Foreword

This paper tells the story of how, more than 15 years ago, authors were encouraged to participate in the production of the academic journal *Electronic Publishing — Origination, Dissemination and Design* (*EP-odd*) which was published by John Wiley Ltd from 1988 to 1998 and for which I was the Editor in Chief.

EP-odd fell into the category of a 'hybrid journal' in that its production was via traditional typesetting but crucially, Wileys agreed to use a typesetting supplier equipped with a PostScript-based typesetter, even though this raised the production costs of the journal. Secondly the publishers and the typesetting company agreed to install LaTeX, UNIX *troff* and Ventura Publisher software so that they could do final corrections on accepted papers using the author's choice of originating software from those three systems.

The important common factor for all three pieces of authoring software was that each of them could routinely produce PostScript output and all of them could be configured with macro sets for the page layout and general 'look and feel' of *EP-odd*. These macro sets were, of course, installed at the publisher's and at the typeset supplier's premises. In addition, by using these macros, authors could submit their papers with an approximately correct layout and, if accepted, they could also help in getting the final page layout correct. The necessary common factor in the above approach was the use of PostScript as the final output format; this was a format that could be proofed by authors on PostScript-based laser printers and by the typeset suppliers on PostScript-based laser typesetters.

Our choice of *Ventura Publisher* as the supported WYSIWYG software (largely for its 'better structuring') now seems bizarre but one has to remember that, in 1990, Microsoft Word had not achieved the dominance that it enjoys today. Instead a battle was in progress between *Word* and *Word Perfect* for market share but neither of them , at that time, was seen as a serious vehicle for creating technical documents.

It is also hard to remember, now, that this paper was published three years before the advent of Adobe's PDF. In effect, we were attempting to use PostScript in the same way that PDF is now used i.e. as a final-form interchange format. Indeed, when PDF came on the scene *EP-odd* was the first journal in the world to adopt it and to put out, as early as 1993, a CD-ROM of *EP-odd* papers in PDF. The entire *EP-odd* archive had its copyright transferred, in 1998, from John Wiley Ltd to the University of Nottingham and it is available online via:

`http://cajun.cs.nott.ac.uk/compsci/epo/papers/epoddtoc.html`

The only reason that we could move to PDF so quickly was that we had carefully archived all of the final PostScript files for the accepted papers. The vast majority of the PostScript-to-PDF conversion, using Acrobat Distiller, was problem free and it was helped by the fact that we had pushed very hard, in all accepted papers, to achieve integrated page makup based solely on PostScript. This included all diagrams etc (i.e. no stripped-in figures or illustrations) which, in turn, made the production of PDF relatively straightforward. The whole exercise underlined, for editors and publishers alike, the virtue of archiving every single electronic file connected with the production of a given journal paper.

Colophon

This paper appeared in "Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography" (EP90). published by Cambridge University Press. In this conference we attempted, for the first time, to emulate the *EP-odd* approach by supplying LaTeX and *troff* macros to emulate the CUP style for Conference Proceedings. Accordingly the enclosed paper has been re-typeset from the original *troff* source code using the *cup* macros.

Electronic Publishing — Practice and Experience

David F. Brailsford[†], David R. Evans[†], and Geeti Granger[‡]

[†] *Electronic Publishing Research Group*
Department of Computer Science
University of Nottingham
Nottingham, NG7 2RD
England

[‡] *John Wiley & Sons Limited*
Baffins Lane
Chichester, PO19 1UD
England

ABSTRACT : *Electronic Publishing — Origination, Dissemination and Design* (‘*EP-odd*’) is an academic journal which publishes refereed papers in the subject area of electronic publishing. The authors of the present paper are, respectively, editor-in-chief, system software consultant and senior production manager for the journal. *EP-odd*’s policy is that editors, authors, referees and production staff will work closely together using electronic mail. Authors are also encouraged to originate their papers using one of the approved text-processing packages together with the appropriate set of macros which enforce the layout style for the journal. This same software will then be used by the publisher in the production phase. Our experiences with these strategies are presented, and two recently developed suites of software are described: one of these makes the macro sets available over electronic mail and the other automates the flow of papers through the refereeing process. The decision to produce *EP-odd* in this way means that the publisher has to adopt production procedures which differ markedly from those employed for a conventional journal.

KEYWORDS : journal production, computer aided refereeing system, remote file access.

