Acta Crystallographica Section D
Biological Crystallography
ISSN 0907-4449

HalX: an open-source LIMS (Laboratory Information Management System) for small- to large-scale laboratories

Jaime Prilusky,[a] Eric Oueillet,[b] Nathalie Ulryck,[b] Anne Pajon,[c] Julie Bernauer,[b] Isabelle Krimm,[d] Sophie Quevillon-Cheruel,[b] Nicolas Leulliot,[b] Marc Graille,[b] Dominique Liger,[b] Lionel Trésaugues,[b] Joel L. Sussman,[a] Joël Janin,[e] Herman van Tilbeurgh[b] and Anne Poupon[b]*

[a]Israel Structural Proteomics Center, Weizmann Institute of Science, Rehovot 76100, Israel, [b]Yeast Structural Genomics, IBBMC Bâtiment 430, CNRS-Université Paris-Sud, 91405 Orsay CEDEX, France, [c]EMBL Outstation Hinxton, European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridge CB10 1SD, England, [d]RMN Biomoléculaire, Université Claude Bernard–Lyon 1, Bâtiment 308 G, CPE–Lyon, 43 Boulevard du 11 Novembre 1918, 69622 Villeurbanne CEDEX, France, and [e]LEBS, Bâtiment 34, CNRS, 1 Allée de la Terrasse, 91190 Gif-sur-Yvette, France

Correspondence e-mail: poupon@lebs.cnrs-gif.fr

Received 22 July 2004
Accepted 13 January 2005

Structural genomics aims at the establishment of a universal protein-fold dictionary through systematic structure determination either by NMR or X-ray crystallography. In order to catch up with the explosive amount of protein sequence data, the structural biology laboratories are spurred to increase the speed of the structure-determination process. To achieve this goal, high-throughput robotic approaches are increasingly used in all the steps leading from cloning to data collection and even structure interpretation is becoming more and more automatic. The progress made in these areas has begun to have a significant impact on the more 'classical' structural biology laboratories, dramatically increasing the number of individual experiments. This automation creates the need for efficient data management. Here, a new piece of software, *HalX*, designed as an 'electronic lab book' that aims at (i) storage and (ii) easy access and use of all experimental data is presented. This should lead to much improved management and tracking of structural genomics experimental data.

1. Introduction

Structural genomics projects, even those labelled as 'medium-throughput' like the Gif/Orsay Yeast Structural Genomics project (http://genomics.eu.org) and the Israel Structural Proteomics Center (http://www.weizmann.ac.il/ISPC), create a vast amount of data of very diverse nature. The large number and variety of proteins treated in these projects, as well as the diversity of experimental steps leading to the final structures, create a need for a high-level information storage and tracking system. Even 'classical' structural biology laboratories now use crystallization robots, leading to very large numbers of crystallization tests. These make it very hard to follow and analyse all the experiments without the use of an informatics tool.

Apart from automation, new laboratory practices and new protocols are also sources of an increase in the number of experiments realised, such as the use, now very common, of different expression systems, which change the 'crystallizability' of the protein (Yokoyama, 2003), the use of many different constructs and the study of protein complexes. In this latter case, the method of reassembling the complex is essential and many methods have to be tested. This increase and diversification of data, even for classical structural biology laboratories, make the use of traditional laboratory notebooks inefficient when it comes to knowing what has been done, when and how. Indeed, laboratory notebooks can only be indexed either by protein or by date. Both are inefficient because, on one hand, most experiments are done in parallel (many different proteins at the same time), which makes protein indexing impossible. On the other hand, going through date-indexed notebooks implies having an idea of the date for a given experiment.

It is recognized that data mining of experimental observations may also improve the yield of successful protein production and structure determinations. The North-East Structural Genomics centre has for example derived a decision tree from their experimental data, allowing the choice of targets which have a greater chance of being soluble and stable (Goh *et al.*, 2003).

Although some commercial solutions for laboratory information management exist, they are often onerous and frequently designed

for industrial use. They lack the flexibility that is necessary and demanded in an academic research environment.

Whereas some areas of experimental biology [microarrays (Fellenberg *et al.*, 2002; Bono *et al.*, 2002), proteomics (Cho *et al.*, 2002; Fenyö & Beavis, 2002), genetics (Imbert *et al.*, 1999), sequencing (Dedhia & McCombie, 1998), X-ray data (Harris & Jones, 2002)] have implemented data-management systems, these approaches still did not penetrate structural biology laboratories. The existing systems are either too heavy or not flexible enough (Haebel *et al.*, 2001; Goh *et al.*, 2003; Zolnai *et al.*, 2003). The system we propose here, named *HalX*, intends to register all types of experiments on the way from cloning through to structure determination in a structured manner allowing extensive data mining. User friendliness and flexibility were major concerns in the set up of the management system. This software is already in use in our laboratories (Yeast Structural Genomics project and Weizmann Institute) and in others (IBS Grenoble, EMBL Hamburg, CBS Montpellier). It can be freely downloaded from our website (http://halx.genomics.eu.org) and an unlimited trial version is available.

2. Material and methods

HalX is built around a three-tier architecture model. A web browser running on the client's computer represents the first tier. The second tier holds the shared parts of application and business logic, and is PHP based supported by an Apache web-server. PostgreSQL manages the storage and DB-server third tier. All software components are freely available under GPL licence. The data model used (the structure of the database) will be discussed in §3.

