

Introduction

The Internet contains an enormous amount of information which is updated frequently. However, there are not many effective ways for Internet users to locate and follow such information. The Atom standard, an implementation of the Extensible Markup Language (XML), is a solution to this problem, as it systematizes the format for informing users of updates to periodical websites and other information available over the Internet.

In compliance with the Atom 1.0 format, the GIST generates Atom feeds for HyperText Markup Language (HTML) and Extensible HyperText Markup Language (XHTML) documents, edits existing Atom feeds, and uses an autodiscovery mechanism to seek Atom, Really Simple Syndication (RSS), and Resource Description Framework (RDF) feeds on the Web.

The Web syndication project comprised four subprojects, namely, Hydrogen, Helium, Lithium and Neon. Hydrogen and Lithium generate feeds, with Lithium providing feed-entry editing options. Helium and Neon deal with extracting information from feeds. Neon also incorporates a distributed search engine that discovers feeds from the Web and displays them. The project entailed extensive use of the Simple API for XML (SAX) and HTML parsers, the FatDog XQuery Engine and the Java Swing utilities. Neon also makes use of the Plazoo feed search engine. All software development was carried out in the spirit of Extreme Programming, code generation and implementation relying heavily on pair programming and unit testing.

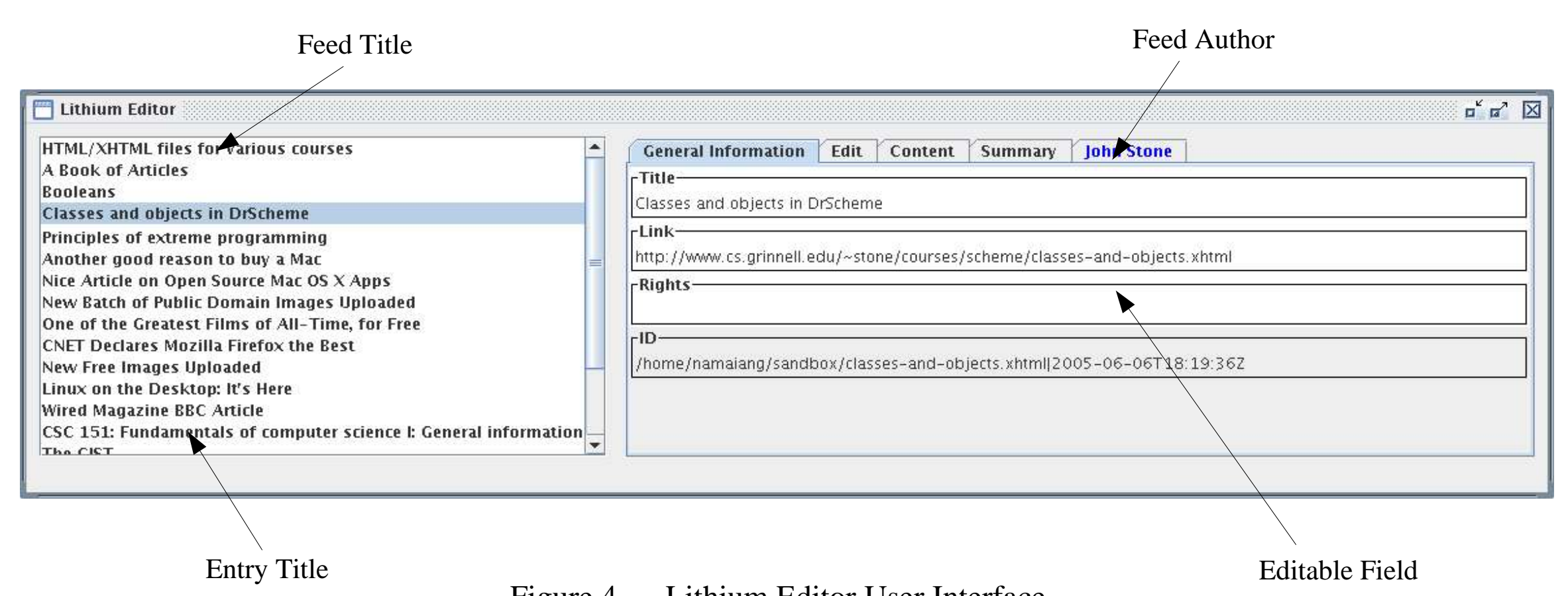


Figure 4. Lithium Editor User Interface

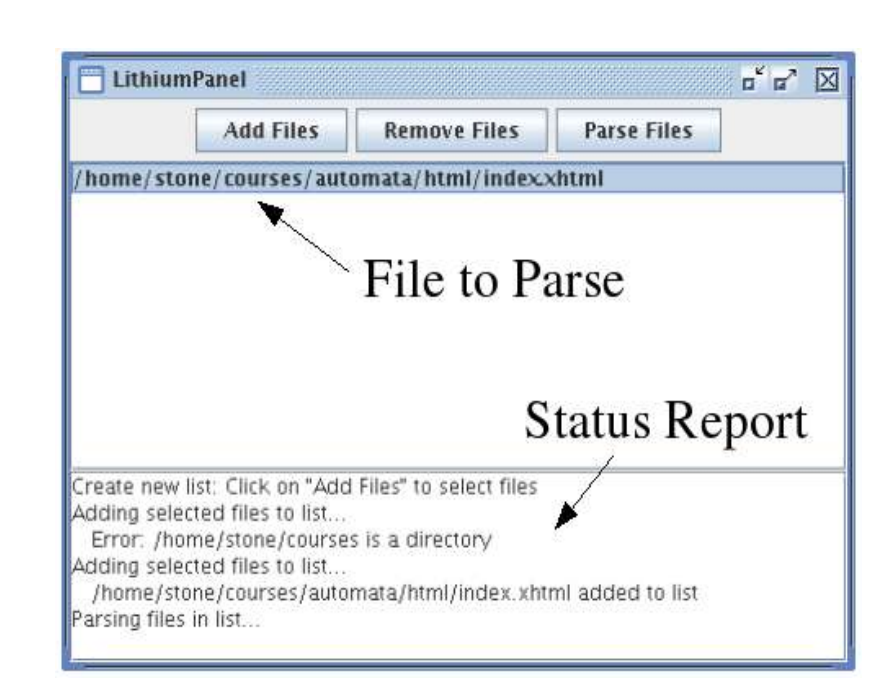


Figure 3. Lithium Panel User Interface

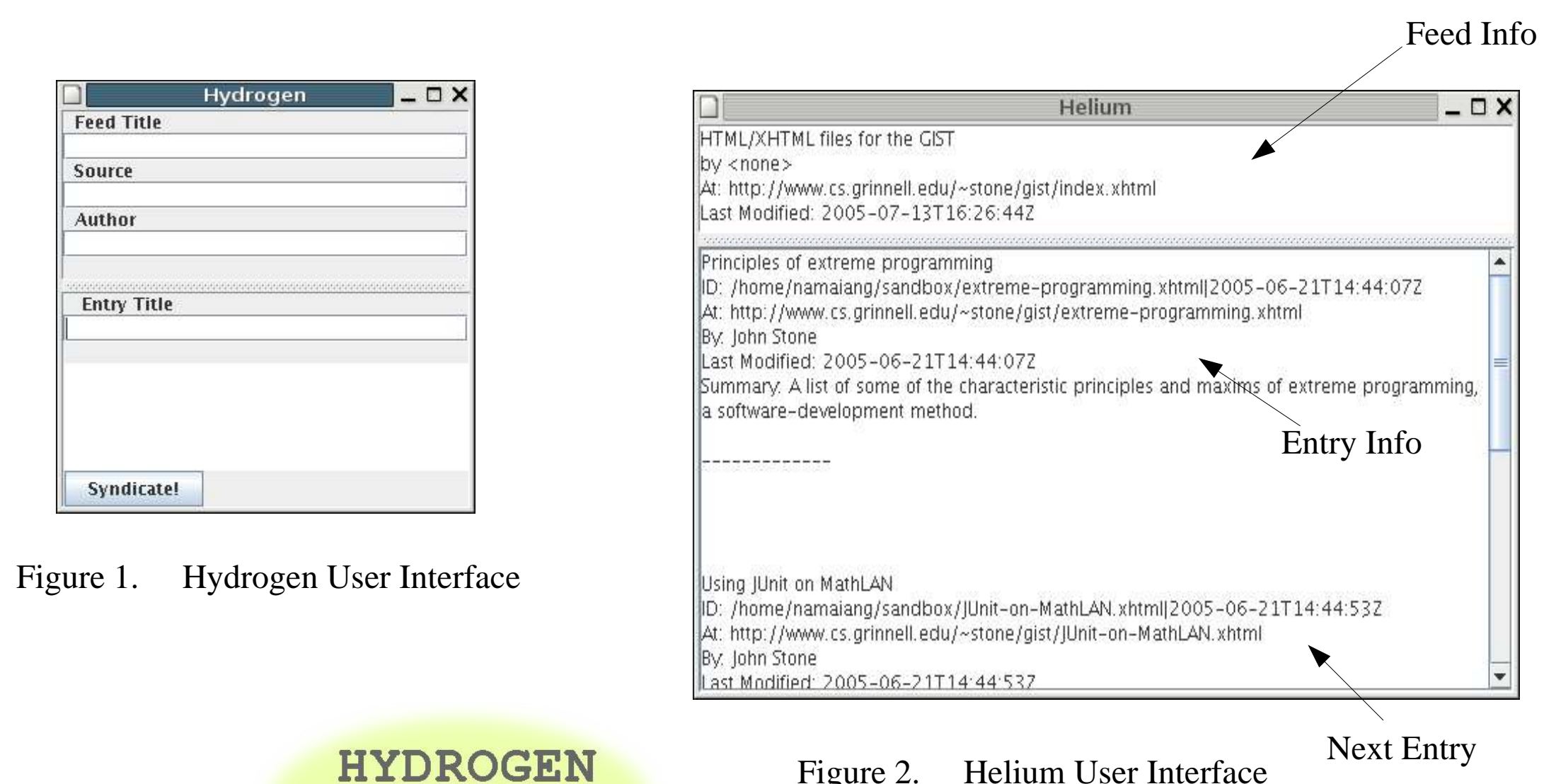
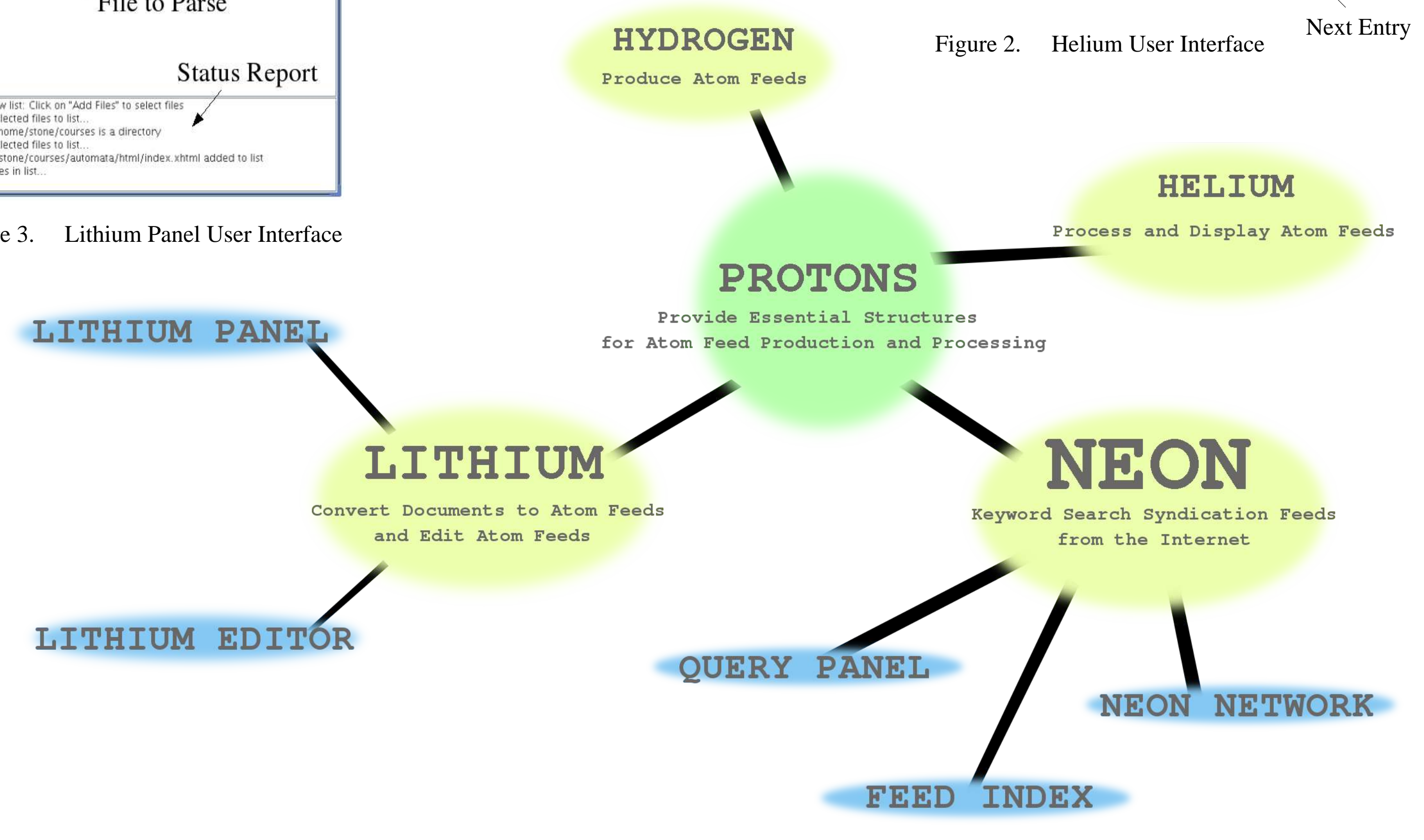


