Active documents and reproducible results

Jon Claerbout

ABSTRACT

A revolution in education and technology transfer will follow from the marriage of word processing and software command scripts. In this marriage, here being called an *active* document (a-doc), an author attaches to every figure caption a pushbutton or a name tag usable to recalculate the figure from from all its data, parameters, and programs. An a-doc provides a concrete definition of reproducibility in computational oriented research. Given suitable interactive software, an *active* document is easily converted into an *interactive* document (i-doc). I have two textbooks undergoing conversion to i-books.

INTRODUCTION

The principal goal of scientific publications is to teach new concepts, show the resulting implications of those concepts in an illustration, and provide enough detail that the work is reproducible. In real life reproducibility is haphazard and variable. We rarely see a seismology PhD thesis being redone at a later date by another person.

The reproducibility problem can be largely overcome by standardized software generally available that is not hard to use. A new form of documentation is coming into existence. We call it an active document (a-doc). Active documents will serve us far better than paper documents. In an a-doc the author provides programs and a command script for every figure. Readers, students, and customers, can thus verify the calculation *and adapt it to new circumstances* without laboriously recreating the author's environment.

An a-doc could take many forms. It need be no more elaborate than this:

1. Each figure in the document has a name given in the figure caption.
2. Each figure in the document is made by a computer command script of the same name.
3. The document be distributed along with the command scripts and underlying programs, written in languages and using plotting code that is reasonably accessible.

Interactive documentation

Once an a-doc has been prepared another frontier immediately opens up, *interactive* documentation (i-doc). Interactive documents allow many features, the most obvious of which are (1) pushbuttons in the text allow rapid jumping from page to page, (2) figures can be converted into movies. While these features are dazzling, they are a distraction, I believe, from the truly revolutionary feature of an a-doc which is the ready *reproducibility* of the results it describes.

From the system programmers' view, the transition from an a-doc to an i-doc is a huge one (described elsewhere in this report), but from the authors' view the transition from an a-doc to an i-doc is easy.

EUREKA

I didn't recognize the concept of an a-doc until I had nearly built one. My third textbook on earth soundings analysis consists of about 300 pages, 120 figures, and listings of 40 terse subroutines. The subroutines were used in preparing the figures. Each of the figure captions contains a one-word name for the figure. Behind the visible book are plot files for each figure as well as main programs and command scripts (makefiles) that I used to make each figure. I suddenly realized that I had all the materials needed by any reader or student to recreate my work. This would give a tremendous boost to anyone wishing to go beyond me. At present the reader would need all my files and the local computing environment, and a little imagination to fill in a few steps that I did manually (mostly file copying and figure scaling). But with a little effort, the figure name could be connected to the command script that generates the figure. I have begun this effort. This will create an a-doc for all those Stanford people who share my environment. With time we should learn how to encapsulate out local software environment in a more portable way thereby assisting graduating students, sponsors, and eventually the research community.

From my author's view, the transition from an *active* book to an *interactive* one was an easy step. But giant steps were made by Steve Cole, Martin Karrenbach, and Dave Nichols outside my view and are described elsewhere in this report.

What I see is that with a little extra effort on my part, my book can appear on a screen. The names in the figure captions become pushbuttons that access my command scripts. Other pushbuttons automatically appear where ever a figure or equation on one page is referenced on another page.

The little extra effort required on my part to see the document in its interactive form is exactly that required of me to be assured that the figure captions buttons do actually point to the programs that generate them. This plugs the final gap of undocumented information about how to redo a result. A document and a makefile with programs doesn't become an "active document" until this last connection is made.

How reproducible is reproducible?

Compiling a program is one test of its completeness. Simply viewing an i-doc can be proof of the completeness of its underlying a-doc. You won't see a figure in the document unless its caption has the required pointer to the command script that creates it. Further, with the i-doc software we are developing (see "cake" elsewhere in this report), a figure *cannot be viewed* until the figure is up-to-date with the stated subroutine dependences. If an author makes a last minute change to a subroutine that a figure is based on, that figure will be automatically rebuilt by the process of viewing or printing the document.

Impact of reproducible publication

The human consequence of reproducible publication have hardly been considered. Yet this medium of scientific communication is nearly upon us. With workstations now becoming widespread and software coming available, the burdens imposed on the author to create reproducible figures can be little more than the burdens of paper publication. We are nearing a time when it will be simply the author's choice whether to communicate fully, in which case reproducible documents will be chosen, or whether to provide sketchy advertisements of scholarship, keeping confidential the detailed means to results, in which case traditional publications will be used.

OBSTACLES

Although active publication seems like a panacea, the idea seems utopian. It is worthwhile recounting the obstacles so we can recognize the most serious ones first. The first question is economic.

What amount of effort is warranted?