3. Results and discussion

3.1. Objectives

The question that has led us to develop this software and that drove its layout and functionalities was: why are structural biology laboratories not using LIMSs? There are multiple answers to that question: because most of the existing LIMSs are too onerous (or not distributed at all), they are not flexible enough to allow all the protocols used or the entry of the data is lengthy and repetitive, and finally, the tools provided are not powerful enough. Thus, the solution we proposed to develop had to be freely distributed, allow rapid definition of new experiments, allow chaining the existing experiments in any reasonable order and extensively use default values to reduce as much as possible the number of data to enter for each experiment. Finally, it has to provide ways to browse through the data that help the user in his/her research work.

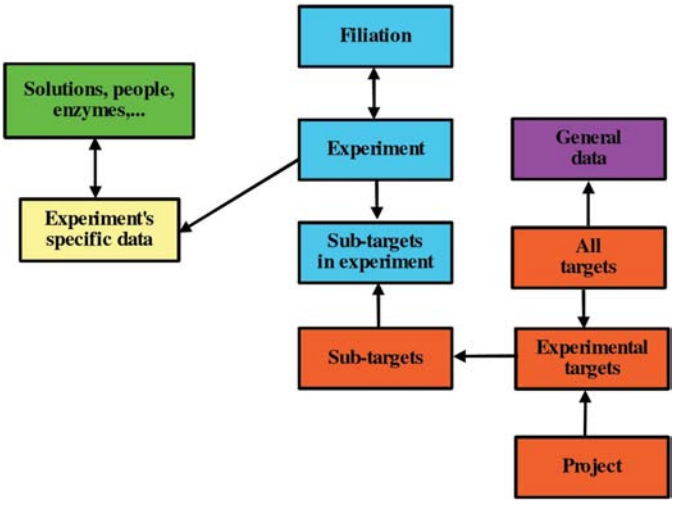

Figure 1
Overview of the data model. The tables of the database are classified into five categories: (i) target tables (red), (ii) data from public databases concerning targets (purple), (iii) tables containing data specific to each type of experiment (yellow), (iv) data common to all types of experiments (green) amd (v) core tables (blue). Colour versions of the figures in this article are available in the online edition of the journal.

HalX allows the easy creation of any protocol by offering a 'brick' based system. Each experiment is regarded as a brick and to construct a protocol users just pile up the bricks. The experimental parameters (temperature, volumes, concentration *etc.*) are precisely defined only once and the protocols can be reused at will to enter real experiments. This way, for most experiments, users need only to enter the results. The protocols can be edited easily to make new, slightly different versions.

3.2. Data model

The data model used for *HalX* (*i.e.* the way data are stored in the database) consists of 123 tables belonging to five different categories (Fig. 1).

(i) The red tables concern the targets.

(ii) The purple tables contain information gathered from public databases concerning targets.

(iii) The green tables contain information that can be used in any experiment (like solutions, chemicals, enzymes *etc.*)

(iv) The yellow tables contain specific data for each type of experiment.

(v) The blue tables represent the core of the model: they store the way experiments are linked together and which proteins are present in a given experiment.

3.2.1. Target tables. There are four tables in this category. The proteins considered as 'targets' are not necessarily proteins for which experiments are realised; they are all the proteins in which the laboratory is interested. This can for example consist, apart from the proteins on which experiments are conducted, of all the potential partners. In the Yeast Structural Genomics project for example, all the yeast proteins are in this list, which allows us to link our targets with their potential partners, with their homologs and with the proteins belonging to the same metabolic pathway. This greatly facilitates the choice of new experimental targets. The second table ('experimental targets') contains all the full-length proteins on which experiments are conducted. Each experimental target is attached to a project, chosen in the list contained in the 'project' table. This allows giving different permissions on different targets to the same user (see User Interface).

The table 'sub-targets' contains all the different forms of a protein that are used in the experiments. These can be labelled protein, mutants, domains *etc.* This subdivision into sub-targets, instead of declaring each construct as a different target, allows easily linkage of the experiments realised on the different forms of a same protein.

It is important to note that *HalX* is able to register any experiment on complexes, but the members of a complex are still registered separately as individual targets.

3.2.2. Information tables. There is a possibility in *HalX* to store data coming from public databases concerning each protein: Pfam domains (Bateman *et al.*, 2004), putative interaction partners, putative complexes, paralogs, orthologs, Swiss-Prot IDs, accession and

comments (Bairoch *et al.*, 2004), results of threading programs *etc.* The availability of this information in a single database allows the experimenter to have an accurate summary on a single web page, on the local server.

HalX does not yet contain any mechanism to periodically update this information, but the database can be reached and modified from a different home-made program (as Perl scripts, for example). This will be done in a future version, through web services.

The solution of not storing the data in the database but rather storing pointers to this information in the various public databases presents two major limitations: it does not really allow data mining on this information and it makes presentation on a single page quite difficult.

3.2.3. Common tables. These tables contain data that can be used in various places of the program. They for example contain all the solutions (Fig. 2) linked to all the elements of each solution (chemical and concentration). In the interface, each time a solution is needed, the user has two choices: either choose a solution already in the database (which appear in a drop-down menu) or enter a new one. In the later case, the new solution is stored in the database and is available for re-use. This has two main advantages. Firstly, there are a small number of solutions that are often used and having them already entered in the database saves time. Secondly, there is not ambiguity on the solution used in an experiment, which is often the case in 'classical' laboratory books, since nobody takes the time to rewrite the solution composition.

A second example is the 'observation-type' table. This table contains a list of possible observation of crystallization drops (*e.g.* macrocrystal, microcrystal, plates, needles *etc.*). When entering observations concerning a given drop, the user has to choose from this list and cannot add a different observation type. This allows comparing and classifying the different crystallization conditions, which requires having a limited number of different categories.

