A systematic approach to modeling, capturing, and disseminating proteomics experimental data

Chris F. Taylor[1,2], Norman W. Paton[2], Kevin L. Garwood[2], Paul D. Kirby[1,2], David A. Stead[3], Zhikang Yin[3], Eric W. Deutsch[4], Laura Selway[3], Janet Walker[3], Isabel Riba-Garcia[5], Shabaz Mohammed[5], Michael J. Deery[7], Julie A. Howard[8], Tom Dunkley[8], Ruedi Aebersold[4], Douglas B. Kell[5], Kathryn S. Lilley[8], Peter Roepstorff[9], John R. Yates III[10], Andy Brass[1,2], Alistair J.P. Brown[3], Phil Cash[3], Simon J. Gaskell[5], Simon J. Hubbard[6], and Stephen G. Oliver[1]*

Both the generation and the analysis of proteome data are becoming increasingly widespread, and the field of proteomics is moving incrementally toward high-throughput approaches. Techniques are also increasing in complexity as the relevant technologies evolve. A standard representation of both the methods used and the data generated in proteomics experiments, analogous to that of the MIAME (minimum information about a microarray experiment) guidelines for transcriptomics, and the associated MAGE (microarray gene expression) object model and XML (extensible markup language) implementation, has yet to emerge. This hinders the handling, exchange, and dissemination of proteomics data. Here, we present a UML (unified modeling language) approach to proteomics experimental data, describe XML and SQL (structured query language) implementations of that model, and discuss capture, storage, and dissemination strategies. These make explicit what data might be most usefully captured about proteomics experiments and provide complementary routes toward the implementation of a proteome repository.

The burgeoning of the various gene and genome sequence databases is well documented. In recent years, we have also witnessed an increasing interest in functional genomics. Now proteomics, the study of the protein complement of a cell, has begun to mature with the development of high-throughput proteome analysis pipelines. These pipelines roughly consist of physical separation of samples by gel electrophoresis, size-exclusion and/or affinity chromatography, followed by mass spectrometric examination of separations and protein identification by bioinformatic analysis[1,2] (examples of the kinds of data produced are shown in Fig. 1).

The representation of gene and genome sequence data is fairly well standardized, and the databases and tools for their analysis are widely used[3]. However, the situation is less developed for transcriptome, and especially proteome, data. This is predominantly because both fields are young and rapidly evolving; their dynamism makes it difficult to define the key data in a set of results. Both fields also produce data that are only meaningful in context[4]. For example, there are many different subsets of the total proteome of an organism, just as there are many different (and not necessarily correlated) patterns of transcription, distinguished by cell type and condition. This necessitates a more complex set of metadata (data about the data) than is required for gene sequences, where, usually, knowing the organism of origin will suffice. For example, it is not possible to reliably compare images of two-dimensional gels without knowledge of their mass and charge ranges, or database searches where the database version is not known. In addition, data and metadata produced in different places may be in different formats, making comparison and exchange difficult.

There are many two-dimensional gel image repositories on the web (http://ca.expasy.org/), with SWISS–2DPAGE[5,6] being one of the most developed in terms of the search tools available at the site. In addition, several sites allow the analysis of the results of mass spectrometry (http://ca.expasy.org/tools/-proteome), although few actual mass spectrometry data sets are publicly available. Many of the gel databases allow the user to click on some of the spots on a gel image to obtain the appropriate entry in one of the major sequence databases, which usually means a paper citation of some sort. However, this is not a reliable way to allow rigorous comparisons to be made between data sets. The use of a paper citation as a proxy for the actual metadata is an unnecessary hindrance to users who want all the relevant information at their fingertips, to quickly assess the value of a data set or to perform a nonstandard search (for example, by sample extraction technique). The requirements of the various journals will also differ, so the necessary information may be lacking in some cases, or presented in ways that are difficult to relate.

Thus, there exists a need for public repositories that contain details of whole proteomics experiments, as opposed to the gel image databases that exist currently, which offer little readily accessible information about where samples came from or how, and by whom, spots from a gel were analyzed. Therefore, it is appropriate to attempt to define the minimum set of information about a proteomics experiment that would be required for such a repository.

[1]*School of Biological Sciences and* [2]*Department of Computer Science, University of Manchester, Oxford Road, Manchester M13 9PL, UK.* [3]*Department of Molecular & Cell Biology, Institute of Medical Science, University of Aberdeen, Aberdeen AB25 2ZF, UK.* [4]*Institute for Systems Biology, 1441 N 34th St., Seattle, Washington 98103. Departments of* [5]*Chemistry and* [6]*Biomolecular Sciences, UMIST, PO Box 88, Manchester M60 1QD, UK.* [7]*Inpharmatica Ltd, 60 Charlotte Street, London, UK.* [8]*Department of Biochemistry, University of Cambridge, Building O, Downing Site, Cambridge CB2 1QW, UK.* [9]*Department of Biochemistry & Molecular Biology, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark.* [10]*Department of Cell Biology, Scripps Clinic & Research Institute, La Jolla, California 92037. *Corresponding author (steve.oliver@man.ac.uk).*

