Short Note

Why active documents need cake

Jon Claerbout and Dave Nichols

INTRODUCTION

An active document (a-doc) is software that reproduces a document including its plots. When plot files are absent the a-doc software should regenerate them. One reason to author active documents with the freely-available utility **cake** instead of the UNIX utility **make** is because **cake** handles an environment where intermediate files are missing.

Make is a widely known UNIX software maintenance utility. **Cake** is a similar utility with important additional features. **Cake** is publicly available but much less well known than **make**. Nichol and Cole [1989] describe why **cake** is better suited than **make** for multivendor software maintenance. Here we explain why we at SEP are using **cake** in active documents. Besides being a more sophisticated document maintenance language, **cake** saves much storage space by allowing us to abandon many bulky intermediate files if they are buildable from other files. One reason that **cake** is better suited for this task is that it was designed as a general dependency processor for all types of files, whereas **make** was only designed as a tool for compiling software.

First we'll review **make** and see how it differs from **cake** in environments where some files are missing.

REVIEW OF MAKEFILES

A **makefile** is a UNIX text file in a specialized language that assembles everything that is necessary to produce a "target" file which is typically a plot file. Let us consider a one-dimensional example. Consider a simple ratfor program that makes a plot. The files in order of their creation might be called:

`prog.r`	ratfor program
`prog.f`	fortran program
`prog.o`	object
`prog.x`	linked object
`plot.float`	seismogram values

`plot.vplot`	intermediate plot language
`plot.ps`	postscript plot language

More generally there is not a line of files above plot.ps but a *tree* of them. The tree includes parameter files, subroutines, headers, etc. But to understand **make** and **cake** it is sufficient to think about the linear dependences above. The first thing to notice is that each file in the above list should be younger that the one above and older than the one beneath. If this is not true, then a file is said to be out of date. When a file is out of date, then invoking **make** or **cake** will fire off the minimum number of commands needed to bring everything up to date. For example there might be a plot parameter that is used in the creation of the vplot file from the float file. When the parameter file is edited its date changes and thereafter an invocation of **make** or **cake** will rebuild only the plot files without recompiling anything. In summary, a **makefile** is a command sequence with the extra feature that any program will not be run its output is already younger than the program and all its inputs. For example, the command "**make figure.ps**" does nothing to a figure file that is up to date, but otherwise it does everything needed to rebuild the figure. A pushbutton in a figure caption can fire off the make command.

LIFE WITH MISSING INTERMEDIATE FILES

Make and **cake** do the same job, examining dates of files and then recompiling and invoking required programs. They differ in what they do when intermediate files are missing. **Make** will rebuild everything starting from the first missing file. **Cake** doesn't care about intermediate missing files so long as they can be created if need be. Looking to the above list of files you will recognize that it is foolish to store many bulky intermediate files, particularly if they can easily be regenerated. In the final analysis, there is a trade off between compute time and available storage space.

In the above example with **cake** you might choose to save only the original ratfor program and the intermediate plot file `plot.vplot`. This would enable a reader of the active document to read it on various devices (hard copy or soft copy) and rebuild the figure only if program changes were made. The bulky intermediate files could be removed. The reader of the document is secure. If he/she is reading it, then it is up to date. If the reader changes a program or parameter file, plots will be rebuilt to the new specifications.

WHAT'S WRONG WITH MAKE

On the other hand, with **make** the author of the document has a difficult decision. Either all intermediate files are kept or else the reader must be left in one of three unfortunate conditions, either (1) the reader must build all the plots to see them, or (2) the reader cannot be assured he/she is reading an up-to-date document, or (3) the reader cannot rebuild the figures without studying and revising the command scripts.

REFERENCE

Nichols, D., and Cole, S., Device independent software installation with CAKE: SEP-**61**, 341-344.

The author of **cake** is:

```
Zoltan Somogyi    (zs@cs.mu.OZ.AU)
Department of Computer Science
University of Melbourne
Parkville, 3052 Victoria, Australia
```