the Functional Genomics Experiment Object Model
(FuGE) which provides a model of common components in functional genomics investigations, such as materials, data, protocols, equipments and software. It can be extended to develop modular data formats and provides a consistent framework to capture complete lab workflows [19]. It is already used in PSI GelML, analysisXML and spML and the porting of mzData is planned in a future release [7]. The goal of this project is to enable future integration of all biological data within a single format, which would eventually enable the capture of a complete proteomic workflow in a single, easy-to-handle file.

Another initiative that coordinate different community standards is MIBBI (Minimum Information for Biological and Biomedical Investigations). It is a web-based resource that focuses on the minimum information (MI) checklists and aims to address the difficulty of the researchers to locate the relevant checklists in their domain, but also to coordinate the development of the checklists to avoid overlap and conflicts, and to facilitate their use in combination [20].

The third problematic point concerns ontologies. To enable efficient data retrieval and search, it is of crucial importance that the vocabulary used be perfectly controlled. Thus, the emergence of several competing controlled vocabularies is dangerous. GO (Gene Ontology) is a well-established biological ontology, which defines entities studied in biological experiments (classified under biological process, cellular component and molecular function). On an other hand, OBO (Open Biomedical Ontologies) project, which aims to bring together ontologies from different areas to develop non-redundant, well-structured controlled vocabularies for shared use across different biological and medical domains. The relationship between these two projects and whether or not they will overlap remains unclear. As a matter of fact, OBO looks more like an uncoordinated public repository for ontologies of several domains than an organisation that would develop ontologies themselves. This is shown, for example, by the fact that the controlled vocabulary for units is developed in the phenotypic domain, whereas it ought to be a project itself, or at least should be placed in a generic group, or that the ontologies are provided in different inconsistent formats (owl, protégé, plain text, obo, etc.). However, the OBO Foundry, a subset of OBO projects, may produce higher quality ontologies.

1.1.5 Standards adoption and evolution

When the standard is finally published, several issues remain that may prevent its wide adoption. First, researchers will not adopt it by themselves if the benefits they can expect from using it do not exceeds its costs. Indeed, standards are intended to make the data interpretable and useful for other researchers. Thus, journals and funding organisations will likely need to enforce standards to see them become commonly used.

Secondly, standards must evolve to follow technological innovations. This is quite contradictory with the fact that standards are most valuable when they are stable and permanent. Frequent changes will inevitably lead to incompatibilities between the software tools handling different subsets of versions. It will be a challenge for standards developers to find the right balance between stability and evolution of the standards.

And finally, tools must be available which facilitate the reporting of the data. Import/Export function of the software tools and LIMS will be of the highest importance, as will be the appearance of proteomics repositories enabling researchers to quickly publish their data [21].

Some scientists argue that standards are pointless or will prevent innovation in proteomics [22], [23]; others think that they should go further (for example, MIAPE should not merely be a guideline but rather a true standard [24]); some think that they are not well designed enough [25]. It is a fact that standards can always be improved, especially given that they are still young, if not released at all. Additionally, to the contrary of the Gene Ontology (GO) project where the heaviest users are usually the authors of the ontology themselves, those required to insert data to conform to PSI standards did not participate in defining them, and that they will not gain much from using them correctly [26].

Therefore, involving everyone in the move towards standards will be a major challenge. It will be important for standard developers to promote the advantages for the proteomics community at large. For example, the microarray repositories allowed new discoveries that could not have been made otherwise [21]. It is likely that similar proteomic repositories, will allow additional mining of the data that will lead to new discoveries. For example, PRIDE can store a whole mass spectrometry experiment, from the sample collection and preparation to the peak list and identification results, not to forget the potential separations applied [27]. But again, those who enter the data will not benefit from it.

The final success of a standard will certainly depend on the finding of the good balance between practicability and sufficiency to be suitable for everybody, and also on the ability of standard-makers to convince the research community of the usefulness of them.
1.2 Evaluation of the standards

The first point was to evaluate the GelML (at that time, version Milestone 2 release) and MIAPE Gel Electrophoresis (an unnumbered version which most certainly corresponded to the version 1.0) documents available from the PSI website [28]. None seemed to have ever entered real gel data in the GelML object model. All the sample files provided [29] where very superficial; they contained only very general information about the structure of the files, with only small subsets of the object model present in each file; no file contained a whole experiment structure. Thus, we tried to encode a SWISS-2DPAGE analytical protocol [30] using Pedro [31], a tool that enables to enter data in an XML structure using XML Schemas. This was made possible by the availability of a XML Schema version of the GelML OM.

Then, we created a MIAPE document for that same protocol. It was first entered into a plain text file, and when it was complete, we tried to encode it in GelML XML format. We will now list the problems we encountered in these different tasks.

1.2.1 GelML OM

The Gel Markup Language is based on the FuGE object model. It defines an abstract object model that makes use of UML notation and XML Schemas and can be used to describe the process of a gel electrophoresis. The complete documentation is available from the page of the Gel working group on the PSI website [28].

It makes an extensive use of ontologies to encode substances, units or values. This means that nearly each term must be encoded in the OntologyCollection part.

All protocols are split into Protocol, Application and Action. Protocols contain the basic information about what should occur in theory. That information may be in a lab book. Applications report exactly how the protocol was applied, if a duration, a concentration, a solution manufacturer or any other parameter slightly differed from the ideal protocol. Finally, Actions allow to state what was actually done and in which order. Several gels can be contained within a single GelML document.

This entails creating a lot of references and links, and the use of the Pedro software considerably simplified data import. We used a modified version of the XML Schema, which was especially simplified to ease data input in Pedro. It was provided by Andy Jones, who produced the GelML specification, as the chair of the PSI gel group and author of the FuGE specification.

We could outline several generic or specific problems in the specification. They were sent by email to Andy Jones. We will not enter into technical details here.

1.2.2 MIAPE

The MIAPE: Gel electrophoresis document (see Annex 3) covers whole gel electrophoresis experiments until the acquisition of the data (gel image), excluding the sample preparation (which should be covered by the sample processing document, only basic sample identification information are to be given). It is divided into seven parts: General features, Sample, Gel matrix and electrophoresis, Inter-dimension process, Detection, Image acquisition and Image. Each part is then subdivided into numbered items. Finally, a table gives further details about each of the item and the exact information that should be entered in it. Again, several problems in the document were outlined. Here are the most important ones:

- The item 3.2.6 made it possible to enter only gels with bisacrylamide as crosslinker. However, various other crosslinkers may be used in a polyacrylamide gel; for example, second-dimensional SDS-PAGE gels from Swiss-2DPAGE use PDA as crosslinker.
- The modelisation of the sample application (item 3.2.9) is not correct. One has to enter the quantity of sample and the volume of loading buffer that where loaded into each lane of the gel, but this is not applicable. Were sample and loading buffers mixed in different quantity measurements, for example a certain mass of sample with a volume of loading buffer, and then if only an aliquot of it was loaded, it is impossible to know the exact amount of sample and loading buffers that were loaded. This may be a bad practice, as it prevents later quantification, but it is not the role of the PSI to rule that.
- Electrophoresis protocol lacks information about material used, which might be of some importance.

The two first problems can be avoided easily by allowing users to select the crosslinker used, and by adding an “Application” input where the users can enter the volume of the mixture they actually loaded, respectively.
By the way, the last version of MIAPE (version 1.2) has already fixed the crosslinker problem the same way we did. Concerning the electrophoresis protocol, it is only a problem of lack of required information that will not prevent users from entering their data, thus we chose to leave this gap in the application and remain consistent with the guidelines.

It appears that all these issues, along with a few others, had already been addressed during the expert review process of MIAPE, but were not taken into account, except for the crosslinker one that was corrected in the paper submitted to Nature Biotechnology for publication (version 1.2 of MIAPE – Gel electrophoresis).