1 Introduction

Over the last ten years electronic publishing (EP) has emerged as a research area in its own right and in the last five of these it has become one of the most active areas of applied computer science. This increased activity led to the notion that the EP community should have a regular publication of its own, featuring refereed papers on all aspects of EP research. In late 1987 one of us (DFB) founded just such a new journal and became its editor-in-chief. The title *Electronic Publishing — Origination, Dissemination and Design* (‘*EP-odd*’ for short) was chosen and John Wiley Ltd. agreed to be the publishers. An initial ‘pilot issue’ was produced in January 1988 and the journal now appears four times per year. The decision to cover EP in its full generality is reflected in the statement on ‘Aims and Scope’ which reads as follows:

> "*EP-odd* is an International journal which publishes refereed papers on all aspects of electronic publishing. A broad interpretation of the topic is taken and we wish to encourage articles in such areas as structured editors, authoring tools, hypermedia, document bases, production of concordances and indexes, document displays on workstations, electronic documents over networks, integration of text and illustrations, typeface design, imaging hardware relevant to electronic publishing etc. (although this list is far from exhaustive)."

The editor-in-chief of *EP-odd* also acts as the UK/Europe editor and he is assisted by a US co-editor — this latter post being occupied, for the first two years of the journal's existence, by Richard Beach of Xerox PARC. More recently, pressure of work has caused him to resign and to hand over to a new US editor who, nothing daunted by his workload as Programme Chairman and Local Organiser of this present EP90 conference, brings to *EP-odd* his many years of experience in the EP field. It has been noted elsewhere [1] that this new editor combines continuity, by having the same first name as his predecessor, with subliminal typographical allusions, by having a surname which is an anagram of "the archetypal German (geometric) sans, and perhaps the least ungainly of this rather austere species" [2]. Richard Furuta takes over as *EP-odd* co-editor starting with Issue 1 of Volume 3.

The sub-title of the present paper tries to combine descriptive accuracy with a gentle nod in the direction of John Wiley's existing journal, *Software — Practice and Experience*. Later sections of the paper will reveal that the decision to allow authors to participate in the production cycle of their accepted papers has not been without its ups and downs. An alternative sub-title could well have been 'Tales from the War Zone'.

In a previous paper on the production of *EP-odd* [3] there is some extra background on laser-printer and electronic mail technologies together with some notes on two other experiments in 'electronic publishing'. In the present paper we focus, in Section 2, on the strategic decisions about the production of *EP-odd*. Section 3 gives more details of our experiences, so far, with the macros and document styles for *troff*, TEX and LATEX, all of which can now be acquired over the computer networks using electronic mail (e-mail). The problems of tracking papers through the refereeing process are recounted in Section 4 and some initial software to address these problems is described. Finally, Section 5 describes the production process for *EP-odd* at the Chichester (UK) office of John Wiley Ltd., and the ways in which this process differs from that used in other journals.

2 Some strategic decisions

From the start it was realised that a new journal about EP might be expected to employ 'leading edge' EP technology for its own creation and production.

Equally, the editors and production staff were aware that beyond every leading edge lies a bottomless abyss and so the idea of a 'truly electronic' journal, with optional video and voice inserts, disseminating full-colour browsable hypertext over the computer networks, was dismissed at the outset for all the boringly predictable technical reasons. Instead, we decided to stay with a traditional hard-copy published form but to use as many computerised aids as we could in the pre-publication stages.

The three text-processing systems initially supported for the authoring of *EP-odd* papers were *troff* [4], TEX [5] and Ventura Publisher®. Bearing in mind that many of our potential authors would be computer scientists, the ready availability of the first two systems made them an almost automatic choice. At the same time we did not want to seem indifferent to the rise of Desktop Publishing programs and, of these, Ventura Publisher seemed to offer facilities for longer and more structured documents by virtue of its tag sets.

All three systems were potentially capable of being processed into the PostScript® page-description language, which combines comprehensive facilities for imaging characters, line diagrams and half-tones with the very potent advantage that its imaging model is independent of device resolution. This enables papers to be proofed on laser-printers during copy editing, before being sent to a typesetting bureau for final output, on bromide, from a PostScript typesetter. In retrospect the decision to standardise on PostScript as the common final form has been every bit as significant as the choice of authoring software.

From the very beginning we were prepared to distribute to our authors, on request, the same set of macros and tags, for each of the authoring systems, that would be used in the final production of the journal. This approach has also been used, quite recently, for various EP conference proceedings including those that you are now reading, The advantages are very clear because, in short, one cannot divorce form and content in the world of electronic publishing. Authors need to become involved in the production cycle of their paper, knowing from the outset whether they have abided by its page limits, and with all the advantages of seeing line diagrams, equations, photographs and tables at the right size and in the right positions.