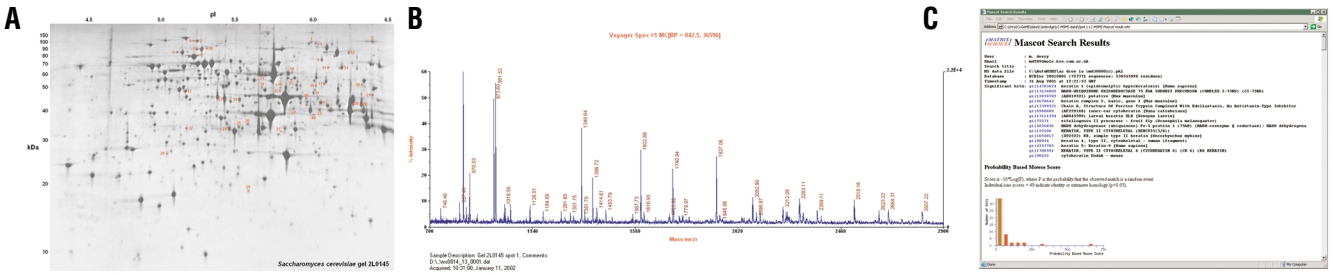

Figure 1. Examples of the types of data generated by proteomics experiments. (A) An annotated two-dimensional gel. (B) A mass spectrum. (C) A protein identification search over a peptide fragment database.

Such a definition can facilitate information exchange, because users (be they end users or tool developers) know what to expect from a data set (that is, what information is present and in what format). It also facilitates the development of effective search tools, by forestalling the disparities that would arise between different repositories, in terms of the data and metadata that would be captured, were they developed independently.

Recently, some journals have begun to require that papers reporting transcriptome experiment results be accompanied by the MIAME[7]-defined minimum set of information about the microarray experiment (http://www.mged.org/Workgroups/MIAME/miame.html) as a standard part of the publication process. An equivalent requirement for the publication of proteomics data sets is clearly desirable. Of course, the precise level of detail required is open to debate. Most mass spectrometers have a long list of machine settings, many of which are unlikely to be of subsequent interest; this is also true for gel electrophoresis and liquid chromatography. Individual pieces of information about the source organism, the sample preparation techniques, and the database version and search parameters used in the protein identification process must also be classified as important or otherwise.

There is currently no such definition of the minimum set of information about a proteomics experiment that would be required by an 'ideal' repository. Examples of standards already in place for the protein structure and microarray communities can be found at the European Bioinformatics Institute's (Hinxton, UK) database submissions page (http://www.ebi.ac.uk/Submissions/index.html). Efforts have also been initiated to develop comparable standards for the metabolomics community. So, there is an urgent need for a definition of the minimum set of information required about a proteomics experiment.

The data model described here is offered as the starting point for a discussion about the level of detail deemed sufficient in such a definition. In deciding what should be included in a proteomics experiment data repository, two general criteria were used:

The repository should contain sufficient information to allow users to recreate any of the experiments whose results are stored within it.

The information stored should be organized in a manner reflecting the structure of the experimental procedures that generated it.

In the next section, we describe the UML (universal modeling language[8]) approach for the Proteomics Experiment Data Repository (or PEDRo; Fig. 2). The model describes, in an implementation-independent manner, the data that are required to be captured from a proteomics experiment (both results and metadata). The attributes for the fields in each class (including data types and which fields are compulsory) are described in the corresponding relational database definition in Appendix 1 (see author's website; URL at end of Discussion). The requirements for PEDRo resemble those of a minimal Laboratory Information Management System (LIMS) in that, in addition to the results themselves, they capture much of the information that would normally only be kept in the laboratory in which the data were produced, such as who performed the work, what hypothesis drove it, and so on.

As the PEDRo model is independent of any particular implementation, several implementation structures can be derived from it for use in different settings. For example, in a later section we describe a PEDRo-compliant, Java-based data entry tool currently under development at Manchester University (Manchester, UK). This tool collates the data and metadata from an experiment into a single XML file (http://www.w3.org/XML/Schema) for submission to a PEDRo-compliant repository.

Appendix 2 (see author's website) details an XML Schema representation of the PEDRo model (PEML, the proteomics experiment markup language, is essentially a set of predefined tags with which to structure, or 'mark up', the raw data in a file) for use as a data interchange format. Incidentally, the use of XML with XSLT (extensible stylesheet language transformations; a styling language that can, among other things, transform XML-encoded data into HTML (hypertext markup language)) offers a simple route for dissemination of information across the world-wide web. Appendix 1 details a relational database schema (the PEDRo itself) in the form of 'Create Table' statements in SQL. Both appendices feature brief discussions of the issues that arose in making the mapping from the UML approach to each of these two implementations.

The PEDRo schema

The PEDRo is intended to capture all the relevant information from any proteomics experiment, such as details of the experimenter, the sample source, the methods and equipment employed, and (of course) any results and analyses. This could be, for example, data from a laboratory performing matrix-assisted laser desorption/ionization (MALDI) mass spectrometry on spots of interest from comparative two-dimensional gel experiments, or from a high-throughput screening facility using multi-dimensional liquid chromatography fed directly into a tandem mass spectrometer. In addition to our stated requirements of sufficiency and natural organization, we endeavored to produce a model with a degree of flexibility. Proteomics technology is still rapidly evolving, and the repository should anticipate, and eventually accommodate, proteomics data generated by novel experimental approaches, such as protein chips.

