Publication of the Proceedings of the 1996 DPB/DPF Summer Study on New Directions for High-Energy Physics

L. Trindle Gennari
Stanford Linear Accelerator Center
Stanford University, Stanford California 94309

Production of the Proceedings of the 1996 DPB/DPF Summer Study in High-Energy Physics built on the methods, lessons, and technology of the production process used for the Proceedings of the 1995 Particle Accelerator Conference (PAC95).

For PAC95, authors were asked to submit PostScript, source files, and hard copy. Producing the proceedings took 16 months of official FTE in addition to countless hours of volunteer work by two additional people — the entire production process was accomplished in just under 9 months. In that time, the PAC95 production team quality checked every file (1099 submissions, totaling 3429 pages) and rebuilt about two thirds of them to fix problems with fonts and figures, and various other problems.

Needless to say, this was an expensive process.

The Snowmass proceedings project started with a much smaller budget and a shorter production schedule, resulting in a much more ambitious plan. The goal was, as for PAC95, to produce both a paper and a CD version of the proceedings. This time, the goal was to complete the project in only 6 months, using half of a staff person from the SLAC Technical Publications Department (responsible for technical design and implementation as well as project management), along with 6 months of a full-time temporary employee to answer the phone and coordinate author support.

The conference editors and I decided on a strategy using the World Wide Web for submission and quality assurance testing. The resulting procedure allowed authors to check the quality of their own PDF files, prevented random browsing of papers before publication of the proceedings, and required minimal human intervention (though we easily could have used a few more bodies manning the help lines!). To this end, we set up a Power Macintosh running FileMaker Pro 3.0 (a relational database application), WebSTAR (Macintosh Web server software), WEB/FM (CGI package for FileMaker Pro/WebSTAR interface), and NetPresenz (Macintosh ftp server) and created a gatewayed mailing list and newsgroup for authors needing technical support.

I. THE PLAN

An ideal submission went as follows:

1. Author got a filename.

   The author filled out our Web form, entering his/her last name, first name, institution, email address, and (added midstream) clicked radio buttons to indicate if the paper was a plenary talk (opening or closing) or summary paper (group or subgroup), to what group and subgroup it pertained, whether this author had gotten a filename before, and whether this was for a test file or a conference submission.

2. File submitted via FTP

   The author logged on to the Snowmass 96 Proceedings Server (using the conference name and password) and put his/her file in the `in` directory.

3. PostScript file distilled

   The Adobe Acrobat Distiller was set to watch the `in` folder and processed any files it found there. Once the paper was processed, the Distiller put it in an `out` folder, which was visible through the Web.

4. Quality check

   For this proceedings, the quality of the PDF was in the hands of the author rather than the proceedings staff. The author viewed his/her PDF file through the Web, checking for fonts, figure problems, and general onscreen readability. If the author found problems, the paper had to be fixed and the process repeated from step 1.

5. Declare Done

   The author filled out a form with his/her last name, the assigned filename (without `.ps` extension), and the title of the paper. The database used this information to look up the author's record. Using this method, authors could access only their own records. When the form was submitted, WEB/FM updated the database with the paper title, marked the paper done, and launched a FileMaker script that in turn launched an AppleScript that moved the paper to a non-Web-visible `done` directory. The server then sent an HTML version of the copyright form to the author's browser, personalized with the author's name, paper title, and filename. The author was then responsible for printing, signing, and returning this form to the proceedings staff with a hard copy of the paper.

II. TECHNICAL SUPPORT

The original plan was to set up a gatewayed mailing list and newsgroup where authors could post their problems and questions. This list was monitored at all times by proceedings staff, but it was our hope that authors would use these to help each other as well as get help from us. As it turned out, most authors unsubscribed to the mailing list immediately, and many emailed or telephoned staff directly rather than posting their questions to the list. This often meant we answered the same question many times, an unanticipated burden for the proceedings staff. To the extent that the proceedings staff did not disseminate every solution to the list and newsgroup, they share responsibility for some process inefficiency.

The lesson here for future efforts of this sort is that such an arrangement must be used consistently by both groups, authors and proceedings staff.

```
#!/usr/local/bin/perl

#first page number
$pageno = 230;

opendir(DIR, "rawfiles") || die "Cannnot opendir
pages\n";
while ($old = readdir(DIR)) {
if ($old =~ /^\./){ # skip the dot files
}else{
    $new = $old.".paged";
   open(PSFILE, "rawfiles/$old") || die "Cannot open
PSFILE\n";
     open(NEWPSFILE, ">>$new") || die "Cannot open
NEWPSFILE\n";
    while ($line = <PSFILE>) {
        if ($line =~ /^%%EndPageSetup/ ) {
            print NEWPSFILE $line;
               print NEWPSFILE "%%ltg pagenumber is
here\n";
                   print NEWPSFILE "290 750 moveto
\($pageno\)show\n";
            $pageno = $pageno + 1;
        } else {
            print NEWPSFILE "$line";
        } # end if
    } # end while
} #end if
} # end while
```

Figure 1. Paging Script for Snowmass96.