The main problem with this approach is that macros and document styles cannot automate every last context-sensitive detail of page design. Ideally, authors need to learn, very rapidly, the aesthetic considerations which underlie good page layout. In a 'one-off' set of conference proceedings the occasional badly-placed diagram or awkward 'widow' can be tolerated and laser-printed camera-ready copy is usually acceptable. For a journal it is important that the layout standards and attention to detail should not fall below what could have been achieved by the conventional production mechanisms. In asking John Wiley Ltd. to publish our

journal we were very aware that text processing schemes such as TEX and *troff*, having been written by computer scientists, would probably not be the systems of choice if the publishers were given a free hand. On the other hand our plans depended on the publisher being prepared to produce the papers using the same text processor that the author had already chosen. Fortunately, one of us (GG) is a senior production manager at Chichester and had already used TEX extensively; the other two of us have had several years of experience with *troff* and its macro sets.

3 Macro sets for *EP-odd*

The difficulties of using either *troff* or TEX in their raw state are well known. The bewildering number of typesetting commands, and their very low-level nature, leads very quickly to an urgent need for macro packages so that the appalling machinations of the basic commands can remain mercifully obscure. Jonathan Seybold once said that 'TEX is not for the faint-hearted' and, even more tellingly, a humorous item was received over the mail networks some time ago, which, in defining UNIX† expertise, gave hallowed pride of place to the ability to write *troff* macros. The eight levels of expertise were divided into *beginner*, *novice*, *user*, *knowledgeable user*, *expert*, *hacker*, *guru* and *wizard*. The required achievements for the last-named category were:

wizard Writes device drivers with '**cat >**'
Fixes bugs by patching the binaries
Can answer questions before you ask them
Is on first-name terms with Ken, Dennis and Bill
Writes his own *troff* macro packages

We were lucky to acquire a starting set of *troff* macros, initially written by Brian Kernighan and Mike Lesk for the journal style of *Software — Practice and Experience*, which have now been extensively adapted to form the *ep* macro set for *EP-odd*. An equivalent set of TEX macros was developed at Chichester and have been used to produce 7 of the papers published so far.

Although the *ep* macros for *troff* have been used by various authors the TEX macros have not been so much in demand. The reason for this seems to be the rising popularity of a more heavily configured version of TEX known as LATEX [6]. In response to author demand we have developed a LATEX document style for *EP-odd* which is an adaptation of a style originally written by Richard Furuta for the book on 'Structured Documents' [7] (this same style being the forerunner of the one used for these conference proceedings).

† UNIX is a trademark of Bell Laboratories.

To date we have published 7 regular issues of the journal comprising some 26 papers. Of these 13 have been produced using *troff* (11 were supplied by the authors in that form; 2 were converted from other word processors, into *troff*, by the editor-in-chief), 7 have been produced in TEX (6 were submitted in that form; 1 was converted from LATEX to TEX), 5 have been produced using LATEX and 1 using PostScript. So far no paper has been submitted simply as a typescript, and there has been very little interest in Ventura Publisher.

The Author Guidelines for *EP-odd* allow for camera-ready copy to be accepted though the conditions laid down are very stringent; page layout details have to be followed exactly and the final submission must take the form of a typesetter bromide. Although we have not yet had copy in this form, a recent paper from Zeev Becker and Dan Berry [8] was submitted as typesetter-ready PostScript — which in some ways can be thought of as a form of electronic camera-ready copy. This paper describes the typesetting of Hebrew, Japanese and Chinese using *triroff*, which is an adapted version of the UNIX typesetter-independent *troff* program. The paper needed to be typeset using *triroff* itself but time constraints did not allow this extra software to be installed at Nottingham. Instead, the authors were given the page range to be used and spent many hours in adapting the *ep* macros for use with *triroff*. Copy-editing corrections were sent to the authors, who were also left in charge of page layout. The final version of the paper was submitted as a PostScript file to the editor-in-chief by electronic mail. The large number of Chinese, Japanese and Hebrew characters in this paper classified it, beyond doubt, as 'difficult copy' but the policy of adopting PostScript as the common final form allowed us to publish it with relative ease.

3.1. Incorporating Diagrams

There are facilities in TEX, LATEX and *troff*, either built-in or made available by preprocessors, for setting out mathematics, diagrams and tables. However, these simple descriptive schemes cannot cope with illustrations such as photographs, screen dumps and complex line-diagrams. Fortunately all of this material can be represented as PostScript and it can be generated from software such as CricketDraw®, or Adobe Illustrator®, or it may take the form of bitmaps in PostScript 'image' format. An increasing number of authors are choosing to send at least some of their illustrations in this form.

Whatever the source of the PostScript it still has to be incorporated into a document whose final output form may also be PostScript, but whose source format uses a very different text-processing model. If one can define start and end markers at the source level, which denote where the PostScript is to be inserted, together with some means of telling the text processing software what the overall size of the diagram will be, then the PostScript insert can be passed on, unaltered,

to the back-end device-driver module. This merges the insert into the PostScript already produced from the surrounding material. These facilities are provided by *psfig*[9] which allows PostScript inserts to be included in TEX and *troff* source code. The *troff* markers used by *psfig* are `.F+` and `.F-` and between these a small set of commands is allowed for indicating such things as the height and width of the diagram. To reinforce this information there must be a `BoundingBox` comment at the head of the PostScript file, which must also conform to Adobe's standards for Encapsulated PostScript [10].