We have also attempted to limit the proportion of the model for which users are compelled to provide data, without compromising the integrity of the resultant data set (for example, the model requires that details of a spot analysis be accompanied by information about the 'parent' gel, and about the sample that was run on that gel). Note that the PEDRo model only occasionally uses user-definable fields (linked to ontologies), in contrast to MAGE (ref. 9; http://www.mged.org/Workgroups/MAGE/mage.html) and some software packages (http://www.expasy.org/melanie/Melanie_descrip-

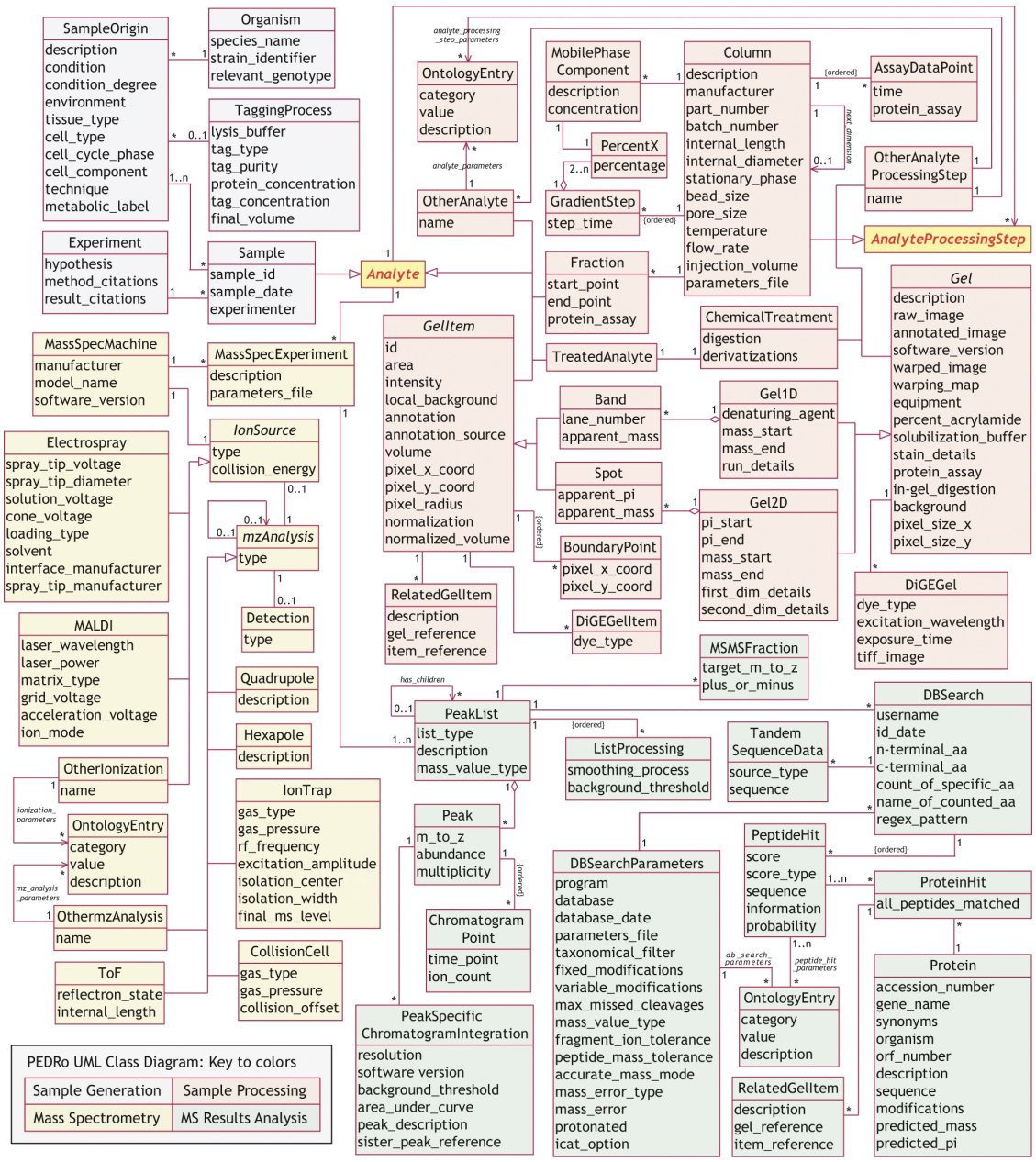

Figure 2. The PEDRo UML class diagram provides a conceptual model of proteomics experiment data, which form the basis for the XML and relational schemas. Colors denote sample generation (blue), sample processing (pink), mass spectrometry (yellow), and *in silico* MS results analysis (green). The names of abstract classes appear in italics. All attributes (including which fields are compulsory) are described in the relational database definition in Appendix 1 (Supplementary Information).

tion.htm - Annotation). This is partly because the scope of the model is reasonably well defined, and partly to act as a spur for discussion about what data should actually be captured by the PEDRo model, and in how much detail. The relationship between PEDRo and MAGE is further discussed below.

UML is the industry-standard, object-oriented modeling language. In this context, it allows us to describe experimental methods, results, and subsequent analyses in an implementation-independent manner. UML schemas, like that in Fig. 2, are referred to as class diagrams. They consist of boxes (classes), representing important entity types, connected by various types of lines and arrows signifying the relationships between them. For a brief tutorial, explaining some of the conventions of UML schemas, see "A brief UML tutorial". Figure 2 shows the UML schema for PEDRo. The schema falls fairly naturally into four sections (color-coded blue, pink, yellow, and green), each of which will now be described in turn.

Sample generation (blue). In the upper left quadrant of the schema (Fig. 2), there are five classes associated with the generation of a sample. 'Sample' itself simply holds an identification code (probably laboratory-specific), the production date, and the name of the responsible person. 'Experiment' captures the rationale behind the work being described, as a 'free text' hypothesis, and whatever descriptions of methods ('methods_citations') and results ('results_citations') may be appropriate. Note that the multiplicities show that one experimental 'hypothesis' may drive the generation of many 'Samples'.