Figure 1. Hydrogen User Interface

Figure 2. Helium User Interface



Discovering and Parsing Feeds

Helium uses the SAX parser to extract information from an existing Atom feed. The feed's title, author, link, and most recent date of modification, and each entry's title, ID, link, author, most recent date of modification and summary are extracted and displayed on the Helium GUI. These data are then stored in an Atom feed container to provide access to feed elements for future feed editing using Lithium.

Neon is a connectionless, peer-to-peer search engine for Atom, RSS and RDF feeds. A Neon user relays his or her search parameters to other users, who then return feeds which meet the given parameters back to the searcher. If no appropriate feeds are found, the searcher will then query the Plazoo feed search engine with the same query and scan for feeds. During the scanning process, Neon indexes the list of feeds (whether relevant to the search terms or not) by keyword, and adds these keywords to the library. If feeds that in Neon's judgment match the query are discovered, they are displayed on the QueryPanel for the user. If not, at least there are now more feeds indexed in the Neon collection for other users to benefit from. Since it is unrealistic to send queries to all Neon users, a given user queries only a small subset of the entire Neon community. If the user finds that he or she is consistently not getting results from some community member, or that the results are not worth reading (or worse yet, do not even exist), Neon will attempt to rectify the situation by asking more prominent members of the community who they query and following those suggestions. In time, this will adapt a user's list of other users to query to one which is optimized for the search patterns of that user.

If search results are returned to the QueryPanel for display, the user will be able to follow the links to view feed information. These links lead to a list of links to the feed entries on which the user may click to view the entries in a browser. The user may also choose to view the feed as an XML document by clicking on the link provided as the header of the entry links window.

Potential and Actual Uses

- ★ Users can create minimal Atom feeds using Hydrogen.
- ★ Helium reads any Atom file and compactly displays feed information for the user.
- ★ Grinnell Math/CS faculty use Lithium to retrieve meta data from Web documents, from which they create and maintain associated Atom feed documents containing course information.
- ★ Neon will assist students and faculty in obtaining and subscribing to informative feeds from the Web.

Future Extensions

- ★ Implement a smoother-working Lithium interface that overcomes the minor technical problems of the Java Swing classes. For example, the current file chooser does not consistently respond to double-clicks when opening directories.
- ★ Improve the speed and accuracy of the Neon search engine.
- ★ Find alternative ways to build the Neon library. At present, the library obtains links to feeds only from the Plazoo search engine, and does not connect to other feed-rich sites.
- ★ With the ever-increasing number of links to feeds found on the Web, it might be necessary to move the Neon library from a user's account to a stable database.
- ★ Integrate the four sub-projects to enable the user perform all Web syndication tasks from one application.

