Stanford Exploration Project, Report 93, November 11, 1997, pages 209–215

Short Note

SEP documents and software on the web

Matthias Schwab and Joel Schroeder[1]

keywords: *graphics, SEPlib, algorithm, processing*

The SEP web site offers SEP reports 70 through 91 to our sponsors. Furthermore, it presents older reports and Jon Claerbout's books to the general public. Claerbout's books and the most recent reports are offered in a simple, albeit powerful format: table-of-contents pages present a short summary of each chapter, a hyperlink to the Postscript version of the paper, and another hyperlink to a tar file that contains the author's entire source directory. In principle, such a source directory enables the reader to recompute effortlessly the author's results, if the reader has a basic set of SEP tools.

SEP also offers all of its software tools on its web pages. These tools include the basic set necessary to reproduce SEP documents. For each package, a short HTML page describes its purpose, installation, and usage. Conceptually, the SEP document and tool pages export the entire SEP software environment to our sponsors and (with the exception of the recent reports) to the rest of the world.

FREE SEP SOFTWARE PACKAGES

SEP offers the following free software packages on its web page. Each item is documented in more detail on an HTML page at the SEP web site.

- **ReDoc Rules:** The Reproducible Document Rules are makefile rules that implement a consistent and simple interface for deleting and recomputing a document's results. The ReDoc rules work best with an updated version of GNU make, which we also offer on our web site. Additionally, we distribute make rules for the compilation, linking, and installation of programs.

- **SEPlib:** SEPlib is SEP's processing system. It currently consists of the traditional SEP77 part that deals with regular data geometry and a new SEP90 part that handles irregular data geometry. SEPlib also contains an entire graphics package, called vplot, and extensive I/O routines.

[1]**email:** matt@sep.stanford.edu, schroeder@alumni.stanford.org

- **LaTeXmacros:** The TeX web page includes SEP's local LaTeX macros, e.g. the `activeplot` macro that includes figures and pushbuttons into our online LaTeX documents.
- **SEP Scripts:** The SEP shell scripts serve many purposes: some test a reproducible document, others print a LaTeX document. If nothing else, they might serve as sample Perl scripts.
- **Xtpanel:** Xtpanel is SEP's language of choice for scripting simple graphical user interfaces.
- **Ratfor:** Ratfor is a preprocessor for FORTRAN code that allows us to use C-like flow expressions. Jon Claerbout believes that Ratfor is the best available expository language for mathematical algorithms. Ratfor is widely used by SEP researchers.

Software environment

At the Stanford Exploration Project, we have combined various software tools into a software environment for research of seismic data processing. Figure 1 shows the electronic version of a reproducible research document. Such an electronic document represents a directory that contains the traditional document and all the software needed to compute the included results.

The window at the left displays a TeX document. Any figure caption in that document includes a push-button that, when pressed, issues a system command to the displaying computer. By default, that system call starts the graphical user interface that is shown to the bottom right of the screen-dump. The graphics interface allows a reader to choose several actions: by default the reader can remove, build, and view the figure. Additionally, a reader can clean the directory of all unnecessary, intermediate files or individually inspect the files underlying the document.

A typical project at SEP starts with an idea and a seismic data set to test that idea. Usually, the data set is already stored in our preferred, home-brewed SEPlib format. An input data set is processed by a sequence of simple, standard SEPlib routines that are governed by command line parameters and combined by UNIX pipes. A researcher's new process idea usually requires a new program. If implementing this program in C or Fortran, the researcher can take advantage of a rich subroutine library that allows him to integrate his own program seamlessly into the SEPlib processing sequence.

The author stores the commands that constitute such a processing sequence in a GNUmakefile. The GNUmakefile includes all the rules to manage the resources the author needs. These resources may include a C or Fortran compiler, Perl interpreter, or various preprocessing programs such as Ratfor. To avoid rewriting the same rules over and over again, every researcher includes a set of SEP-wide rules and variable definitions into his GNUmakefile.