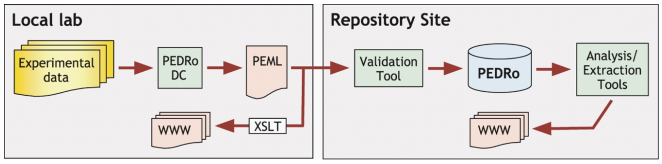

Figure 3. One possible flow of information from and about a proteomics experiment, from generation and encoding in a local lab, to storage and dissemination at the repository site.

'Sample Origin' holds basic information, such as the specific biological material used, what tissue or subcellular fraction was studied (if appropriate), and the experimental conditions to which the organism was subject. 'Sample Origin' also has two offspring (that is, classes dependent on a 'parent'): 'Organism' holds the name of the species/strain used and a list of the relevant genes/mutations carried (in 'relevant_genotype'); and 'Tagging Process' describes the labeling of the parts of a combined sample for differential expression studies, such as difference gel electrophoresis (DiGE)[10] or isotope-coded affinity tag (ICAT) mass spectrometry[11].

It should be noted that the multiplicity indicators ensure that a 'Sample' can only be associated with one 'Experiment', whereas a single 'Experiment' can generate many 'Samples'. For example, these could be the replicate analyses required to highlight statistically significant changes in the proteome in response to a particular condition. A 'Sample' can also be associated with one or more than one 'Sample Origin'—the former case (one-to-one) represents a classical proteomics investigation, and the latter (many–to–one) captures the process of combining differently tagged extracts into one sample, as in a differential expression study.

Sample processing (pink). To the right of 'Sample' in the schema is *Analyte*—an abstract class (thus, in the class diagram, its name is in italics). *Analyte* is effectively a placeholder, to be replaced by one of its subclasses ('Sample' initially, then either 'Fraction', 'Band', 'Spot', 'Treated Analyte', or 'Other Analyte'). An *Analyte* can immediately be used as the source for a 'Mass Spec Experiment' (described in the next section), or it can be put through one or more *Analyte Processing Steps*, represented by the cycle in the top right quadrant of the schema. This cyclical design enables a complex series of catenated processes to be easily described, because the result of one *Analyte Processing Step* can be fed back into the cycle as the next *Analyte* (for example, running a two-dimensional gel with a 'Sample', then putting a 'Spot' from that 'Gel' through two-dimensional liquid chromatography, before moving on to run a 'Mass Spec Experiment' with a particular fraction).

The schema currently allows five subclasses of the abstract superclass *Analyte Processing Step*, four of which are 'Gel1D', 'Gel2D', 'Chemical Treatment', and 'Column'. The fifth subclass, 'Other Analyte Processing Step', provides a mechanism to capture any other form of analyte processing by linking to a series of entries in an ontology (a controlled, structured vocabulary); this is exactly the approach taken by the designers of the MAGE object model, but on a reduced scale. Note also that the modular nature of the model allows the addition of further explicitly described processing steps (for example, capillary electrophoresis) should they be needed.

Gel is an abstract class, from which both 'Gel1D' and 'Gel2D' inherit. The first six fields of *Gel* (and, therefore, of 'Gel1D' and 'Gel2D') capture the (free text) description of the gel, the image analysis software used, and whatever images of the gel are available, referred to by URIs (universal resource identifiers—a more general form of the standard internet address otherwise known as a URL or uniform resource locator). The raw and annotated images are captured, as are the warped image and warping map that would result from the use of gel image comparison software (on the schema, corresponding to

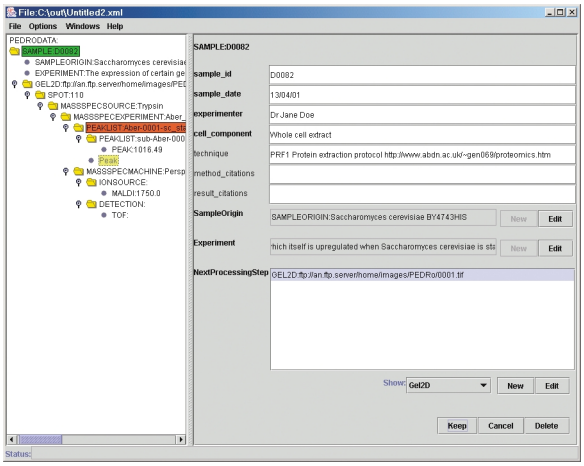

Figure 4. A developmental version of the PEDRoDC graphical user interface.

'raw_image', 'annotated_image', 'warped_image', and 'warping_map', respectively). There are also several parameters describing the gel itself (for example, 'percent_acrylamide' in the mix, the 'solubilization buffer' and 'stain' used, a measure of the total protein on the gel, the 'in-gel digest' (if performed), and the image's average 'background' intensity and total size in pixels ('pixel_size_x' and 'pixel_size_y')). 'Gel1D' and 'Gel2D' inherit most of their properties from *Gel*, the only additions concern the ranges of the gels, subsidiary information about the run, and, in 'Gel1D', a field to describe the denaturing agent (if one is used). The class *Gel* has one further association: 'DiGE Gel', which describes a 'sub' gel made visible by using a laser at the appropriate wavelength to excite a dye, generating a TIFF image for that wavelength, which is again referred to by its URI.

The class 'Column' describes the equipment's origin, its dimensions, the stationary phase, the bead size of that stationary phase, the pore size in the beads, the temperature at which the column was run, the flow rate of the mobile phase, the total injection volume, and the amount of protein against time captured as a series of 'Assay Data Points'. A 'Column' can be associated with one or more 'Mobile Phase Components', which contain a 'description' of each substance used and its 'concentration'. These components are then combined, according to the information captured in an ordered series of 'Gradient Steps', with each step specifying the percentage of the total mobile phase made up by each component at the end of that step ('step_time'). The self-reference on 'Column' (*next_dimension*) allows two (or more)-dimensional liquid chromatography to be simply represented, without the need to run the parameters again or to specify a 'dummy' fraction from the first *Column* as the Analyte for the second (or higher) dimension.