### 1.2.3 Interface between GelML OM and MIAPE

GelML OM was not defined with MIAPE in mind — paradoxically it was even impossible to encode the MIAPE requirements in a GelML document…

17 out of 49 MIAPE items were rising a problem one way or another. Again, we will not list them all here, but we will just mention that they are spread through the whole MIAPE document, and were of all sort, from a missing element to incompatible design and insufficient documentation making it unclear how to encode the data.

All the comments concerning GelML, MIAPE or their use in combination have been sent to Andy Jones.

### 1.2.4 Controlled vocabulary

The controlled vocabulary (abbreviated CV) for gel electrophoresis is a common ontology with sample processing called Separations CV (abbreviated SepCV) [32]. It currently only contains ontologies for gel documents, but will also be extended to those needed for sample processing. It was planned to have two separated ontologies, but due to the high overlap between them, they were finally merged.

What it will exactly contain eventually is not clear. The design principles for the controlled vocabularies [33] do not constraint what the ontologies should contain or not, but states a few principles, for example that the terms should be lower-case, containing only alphanumerical characters, and that words should be separated by spaces. Together with the already-published SepCV, it seems to suggest that this ontology will be a machine-readable ontology rather than a human-friendly one to be used, for example, in articles. Furthermore, according to Andy Jones, chairman of the PSI Gel working group, each implementation will have to maintain its own ontology concerning models (names and catalogue numbers), manufacturers and substances.

For a use in our interface and in MIAPE documents, we will prefer using more human-readable terms by capitalising some letters (for instance in the beginning of the words) or preferring to use the greek letters directly, as its users (either those entering the data or using the documents) will mostly be biologists without a computational background.

According to the PSI-CV page [34], the measurement units should refer to the units ontology by PATO (Phenotypic qualities (properties)) ontology [35]. Unfortunately, it is far from final yet and it has got lots of lacks and errors. For instance, some units are missing, misspelled, represented with wrong symbols, or even conflicts with official SI unit recommendations. Some units are redundant, which is indeed in contradiction with the design principles for PSI ontologies [33]. Finally, the unit symbols are not consistently annotated and are actually treated as mere synonyms. Anyway, the project evolves quite quickly, new versions are released frequently and the author promptly corrected most of the mistakes I outlined (often introducing new ones), so we can hope that a usable version of this ontology be available soon enough.

### 1.3 Objectives

The goal of this internship is to create a tool that will allow researchers to input data required by MIAPE in a document through a web-based interface. Documents containing all the required information will be generated in text and/or XML format and made available with no restriction. It will focus on the MIAPE Gel Electrophoresis document, which will be shortly released.

First, checklist and controlled vocabulary corresponding to the chosen MIAPE document must be defined. Then, the corresponding web interface to submit data must be written. Finally, documents must be produced in different formats (plain text, XML).
2. Implementation

The application was called MIAPEditor. It can be accessed from the following address: http://seti.isb-sib.ch:8080/.

2.1 Requirements

Like for MIAPE, the web interface must be as easy to use as possible by biologists. Our aim is to implement a tool that can be used without spending hours reading the documentation. Most of the information, for instance MIAPE requirements or additional application-specific information stating how to fill in the fields, should then be included directly in the forms so that what to enter in each field is straightforward.

Another important requirement is that it should be easy to update. MIAPE guidelines will evolve through time to fit to new experimental practices and to correct some flaws that are still in it — it is planned to have a new version of MIAPE each year [11]. Therefore, the interface should make simple fitting to the new guidelines simple while keeping old documents working. It is not easy to guess what parts of MIAPE will change and towards which direction. But at least can we try to make the application look as robust as possible so that changes can be performed as easily as possible.

The last requirement is the output in different formats, especially text and XML. Even if we are not going to implement it right now (see chapter 1.2.1 for more details), we should still keep in mind what kind of data can fit into GelML, at least for the stable enough parts of the specifications that will probably stay unmodified in the future. Regarding this, HTML and text output will be easier to adapt and will not be a problem, thus justifying our choice of these output formats.

2.2 Choice of the tools

The only constraint we had set for the tool development was that it had to be written in perl. Perl programming language offers several ways for building a web interface. The most straightforward one is the CGI method. However, for bigger applications, it has several weaknesses, for instance the code mixes application logic and display code, it is difficult to handle complicated flow controls and parts of code are often repeated. The second approach is to use one of the many web application framework existing in perl: CGI::Application (probably the first one in perl, and still widely used), AxKit (which is designed for use with XML files and running under mod_perl), Maypole, Jifty or Catalyst (which all three are MVC frameworks).

MVC stands for Model-View-Controller. The basic idea of MVC is to separate the view (the output seen by the user) and the model (the accession and modification of the data), and to link them together using controllers. All parts being separated, it is easier to read (and thus also to code), to maintain and to update the code. Furthermore, relying on a strong code base ensures regular updates of the application together with the ability to reuse code when existing.

Finally we chose the Catalyst framework. It seemed to be the most active, most powerful and most supported perl framework available. It can be used as an Apache mod_perl application or with FastCGI, but also comes with a built-in development server daemon. Normally, switching from one of these solutions to another should not require to issue any changes to the application code. It has many plugins that can extend its functionalities, although any perl module can be directly used. Catalyst has been successfully used in several large projects such as EditGrid.com (an online spreadsheet service), the Oxford English Dictionary Science Fiction Quotations Database or Handel (a commerce framework used on many websites).

The standalone development server is a script that launches a server and loads the application. The same script can be launched as mod_perl script. A helper script also exists that allows the seamlessly creation of models, views or controllers.
2.2.1 The URL scheme

In Catalyst, each controller takes the form of a perl package (.pm) and contains one or several actions that are actually subroutines. On each request, Catalyst will dispatch to the action(s) which path corresponds to the request. If it can not find anything, the default action will be executed — it typically corresponds to a 404 error page.

Each action has a defined type which states how to compute its path. A private action can be called only by forwarding from another action, when public actions can be accessed directly upon user request (therefore, only public actions do have a path). Public actions comprise several types: local actions, that will be accessible only under the specific namespace path defined for the controller (by default, it corresponds to the lowercased name of the package with :: replaced by /); global actions, that are accessible from the root of the application; both local and global ends by the exact name of the subroutine, except if a Path is specified; regex actions take a regular expression as argument and can match paths while capturing arguments; finally, Catalyst provides a method to build chains of actions, that are processed sequentially and that can also capture arguments.

Action chains are a very convenient way to handle paths in a Catalyst application. This action type allows to build paths of type /experiment/*/gel/*. Here two actions, experiment and gel, take one argument each, the experiment_id and gel_id respectively. For example, the path /experiment/3/gel/10 would refer to the gel number 10 contained in experiment 3. It would fail if the gel number 10 does not exist or belongs to another experiment. As for the experiment, it could be accessed via the path /experiment/3.

Despite its name, an action chain is not a linear chain, rather a tree structure. Several actions can branch on experiment, which means that it will be simple to add other document types in the future. Experiment is processed first, then gel. This means that we can organise our application in experiments that will contain different types of documents (here, gels). Of course, all these action types can be used together in an application, when their use is the most appropriate.

2.2.2 Templates

The templates are used to generate the HTML code that will be sent to the browser eventually. Several template engines are available for use as a Catalyst View: Template::Toolkit, HTML::Template or HTML::Mason. Template::Toolkit has a very simple and powerful syntax. It can handle perl data structures and comes with several functions to work on strings, lists and hashes to replace those of perl (which are not accessible in the template). The view comes with a helper script that creates the view file together with a set of templates and wrapper that is called automatically on each request to build the HTML layout (header and
used through the application, it was easier, and it increased readability, to define them once. The Smallint
the form with database content and population of the database with form values.

the template, along with a better control over the generated HTML. It should also enable automatic filling

of gaps of HTML::FormFu fields. All the same, the combination of HTML::FillInForm and Data::FormValidator, CGI::FormBuilder or HTML::Widget. As the input is long and difficult for the users, we need to be able to add comments to the form, to remind
users about the MIAPE guidelines, and also to add additional comments about how to fill in the fields. We also
need to group some fields (using HTML fieldset tag) for better readability. Therefore, it is preferable to import
each field individually in the template than to use the “print whole form” function that comes with
CGI::FormBuilder and HTML::Widget. The validation of user input is also a critical point. Field restrictions
must be reliable, easy to set and cover all the issues we can face, from restricting the input to a number to
more complex constraints involving several fields.