3.2.4. Experiment-specific data. In these tables all the experimental details are stored. Because the details are very different between two different types of experiments, there are one or more specific tables for each type of experiment. Some experiments, like crystallization, are quite complicated and require more than one table (Fig. 2). Having the data specific to each experiment type in separate tables is very convenient when it comes to adding new experiment types. It just requires creating a new table containing the fields needed for that experiment and creating an interface page for entering its data. This means that we can add a new experiment in a few hours time without changing anything crucial in the software or the database, which allows keeping up with the new protocols very easily and efficiently.

3.2.5. Core tables. There are three core tables. The 'experiment' table gives a unique reference to each experiment, and registers its date, type (*e.g.* PCR, DNA purification *etc.*), the person who has done this experiment and the holder in which it took place (*e.g.* tube, deep-well plate, Hampton 96 wells – flat bottom *etc.*) and if needed bar code.

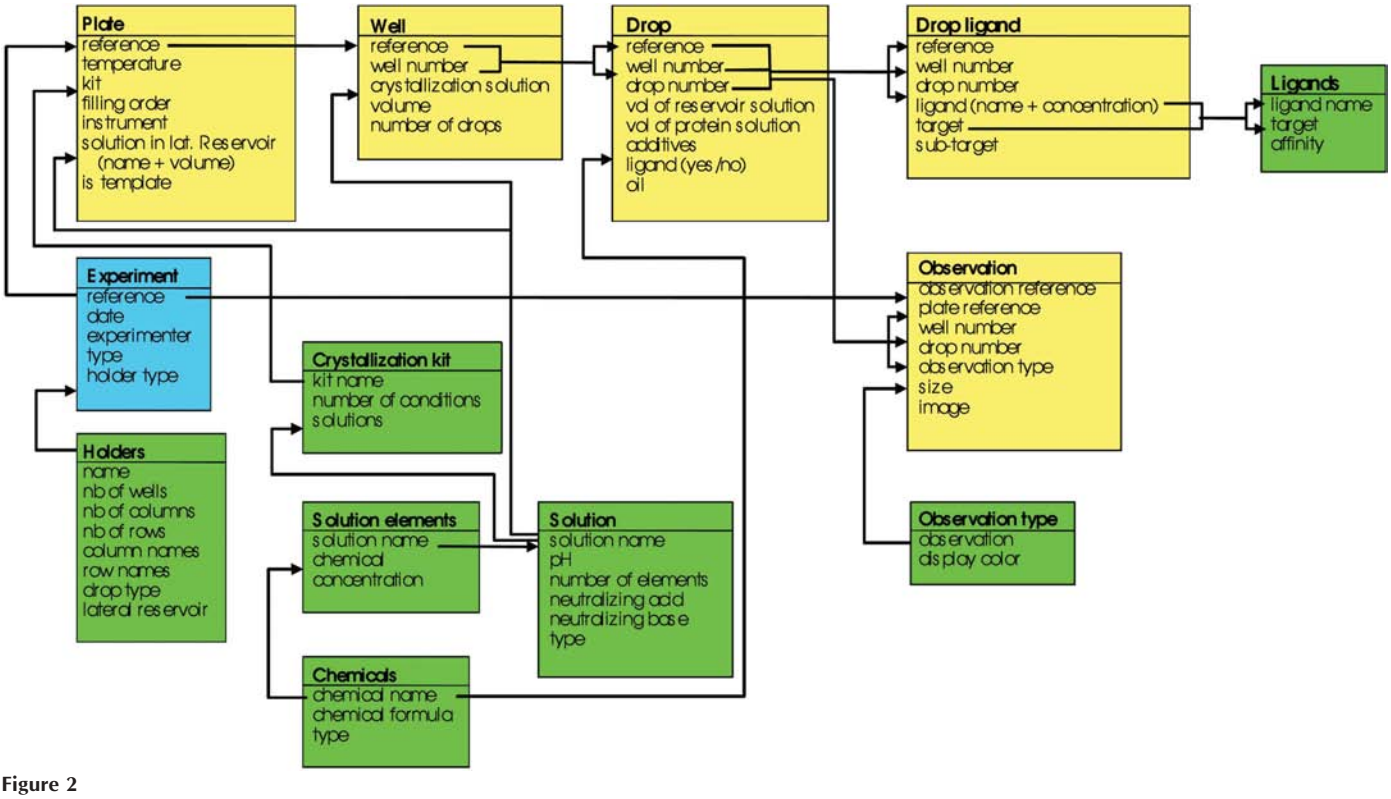

Figure 2
Registering the data for crystallization plates. Each plate is given a reference number, which is registered in the table 'experiment' together with the date, the experimenter, the type (here would be 'CRYSTALLIZATION') and the type of plate used (*e.g.* 'Hampton 96 well – flat bottom'). More specific data concerning the plate are registered in the table 'plate': temperature, kit used, filling order (either horizontal or vertical), if needed the solution in the lateral reservoirs and if this plate is a template. Data concerning each well are registered in the table 'Well': crystallization solution, volume and number of drops. Data concerning each drop are registered in the table 'Drop': volume of crystallization solution, volume of protein solution(s), additives (which are not already in the crystallization solution), ligand can be set to yes or no, and oil (fields are not all shown: the oil can be a mix of different oils, and the volume can be entered). If ligands are added in a drop, a line is inserted in the 'Drop ligand' table, together with the sub-target to which this ligand is supposed to link. The tuple (ligand, sub-target) must already be in the 'Ligands' table. When adding an observation, a new experiment is created (in the table 'Experiment') and lines are inserted in the table 'Observation' for each observed drop. The observation must be chosen in a predefined list stored in the table 'Observation type'. An image of the drop can be stored.