'Chemical Treatment' exists to capture any of a range of chemical derivatizations, or a protein digestion, at any stage during processing, and results in a product 'Treated Analyte'. In fact, all the *Analyte Processing Steps* have their own product (each being, like 'Treated Analyte', an instance of the abstract class *Analyte*).

One *Gel* contains many *Gel Items* (another abstract class of which 'Spot' and 'Band' are instances that simply add appropriate coordinates—pI, mass, or lane number and mass). *Gel* also offers two other methods to describe a particular *Gel Item's* whereabouts, both of which use pixel-based coordinates either to give the center point and radius of an item, or to outline the spot with a series of 'Boundary Points'. A *Gel Item* has several attributes such as 'area', 'intensity', and 'local background' intensity (as opposed to the average over the whole gel). This class also allows an identification tag (assigned by the labo-

A brief UML tutorial

In this tutorial, we present the basic constructs of a UML class diagram. The example diagram in Fig. 5 indicates that each 'Author' (who possesses several attributes—'name', 'institution', and 'email_address') is associated with (the connecting line) an unspecified number of mjournals (also with their own attributes—'Journal' has attributes of 'name', 'publisher', and 'contact_address'). The asterisk at the left end of the association signifies that a particular 'Journal' may have published zero, one, or many authors ('Author'); the asterisk at the right end of the association signifies that an 'Author' may have been published in zero, one, or many journals ('Journal'). Replacing the right-hand asterisk with a '1' would state that an 'Author' must have published in exactly one journal; replacing with '0..1' would mean an association with zero (remains unpublished) or one 'Journal'; replacing with '1..n' would mean an association with at least one 'Journal'.

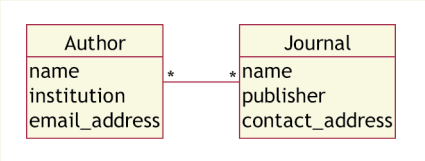

Figure 5. A many-to-many association.

The line in Fig. 5 represents a type of bidirectional association (that is, ignoring cardinalities, it does not 'point' in either direction). However, the arrow in Fig. 6 represents a necessarily unidirectional association—an inheritance relationship.

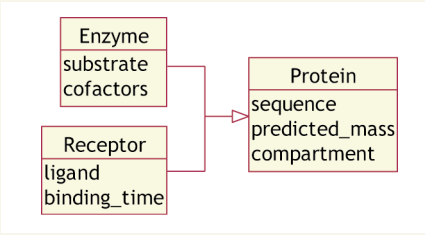

Figure 6. An example of inheritance.

In Fig. 6, 'Protein' is a general class of which 'Enzyme' and 'Receptor' are more specialized variants. The generic class 'Protein' holds attributes shared by all subclasses of that class, and is known as the superclass; those more specialized subclasses contain information specific to each (in addition to the attributes they automatically inherit from 'Protein'). The rhombus-plus-line configuration in Fig. 7 signifies that two-dimensional gels ('2D-Gel') contain spots ('Spot'); the class at the rhombus end of the association does the containing, and the class at the plain end is contained. The '1' and the '*' specify that one '2D-Gel' contains an unspecified multiplicity of spots (none, one, or more than one). Note that containment is directional—a 'Spot' cannot contain a '2D–Gel'; neither can a 'Spot' be in more than one '2D-Gel' at one time.

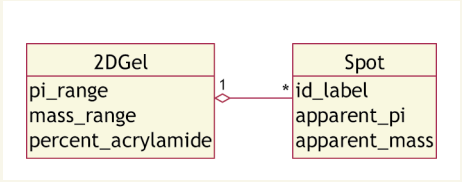

Figure 7. An example of containment.

The open-headed arrow in Fig. 8 indicates a unidirectional association between the two classes. This is really a suggestion for implementation (for example, as a relational database) rather than a feature of the data, suggesting that 'Raw Data' be linked to 'Ontology Entry', but that the reverse link need not be implemented.

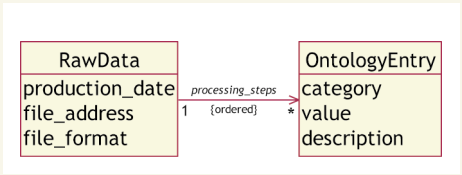

Figure 8. A directed association with a role name and a constraint.

The italicized text (*processing_steps*) gives the name, or 'role', assigned to this association to make its function explicit. The text between the curly braces ({ordered}) is a constraint—a rule that must be obeyed—which in this case decrees that the ordering of the particular 'Ontology Entries' is important, and should be preserved.

ratory) and a putative protein identification, along with a source for that identification information. A *Gel Item* can be linked to another *Gel Item* through 'Related Gel Item', which contains a 'description', a URI, and a reference number. A particular *Gel Item* can also be associated with one of the fluors in a DiGE experiment through 'DiGEGelItem'. 'Column' produces (but note that it does not contain) many (contiguous) 'Fraction' items, described by their start and end points in time, and the corresponding assayed amount of protein. Last, the generic class 'Other Analyte Processing Step' is associated with a similarly generic product, 'Other Analyte', which is again a named binder for the series of ontology entries that describe it.