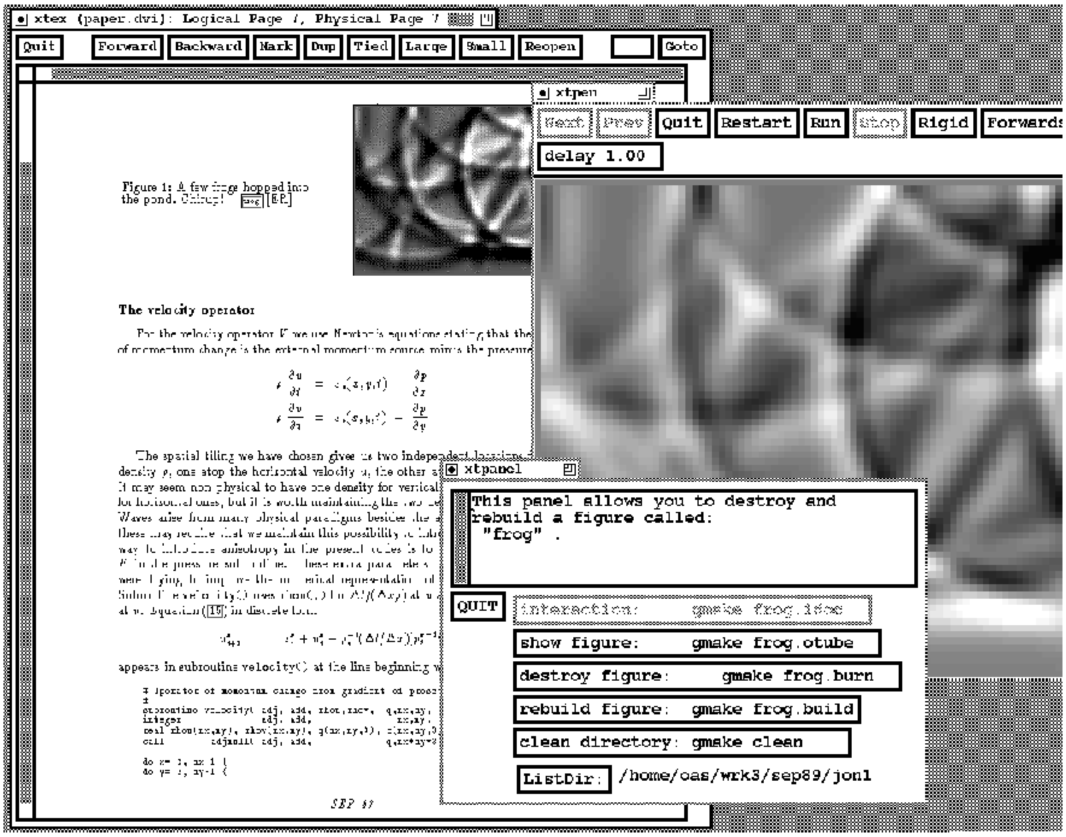

Figure 1: The reader interface for reproducible research is only one component of SEP's current computational research environment: A research document at SEP is written in LaTeX (Postscript version visible in the background to the left). Using SEP's own LaTeX macros, a push-button in each figure caption invokes a graphic user interface written in a script language called *xtpanel*. The graphic user interface enables a reader to interactively execute the `burn,` `build,` `clean,` and `view` commands for each individual figure. (The panel is shown in the foreground. The result of `make view` is shown towards the right.) SEP's ReDoc rules allow an author to easily extend the interactivity of a result figure to additional, application-specific actions. matt3-xtpanel [NR]

Additionally, authors write a LaTeX document that describes their research and their findings. As in many fields of computational sciences, the results in seismic processing are illustrated by figures. At SEP, special TeX macros include a push-button in any figure caption. That push-button, when pressed, issues commands to the GNUmakefile of the document's underlying software directory. It is that connection which allows a reader to inspect and reuse an author's software efficiently.

Software installation

SEP's software installation is comparable to installations of similar, publicly available packages. All the software employs makefiles and common commands such as `make install` or `make clean`.

However, improvements in the installation of GNU software (such as Perl, Emacs, and gcc) and the dynamics of the World Wide Web are challenging SEP to simplify its software installation mechanisms. SEP's major obstacles to more seamless installations stem from differences in platforms and user environments, and from the interdependencies of our software tools.

