Journal

Bioinformatics, 27(3)

ISSN

1367-4803

Authors

Huang, Y Wayne

Arkin, Adam P

Chandonia, John-Marc

Publication Date

2011-02-01

DOI

10.1093/bioinformatics/btq676

BIOINFORMATICS APPLICATIONS NOTE
Vol. 27 no. 3 2011, pages 437–438
doi:10.1093/bioinformatics/btq676

Databases and ontologies

WIST: toolkit for rapid, customized LIMS development

Y. Wayne Huang[1,2], Adam P. Arkin[1,2,3] and John-Marc Chandonia[1,2,4,*]

[1]Physical Biosciences Division, Lawrence Berkeley National Laboratory, [2]Virtual Institute of Microbial Stress and Survival, [3]Department of Bioengineering and [4]Department of Plant and Microbial Biology, University of California, Berkeley, CA 94720, USA

Associate Editor: Martin Bishop

ABSTRACT

Summary: Workflow Information Storage Toolkit (WIST) is a set of application programming interfaces and web applications that allow for the rapid development of customized laboratory information management systems (LIMS). WIST provides common LIMS input components, and allows them to be arranged and configured using a flexible language that specifies each component's visual and semantic characteristics. WIST includes a complete set of web applications for adding, editing and viewing data, as well as a powerful setup tool that can build new LIMS modules by analyzing existing database schema.

Availability and implementation: WIST is implemented in Perl and may be obtained from http://vimss.sf.net under the BSD license.

Contact: jmchandonia@lbl.gov

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on December 23, 2009; revised on November 11, 2010; accepted on November 26, 2010

1 BACKGROUND

Laboratory information management systems (LIMS) are important in management and tracking of laboratory processes, not only to provide status and results in an organized and consistent manner but also to assist in quality assurance of lab experiments. LIMS capture experimental data in a structured format, e.g. a relational database, to enable data validation and facilitate automated analyses. Large multilab research projects often have very complex LIMS requirements that may evolve over time as the laboratory processes within the project mature. Customizing a LIMS to store and display data specific to each lab's experimental protocols can be costly (Morris *et al.*, 2008), especially when there is rapid turnover in experimental procedures, as is often the case in academic or research prototyping environments. WIST is a toolkit that enables customized LIMS modules for data storage and display to be rapidly developed, deployed and modified in response to changing experimental designs. Although WIST is useful for efficiently developing LIMS prototypes, it may not be suitable for commercial production facilities, as it does not include typical enterprise features such as electronic laboratory notebooks, project tracking, automated data import from instruments, automated reports or tools to ensure regulatory compliance.

WIST enables developers to rapidly build a modular LIMS, with individual web-based forms to capture and display data related to each experiment or group of closely related experiments (called a 'workflow'). These data are generally stored in a relational (SQL) database, for which the schema must be designed prior to using WIST to create a LIMS user interface. WIST also provides a repository to store larger objects, such as uploaded files. All WIST-based LIMS modules share a common user interface, which links data from related workflows (e.g. a prior or subsequent stage in a pipeline) and provides common features such as searching, data filtering and sorting that are essential for navigating the volumes of data produced in a high-throughput research environment.

WIST modules are configured using text-file templates, which may be generated automatically using the included *WIST Builder* tool, or written in a flexible language that specifies only the structure of the workflow and the visual and semantic attributes of each input component. This greatly facilitates the process of building and modifying modules, which is essential for the iterative process of defining and refining workflow specifications. Rapid template-based creation and customization of new modules is a key feature of WIST that distinguishes it from other open source LIMS software such as Sesame (Zolnai *et al.*, 2003) and free-to-academics systems such as PIMS (Troshin *et al.*, 2008).

2 FEATURES AND CONFIGURATION

WIST includes a set of web applications and tools to perform the most common LIMS tasks, including adding, editing and viewing single workflow data entries, browsing and searching data and creating new workflows from existing database schema. Several of these applications are shown in Figure 1. More details and examples are provided in the Supplementary Material.

Users upload data through the *WIST Upload* tool, which enables data entry using a web form that may include a variety of component widgets. These widgets are described in the template, and are customized by the developers to correspond to various data types in the corresponding SQL schema. For example, to enter a date, either a pop-up calendar widget or a text box may be used. Drop-down menus allow users to choose options from a list that may be enumerated in the configuration file or retrieved dynamically at run time from the database. These menus may also be configured to allow users to add new values to the list of options, which can be stored in the database for subsequent use. Other widget types include checkboxes, radio buttons, file upload boxes and various text entry fields. Several of these widgets support multiple inputs; e.g. the multifile upload allows users to upload a fixed or variable number of files, within a range specified by the developers. All widgets include AJAX functionality, which allows data input into one widget to immediately change the values or number of inputs in another widget, based on a function and/or database query. Data entry may be validated (e.g. allowing only a range of values or preventing

*To whom correspondence should be addressed.