When a new experiment is entered, the user gives the references of the sample(s) used for this new experiment. These samples are considered as the 'parents' and the new sample is the 'child'. Each (parent, child) tuple is registered in the 'filiation' table. This system allows two very important things.

A given experiment, denoted (child), can have any number of parents, denoted (parent1, parent2, ..., parent*n*), by registering *n* lines in the filiation table: (parent1, child), (parent2, child), ..., (parent*n*, child). Reversely, one parent can have *n* children by registering *n* lines in the filiation table: (parent, child1), (parent, child2) ... (parent, child*n*). Both of these cases do happen frequently in crystallization: one single protein purification might be used for many crystallization plates and one plate often contains samples coming from different protein purifications.

The parent–child relationship does not depend on the type of each experiment. This means that, even if in most cases, crystallization plate directly follows concentration, a dialysis is sometimes needed. Most importantly, this allows the addition of new types of experiments without changing the core of the data model.

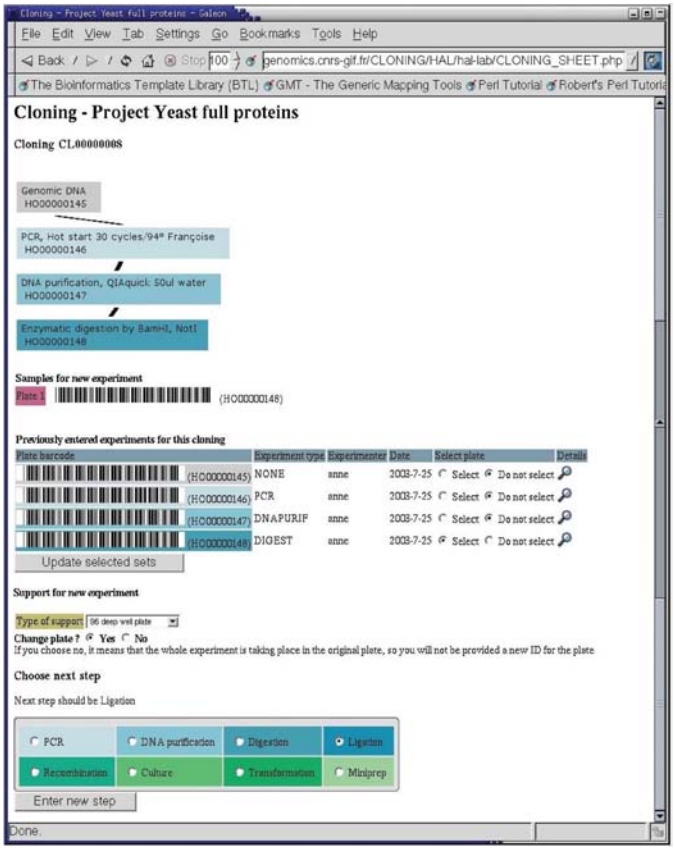

Figure 3
Cloning page. CL00000008 is a reference number given to this cloning experiment. As for all reference numbers in the database, it is unique. The graphic represents the experiments already entered in this cloning. For each of them, the main feature is given (protocol for PCR and DNA purification, enzymes for digestion ...) with the reference of the plate. Each box is clickable, opening a popup window with all the experimental details. In 'Sample for next experiment', the reference and barecode of the plate that will be used for next experiment are given (when tubes are used, instead of 96 deep well plates here, the reference of the experiment is also given). In 'Previously entered experiments for this cloning' more details are given, and any sample can be selected or unselected for next experiment. The type of support for next experiment can be changed. The 'change plate' option allows to keep the same plate through many experiments, as it often happens when using robots. In 'choose next step' all the experiments available for cloning can be selected. When using the 'Enter new step' button, the user will be ask to enter (or confirm if using default files) the values for the experiment's parameters.

The third core table is 'sub-targets in experiment' and registers, for each sub-target (sub1, sub2, ..., sub*n*) present in the same experiment (reference) the tuples (sub1, reference), (sub2, reference), ..., (sub*n*, reference). This allows there to be any number of sub-targets in the same experiment.

This data model will, in the near future, be slightly modified to merge with that designed by the EBI in the SPINE project (Pajon *et al.*, 2005; http://www.ebi.ac.uk/msd-srv/docs/ehtpx/lims/). The philosophy of this new model is the same, the only important change is the definition of explicit samples, which is defined as the holder and what it contains. In our current model, the sample is implicitly defined by the description of the experiment that yielded it. This will not change the way the user enters or views its data, but will facilitate data exchange, for example between the laboratory and the synchrotron center.

3.3. User interface

The user interface is divided into six main parts.

(i) Add experiment.

(ii) Modify experiment.

(iii) Defaults and templates, where the users can define default protocols, or define a given experiment as a template that can be reused for new experiments.

(iv) View experimental results, where the data-mining tools are.

(v) Superuser. The superusers are defined as project managers.

(vi) Admin.

3.3.1. Data-access restrictions. To be able to restrict the accessibility of the data in a flexible manner, we have implemented the notions of project, administrator, superuser and experimenter. One project is a list of targets and experiments on these targets, which accessibility can be globally set for the three different types of users.

The administrator can add new superusers and perform updates of the database and software. This has to be used very carefully, since this is the only user category that has direct access to the database. The administrator can modify any data in the database.

The superuser, whose role is project manager, can add new projects, add/delete/modify targets in his/her projects, add new users and add new crystallization kits. The superuser can modify some of the data for experiments of his/her projects. For data integrity reasons, the data are not all modifiable. For example, the sub-target used in a given experiment cannot be modified. Though, an experiment can be deleted. This has to be used with great care, since deleting one experiment deletes all of its children.

