Laboratory experiment design: Using Protégé interactively in a runtime-extensible setting

Stephen Meehan, James Tung, David R. Parks,
Wayne A. Moore, Mark Musen and Leonore A. Herzenberg
Department of Genetics, Stanford University School of Medicine

Background

FacsXpert is the first of a planned set of laboratory experiment design tools that use Protégé to capture, express and utilize expert knowledge about experiment details and make that constantly-evolving knowledge available to novice planners. The research workflow for designing complex experiments involves details such as:

- the names and origins of the reagents available for use in the experiment
- the amount of a given reagent to be added to a sample
- the species and organ origin of the sample
- the treatment of the sample donor
- the staining and incubation techniques available for use
- the properties of the instruments available for data collection

Researchers draw on experience and accumulated knowledge to weave these details into a protocol that specifies how and with what an experiment will be conducted. *FacsXpert* facilitates this protocol design by incorporating the experience and knowledge available to expert researchers and provides this experience and knowledge to all users via a highly flexible GUI that mediates between the user designing a protocol and a Protégé knowledge base.

The enthusiasm with which *FacsXpert* has been greeted demonstrates the feasibility of modeling the knowledge necessary to do experiments that, with minor variations, are routinely performed in virtually all biomedical research laboratories. Because we have built this application "iteratively" with the help of bench scientists who are not shy about requesting what they need, this protocol design tool now satisfies many demands. In essence, although still far from its final version, this working system already demonstrates that modeled procedural knowledge can be used to provide real and necessary support for experiment planning in a working research environment.

In the kinds of experiments that *FacsXpert* is designed to design, a set of cell samples are incubated with selected elements of a set of reagents in a microtiter plate or a rack of tubes (first step) and evaluated with a simple instrument, such as an ELISA reader, that generates a single value for each well/tube, or a more complex instrument such as a flow cytometer (FACS)or a DNA microarray reader that generates a separate data file for each well/tube. Variations include multistep incubations in which the reagents are "washed" away and a new set of reagents added; changes in incubation conditions at each step; fixation and permeablization at a particular step; and, the use of different instruments to collect the data.

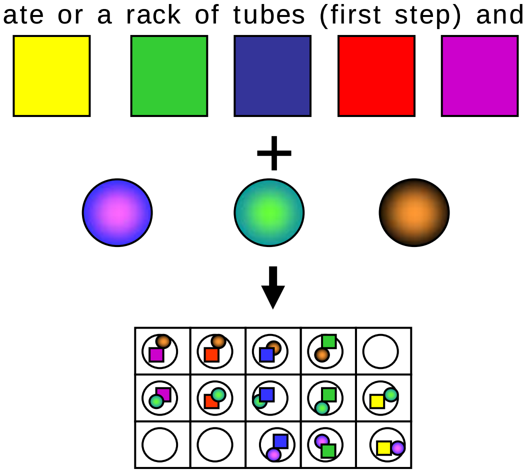

Figure 1. Typical pipetting plan.
Reagents (squares) and cells (spheres) are combined (pipetted into) in plate wells or tubes and incubated. Some combinations may be omitted.

Thus, to plan an experiment, it is necessary to have a knowledge base that "knows", for example, that reagents must be selected from a list of those available; that samples have identifiers; that reagent and sample addition volumes must be calculated; that data collected for each well/tube must be associated with the sample and reagent descriptions for the well/tube, etc. This knowledge base constitutes a generic superclass to which knowledge specific to a given instrument domain is subclassed.

The FacsXpert Architecture

FacsXpert provides a JAVA swing interface that hides much of the Protégé mechanics and presents knowledge to the user in a way that allows creation of experiment protocols in a "language" familiar to the scientist. Some additional features of this architecture are summarized below.

Run-time extensibility. The dynamic runtime modeling capability of Protégé assists FacsXpert in adapting to the FACS expert's constantly evolving knowledge. By basing much of its behavior on the runtime interpretation of the Protégé knowledgebase, FacsXpert is more adaptable to a wider range and more rapid pace of knowledge by allowing end users to evolve the schema for the knowledgebase. For example, by extending the species class hierarchy, scientists can capture and use more detailed information about their subjects and reagents than the software engineers foresaw as useful. To make such flexibility safe, *FacsXpert* embeds its own domain-independent Protégé tool named Darwin, which safeguards model evolution by clarifying the direction/pathway along which a given model can evolve without breaking any expectations defined by JAVA or other code used, for example, to validate data integrity.