As we can see, the evolution of computer technology has reached a point where the extra effort of making an a-doc is little more than making the document itself. the only effort is keeping a record of all the steps in preparing figures. Nearly everyone already does this anyway. The extra effort required now is mainly because older documents and computer files omitted a few small steps that were done manually, principally running a plot program and inserting the figure in the text.

What degree of openness is desired?

Another limit to the future of a-docs is the authors' desire for openness. Plainly stated, the main limit of openness is people's desire for it. An active document is software and some people sell software. Active documents will be welcomed by those who want more openness and rejected by those who don't.

OBJECT ORIENTED DOCUMENTATION

Programming by the "object oriented" method is now fashionable. It is said to be especially important in dealing with large programs, in integrating the work of many people, and in making code reusable. An a-doc is really a lot of computer code and object-oriented philosophy can affect the organization, particularly of large documents like books.

An a-doc consists of two parts, one is system supplied, and the other is author supplied. For the author supplied part we envision reports, theses, and books. At the top directory of my book (as it is organized now) I see the subdirectories:

class	Exercises, answers, quizzes.
lib	Subroutine library included in text and used for computation.
prog	Directories with programs that make figures and test routines in **lib**.
tex	Text, a file for each chapter.
proc	Utility programs based on **lib**.

According to the object oriented philosophy (OOP) I have done it wrong. As a result, if I want to add or subtract a chapter of my book then I need to do something to each of the above directories. If I forget one I can make a mess.

To convert to OOP everything should be organized in subdirectories by geophysical topic. My **prog** directory is already organized that way. A typical subdirectory in **prog** makes a few figures in the book. Into each of these subdirectories I should put the text that describes its figures; I should put in the subroutines contributed to the library, and the utility programs (if any) that are provided for the benefit of other directories, and the class exercises and quizzes related to its figures. With this organization I could give one specially annotated book subdirectory to each new student, for training in use of all things I have had to learn to make the active book.

If OOP organization is desirable then what are its immediate obstacles? the system supplied portion of the a-doc must support certain features. For example, can the text processor deal with text, figures, and programs (for listings) scattered over multiple directories? Is it easy or difficult to deal with the renumbering implied when a new chapter or section or figures is included? Can the system build libraries and utilities from parts that are scattered all around? What about subdirectories within subdirectories? We are in the process of learning the answers.

STAGES OF ACTIVE DOCUMENTATIONS

The form of a-docs will change as we learn more about them. Here are the stages of conversion to a-docs:

Reproducing your own work

The first stage of active documentation is to prepare a document to a form where you yourself can reproduce your own work a year or more later by "pressing a single button". Many SEP students are now at this stage. Of those who are not, reasons can be

- data sets that are too large to be kept online
- limited knowledge of software for figure inclusion in documents
- lack of desire to use figure inclusion software because it slows the document printing time.
- use of interactive programs that do not save the steps required to reproduce a result.

Reproducing the work of coworkers

Now we are working on the next stage of active documentation. This stage is learning how to leave finished work in a condition where coworkers can reproduce it. We are experimenting with a variety of documents and learning various ways to connect the figure caption to the makefile.

Merging documents

As documents grow, they need to be handled in parts. For more complicated documents such as a thesis, textbook, or a large report including multiple topics such as by multiple authors, we developed the concept that *sections* of the document be associated with *directories*. Analogous to a table of contents is a file written in LaTeX language that "includes" the various directories. An unresolved issue in our present work is how to allow directories within directories.

Exporting a document

The next stage will be exporting an active document to another site. We'll need a receptive site willing to install all the software that we use–environment, file system, compilers, graphics, textprocessor, fonts, etc. Probably the first export will be a former SEP student. After a few such exports we may attempt to export an active report to sponsors, knowing full well that most will be unable to run it all, though all should appreciate the organized files.

Interactive document workbench

The "interactive document" is a concept. Beyond this concept are realizations of ever increasing completeness and utility. As underlying interactive software becomes more powerful and more friendly, we may see the i-doc become the "desktop" of interactive computing.

ACKNOWLEDGMENTS

Parallel efforts by Steve Cole, Martin Karrenbach, and Dave Nichols boosted my morale immensely on what originally looked like a huge volume of drudgery, converting it to a fun project. Mark Chackerian converted by second book "Imaging the Earth's Interior" (IEI) from the troff typesetting language to the LaTeX language, including scanning the figures. This was a great boost to my morale. Before scanning, improvements to the book were discouraged by the effort of repasting all the figures. Like Lazarus, IEI has come alive again, susceptible to frequent improvements.

REFERENCES

Axelrod, R., 1984, The Evolution of Cooperation, Basic Books, New York.
Cole, S., 1990, An interactive processing environment: SEP-**65**, 305-314.