The experimenter can enter experiments in a project for which he/she has permissions (which are decided by the superuser of each project). The experimenter can modify some of the data describing his/her own experiments. For the same reasons described for superuser, the data are not all modifiable. A user can delete one of his/her own experiments only if all the children of this experiment also belong to him/her.

3.3.2. Add experiment. This is of course the main part of the software, where users can enter their experiments. The experiments in structural genomics protocol pipelines were divided into five main categories, called 'meta-experiments': cloning, expression, protein purification, X-ray structure resolution and NMR structure resolution. Each meta-experiment regroups all the experiments associated to this step. For example, the 'cloning' contains modules for PCR amplification, DNA purification, digestion, ligation, recombination, transformation, culture and Miniprep. These individual steps can

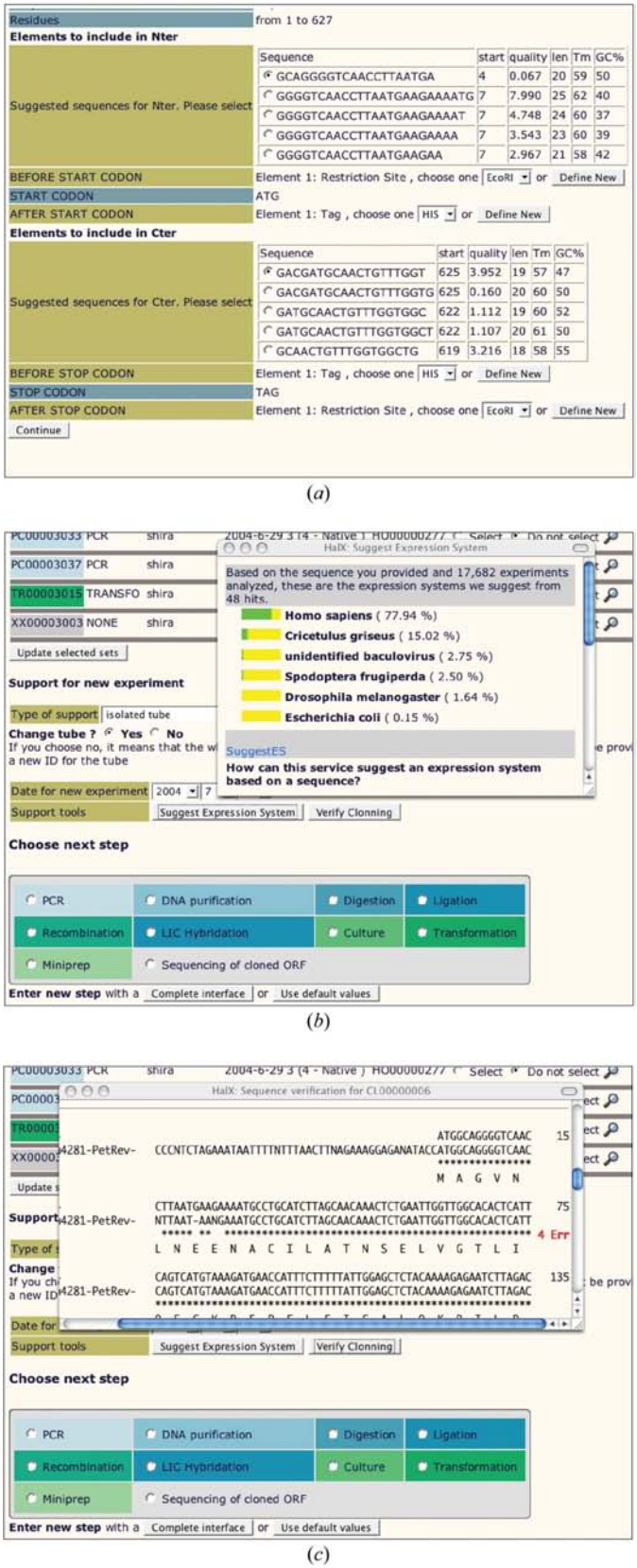

Figure 4
Web services. Here are presented three examples of web-services included in *HalX*. BestPrimers (*a*) proposes sequences for primers, and is included in the primer design tool. suggestES (*b*) suggests expression systems from the cloning page. verifyCloning (*c*) is also called from the cloning page and checks the sequence obtained after sequencing against the sequence registered in the database.

then be chained into any cloning protocol, such as Gateway, pET or any custom cloning protocol.

All the categories are designed on a common framework: the experimental work is considered as a graph, where the nodes are individual experiments. Each node is identified by its type (PCR, ligation *etc.*), parameters, parents [the experiment(s) that yielded the sample(s) used at this node] and children [the experiments that use the sample(s) that stem from this node].

Ideally, all experiments except PCRs should have a parent experiment, which is registered in the database. However, because some samples come from outside of the laboratory, any experiment can be entered even without a parent's reference. If the information concerning these parent experiments is entered later than the child experiment, it is also possible to make the link afterwards.

This design allows any variation in the protocols, because any number of parents and/or children can be linked to one node, any type of node can be integrated in the graph. Introduction of a new type of experiment in the general pipeline just requires creating a new node without modifying the architecture of the program.

The interaction of *HalX* through the database's interface was considered crucial during the development. The same general layout is used for the different pages and a typical example for the cloning page is given Fig. 3. On this page the user always finds which experiments have already been entered for this cloning and how they are linked to each other. The user can at any moment retrieve all details of a given experiment and choose a starting sample for the next experiment. In a given experiment, any relevant images can be easily uploaded and stored (gel images, elution profile, drop images *etc.*).

Another very important feature is that the interface allows the entry of many similar experiments at one time, either in separated tubes, or in plates. IDs and barcodes are given to any holder used.