Mass spectrometry (yellow). As stated, 'Spot', 'Band', and 'Fraction' are, along with 'Treated Analyte', 'Other Analyte', and 'Sample', subclasses of the abstract superclass *Analyte*, and can therefore be fed back into the processing cycle, passed on for mass spectrometric analysis, or both. The class 'Mass Spec Experiment' provides an exit point for the analyte processing cycle, and is associated with all or part (thus the one-to-many relationship) of the *Analyte* that is to be used. 'Mass Spec Experiment' has two fields: 'description' provides the facility to name the particular machine setup used, and 'parameters file' may contain a URI pointing at the machine-generated parameters list. 'Mass Spec Experiment' has a many-to-one association with 'Mass Spec Machine', which details 'maker', 'model', and 'software_version', reflecting the fact that many experiments can be run on the same piece of apparatus with broadly the same settings. Detail about the makeup of the 'Mass Spec Machine' is stored in three further classes: *Ion Source* is an abstract class that will, in practice, be either 'MALDI' or 'Electrospray', each of which has its own set of fields (voltages of various kinds; tip, solvent, and interface details for electrospray; laser wavelength and matrix type for MALDI runs). 'Other Ionization' exists to capture less widely used, or novel, ionization strategies; it has links to a series of entries in an ontology,

and functions in the same way as 'Other Analyte Processing Step' and 'Other Analyte' from the sample processing portion of the model. 'mz Analysis' (a corruption, due to naming constraints, of *m/z* analysis) represents the mass analyzing and fragmentation section of the mass spectrometer (for example, 'Quadrupole', 'Ion Trap', or 'Collision Cell', each with its own parameters); it too has a generic instance ('Other mz Analysis') with which to describe analyzers not explicitly captured by the model. The self-association on 'mz Analysis' allows several analyzer stages to be catenated to describe more complex scenarios, such as are found in tandem machines. Finally, 'Detection' describes the ion detector in the machine: photomultiplier, electron multiplier, or micro-channel plate. It is worth noting again at this point that the modular nature of the model allows the addition of further, explicitly described, mass analyzing steps should they be needed.

MS results analysis (green). A 'Mass Spec Experiment' generates a chromatogram, from which is extracted a list of peaks (although in many MALDI setups the chromatogram, such as it is, is largely ignored, the user simply being presented with the summed peak list). A 'Peak List' could be the 'raw' (but heavily machine-processed) output from the spectrometer or a human-edited list (for example, prepared for submission to a search engine) associated with the original unedited list through the self-reference 'has_children'. Processing of the list is captured in the associated class 'List Processing'. The origin of the 'Peak List' is captured by 'list_type', whereas 'description' is for a free text annotation of the list by the user; 'mass_value_type' serves to flag the peaks in that 'Peak List' as averaged or monoisotopic. A derived 'Peak List' (again linked through 'has_children') could be generated by the second stage of a tandem mass spectrometer run. 'MSMS Fraction' describes, by center and width, the 'window' of the initial mass spectrum that was passed to the second stage of a tandem mass spectrometer, to generate another 'Peak List'. To recap, this second 'Peak List' would have associations with the parent 'Peak List' (through 'has_children'), an instance of 'MSMS Fraction', and any database search that used that peak list.

The individual peaks (under 'Peak') in a list are described by mass-to-charge ratio ('m_to_z'), 'abundance' (literally, the peak height), and 'multiplicity' (that is, the isotopic pattern around the main peak, if known). Given that the 'multiplicity' of a particular 'Peak' is usually greater than one, the mass-to-charge ratio must refer to the monoisotopic peak, rather than the most abundant, or the average. If an ICAT experiment is being performed, the expression information for a peak (that is, the area under the chromatogram specific to that particular peak), and the software used to generate it, are captured in 'Peak-Specific Chromatogram Integration'. The peak-specific chromatogram itself is represented by a vector of 'Chromatogram Points', to avoid loss of information.

To perform a protein identification, a particular 'Peak List' (usually an edited list) would be submitted to an identification tool, such as Sequest (ref. 12; http://fields.scripps.edu/sequest/), Mascot[13], or PepMapper[14]. The classes 'DB Search' and 'DB Search Parameters' capture information about who did the identification, when they did it, what program they used, what database (of theoretical proteins from an *in silico* digest of an organism's predicted proteome) was used, what errors were taken into account when searching, what potential modifications were allowed on proteins from the sample that generated the peak list, any additional information gleaned by Edman sequencing or another chemical analysis, and whether the ions carry ICAT labels (such that only cysteine-containing peptides should be searched against). One identification process will usually generate several 'Peptide Hits'; the link from 'DB Search' to 'Peptide Hit', which stores some confidence measures for the identity assignment, the 'sequence', and some annotation 'information', is, therefore, a one-to-many relationship. 'Protein Hit' represents the proteins against which all or some of the peptides have been aligned, and links to some (locally stored) standard information about the protein itself ('Protein'): a 'description' of the protein, an 'accession number' (from GenBank or EMBL), the 'predicted mass' and 'predicted pI' of the protein, its amino-acid 'sequence', any common *in vivo* 'modifications', the 'organism' in which it is to be found, and an open reading frame identifier ('orf_number') if possible (as is the case in *Saccharomyces cerevisiae*[15]). Note that, for convenience, a 'Protein Hit' may also have a link back to the spot that generated it, through the associated class 'Related Gel Item'.

The issue of whether a database search has identified a protein that was actually in the sample, or has made a false identification, is obviously important—but difficult to answer. Currently, threshold models are used in which a cut-off probability score is arbitrarily chosen, above which the identification is believed to be true. Statistical tools are being developed[16] to determine the probability that a peptide or protein is indeed a 'true positive'. In principle, these tools are transparent to the type of mass spectrometer, the search tools used, and the database searched. We could envision adding several additional fields to the PEDRo model to capture information from data validation processes. However, such quality-control tools have yet to be fully accepted in the proteomics community at large, so (for now) we have restricted ourselves to including a single field—'probability'—in the class 'Peptide Hit'.