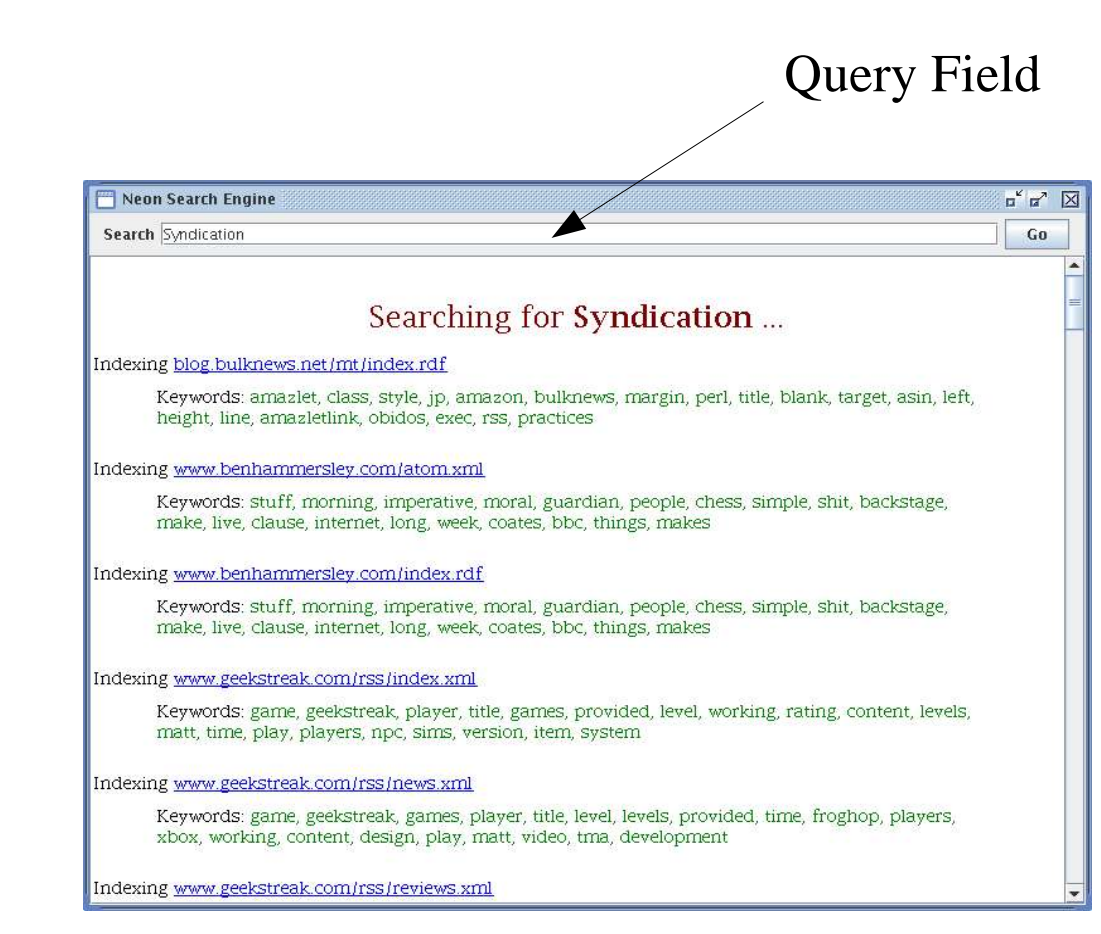


Figure 5a. Neon: Indexing Links to Feeds

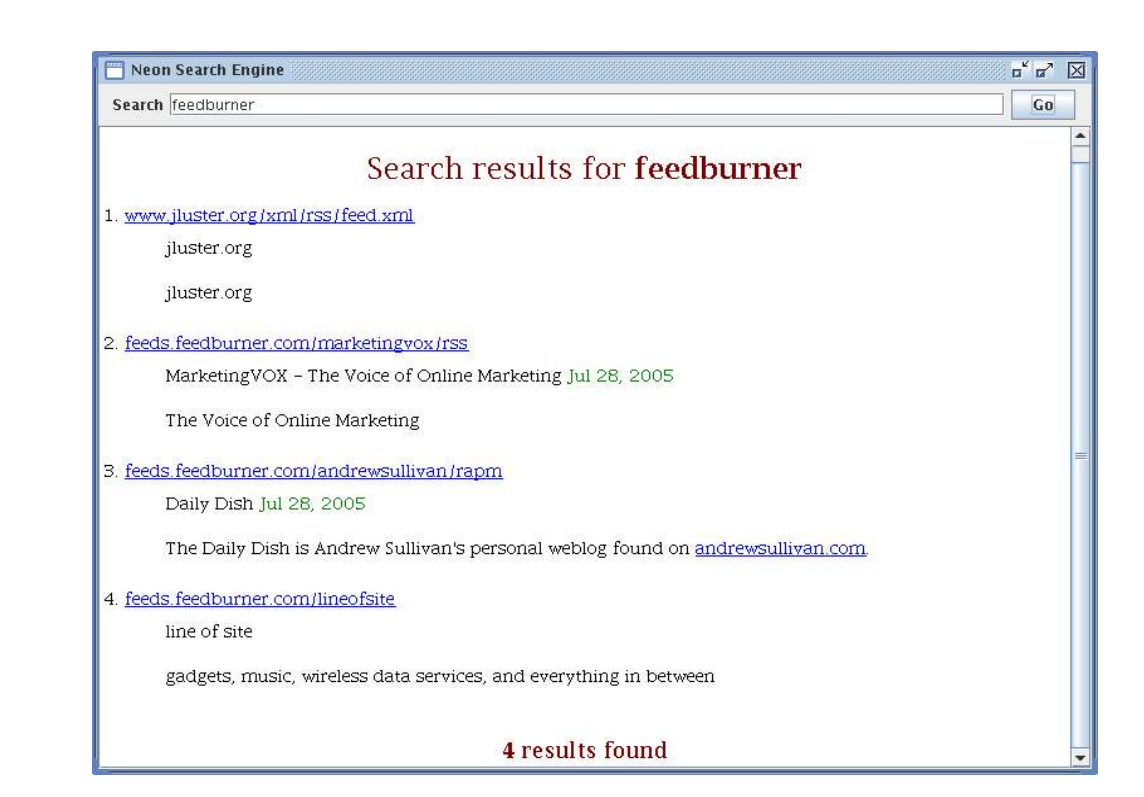


Figure 5b. Search Results

Generating and Editing Feeds

Hydrogen is a simple Atom feed generator whose interface allows the user to specify a new feed's title, link, author, and initial entry set-up information while automatically collecting the feed creation time. Since Hydrogen is a prototype, it only supports the creation of single-entry feeds.

Lithium extends Hydrogen to generate multiple-entry feeds and to provide a feed-entry editor. The Lithium interface allows users to create feeds from readable Atom, HTML, and XHTML files. The user first selects a list of files to be parsed, which may be specified through a terminal command line, or through the LithiumPanel startup window. Lithium then uses an HTML parser to extract the title, source link, creation and modification dates, summary, and author information from any HTML and XHTML files specified. The data for each of these files are stored in an Atom entry container. If there are multiple Atom files in the list, entries from these as well as those corresponding to the HTML and XHTML files are added as entries to the new feed. If no Atom file is specified, the entries derived from the HTML and XHTML files are inserted into the new feed. Otherwise, the single Atom file specified in the list is the feed to which produced entries are added.

Upon feed generation, the feed-entry editor allows users to make changes and corrections to the feed and entry titles and source links. The Lithium editor also provides the option of editing the feed's rights field (which specifies the author's copyright terms), and saving those rights to all entries. The user may also edit author information to be saved to all entries that share an author.

Acknowledgments

The GIST team would like to recognize their mentor, Dr. John Stone.

Bibliography

Albing, Carl, and Michael Schwarz. *Java Application Development on Linux*. Boston: Prentice Hall Professional Technical Reference, 2005.

Ayers, Danny, and Andrew Watt. *Beginning RSS and Atom Programming*. Indianapolis, IN: Wiley Pub, 2005.

Cornell University. *SMART* (list of common English words)
<http://ftp.cs.cornell.edu/pub/smart/english.stop>

Nottingham, M., and R. Sayre, eds. "The Atom syndication (draft version 10)." July 11, 2005.
<http://www.ietf.org/Internet-drafts/draft-ietf-atompub-format-10.txt>

Wake, William C. *Extreme Programming Explored*. Boston: Addison Wesley, 2002.