3.3.3. Modify experiment. As explained above, the different types of users have different permissions. The administrator can modify any experiment, the superuser can modify any experiment within his/her project and the experimenter can modify only his/her own data.

For data-integrity reasons, some data cannot be modified. For any experiment, the name of experimenter, reference of the experiment and holder (type and barcode) are not modifiable, because too much other information is referring to these. For crystallization plates, all the other experimental parameters are modifiable. This allows adding new drops in a plate that was not full, which happens quite often in the laboratory.

In the near future, these modifications will all be registered, so that the original information is not lost.

3.3.4. Defaults and templates. One obstacle to the use of electronic registration for experiments is the very detailed level of the information asked for. To avoid lengthy and repetitive entries, we have set up a system of default protocols. In any laboratory, the number of different protocols for each step from cloning through to crystallization is very limited; only a few details change from one protein to the other.

There are three options in the default and templates section.

(i) Create a new default file. This allows the definition of a succession of experiments, not necessarily linear (one parent can have more than one child, for example one plasmid might be systematically transformed in two different cell lines) and all details (volumes, temperatures, durations . . .) for each individual step.

(ii) Edit a default file. In this case, an existing default file is opened, the experimental parameters can be modified and the resulting protocol is registered in a new default file. This allows the making of a slightly different version of a same global protocol very easily.

(iii) Use exiting plate as template. This option concerns only crystallization plates; it allows the selectionof existing plates as templates for new plates. One template might be defined as a template for a single user or for all users. When using a template, the experimenter has only few details to define: the proteins or samples used. The volumes, crystallization solutions and drop definitions are copied from the template.

When using a default file, the user does not have to enter any parameter to enter an experiment. However, there is always the possibility to change them when entering the experiment. The default files are there as guidelines and not as mandatory values. This means that if an experiment is performed just once differently from how it is defined in the default file, there is no need to make a new default file for it.

Because of the data model, the designed protocols are graphs and not linear successions of experiments. This means that protocols can have 'branches' that are used or not, depending on the results obtained in the first steps, when entering the experiments. For example, a protein-expression default protocol might contain three parallel transformations in three different cell lines. However, when using this default protocol, an experimenter can use only one of the three. Conversely, if in one particular case, an experiment that is not in the default protocol is needed, it can be added, as a new branch or between two bricks of the default protocol.

3.3.5. Web services. We have also started to include web services. For now, three web services are available and more will come soon. The tools that *HalX* currently offers through web services are all related to cloning.

(i) BestPrimers (http://bip.weizmann.ac.il/sqfbin/bestPrimers) is used to propose primers from the sequence of the target. Fig. 4(*a*) shows the integrated use of BestPrimers in the design of new primers

(ii) suggestES (http://bioportal.weizmann.ac.il/expsysb/suggestES) is a piece of software that suggests an expression system depending on the DNA sequence (Fig. 4*b*).

(iii) verifyCloning (http://bip.weizmann.ac.il/vfclonb/main) checks the sequences obtained after sequencing against the theoretical sequence registered in the database (Fig. 4*c*).

3.4. Viewing and mining data

HalX provides a 'progress page' which allows the user to see the global progress for all targets in all projects. For each target, links are available for a 'detailed progress' page and a page summarizing the data known about the target. This 'target page' contains information from public databases like Pfam, Swiss-Prot, SGD for Yeast *etc*. and links to the same target in these databases.

HalX also provides convenient tools to overview the stored data. For any target, the user can retrieve the list of all experiments on a given target, for all subtargets or for a single subtarget. The list is