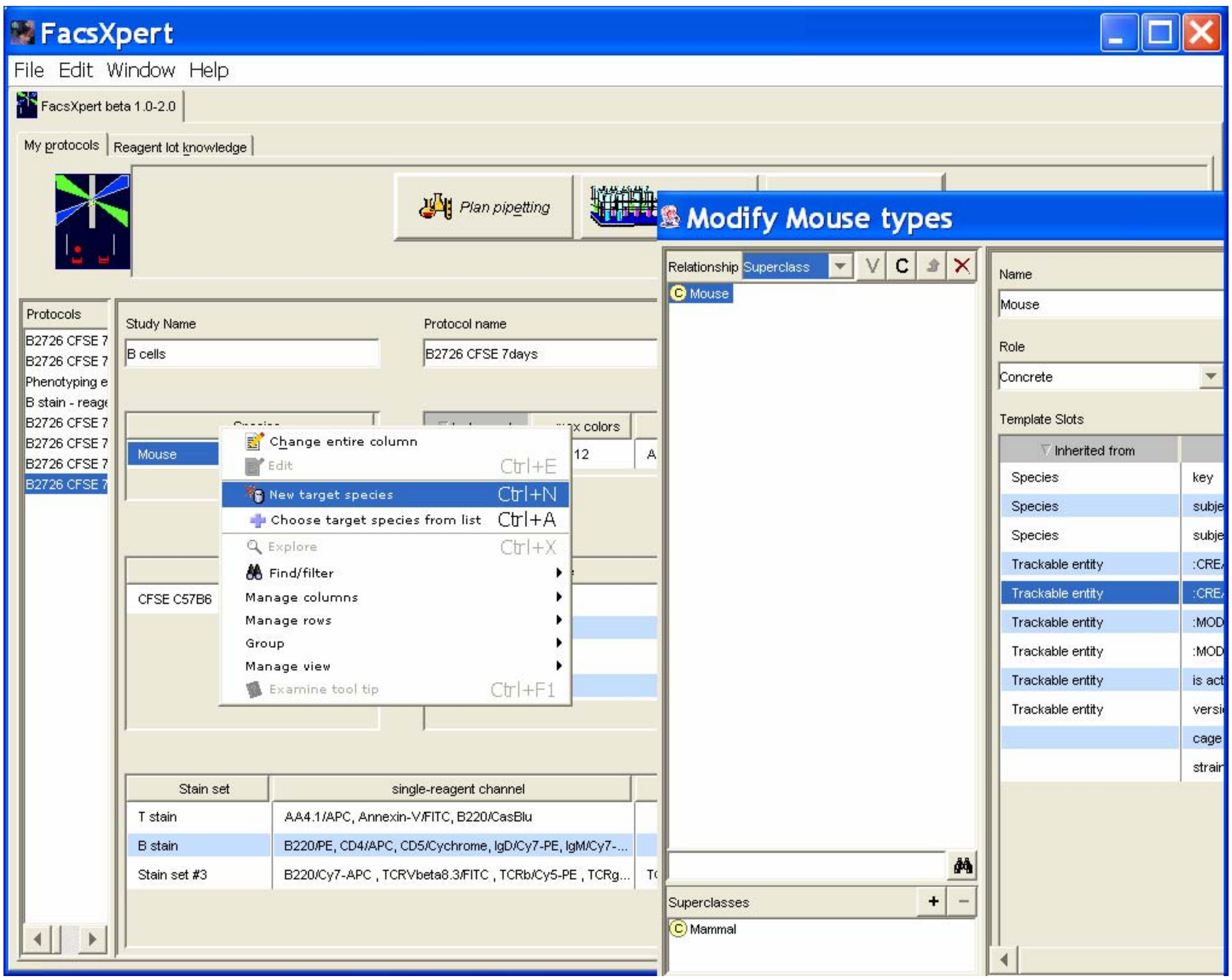

Figure 2. Runtime extension of the Protégé knowledge base.
On this *FacsXpert* Protocol master form, the user can evolve the species class hierarchy by right clicking on the target species slot widget and choosing **New target species**. This action pops up a dialog box containing the standard Protégé class panel. Here the class panel supports the normal modification operations but it confines the user to specific subset of the model starting at the class: Mouse.

Validity checking. FacsXpert also has a domain-independent framework to simplify validity-checking. With this tool, the engineer associates a validity test with a specific model context. At runtime Protégé decides when and how to execute these tests. For example, the Protégé forms and pick lists use this framework to enforce data integrity. This not only makes these visual devices more useful, but it also makes them more stable when faced with new validation requirements.

While this framework does not remove the need to program a module to enforce a new "rule", it eliminates much of the programming needed to:

- Ensure that this rule is consulted at the appropriate time
- Bring this rule to the user's attention in a consistent and clear way.

Reusable GUI widgets. To increase scientist productivity, FacsXpert has a number of new Protégé widgets that simplify user interactions and allow more personalization of the look and feel. These include a tree/grid widget which is convertible by the user, while *FacsXpert* is running, to a tree structure for readily visualizing, accessing and modifying the contents of a particular hierarchy.