The PEDRo UML model represents a subset of the total information available about a proteomics experiment. However, we believe it fulfills, but does not exceed, our stated criterion of sufficiency, and therefore it offers a sound base from which to develop both repositories and the tools to stock, maintain, and interrogate them. The next section gives an overview of the operation of a PEDRo-compliant repository, and describes in some detail a tool (PEDRoDC) by which XML files can be generated that conform to the PEDRo model.

Using PEDRo

A repository such as that specified by the PEDRo UML model cannot exist in isolation. It requires a suite of tools to stock and curate it, and to disseminate the information it contains. Figure 3 shows one possible framework around such a repository, by describing graphically the flow of information from generation to dissemination. Under this scheme, a laboratory generates data from, and about, a proteomics experiment (far left). These data are then encoded by the data entry tool (PEDRoDC), which produces an XML file (specifically a PEML file). Note that, by applying a predefined transformation expressed in XSLT (which is one of the series of languages that relate to XML data description and manipulation; http://www.w3.org/TR/xslt), PEML files can be read directly, as HTML, using a normal web browser. XSLT files can also be written to transform old-format PEML files when the schema is modified, thereby avoiding 'versioning' conflicts with newer software. The PEDRoDC-generated PEML file is then submitted to the repository site, whereupon a validation tool checks the correctness of the submitted file, before allowing its entry to the relational database that acts as the repository (in this example, direct submission to the repository is forbidden). Once in the repository, the data potentially becomes publicly available (subject to proprietary constraints) for use by PEDRo-compliant query, search, and analysis tools.

The PEDRo data collator (PEDRoDC) is an example of a tool with which a user might enter information about and data files from proteomics experiments. The tool collates these data into a single XML file for submission (by some route) to the repository. Note that the details of the repository's implementation are unimportant to PEDRoDC, although some familiarity with the PEDRo model will be beneficial to users.

Figure 4 shows a screenshot of the PEDRoDC. Files are loaded from the 'File' menu (which also allows the initiation of a new file, data importation, saving of the current file, and saving and loading of 'tem-

plates'). A simple graphical browser of all the records present within a file (concrete instances of the UML classes described above) is presented in tree form (left panel); this normally serves as a position indicator during editing. However, in conjunction with the options menu, the tree can display validation errors in the current file (for example, records that lack required children), indicate changes made since the last file save, and highlight the records that match in a free-text search of the file.

The right panel shows an actual record (an instance of 'Sample'). The buttons at the bottom of the window relate to the current record: 'Keep' updates the tree with the current record (in memory only—'Save' must be selected from the 'File' menu to update the saved version), 'Delete' removes the current record from the tree, and 'Cancel' closes the particular record with no action taken. There are seven simple data entry fields (the first four of which are compulsory—indicated by the bold typeface). These fields perform simple validation checks on the fly, and if data is incorrectly entered to a field, or a compulsory field is left blank, the offender's label is colored red, and the record cannot be kept. The right panel also has slots for two required (that is, compulsory, single-choice) child records ('Sample Origin' and 'Experiment') to be attached, and a multiple choice list—this is because 'Sample' also has optional children ('Gel1D', 'Gel2D', 'Chemical Treatment', 'Column', 'Other Analyte Processing Step', and 'Mass Spec Experiment'). The list of optional children, and the two compulsory slots, allows existing children to be edited or deleted, by selecting a record and clicking on 'Edit'. A new child can be added by selecting the record type from the combination box (for the list of optional children) and clicking on 'New'.

Other features of the software include the following: first, the ability to open multiple files and copy/paste between records in them; second, the manufacture and use of 'templates'—saved versions of a particular record and (optionally) its subtree, intended to facilitate rapid completion of frequently used records (for example, details about the mass spectrometer or the database search parameters, which vary only rarely); third, support of the use of ontologies, through an intuitive right-click interface that can be invoked on certain fields; fourth, context-sensitive help can be turned on through the 'Help' menu, offering guidance about both the interface itself and the nature of the data required by it; and fifth, a user's manual and tutorial with example data sets (currently under development).

We believe PEDRoDC to be a reasonably intuitive data entry and editing tool. It is flexible (due to the ability to 'jump in' at any point in the file hierarchy), efficient (because of the templating facility), and rigorous (because of the enforcement of an audit trail for data through the hierarchical structure of PEML—for example, information about a gel cannot be entered without describing the origin of the sample that was run on it). PEDRoDC, along with the other components of the PEDRo repository system (Fig. 3), will facilitate the proteomics community's ability to capture and disseminate proteomics experimental data in a systematic way. The PEDRoDC program is publicly available at the PEDRo website (see URL at end of Discussion), along with the PEML schema and some sample data files.

Discussion

Our goal was to describe the PEDRo model, together with the associated implementations and tools, in sufficient detail to engender a full discussion of their features among the wider proteomics community. This discussion will inevitably highlight potential improvements to the model—trimming in some places, expanding in others—with a view to reaching a standard representation that will bring benefits to all.

The PEDRo model requires a fairly substantial amount of data to be captured. However, much of this information will be readily available in the laboratories that generate the data, and a substantial portion will be common to many experiments and so will only have to be entered once (then saved as a template in PEDRoDC). There are several advantages to adopting such a model. All data sets will contain information sufficient to quickly establish the provenance and relevance (to the researcher) of a data set, and to allow non-standard searches. Tools can be developed that allow easy access to large numbers of such data sets, and information exchange between researchers will be facilitated through the use of a common interchange language (PEML).