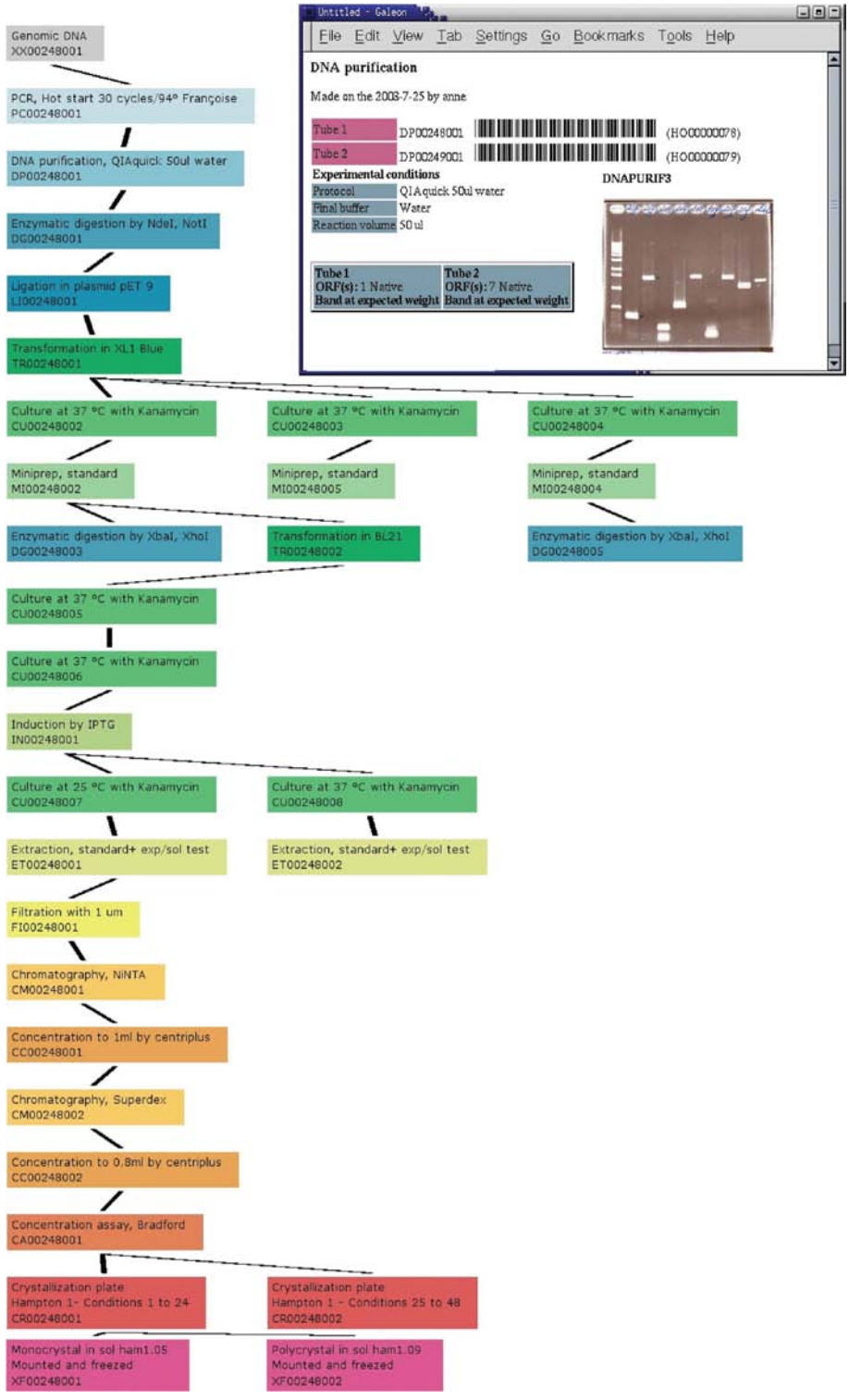

Figure 5
Graphical display of all the experiments made with the protein YPR062w native. Each box corresponds to a single experiment and one main feature is given for it, together with the reference number of the experiment. When clicking on one of these boxes, the user is provided the experimental details (top right).

show as a graph (Fig. 5) illustrating the links between the experiments. Clicking on the corresponding box can retrieve the details of each experiment.

HalX can also offers a global view of all the crystallization experiment made with a given protein (Fig. 6), a summary of all the plates for a given protein can be listed on the screen, together with a table regrouping all the drops containing the protein. This table gives some details about the crystallization experiment such as the composition of mother solution, protein concentration of and any observations on the outcome of the crystallization drop. It also gives information related to the optimization: has this condition been optimized or is this condition an optimization of a previous one? This