The combination of HTML::FillInForm and Data::FormValidator, and more precisely its use in Cata-
lyst, is not well documented. It looks like the HTML code should be directly written in the template, and then
the output document is automatically parsed before being sent to the browser and the default values added. If
it is the case, it seems rather inefficient and weak. Furthermore, select lists are not very easy to generate with
Template::Toolkit, as it requires quite a lot of code, so it is probably better to avoid this option.

With CGI::FormBuilder, forms are generated using large hash data structures. This allows any element,
attribute or text to be accessed individually. It is well documented. The built-in validation is actually done on
server side by Data::FormValidator, as well as on client side by automatically-generated javascript, at least
for simple one-field validations.

HTML::Widget is an object-oriented way of defining fields, constraints and filters, in a concise syntax. Un-
fortunately it allows to access only the whole form or entire input elements, which is a problem for groups of
radio buttons that are inserted in a fieldset and cannot be extracted easily (it would involve creating a custom
HTML::Element package which would duplicate the existing functionalities and go against the MVC
philosophy). HTML::Widget is poorly documented, especially concerning validation but the dedicated mailing
list is very active, which compensates for this lack. Anyway, its use is very simple and straightforward. The con-
straints, which can be set on one or several fields, are powerful and most of all, easy to extend. We can set de-
pendencies between fields in a very simple manner, using a DependOn constraint. Callback constraints allow
to check for nearly anything, using a callback subroutine. For custom constraints that are used more often,
new custom constraints can be created with only a few lines of code and can then be used like any other con-
straint.

Finally we chose to use HTML::Widget, which was also the solution explained in the Catalyst tutorial (by
the way, the original author of HTML::Widget is now highly involved in the development of Catalyst), and
seemed to be the only one to be actively developed. The fact that we did not use the CGI::FormBuilder plug-

in was fortunate indeed, because it has been now deprecated by a new controller that merges the functional-
ity of CGI::FormBuilder with Catalyst in a very nice manner, using separate configuration files to define the
fields. All the same, HTML::Widget may also become obsolete in a more or less distant future because of
HTML::FormFu, that is currently being developed by the author of HTML::Widget. It is expected to fill
the gaps of HTML::Widget that were reported above, especially by allowing to print elements independently in
the template, along with a better control over the generated HTML. It should also enable automatic filling of
the form with database content and population of the database with form values.

We had to define some new custom constraints to restrict the input: DependOnOther, DependOnValue and
Smallint. All these constraints could have been defined through callback subroutines, but as they are widely
used through the application, it was easier, and it increased readability, to define them once. The Smallint
constraint checks that the field will be accepted in a smallint column of the database and can optionally be restricted to a certain range (it is based on the Range constraint by HTML::Widget, thus accepting minimum and maximum values for the range), for example to accept positive values only. The DependonValue constraint allows to create a dependency between a field and a specific value in another one. If the field has the specified value, all the other fields are going to be required. Strangely enough, this does not seem to be possible with the default set of constraints of HTML::Widget. DependonOther is a specific DependonValue constraint that will check if the first field value is “o” (for ‘other (please precise below)’ options) and if it is, all the following fields are required. It could have been achieved either by a Callback constraint or an All (required) constraint in an if statement, but it was shorter to define a custom constraint optimized for that task, as it is widely used throughout the application.

2.2.4 Authorisations

It is preferable to impose editing restrictions. All users should be able to view published documents without needing to register. Additionally, registered users should be given the ability to modify only their own gels, not those of other users. Thus, we can not use role-based authorisations which, at the database level, deals with authorisations over a whole table, and at the application level, with specific pages. The several authorisation Catalyst plugins all seemed to handle role-based authorisations only and thus could not be used.

We use of the Catalyst::Plugin::Authentication plugin that greatly simplifies the login and logout of users. It checks on each request if the user is logged or not. Then, we have to check at the application level (in each requested controller) if the user is allowed to view the page or not. The login is remembered between the pages using sessions. A cookie containing a unique random identifier is sent to the user, and the login data is kept locally using FastMmap.

2.2.5 Database model

In a MVC application, the preferred way of interacting with the relational database model is Object-Relational Mapping (ORM). Instead of issuing SQL commands, data structures are passed to the model which will use them to update the model or fetch information from it. Even if it can be slower concerning setup, definition and execution, it is then much easier to use and allows commands to be issued faster. For example, a single method enables to find an entry if it exist or create it if it does not, and then retrieve the data (in the form of an object) for further use. With this, we do not need to check if the data already exists or not. Another important advantage is the ability to retrieve the data by walking through the tables in a very simple and convenient way, doing all necessary joins automatically.

Catalyst uses DBIx::Class for ORM by default. It is very well documented and widely used and supported by a very active mailing-list, even though the syntax for complex commands can sometimes be a little bit cryptic. Other ORM perl modules exists, for example DBIx::DataModel, created by Laurent Dami or Class::DBI, but they seem either much less complete than DBIx::Class (especially for complex commands) or less supported.

We studied the possibility of loading the database model automatically using DBIx::Class::Schema::Loader. Unfortunately we could not manage to make it load the tables, and it is said to be quite weak, as the loading could be altered by updates of the database. Additionally, it is nearly impossible to tune it finely, for example to automatically order the resultsets of some specific joins. Thus, we preferred to load the model manually using the classic DBIx::Class::Relationship way of defining relations between tables.

2.2.6 Database design

Catalyst and its tutorials are created for use with SQLite, a very simple self-containing database. Unfortunately it is not really suited to large applications: among other things, auto-incrementation of primary keys is not available (although DBIx::Class can emulate it), foreign keys are not enforced, and primary keys can be null. Thus, we chose to use a PostgreSQL database, the most commonly used in bioinformatics. Hopefully, it involves minor changes as DBIx::Class is designed in a database-independent manner (actually, one single line in the Model package needs to be changed, replacing SQLite by Pg).
2.2.7 Unicode

Catalyst was not developed with Unicode encodings in mind. In the default configuration, lots of conversion break non-ASCII characters, when redisplaying the form after a validation error, or when loading strings from the database. All pages are sent to the browser declaring an UTF-8 encoding by default, but the application need send these characters consistently, as some of them may be used for example in names or addresses of non-english users, or in samples, buffers or reagents, not even speaking about the units where the greek letter $\mu$ is especially frequent.

Hopefully, several bits of code in the template files, the database model packages, the view and the controllers, together with the Catalyst Unicode plugin, allows to correct these behaviours and have a consistent character handling.

2.3 Construction of the application

2.3.1 Checklist definition and database design

We started with defining the checklist of the data a user has to enter. This has to be done together with the design of the database. It was a subtle balance between:

- Easy data input, for the user and the developer, and easy retrieving for display.
- Minimal redundancy in the database.
- Data that can be represented in GelML OM format.

The fact that the data is entered step by step must be taken into account when defining non null constraints: most of the time, the content of an entry is not entered upon creation of the entry, but updated later on. A non null constraint would prevent entering the data.

As an example, the duration of an electrophoresis step may be represented in the database with the data type interval that stores duration with units. But GelML OM expects a couple “value-unit”, which would impose the use of a regex for conversion. Furthermore, it is quite difficult to validate such data on user input, even if PostgreSQL seems to be very liberal about the accepted formats. Thus, the duration was finally encoded as a numerical value and a reference to a time unit. For users, it means a text field in which a number is selected from a drop-down list with corresponding time units.

Illustration 2: The base organisation of the database.