III. PAGE NUMBERS

In keeping with tradition in advanced technology, the biggest problem came from the smallest element, the page numbers. For the PAC95 proceedings, we were forced to pay the printer to add page numbers to the document. By the time production of the Snowmass96 proceedings started, a commercial solution for numbering pages was available. However, because it would have put all of the papers into a single PDF file, and because it was available only for Windows, it was not an option for this project.

Since we planned to create PostScript files of all of the PDF files, page numbering the PostScript files was the most logical option. When a PDF file was received in the `done` folder, it was printed to file. When the final contents of the document were set, the PostScript files (182 submissions, totaling 1123 pages) were transferred to a Unix machine, and a perl script run to add page numbers to the PostScript (see Figure 1). The paginated PostScript files were then brought back to the Macintosh and put into a folder watched by the Acrobat Distiller.[1]

IV. TABLE OF CONTENTS

The table of contents was produced using the conference proceedings database, Textures (a Macintosh TeX compiler), and some by-hand PostScript editing.

[1] Three of the papers had problems with the page numbers being covered by white space when the page was drawn onscreen or printed on paper. Each page number had to be placed on the page by hand using the coordinate translation and printing macros that were used to print the text rather than the generic PostScript code used to page number all of the other documents.

```
%This block defines variables used to make pdf links
in Textures
%Also necessary to remove the lines surrounding the
code inserted
%around the code specified in the \special and rede-
fine s (below)

%Allow use of pdfmark
systemdict /pdfmark known not
{userdict /pdfmark systemdict /cleartomark get put}
if

/MyBottomy { FirstBottomy 12 sub } def
/MyRightx { FirstRightx 34 sub } def

%Hardcode top of box to be 12 points above bottom
/MyTopy { MyBottomy 12 add } def

%Hardcode left of box to be 000 right of right
/MyLeftx { MyRightx (000) stringwidth pop sub} def

% procedure to create LaunchMe
/MakeLaunch {
/Base (../papers/) def
Base length ToBeLaunched length add
% top of stack should now contain entire string
length
/LaunchMe exch string def % make the string
LaunchMe 0 Base putinterval % put Base in front
Base length % get position
LaunchMe exch ToBeLaunched putinterval
} def

/MyMakepdfmark {
MakeLaunch
/NewRightx { MyRightx 65 add } def
/NewLeftx {MyLeftx 65 add } def
[ /Rect [ NewLeftx MyBottomy NewRightx MyTopy ]
  /Border [ 0 0 0 ]
  /Action /Launch
  /File LaunchMe
  /LNK
pdfmark
} def
```

Figure 2. Custom PostScript Code.

I created a calculation field in the database to make the entries for the `.toc` file. These entries look like

```
\contentsline{section}{\protect\hskip -2emA
High Energy Physics Perspective---W. Mar-
ciano}{1\special{postscript /ToBeLaunched
(OPS001.PDF) def MyMakepdfmark}}
```

This `.toc` file was then used as input to a LaTeX document to typeset the table of contents. The typeset document was then printed to file. It was then necessary to add custom code to the top of the PostScript file (see Figure 2), redefine the PostScript procedure `s` (see Figure 3), and remove the lines above and below the PostScript lines generated by the `special` commands in the LaTeX code to create hyperlinks to the papers in the PDF table of contents.

V. INDEX

The index was produced using the conference proceedings database, MakeIndex on a Unix machine, Textures, and Acro-

```
%my version of s, redesigned to pass the
%currentpoint to MyMakepdfmark
/s{exch currentpoint /FirstBottomy exch def
/FirstRightx exch def FirstBottomy moveto show}bd
```

Figure 3. Redefinition of `s`.

bat Exchange. The oddities of TeX's PostScript coordinates and the limitations of MakeIndex unfortunately did not allow the hypertext links in the index to be generated in the same way as those in the table of contents. The index entries were pulled from the database using a calculation field:

```
\indexentry{Abdullin, S.}{541}.
```

The list of indexentries was then run through MakeIndex to create a `.ind` file, which was then used as input to a LaTeX file to typeset the index. Since time was short, efforts to generate hyperlinks with FileMaker and LaTeX were abandoned for the tedious, but guarenteed by-hand method. All hyperlinks in the Index were added in Acrobat Exchange.

VI. Conclusions

Overall, the methods used to produce this proceedings were considerably more efficient than those used in the past. The time from receipt of the last submission (April 2, 1997) to sending the final product to the publisher took one person less than two months (including some vacation days!). The time elapsed between first and last submission delayed the final printing by six months. Had the submissions been complete by the original deadline of October 15, the production of the proceedings would have been completed by mid-December.

One reason for the very long submission period was that many of the group summaries were dependent on the content in their subgroups' summaries, which were in turn, dependent on some of the individual reports. If the deadlines for individual papers, subgroup summaries, and group summaries had been staggered from the outset, the planning, writing, and production of the proceedings would have happened much more quickly and easily.

I am pleased to have had the opportunity to develop and implement this unique publishing process. Comments and/or questions about the process or the technology behind it are welcome (gennari@slac.stanford.edu).