The PEDRo schema is large because it is an explicit model of the data and metadata. This was seen as the correct approach while in this developmental stage, both to make the scope of the model explicit and to facilitate discussion of its content. However, as time passes, technologies evolve. In response, we propose XSLTransformations to transform old PEML data files when the Schema is modified.

Another approach to 'future-proofing' is that taken with MAGE (OM/ML), which is used only sparingly in PEDRo. The individual sections of the MAGE object model are essentially generic process templates, with all the specific information being kept in a series of ontologies. The MAGE model is extremely robust against technological evolution and also offers the facility to describe rare technologies not necessarily covered by an explicit (and therefore limited) model. However, there is already some evidence that MAGE's flexibility is producing some cross-compatibility issues between different software producers' allegedly MAGE-compliant export formats. Also, it should be made clear that the explicit detail about technologies does exist, in the ontologies, and does need to be updated as those technologies evolve, even if the model itself remains stable.

Whether or not future incarnations of the PEDRo schema should more closely reflect the design philosophy of the producers of MAGE is a question open to debate. There is certainly a clear case to adopt the same description of the production of the initial experimental sample and the motivation for the experiment (Fig. 2, blue), as these data would provide the common handle for 'grand experiments' across genome, proteome, and, eventually, metabolome. In the meantime, an XSLT mapping from PEDRo to MAGE is a possible answer (although the reverse mapping is less straightforward).

We have attempted to provide an overview of the suite of tools required to maintain a PEDRo implementation and described one (PEDRoDC) in detail, to provide as full a context for discussions as possible. Once the PEDRo model (or its successor) has stabilized, the development of such applications can proceed, as can the integration of proteomics databases with each other and with other resources (for example, the major sequence databases), providing sophisticated search and analysis tools to practitioners of proteomics and the wider research community. In this age of genome-scale experiments, the need to establish best practice in data capture and dissemination is very great if we are to extract full value from our experimental activities. It is our hope that this paper will bring the fulfillment of this need a step closer for the proteomics community.

URL. For PEDRo website, see http://pedro.man.ac.uk/.

Acknowledgments

Special thanks go to Francesco Brancia, Jenny Ho, and Sandy Yates for their critical appraisal of the Schema at various stages. This work was supported by a grant from the Investigating Gene Function (IGF) Initiative of the Biotechnology & Biological Sciences Research Council to S.G.O., N.W.P., A.B., S.G., S.H., P.C., and A.J.P.B. for the COGEME (Consortium for the Functional Genomics of Microbial Eukaryotes) program. D.B.K. thanks the BBSRC for financial support, also under the IGF initiative. K.L.G. is supported by the North West Regional e-Science centre (ESNW), within the UK eScience Programme. Many people have contributed their advice and expertise to the design of PEDRo, at various meetings formal and otherwise, notably attendees at the 2002 Proteomics Standards Initiative meeting of the Human Proteome Organisation at the European Bioinformatics Institute.

Received 3 January 2003; accepted 27 January 2003

1. Wilkins, M.R., Williams, K.L., Appel, R.D. & Hochstrasser, D.F. (eds.) *Proteome Research: New Frontiers in Functional Genomics* (Springer, Berlin, 1997).
2. Pennington, S.R. & Dunn, M.J. (eds.) *Proteomics. From Protein Sequence to Function* (BIOS, Oxford, UK, 2001).
3. Attwood, T.K. The quest to deduce protein function from sequence: the role of pattern databases. *Int. J. Biochem. Cell. Biol.* **32**, 139–155 (1999).
4. Oliver, S. Guilt–by–association goes global. *Nature* **403**, 601–603 (2000).
5. Hoogland, C. *et al*. The 1999 SWISS–2DPAGE database update. *Nucleic Acids Res.* **28**, 286–288 (2000).
6. Sanchez, J.C. *et al*. The mouse SWISS–2DPAGE database: a tool for proteomics study of diabetes and obesity. *Proteomics* **1**, 136–163 (2001).
7. Brazma, A. *et al*. Minimum information about a microarray experiment (MIAME)—toward standards for microarray data. *Nat. Genet.* **29**, 365–371 (2001).
8. Booch, G., Rumbaugh, J. & Jacobson, I. *The Unified Modelling Language User Guide* (Addison Wesley, Massachusetts, 1997).
9. Spellman, P.T. *et al*. Design and implementation of microarray gene expression markup language (MAGE-ML). *Genome Biol.* **3**, 0046.1–0046.9 (2002).
10. Unlu, M. *et al*. Difference gel electrophoresis: a single gel method for detecting changes in protein extracts. *Electrophoresis* **18**, 2071–2077 (1997).
11. Gygi, S.P. *et al*. Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nat. Biotechnol.* **17**, 994–999 (1999).
12. Eng, J.K., McCormack, A.L. & Yates, J.R. III An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass Spec.* **5**, 976–989 (1994).
13. Creasy, D.J., Cottrell, D.M., Perkins, J.S. & Pappin, D.N. Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis* **20**, 3551–3567 (1999).
14. Sidhu, K.S. *et al*. Bioinformatic assessment of mass spectrometric chemical derivatisation techniques for proteome database searching. *Proteomics* **1**, 1368–1377 (2001).
15. Mewes, H.W. *et al*. Overview of the yeast genome. *Nature* (Suppl.) **387**, 7–65 (1997).
16. Keller, A., Nesvizhskii, A.I., Kolker, E. & Aebersold, R. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. *Anal. Chem.* **74**, 5383–5892 (2002).