The database is organised in experiments. Each experiment belongs to a user and can contain several documents (here, gel1s document). The gel table has a central role. Illustration 2 shows this basic structure with all the tables that corresponds to a MIAPE title.
2.3.2 User management

All users can view all published documents, but only the owner of the experiment can modify the documents contained in his/her experiment. This control is done at the experiment level of the action chains, and all child actions can know if the user is the owner of the experiment or not. So the pages can adapt to the context and, if necessary, show add/edit/remove actions to the owner, or refuse access to unauthorised people.

To register, a user must provide a username (that is not already used by another user), a password containing 4 to 40 alphanumerical characters and an email address. The password is verified by a second field that must be identical. All this information can be changed afterwards, provided the user enters his/her password again. As the password is stored in clear in the database, it can be easily resent to the user by email.

2.3.3 Controlled vocabulary

The controlled vocabulary for the gel was not available at the very beginning of the implementation of the application. We had to define it ourselves using the terms proposed in the MIAPE documents and those we thought may be needed.

For the units, because of the inconsistencies of the PATO ontology explained in chapter 1.2.4, we had to extract the units manually, and then edit the list by adding missing units or removing those that were obviously useless — megavolt, nanosecond, century, picometer or femtolitre will never be used in gel context — and finally add the correct symbols. It would have been better to extract them using a script, which would have made it easier to update, but a complete, rigorous, stable and reliable file would have been required.

To delay the integration of units in a XML document, we will need to use the accession numbers of the ontology. This will require to add a column in the unit table. We did not do it, because the PATO ontology is not stable enough and accession numbers are not thoroughly conserved. Adding this information later is easier than checking and modifying it afterwards.

At the end of the construction of the application, we were finally able to access a draft of the SepCV ontology [32] containing the vocabulary needed for the gel. It appeared to be only of little use, as the only terms we used from it were the image formats and the names of detection process. Other terms are either asked in free-text fields or not required in MIAPE documents (most of them define abstract concepts rather than concrete procedures).

2.3.4 Data saving

When the data is entered and validated by an authorised user, it is immediately saved into the database. The generation of the HTML output is rather slow, as it involves issuing request to nearly all tables of the database. Consequently, it should be avoided to do it each time the page is displayed. On the other hand, saving the output involves storing a few kilobytes for each gel. We chose to save the output to enable full-text search. As a result, we need to (re)generate the HTML output each time the user enters some data into the database.

2.3.5 Example of a page

We will now take the example of the MIAPE part 3.3 concerning the electrophoresis protocol to show the development process with Catalyst. It is a very simple page with few fields, but that illustrates the process quite well.

The first step is to define the data needed to fulfil the guidelines. The MIAPE item 3.3.1 requires a single buffer with components and concentrations and an optional comment stating whether it is an anode or a cathode buffer, if applicable. This can typically fit into a textarea. To spare users the effort of entering all these data for each of their gels, we will also provide a short name field, together with a selection list that will allow later them to re-use this buffer. MIAPE items 3.3.2 and 3.3.3 will require more input from the user before we can show the corresponding form. We will need to ask the user about the number of additional buffers and steps in their electrophoresis experiment. This will allow generating the pages for delaying data input with the right number of fields.

We may also cut the buffer input more finely by asking for the number of components of the buffer. Then users would have to enter each component individually, but we think that, even if the input was a bit more controlled, it would be much more complicated for users and would not be worth the cost.
The second step is to design the database for including that data. Here we need to design the MIAPE 3.3.2 and 3.3.3 items at the same time. The result is shown in Illustration 3.

Illustration 3: The tables involved in the electrophoresis protocol.

The protocol will be a table `electrophoresis_protocol` with a one to one relation to the dimension (however, the dimension entry in the dimension table is created a lot before the entry in `electrophoresis_protocol`). The buffer compositions will be stored in the table `buffer`, and referenced in the table `electrophoresis_protocol` via the table `user_buffer` that allows the association of the buffer with a specific user, enabling him/her to re-use it later in other documents. The additional buffers of item 3.3.2 will be stored in a table `additional_buffers` with a one-to-many relationship to `electrophoresis_protocol`. The number of additional buffers used in the gel needs to be kept somewhere. It will best fit in the table `electrophoresis_protocol`. The column `number_additional_buffers` of `additional_buffers` will be used to order the buffers. Like additional buffers, the running conditions will be stored in a table `electrophoresis_step` with a one-to-many relationship with `electrophoresis_protocol` and a column `step_number` for the ordering of the steps. Each step consists of a duration and an electrical parameter, with a unit in volt or ampere. Finally, we believe that the temperature, if controlled, is applied through the whole protocol and thus this information is placed in `electrophoresis_protocol`.

Now we can create the model of this part of the database. Each table is represented by a perl package (.pm) file that defines the name of the table, its columns, primary key, unique constraints, relations to other tables (including the ordering for of `additional_buffer` and `electrophoresis_step` which is set in Electro­phoresisProtocol.pm) and optionally the fields that are stored in UTF-8 in the database, if any (all the fields that can contain unicode characters, i.e., all the text fields).

The third step is the creation of the controller. We issue the following command (in the MIAPE application directory):

```
script/miape_create.pl controller Experiment::Gel::Dimension::Electrophoresis-Protocol
```

This creates the controller file using the controller helper. We can now edit it to be called when requesting the corresponding address. This is done in the subroutine declaration:

```
sub electrophoresis_protocol : PathPart('electrophoresis_protocol')
  Chained('/experiment/gel/dimension/dimension') Args(0)
```

The name of the subroutine is not important here and it could be anything, as it is overridden in the path by the modifier `PathPart`. Together with the arguments of `Args` and `Chained`, it will mean that the path to call this action will be `/experiment/*/gel/*/dimension/*/electrophoresis_protocol`, where * are the arguments for each part of the chain (the experiment and gel ids, as explained in chapter 2.2.1, and the dimension number). No other path can call this action. The argument of `Chained` is the path to an action link (that itself follows the gel action and recursively the experiment) that was set as a link using the modifier `CaptureArgs` instead of `Args`. As the chain link is defined in another controller, we have to explicitly define its namespace in addition to the name of the subroutine, but actually this is independent of the path. Due to the call of `Args` here, the action `electrophoresis_protocol` is an endpoint, which means it cannot be called by another action as a chain link.
3.3 Electrophoresis protocol

If the protocol is a MIAPE-compliant published protocol then provide a reference to the appropriate protocol(s). If no published protocol is available then record the running conditions as outlined.

The electrophoresis protocol is separated in three points:

1. Running buffer
2. Additional buffers
3. Electrophoresis conditions

The running buffer should be entered on this page directly. For additional buffers and electrophoresis conditions, please enter the respective number of each and fill them in the next pages.

### 3.3.1 Running buffer

Description of the running buffers used, in terms of name of buffer, components with concentrations. State if this is the anode or cathode buffer (if appropriate).

**Running buffer**

<table>
<thead>
<tr>
<th>Component</th>
<th>Concentration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tris-Glycine</td>
<td>125 mM - 100 mM - 0.1% w/v SDS</td>
</tr>
</tbody>
</table>

If the correct running buffer isn’t in the list, please choose “Other” in the scrolling list and precise its components and concentrations together with a short name identifying it in the text fields below.

**Other running buffer**

<table>
<thead>
<tr>
<th>Component</th>
<th>Concentration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tris-Glycine</td>
<td>125 mM - 100 mM - 0.1% w/v SDS</td>
</tr>
</tbody>
</table>

*Components and concentrations. Don’t forget to state if this is the anode or cathode buffer if appropriate.*

**Short name**

<table>
<thead>
<tr>
<th>Short name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Tris-Glycine-SDS 8.3</td>
</tr>
</tbody>
</table>

This short name will allow you to quickly re-use this loading buffer in other documents. 20 characters maximum.

### Additional buffers

**Number of additional buffers**

The number of additional buffers (other than running buffer) used in the electrophoresis.