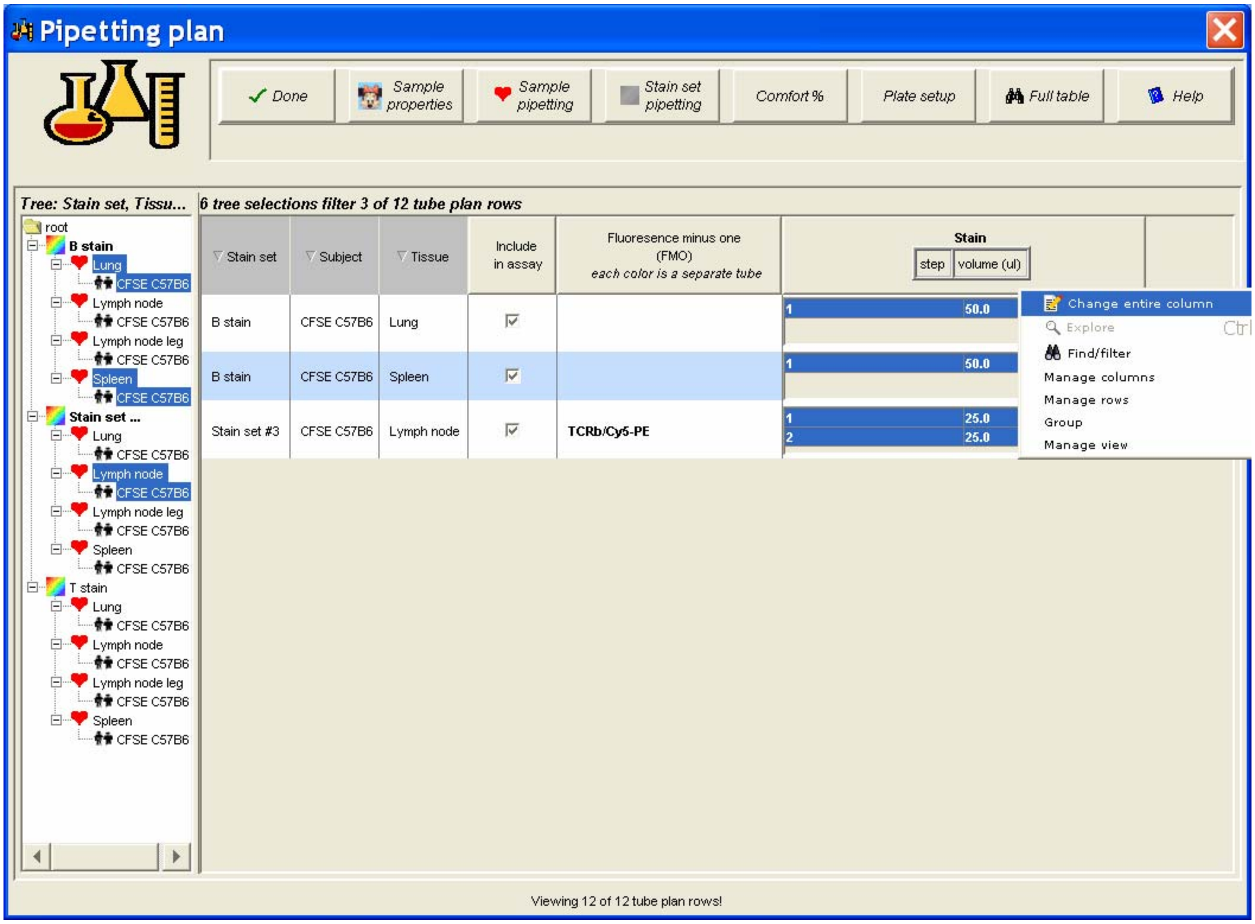

Figure 3. Runtime modification of knowledge display format. The FacsXpert grid slot widget allows users to define tree abstractions of tables for the purpose of visualizing and modifying groups of rows. In this figure, the user is working with a table of tube configuration rows and has arranged them in a 3 level hierarchy using the 3 columns: stain set, subject and tissue.

Conclusion. Experience to date with FacsXpert, which is in beta 2, indicates that laboratory scientists can rapidly build a complex protocol and use HTML reports to provide detailed direction to technical personnel for executing the experiment. In addition to making experiment design easier, estimates for time saved by using *FacsXpert* vary from one half to three hours per experiment. Thus, *FacsXpert* has understandably been well received.

The success of *FacsXpert* demonstrates that Protégé, with the tools we have added, is an effective runtime framework for applications in highly dynamic knowledge domains. The validity checking tool allows engineers to expand the integrity checking of an application with minimal programming. The Darwin tool allows users to safely evolve the model while the application is running and the GUI widgets make it possible for users work productively with the knowledge in the model. Together, these tools bring Protégé to a new level of user interactivity and extensibility demonstrated, in this presentation, by *FacsXpert, a* tool for facilitating laboratory experiment protocol design.