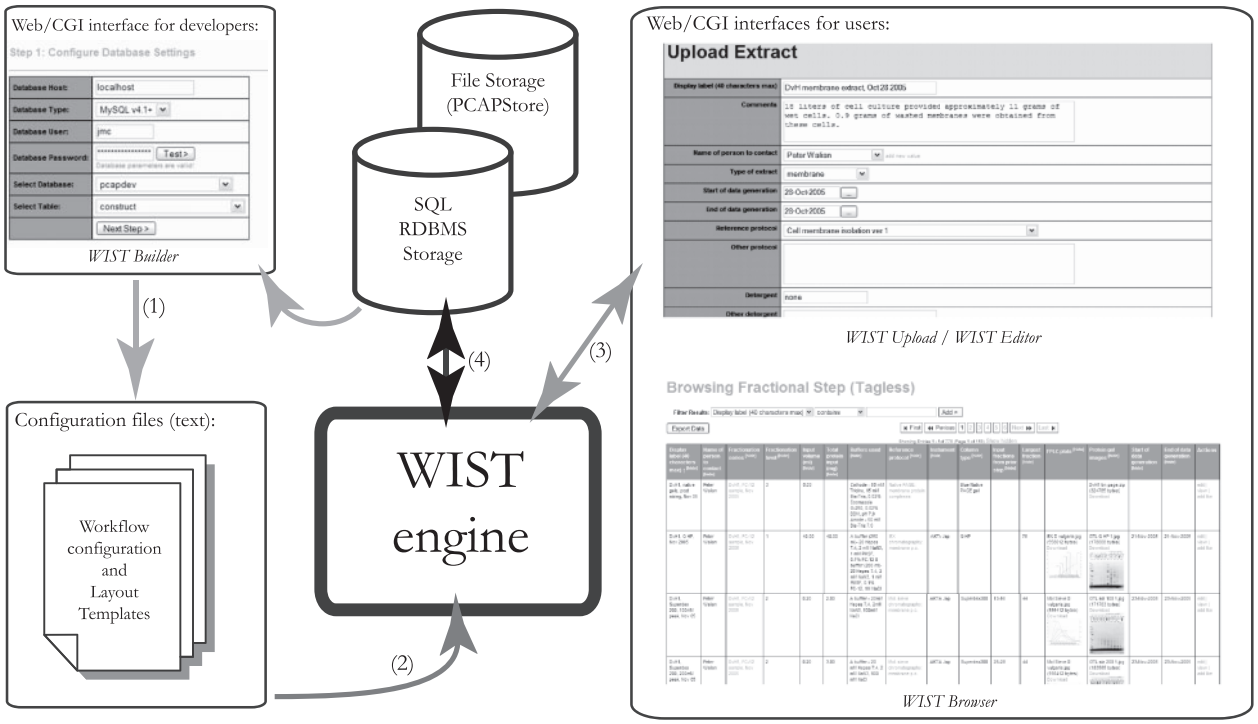

Fig. 1. WIST features and architecture. Each workflow is described in a template (bottom left); visual style may be configured separately using a layout template. Templates are text files that may be built from SQL schema using the *WIST Builder* tool (1, top left). The WIST engine parses the templates (2, gray arrow) and interacts with users through several integrated web/CGI interfaces (3, gray arrows); additional interfaces are described in the Supplementary Material. The data storage subsystem (4, black arrows) allows data to be stored in a SQL database or in the file system.

floating point numbers from being entered as an integer data type); invalid data entry results in a customized, user-friendly pop-up box, and the problematic field being visually highlighted. *WIST Upload* also allows parsed data fields, a feature that allows users to upload a file (e.g. spreadsheet or text file) corresponding to multiple rows in the database; this feature is especially useful for high-throughput data entry.

WIST data entry forms may include multiple steps and branches before data are finally committed to the database. Branching logic allows navigation to subsequent input steps based on data provided in all prior steps. Each step may also specify a custom display template, which is used to render the web-based form corresponding to the step. The included (default) templates automatically provide all necessary navigation links and may easily be customized to alter the look or style of the web pages. Both WIST configuration and layout templates, including the sections that define SQL statements for database operations, are built using an advanced template language that supports basic macro replacement, if/else conditional logic, for each iteration logic, and various string and array transformation functions.

3 REQUIREMENTS AND ARCHITECTURE

WIST back-end installation requires Perl 5, MySQL 4.1+ or PostgreSQL 9+ and Apache 2. All applications are built on WIST's APIs, which allow developers to easily extend the functionality of WIST as needed, e.g. to add new input components and/or validation methods or to add support for new relational databases. Internally, the APIs handle parsing and rendering of workflow templates, tracking session state, rendering and providing validation for input components and interfacing with relational databases and storage subsystems.

WIST also includes a convenient web service-based content store called PCAPStore, which is useful for storing and retrieving large data files outside the relational database.

ACKNOWLEDGEMENTS

WIST was originally developed to support the Protein Complex Analysis Project (PCAP), part of the Virtual Institute for Microbial Stress and Survival (VIMSS). Thanks to all members who contributed suggestions and feedbacks. Ralph Santos wrote PCAPStore, packaged WIST, and helped debug WIST.

Funding: This work conducted by ENIGMA was supported by the Office of Science, Office of Biological and Environmental Research, of the U. S. Department of Energy under Contract No. DE-AC02-05CH11231.

Conflict of Interest: none declared.

REFERENCES

Morris,J.A. *et al.* (2008) A Perl toolkit for LIMS development. *Source Code Biol. Med.* **3**, 4.

Troshin,P.V. *et al.* (2008) Laboratory information management system for membrane protein structure initiative–from gene to crystal. *Mol. Membr. Biol.*, **25**, 639–652.

Zolnai,Z. *et al.* (2003) Project management system for structural and functional proteomics: Sesame. *J. Struct. Funct. Genomics*, **4**, 11–23.