### Running conditions

**Number of running steps**

The number of distinct steps in the running of the electrophoresis. A running step consists of a current or voltage (for a range of 10) applied during a specific duration.

For example, if the voltage was linearly increased from 300 to 3500 V during 5 hours, followed by 5 additional hours at 3500 V, whereupon the voltage was increased to 5000 V, you have exactly 5 running steps.

Save & Continue
We can now create the form elements using \$c->widget (which creates the HTML::Widget object), set the constraints and filters, and then the template. Each field is defined using a single instruction. Constraints are also defined in one instruction and can concern several fields (and even non-existing ones, which can lead to difficult-to-debug errors). We can finally filter all the free-text fields to avoid leading and trailing spaces in user inputs.

The fourth step is the creation of the template. We import each field independently with surrounding HTML formatting. Then we can set additional javascript code to enable the buffer fields only if they are required (i.e. if the user chose “other” in the drop-down list).

The fifth step is the population of the database with the submitted data. DBIx::Class provides a find_or_create function that can automatically handle the finding or creation of the correct table entry. It also returns a DBIx::Class::ResultSet object, containing the id of the newly-created entry or enabling the use of the find_or_create_related function. All these functions are used within the controller by explicitly calling the model. Eventually, the electrophoresis_protocol table entry will be updated or created using the update_or_create function. Unlike find_or_create, this function does not create a new entry if it does not find the exact data, but rather uses a unique key (we have to define which one to use exactly) to find the correct entry and then updates it to contain the specified values.

Finally, in order to fill the fields with the values the users previously entered, we fetch the data from the database and enter it as default values. This must be done before the generation of the form widget in which they will be entered using the value or selected methods of HTML::Widget. In that way, users can safely re-submit the pages without risking any data loss.

Illustration 4 shows the electrophoresis protocol page as redisplayed after data has been entered.

### 2.3.6 Gel duplication

The function that allows the duplication of a gel was requested by the users. It is especially useful for labs in which lots of very similar gels are produced with only a few (if any) properties changing each time (electrophoresis time, physicochemical range of the gel, type of detection, etc.).

To do that, we need to duplicate all entries that (recursively) references the gel entry. We end up with 21 tables in which entries are copied, shown in Illustration 5.

Illustration 5: Tables in which entries must be copied to duplicate the gel.

Every entry column is copied, except the ids and the reference columns that need to reference the newly created entries in parent tables. Gel name and description, as well as the image location, are left blank, because they are unique to a gel, and the user is asked to enter them for the new gel. Finally, the gel is marked as unpublished, in order to allow the user to make changes before its publication. This should also enable to create template gels containing all the generic data and reducing the time needed for creating documents.

The duplication involves several loops to fetch all the data. An especially tricky point is the duplication of entries in the lane table that needs newly created ids from both sample_application/loading_buffer_application and gel_matrix. This cannot be done within a single loop, so a conversion hash has been defined to convert the ids from the old lane entries to the freshly created ones.
The duplication of experiments is also implemented. It involves a loop that calls the function duplicating gels for each gel of the experiment and sets a reference to the new experiment. To delete a gel, we only need to delete the entry in the gel table and all depending entries will be deleted as well thanks to the cascade.

2.3.7 Search

The search function is very basic. It is not intended to allow a complete control over the search, but only to retrieve documents bearing a few characteristics from the database.

The full-text search searches the column html of the gel (which contains the complete HTML output code) for the keyword, using the case insensitive SQL like syntax. It returns all gel entries containing the keyword, and then also searches in the name and description of experiments. Those two parts are searched and returned independently.

Advanced search allows a bit more control over the search. One can still perform a full-text search, but restricted to specific fields, for example name and description or sample, to narrow the search to a certain type of electrophoresis, to force case sensitivity, or a mix of all.

Experiments can also be searched by name and description. However, it is not currently possible to search an experiment based on the contents of one of its document. Except for “name or description”, every added constraint adds an and condition to the where clause and restricts the search. Any space between two words will also be transformed into an and condition. It is not currently possible for the user to change this behaviour to or. Furthermore, quotes are not taken into account.

The search page also provides a way to jump directly to a gel or experiment which number is known. If no gel with the provided id is found, it is used to restrict ids to those containing this number.

2.4 Update

The exact update mechanism to be used when the MIAPE document is updated has not been chosen yet. We can already consider several ways to do it:

1. Switch all older gels to a “read-only” state. They could not be modified anymore, but still viewed. This solution is probably the easiest to implement: a single column to add in the gel table, as well as the corresponding control in the beginning of the processing of the gel pages. The drawback of this method is that gels whose edition is still in progress during the update will be made unavailable for further modification, so that users will need to restart entering data from the beginning.

2. Duplicate all the tables, controllers and the model (or, for the database and model especially, only those tables depending on the gel table) and prefixing everything with the version number. The drawbacks of this approach are the resulting redundancy, the need to change nearly all the redirections and calls to the model and controllers, and the fact that URLs will change, following the new names of the controllers. Integrity of URLs could be conserved though, using a wrapper controller that would check the version number in a column of the gel table entry and then forward to the appropriate controller version.

3. Carefully play with foreign keys and default and not null constraints of each table involved in the update in order to adapt the table to the new requirements without breaking existing documents. It would probably take more time than both previous approaches if the update of MIAPE document is important.

The best way will probably depend on the scope of the update of the document. If it is only a minor update, it may be possible to apply the third possibility without breaking anything. It will also depend on the time available to do the update, then the first solution would probably be the best one.

2.5 Improvements

The following modifications could either improve user experience or simplify future developments of the application.

- After the login, we could redirect users to the page on which they were before, rather than to the home page. Currently users may use the back button and reload the page, but it is not necessarily
straightforward. Another option would be to include the login form directly in the error page whenever an authorisation is refused.

- Set the flow control between the pages in a separate controller. Currently the controls are spread all around the application, and it can be painful to find all the redirections leading to a page if we want to change it. This is even complicated by the fact that some redirections are made at the beginning of the processing of the pages to check that all the data required has been entered, while other ones are performed at the end of the processing, and can even depend on the data entered (on the current page or a previous one).

- Hash the password in the database rather than storing the plain password. This is handled by the authentication plugin (only a change in the config file is needed) but would make it impossible for the user to recover it. Password recovery is not handled by Catalyst, and no dedicated plugin seems to exist. It would then involve the random creation of a new password by the application, sending a confirmation by email to avoid identity spoofing; the new password would be validated only once the user has confirmed it by using the information given in the email.

- Apart from the GelML output, it has been proposed to generate a “material and methods” section for articles, containing only the information usually entered in this section. It would be easy to implement, but would require a probably very complicated template to make it look like real language.

- The find_or_create function does not update fields when the data is changed by the user but rather creates a new entry. This avoids modifying data from other documents at the same time, but may create unused entries, unnecessarily increasing the size of the database. These unused entries could be removed by a trigger or a rule that would remove orphan entries.

- Javascript scripting could be improved. Currently it is applied at the end of the loading of the page, which leads to a small blinking on certain pages on slower computers. Also, even if it works, it is not done in a very clean manner, and adding it on template pages would require a large quantity of code.

- When uploading a file, if the data validation fails, the field is emptied and the user has to select his/her file again. This may be worked around by the use of a hidden field containing the data of the file previously entered.

- Allow the importation of manually-entered documents. It would require parsing non controlled files, but it would save time for users even if a small part is already entered.

- Short names (which allow users to quickly re-use content they entered previously) are ignored if the user has already defined another short name for the exact same content. The correct content is used and displayed, but only the short name that was defined first is kept. The cause of this bug is that unique constraints applies on content and user, preventing to have the same user/content pair used several times.

- A version number should be added to the gel, stating which version of the MIAPE guideline is followed by the document. This would improve the consistency for future mining of the data. It could not be added before, due to the lack of information on the MIAPE guideline document we used in the implementation.

The following perl modules may have simplified the work but were discovered too late now to take advantage of them.