To deal with platform dependencies elegantly, GNU software uses configure scripts. Before installing a package, the configure script queries the system (e.g., it finds the available C compiler) and creates a system-dependent makefile. In general, the user installing the software does not need to think about platform dependencies. Configure scripts have tremendously eased the installation of many public domain software packages.

UNIX software that relies directly on local resources, such as printers and temporary disk space, are inherently difficult to export. On a remote system, the `tmp` disk space might be too small and the printer name is probably different from the local system. Reasonable standards (such as a default printer name "Postscript") could help, but UNIX lacks such standards. Consequently, packages are usually accompanied by README files that instruct users on how to change their environment appropriately. Such changes can be non-trivial since they often depend on the user's shell, and they can conflict with definitions due to other existing software.

On the other hand, SEP could use Sun's public domain software *Modules*. Modules supports shell independent environment definitions for individual packages, avoids naming conflicts, and therefore offers a convenient method of exporting environment variables. Unfortunately, users who download SEP software would be forced to install and use Modules on their computer system.

In contrast to much public domain software, SEP's packages are inherently hierarchical. For example, an SEP document will only function after SEPlib and the ReDoc rules have been installed. These dependencies are unknown to a novice to SEP software. The problems could be avoided if SEP would distribute all of its software as a single monolithic package (something SEP has done when creating CD-ROMs).

However, because of the Internet's limited bandwidth, such large packages are difficult to distribute on the web. Alternatively, SEP could extend the configure concept: when installing an SEP software package, the configure script could check the local system and inform the user about additionally needed SEP software.

Java offers a very different alternative to exporting software on the web. Java's portability avoids the entire problem of platform dependencies. Java's security manager and its limited access to resources on a client recast the traditional environment problem in a very different and flexible way. Finally, Java's ability to dynamically load additional tools on the fly addresses the problem of complex software dependencies.

SEP DOCUMENTS ON THE WEB

The web version of each SEP document (a report or book) contains an HTML abstract, a Postscript version of the complete article or chapter, and a tar file with all the files that constitute the reproducible document directory. Furthermore, a web version of a report or book may contain the source of a program library or a data directory. We have three simple scripts to convert future SEP reports into our web format: `paper2html, book2web, report2web`. SEP reports earlier than report 70 are not reproducible and are represented by their ASCII source files.

We would prefer to display SEP documents in HTML rather than Postscript. Unfortunately, HTML does not yet support mathematical expressions. Since LaTeX typesets mathematical expressions beautifully, SEP will continue to write its documents in LaTeX. To transfer such documents to HTML routinely, SEP needs to maintain a `latex2html` enhancement for each of SEP's LaTeX macros. Consequently, we suggest a drastic reduction in the number of available LaTeX macros to the bare minimum. Furthermore, SEP may want a reliable mechanism to convert its LaTeX documents to PDF format automatically.

As suggested above, the web and Java will probably change SEP's current reproducible document concept. A Java enabled browser and a document applet may replace the current combination of a dvi file, a dvi viewer, a figure's graphic user interface, and a GNU makefile. A web document's hyperlinks will be able to connect to a wide variety of local and remote resources. Java's dynamic loading of classes will possibly replace much of the makefile functionality.

FREE AND OPEN SEISMIC SOFTWARE

The ease of distributing software and the increased need for collaboration between contractors, oil companies, and universities could, with a minimum of organization and money, lead to a free seismic software system.

Such a software collection might resemble the Free Software Foundation's GNU software. The free seismic software system would consist of a set of fairly independent packages, e.g. vplot graphics, I/O routines, and processing applications. Each package would be maintained by an expert somewhere on the web. The project's individual programmers would agree on a few initial standards, such as the seismic file format, or documentation and installation requirements. But otherwise, experts would be free to develop their area of expertise. New packages could naturally emerge and compete with existing ones. The various packages would be freely available at a few anonymous ftp sites. A newsgroup and perhaps a CD-ROM distribution would foster the development. As soon as a certain amount of software exists, more precise standards will emerge.

ACKNOWLEDGMENTS

David Nichols mentioned the `Modules` package for managing and exporting SEP's setup environment.