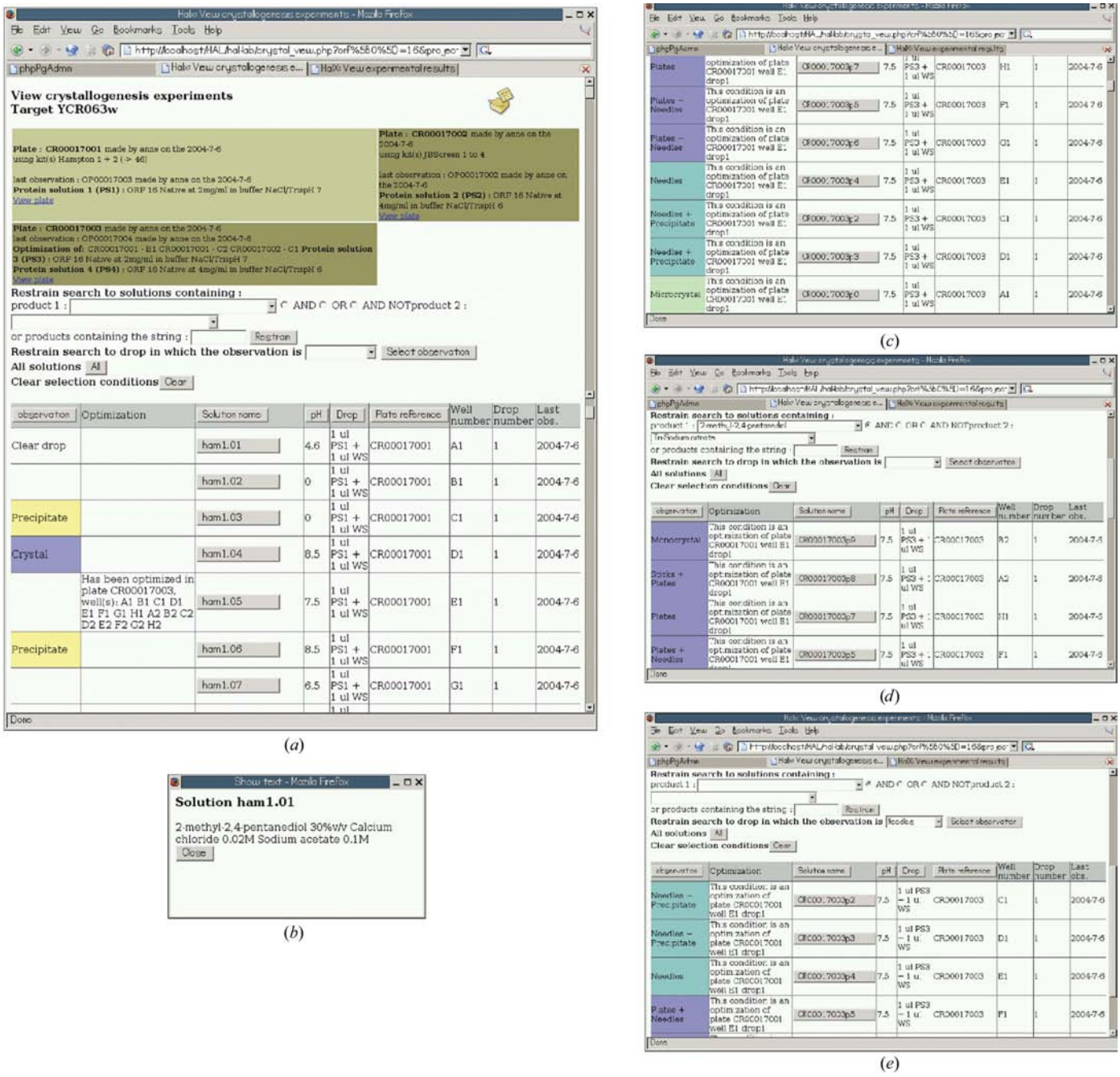

Figure 6
Viewing crystallization experiments for protein YCR063w native. (*a*) At the top of the page, a summary for each plate is displayed, giving the reference of the plate, the date, experimenter and crystallization kit used if any. The reference, date and experimenter of the last observation is displayed, the protein solutions used for the plate are given, and the complete plate can be viewed by clicking on 'View plate'. The table at the bottom of the page contains for each drop in each plate (i) the last observation made, (ii) optimization – this column indicates if the drop has been optimized or if the drop is an optimization of another condition, (iii) name of the crystallization solution – the detailed composition is obtained by clicking on the name and pops up in another window (*b*), (iv) pH of the crystallization solution, (v) composition of the drop, giving the volumes of protein solution(s), crystallization solution, and additives if any, (vi) reference of the plate, (vii) and (viii) well and drop and (ix) date of last observation. The table can be sorted by observation (*c*). The user can select all the drops according to the crystallization solution composition: for the example shown in (*d*) we have asked for all the drops containing 2-methyl-2,4-pentanediol and Tris-sodium citrate. Different Boolean operators can be used (AND, OR, AND NOT). It is also possible to use a string to select for example all the crystallization solutions containing zinc, using the field 'products containing the string'. The user can retrieve all the drops for which the observation is crystal, sticks, plates, needles (example shown in *e*), microcystal, precipitate or other.

table can then be sorted in various ways: according to observation (for instance drops with crystals on top) or according to precipitant or any chemical present in the drops.

HalX is a free-source software (distributed under GPL licence), documentation, demo version and the complete source code are available on the website (http://halx.genomics.eu.org). This software is developed in PHP and the database-management system is PosgreSQL, which are both free-source. New features within *HalX* are forthcoming: the inclusion of new types of experiments (mainly those associated with protein characterization) and installation of new tools (remote control of robots, remote use of data-treatment software).

We are now working on connections to other tools, especially sequence-analysis tools, to be able to update, either automatically or on the fly, the general information concerning the targets.

The research was supported by the CNRS, European Commission Fifth Framework 'Quality of Life and Management of Living Resources' 'SPINE' Project, grant No. QLG2-CT-2002-00988, the Divadol Foundation for Technology Development and the Israel Ministry of Science and Technology Grant for the Israel Structural Proteomics Center. JLS is the Morton and Gladys Pickman Professor of Structural Biology.

References

Bairoch, A., Boeckmann, B., Ferro, S. & Gasteiger, E. (2004). *Brief. Bioinform.* **5**, 39–55.
Bateman, A., Coin, L., Durbin, R., Finn, R. D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E. L. L., Studholme, D. J., Yeats, C. & Eddy, S. R. (2004). *Nucleic Acids Res.* **32**, D138–D141.
Bono, H., Kasukawa, T., Hayashizaki, Y. & Okasaki, Y. (2002). *Nucleic Acids Res.* **30**, 211–213.
Cho, S. Y., Park, K.-S., Shim, J. E., Kwon, M.-S., Joo, K. H., Lee, W. S., Chang, J., Kim, H., Chung, H. C., Kim, H. O. & Paik, Y. K. (2002). *Proteomics*, **2**, 1104–1113.
Dedhia, N. N. & McCombie, W. R. (1998). *Genome Res.* **8**, 313–318.
Fellenberg, K., Hauser, N. C., Brors, B., Hoheisel, J. D. & Vingron, M. (2002). *Bioinformatics*, **18**, 423–433.
Fenyö, D. & Beavis, R. C. (2002). *Trends Biotechnol.* **12**, S35–S38.
Goh, C. S., Lan, L., Echols, N., Douglas, S. M., Milburn, D., Bertone, P., Xiao, R., Ma, L. C., Zheng, D., Wunderlich, Z., Acton, T., Montelione, G. T. & Gerstein, M. (2003). *Nucleic Acids Res.* **31**, 2833–2838.
Haebel, P. W., Arcus, V. L., Baker, E. N. & Metcalf, P. (2001). *Acta Cryst.* D**57**, 1341–1343.
Harris, M. & Jones, T. A. (2002). *Acta Cryst.* D**58**, 1889–1891.
Imbert, M.-C., Nguyen, V. K., Granjeaud, S., Nguyen, C. & Jordan, R. J. (1999). *Nucleic Acids Res.* **27**, 601–607.
Pajon, A. *et al.* (2005). *Proteins*, **58**, 278-284.
Yokoyama, S. (2003). *Curr. Opin. Chem. Biol.* **7**, 39–43.
Zolnai, Z., Lee, P. T., Li, J., Chapman, M. R., Newman, C. S., Phillips, G. N. Jr, Rayment, I., Ulrich, E. L., Volkman, B. F. & Markley, J. L. (2003). *J. Struct. Funct. Genomics*, **4**, 11–23.