- Catalyst::Controller::BindLex for an easier interaction with stash.
- Catalyst::Plugin::Auth::Utils for easier build of login and logout actions.
- Catalyst::Manual::Actions (ActionClasses) to avoid repeating generic parts of actions.
3. Conclusion

The objectives of this internship were fulfilled. MIAPEditor is now working and has been tested by some end-users of the BPRG. Some of them showed an interest for using it as a complement to ProteinScape, the LIMS used by the BPRG [36]. Unfortunately ProteinScape, being a closed-source software, cannot be extended to contain MIAPEditor functionalities. Therefore, some simpler way of interacting should be found, for example setting a link in one of the “comment” text boxes of ProteinScape.

MIAPEditor could also be used within the Make2D-DB project developed by Khaled Mostaguir [37] to help to provide more information about gel and gel informatics. Currently, no interface is available in Make2D-DB to enter data required by MIAPE guidelines.

MIAPEditor may now be released on ExPASy or PSI servers as an apache mod_perl application. We can hope it will be used by journal editors to promote the MIAPE guidelines by requiring authors to publish their data and metadata.
Acknowledgements

● Laurent Dami for sharing his knowledge about Catalyst framework.
● Khaled Mostaguir for his knowledge about databases.
● The people in the BPRG and PIG who tested the application and sent feedbacks, Jean-Charles Sanchez, Alirezah Vazzadeh and Pierre-Alain Binz.
● Frédérique Lisacek for intensive manuscript reviewing.
● All the staff of the SIB for their kindness, patience and for having borrowed all the required infrastructure.
● The members of the forum of Geckozone.org and Forum-openoffice.org for their kind help with javascript and OpenOffice.org respectively.
● And all the people I forget!
Annex 1. Database design

Illustrations 6 – 11 shows the structure of the database.
Illustration 8: Table representation for 3. Gel matrix and electrophoresis
Illustration 9: Table representation for 4. Inter-dimension Process

Illustration 10: Table representation for 5. Detection
Illustration 11: Table representation for 6. Image Acquisition and 7. Image
Annex 2. Page screenshot

Capture of a page of a gel as viewed by its owner. We can especially see the succession of the form pages to edit, here for a typical two-dimensional gel (30 pages).

View gel: Swiss-2DPAGE analytical gel

Name: Swiss-2DPAGE analytical gel
Description: This is the analytical two-dimensional gel used on Swiss-2DPAGE.

Formats

The following formats are available to view the gel:
- html
- text

Actions

Duplicate document

The duplication of a gel will create a new gel as an exact copy of the current one, except its name and description will be emptied, as well as the image. This avoids you the burden of repeating a document if only a few changes has to be made.

Duplicate the current gel

Note that changing the number of dimensions is not supported and can lead to unexpected results.

Delete the document

Delete the current gel

You will be asked to confirm this.

Edit document

The following links allow you to edit your documents.

Please use with care! Changes are irreversible (i.e., no "undo" functionality is provided).

Some changes may make the document invalid. You should do a backup of your document BEFORE trying to edit it (in order to be able to fill a new document easily, no "restore" functionality being provided either). You should also avoid "major" changes when possible, such as the number of dimensions. Even if it should normally work, no guarantee is given that it will work as expected.

- 0. Name and description
  - 1. General Features
    - 2. Sample
      - Number of samples and loading buffers
      - Samples
      - Loading buffers
    - 3. Gel matrix and electrophoresis protocol
      - 3.1 Dimension details
        - 3.2 Gel Matrix
          - Description
          - Gel manufacture
          - Physical and Physicochemical
          - Sample application
          - Sample application comment
        - 3.3 Electrophoresis protocol
          - Electrophoresis protocol
          - Electrophoresis conditions
      - 4. Inter dimension process
        - 4.1 Inter dimension step 1 details
        - 4.2 Inter dimension step 2 details
      - 3. Gel matrix and electrophoresis protocol
        - 3.1 Dimension details
          - 3.2 Gel Matrix
            - Description
            - Gel manufacture
            - Physical and Physicochemical
            - Sample application comment
          - 3.3 Electrophoresis protocol
            - Electrophoresis protocol
            - Additional buffers
            - Electrophoresis conditions
        - 5. Detection
          - 5.1 Detection details (direct or indirect)
        - 6. Image acquisition
        - 7. Image

Note that this list of pages will change over the progression of the data submission.
Annex 3. MIAPE document

Version 1.1 of the MIAPE – Gel electrophoresis document, also available for community consultation, can be downloaded from the following address: http://www.nature.com/nbt/consult/index.html

MIAPE: Gel Electrophoresis

Version 1.1, 20th November, 2006

This module identifies the minimum information required to report the use of n-dimensional gel electrophoresis in a proteomics experiment, in a manner compliant with the aims as laid out in the ‘MIAPE Principles’ document (latest version available from http://psidev.sf.net/miape/).

Introduction

Gel electrophoresis facilitates the separation of protein (or peptide) mixtures. These separations are effected in a gel matrix under the application of an electric field. Proteins with differing physical or chemical characteristics migrate at different speeds through the matrix and may become focused (i.e. cease to migrate) depending on the parameters of the gel matrix and applied electric field. Selecting particular physico-chemical properties for the matrix, chemically modifying the proteins themselves, or solubilising them with a detergent allows the separation to be further tuned. Electrophoresing a protein mixture along a single axis, on the basis of a single characteristic such as molecular weight, results in a one-dimensional separation. Higher-dimensional separations usually separate by different characteristics (for example, charge and mass) along orthogonal axes.

The requirements specification for the gel electrophoresis family of techniques is prescriptive in some respects while maintaining flexibility, allowing the description of a wide range of protocols. For a full discussion of the principles underlying this specification, please refer to the MIAPE ‘Principles’ document, which can be found on the MIAPE website (http://psidev.sf.net/miape/).

These reporting guidelines cover gel manufacture and preparation, running conditions, visualization techniques such as staining, the method of image capture and a technical description of the image obtained. They do not explicitly cover sample preparation, but do require the recording of which samples were loaded onto a gel. They do not include spot detection or other analyses of gel images, nor do they include protein identification procedures. Items falling outside the scope of this module may be captured in complementary modules, which can be obtained from the MIAPE website. Note that subsequent versions of this document may evolve over time as will almost certainly be the case for all the MIAPE modules.

The following section, detailing the reporting requirements for the use of gel electrophoresis, is subdivided as follows:

1. General features
2. Sample
3. Gel matrix and electrophoresis
4. Inter-dimension process
5. Detection
6. Image acquisition
7. Image

The glossary provides the list and definitions of required items in the MIAPE: Gel Electrophoresis guidelines. Examples are given to facilitate interpretation and are not provided as a comprehensive list of the technologies that can or can not be recorded under each section heading.
Reporting requirements for gel electrophoresis

1. General features
   1.1.1 Date stamp (as yyyy-mm-dd)
   1.1.2 Responsible person or institutional role
   1.1.3 Electrophoresis type

2. Sample
   2.1.1 Sample name(s)
   2.1.2 Loading buffer

3. Gel matrix and electrophoresis
   There should be one description for each dimension. Each dimension must have a description for sections 3.1, 3.2 and 3.3. If a gel is composed from two or more matrices (e.g. stacking gel and resolving gel), each matrix must have a section 3.2 description.
   3.1. Dimension details
      3.1.1 Ordinal number for this dimension
      3.1.2 Separation method employed
   3.2 Gel matrix
      3.2.1 Description of gel matrix
      3.2.2 Gel manufacturer
      3.2.3 Physical dimensions
      3.2.4 The physicochemical property range and distribution (as appropriate)
      3.2.5 Acrylamide concentration
      3.2.6 Acrylamide : Crosslinker ratio
      3.2.7 Additional substances in gel
      3.2.8 Gel lane
      3.2.9 Sample application
   3.3 Protocol
      3.3.1 Running buffer
      3.3.2 Additional buffers
      3.3.3 Electrophoresis conditions

4. Inter-dimension process (not applicable for one-dimensional gel electrophoresis)
   This section is used to record any process or processes applied to, or carried out between the dimensions described in section 3. Each inter-dimension process applied must have a description for section 3.1 and 3.2.
   4.1 Inter-dimension process
      4.1.1 Step name
      4.1.2 Inter-Dimension buffer
      4.1.3 Additional reagents
      4.1.4 Equipment
      4.1.5 Protocol

5. Detection (if applicable)
   If detection is carried out there should be a description for the appropriate direct (5.1) or indirect (5.2) detection method. Each detection process must have a protocol description (5.3).
   5.1 Direct detection
      5.1.1 Name of direct detection process
      5.1.2 Direct detection agents
      5.1.3 Additional reagents and buffers
      5.1.4 Equipment
      5.1.5 Direct detection protocol
5.2 Indirect detection

5.2.1 Name of indirect detection process
5.2.2 Transfer medium
5.2.3 Detection medium
5.2.4 Indirect detection agents
5.2.5 Additional reagents and buffers
5.2.6 Equipment
5.2.7 Indirect detection protocol

6. Image acquisition (if applicable)
6.1 Acquisition equipment

6.1.1 Type of equipment
6.1.2 Name of equipment
6.1.3 Software
6.1.4 Calibration (if appropriate)
6.1.5 Equipment specific parameters

6.2 Acquisition protocol

6.2.1 Image acquisition process
6.2.2 Reference to gel matrix

7. Image (as a result of section 6)

7.1.1 Image name (or id)
7.1.2 Dimensions
7.1.3 Resolution
7.1.4 Bit depth
7.1.5 Image location
7.1.6 Standard image orientation

Summary

The MIAPE: Gel Electrophoresis minimum reporting requirements for the use of \( n \)-dimensional gel electrophoresis specify that a significant degree of detail be captured about the gel, running conditions, visualization procedures and the acquisition of raw image data. However, it is clear that providing the information required by this document will enable the effective interpretation and assessment of gel electrophoresis data and metadata and potentially, support experimental corroboration. Much of the information required herein may already be stored in an electronic format, or exportable from instrumentation; we anticipate further automation of this process.

These guidelines will evolve. To contribute, or to track the process to remain ‘MIAPE compliant’, browse to the website at http://psidev.sf.net/miape

Appendix One. The MIAPE: Gel Electrophoresis glossary of required items.
<table>
<thead>
<tr>
<th>Classification</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. General features</td>
<td></td>
</tr>
<tr>
<td>1.1.1 Date stamp</td>
<td>The date on which the work described was initiated; given in the standard ‘YYYY-MM-DD’ format (with hyphens).</td>
</tr>
<tr>
<td>1.1.2 Responsible person or role</td>
<td>The (stable) primary contact person for this data set; this could be the experimenter, lab head, line manager etc.. Where responsibility rests with an institutional role (e.g. one of a number of duty officers) rather than a person, give the official name of the role rather than any one person. In all cases give affiliation and stable contact information, which consists of (i) Name, (ii) Postal address and (iii) Email address.</td>
</tr>
<tr>
<td>1.1.3 Electrophoresis type</td>
<td>The gel electrophoresis type; such as two-dimensional, one-dimensional, difference gel electrophoresis (DIGE).</td>
</tr>
<tr>
<td>2. Sample - The sample preparation should be documented in a MIAPE compliant manner and referenced from here.</td>
<td></td>
</tr>
<tr>
<td>2.1.1 Sample name(s)</td>
<td>Name of sample(s) including any label, marker or tag applied that will be used for protein detection, such as radiolabels or fluorescent labels (by name only). From the sample described above identify control, standard and test samples.</td>
</tr>
<tr>
<td>2.1.2 Loading buffer</td>
<td>The components, with concentrations (excluding the sample) of the loading buffer that is to be loaded onto the gel matrix. In case of more than one loading buffer for the same dimension, describe one loading buffer for each sample.</td>
</tr>
<tr>
<td>3. Gel matrix and electrophoresis protocol — 3.1 Dimension details</td>
<td></td>
</tr>
<tr>
<td>3.1.1 Ordinal number for this dimension</td>
<td>If this is a one-dimensional gel then the dimension is “First”. For a two-dimensional gel there will be descriptions for the first dimension gel matrix “First” and the second dimension gel matrix “Second”.</td>
</tr>
<tr>
<td>3.1.2 Separation method employed</td>
<td>Name of the separation technique employed for this particular dimension; e.g. isoelectric focusing (IEF), Sodium dodecyl sulphate polyacrylamide gel electrophoresis (SDS-PAGE), Non-equilibrium pH gradient electrophoresis (NEpHGE).</td>
</tr>
<tr>
<td>3. Gel matrix and electrophoresis — 3.2 Gel matrix</td>
<td></td>
</tr>
<tr>
<td>3.2.1 Description of gel matrix</td>
<td>Gel matrix being used for this dimension. Include the descriptive name of the matrix (e.g. IPG strip, slab gel) and the type of the matrix used for this dimension (e.g. a native gel, denaturing gel, gradient gel, etc.). State whether the matrix is composed of more than one kind of gel and name the parts (e.g. stacking gel). Give the dimensions of the matrix and associated parts, under the physical dimension section, below.</td>
</tr>
<tr>
<td>3.2.2. Gel manufacture</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>If the gel was purchased pre-cast, then include the model name, model number, batch number and manufacturer. If the gel has been manufactured 'in house' then a reference to published protocol should be given. If no published protocol is available a recipe should be given.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.3 Physical dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>The physical dimensions of the gel matrix and of any sub-matrices described in section 3.2.1. The measurements must be in the form of the Cartesian Coordinate system (x,y,z). According to the standard image orientation described in section 7.1.6, x represents the distance from the anode (+) to the cathode (−). (For example in an IPG strip x = the strip length, for a standard slab gel, x = the width). z = the matrix depth.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.4 Physicochemical property range and distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>As applicable, the details of the pH distribution of the matrix, including the overall pH range of the gel, if known. Details of the molecular weight distribution of matrix with appropriate measurement unit. Examples include linear pH 4-7, logarithmic apparent molecular mass 200-10 kDa.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.5 Acrylamide concentration</th>
</tr>
</thead>
<tbody>
<tr>
<td>The acrylamide concentration of the gel, or each matrix as described in section 3.2.1. In the form of a single percentage (1%) or gradient (1-2%). For gradients include the gradient distribution if appropriate, (e.g. fixed, stepped)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.6 Acrylamide : Crosslinker ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>The acrylamide to crosslinker ratio of the gel, or parts described in 3.2.1 (Name of crosslinker and ratio described as “acrylamide : crosslinker”)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.7 Additional substances in gel</th>
</tr>
</thead>
<tbody>
<tr>
<td>The details of any additional substances or reagents contained within the gel (not given in section 3.2.2), recording the concentration or molarities for each (and pH if appropriate).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.8 Gel lane</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of lanes on the gel matrix.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3.2.9 Sample application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description of the sample (referenced from section 2.1.1) as applied to the matrix, giving: 1. Name of sample (reference 2.1.1) 2. Volume of loading buffer (reference 2.1.2) 3. Sample loaded per lane (if applicable for quantification, in SI measurement) 4. Lane designation (as defined in section 3.2.8) Include a description of how the sample was applied to the matrix; e.g. well loading, rehydration loading, cup loading etc. If the sample is applied using the cup-loading method also state the application point as either anode or cathode.</td>
</tr>
</tbody>
</table>

3. Gel matrix and electrophoresis — 3.3 Protocol. If the protocol is a MIAPE compliant published protocol then provide a reference to the appropriate protocol(s). If no published protocol is available then record the running conditions as outlined

<table>
<thead>
<tr>
<th>3.3.1 Running buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description of the running buffers used, in terms of name of buffer, components with concentrations. State if this is the anode or cathode buffer (if appropriate).</td>
</tr>
</tbody>
</table>
### 3.3.2 Additional buffers

Additional buffers used other than the running buffer described in section 3.3.1. Described in terms of Name of buffer, plus components with concentrations.

### 3.3.3 Electrophoresis conditions

The running conditions applied to the gel described in 3.2. To be given in terms of voltages versus time/kilovolt hours, (or appropriate measurements) and temperature. Include voltage mode profile (step and hold, or gradient).

### 4. Inter-dimension Process — 4.1 Protocol (not applicable for one-dimensional gel electrophoresis). This section is used to record any process or processes applied to, or carried out between the dimensions described in section 3. Each inter-dimension process applied must have a description for section 3.1 and 3.2. This includes processes such as equilibration, reduction and alkylation. If the protocol is MIAPE compliant and published then provide a reference to the appropriate protocol(s) in the standard manner. If no published protocol is available then record the running conditions as outlined.

#### 4.1.1 Step name

A descriptive name of the steps involved in the inter-dimension process. For example, equilibration, or reduction and alkylation.

#### 4.1.2 Inter-dimension buffer

The details of the buffer should be recorded with name, components and concentrations.

#### 4.1.3 Additional reagents

Any additional reagents used should be recorded with name, components, and concentrations. For example, reduction and alkylation agents.

#### 4.1.4 Equipment

Record the Model Name and Model Number and Manufacturer for specialised equipment (note that equipment such as glassware and shakers should not be included unless deemed integral to the result).

#### 4.1.5 Protocol

For the steps named above including, duration and temperature, if appropriate.

### 5. Detection (if applicable) — this section documents the process and the methods employed to allow the proteins which have been separated along the matrix to be detected. This can involve such process as; staining the proteins on the gel (direct detection), exposing the gel which contains radiolabel sample to photographic film, or transfer of proteins to a matrix such as in immunoblotting (indirect detection). If detection is carried out there should be a description for the appropriate direct 5.1 or indirect detection method 5.2.

#### 5. Detection — 5.1 Direct detection

#### 5.1.1 Name of direct detection process

The detection process applied to the gel; for example, silver staining or Coomassie staining.

#### 5.1.2 Direct detection agents

The detection agents employed for the described process above. Such as the type of stain employed or the type of radiolabel. Give model and manufacturer, if appropriate, else describe origin. If any antibodies are used also describe them in terms of polyclonal, monoclonal, species, target and specificity. Identify antibodies as primary/secondary etc.. The amount of detection agent employed during the detection protocol. In terms of volume and concentrations (if appropriate).

#### 5.1.3 Additional reagents and buffers

Additional reagents used other than the detection agent described above; components with concentrations.
<table>
<thead>
<tr>
<th>5.1.4 Equipment</th>
<th>Record the Model Name and Model Number and Manufacturer for specialised equipment (note that equipment such as glassware and shakers should not be included unless deemed integral to the result).</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1.5 Direct detection protocol</td>
<td>Outline of the direct detection protocol, including, duration and temperature, if appropriate. If the protocol is a MIAPE compliant and published protocol then provide a reference the appropriate protocol(s).</td>
</tr>
<tr>
<td><strong>5. Detection — 5.2 Indirect detection</strong></td>
<td></td>
</tr>
<tr>
<td>5.2.1 Name of indirect detection process</td>
<td>The indirect detection process applied to the gel. Such as immunoblot, exposure to photographic film.</td>
</tr>
<tr>
<td>5.2.2 Transfer medium</td>
<td>If the proteins separated along the gel, are transferred to another medium to aid detection (such as a nitrocellulose membrane as used in immunoblots) then give details of the name and type of medium, models and manufacturers, where appropriate.</td>
</tr>
<tr>
<td>5.2.3 Detection medium</td>
<td>If the proteins on the gel matrix must be exposed to another medium to permit detection (e.g. photographic film for radiolabel samples) then give details of name, model, manufacture, of detection medium.</td>
</tr>
<tr>
<td>5.2.4 Indirect detection agents</td>
<td>The indirection detection agents employed for the described process above. Such as the type of stain employed or the type of radiolabel. Give model and manufacturer, if appropriate, else describe origin. If any antibodies are used also describe them in terms of polyclonal, monoclonal, species, target and specificity. Identify antibodies as primary/secondary etc.. The amount of detection agent employed during the detection protocol. In terms of volume and concentrations (if appropriate).</td>
</tr>
<tr>
<td>5.2.5 Additional reagents and buffers</td>
<td>Additional reagents used other than the detection agent described above; components with concentrations.</td>
</tr>
<tr>
<td>5.2.6 Equipment</td>
<td>Record the Model Name and Model Number and Manufacturer for specialised equipment (note that equipment such as glassware and shakers should not be included unless deemed integral to the result).</td>
</tr>
<tr>
<td>5.2.7 Indirect detection protocol</td>
<td>Outline of the indirect detection protocol, including, duration and temperature, if appropriate. If the protocol is a MIAPE compliant and published protocol then provide a reference the appropriate protocol(s).</td>
</tr>
<tr>
<td><strong>6. Image Acquisition — 6.1 Acquisition Equipment</strong></td>
<td></td>
</tr>
<tr>
<td>6.1.1 Type of equipment</td>
<td>The type of equipment used in image acquisition, (e.g. laser scanner, camera, fluorescent scanner)</td>
</tr>
<tr>
<td>6.1.2 Name of equipment</td>
<td>Image acquisition equipment. Give model name and manufacturer.</td>
</tr>
<tr>
<td>6.1.3 Software</td>
<td>The software used in the acquisition process such as scanning software. Give manufacturer, name and version.</td>
</tr>
<tr>
<td>6.1.4 Calibration</td>
<td>Is the equipment calibration Automatic? Value of “Yes” or “No”. If “No” describe the calibration processes applied to the image acquisition equipment in section 6.1.5</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>6.1.5 Equipment specific parameters</td>
<td>Any equipment settings that are necessary to reproduce measurements, such as manual calibration or photomultiplier voltages on imagers, filters used. Include files if appropriate.</td>
</tr>
</tbody>
</table>

**Image Acquisition — 6.2 Acquisition Protocol**

<table>
<thead>
<tr>
<th>6.2.1 Image acquisition process</th>
<th>The protocol for image acquisition; e.g. exposure times, laser types, wavelengths. If any pre-processing of scanner equipment other than calibration is performed such as noise filters, give details.</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.2 Reference to gel matrix</td>
<td>A reference to section 3 to indicate which gel matrix is being used for the image acquisition process</td>
</tr>
</tbody>
</table>

**7. Image (The raw image file resulting from section 6, no post-cropped images should be included)**

<table>
<thead>
<tr>
<th>7.1.1 Image name (or id)</th>
<th>The name or identifier for the image file; include image format (e.g. tiff).</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1.2 Dimensions</td>
<td>The x, y dimensions of the image (in pixels for a digitized image).</td>
</tr>
<tr>
<td>7.1.3 Resolution</td>
<td>The resolution of the digitized image (e.g. micrometers per pixel [µm/pixel]).</td>
</tr>
<tr>
<td>7.1.4 Bit-depth</td>
<td>The bit-depth of the digitized image (e.g. 8bit, 16bit).</td>
</tr>
<tr>
<td>7.1.5 Image Location</td>
<td>The image file location should be made available when the experiment is published, for example, using a Uniform Resource Identifier (URI) or a Digital Object Identifier (DOI).</td>
</tr>
<tr>
<td>7.1.6 Standard image orientation</td>
<td>The orientation of all gels MUST be high molecular weight at the top, low at the bottom. For gels with lanes, the lane numbers MUST increase from left to right. For other gels the lowest value of the physicochemical distribution MUST be on the left with the highest value to the right. For all other gel matrices the anode (+) should be on the left hand side and the cathode (−) to the right.</td>
</tr>
</tbody>
</table>