3.2. Remote Acquisition of Macros via Electronic Mail

In the early days of *EP-odd*, macros were sent to intending authors by the editor, or a colleague, using either electronic mail or a floppy disc.

In addition to the macros themselves there is a variety of further information available about the journal, such as the instructions for authors on the use of the macros, order forms, lists of past papers and so forth. However, distributing these extra files would be very time consuming if every intending author, or interested party, had to be mailed personally. Since many of the authors for the journal have easy access to the academic computer networks, it was decided to provide automatic remote access to these files.

Remote access to information over a wide area network[1] is possible by several different procedures. Files may be transferred using a file transfer protocol (FTP) from one host computer to another [12]. FTP is a protocol in the presentation layer (layer six) of the International Standards Organisation (ISO) reference model for open systems interconnection (OSI). Many computer systems now support this protocol, which allows remote user logging in and authorisation, remote directory listings (to see what is available), and transfer of human-readable or binary files. By the use of *anonymous* FTP, where the user does not need a username on the remote host, it is possible to allow controlled access to a restricted part of the machine, thus enabling files to be transferred to interested parties. At present we can only provide the full facilities for access to the *EP-odd* files, via anonymous FTP, over our *local* campus Ethernet.

To allow wider access to the *EP-odd* files it was necessary to implement a mail server, since many potential authors can send e-mail to our host machine. The mail server is a program which is activated on receipt of a message, and is able to parse a simple language contained within the message body. The language keywords are described in table 1 and they denote operations which the sender of the message wants the mail server to perform.

[1] For a good description of computer networks see [11].

Keyword	**Mail server response**
`help`	Send the help file, and ignore any other requests.
`index [category]`	Send the index for the category, if specified, or else the top-level index. Ignore any other requests to send files.
`send category file`	Send the requested file or files. If the filename ends in '`.Zba`' then `compress` the file and run it through `btoa`, to reduce the size of the returned message.
`path <e-mail address>`	Specify a return mail path to the author for machines which prove difficult to get to.

Table 1: Language syntax and responses for the mail server

The mail server actually contains several different categories of information for retrieval, and stores each of these in a sub-directory. The top-level directory and the sub-directories each have their own indexes, which may be retrieved to see what is available for transfer. Normally the help file would be obtained by the user with his first message, then an index, or indexes, in the next, and then any files that were of interest. As an example, to obtain a set of *EP-odd* macros, one sends the following message:

```
To: ep-server@uk.ac.nott.cs
Subject:

send EPodd epodd.shar
```

It is possible to specify a complete return address with the *path* command for those mail addresses where the mail server cannot reply successfully by using the received mail header information.

The mail server is implemented as a set of ten UNIX shell scripts [13], with the message parsing being done by an AWK program [14]. This allowed the server to be written fairly easily using the existing software tools available under UNIX.

To automate the maintenance and running of the server there are several shell scripts that can be run automatically or manually. There is a daemon which is run automatically at various times throughout the day to check the server's queue of pending messages, and to send out the replies (this is normally executed outside working hours). Statistics of how much it has been used are produced weekly from the log file, and show that approximately 400 files (approximately 5.3Mbytes) have been transferred over the last year.

Each category is easily updated with new information by the use of a

makefile[15] which will rebuild the indexes by running a *makeindex* shell script, and update any other files using the rules in the *makefile*. This makes the general maintenance of the server very simple, and in fact is left to run on its own for weeks without human intervention.

At present the server contains about 300K of information in the *EP-odd* category, ranging from the macros and style files, to the instructions for authors, and the lists of past papers.

4 The Refereeing Process

4.1. Present Procedures

The refereeing process is one of the most time-consuming aspects of an editor's work for any journal. The administrative work load for this process is quite considerable even when the paper flow is low. For *EP-odd*, the rules are that papers must be submitted initially as conventional hard-copy. They will then be sent to two suitable referees, with a request for a report within a fixed period of time. If the paper is accepted it may need to go back to the author for the referees' comments to be taken into account. Only after such revision can the paper be re-submitted by electronic mail, preferably as marked-up source for one of the allowed authoring systems. The editor performs some final tidying up before the paper goes to Chichester for final copy-editing and pagination. At every stage of this process a record is needed of the progress of each paper so that the editors can plan future issues of the journal and track the progress of papers through the system.

The passage of a paper through the refereeing process has, until now, been conducted entirely by paper-based records, but the advantages of a computerised scheme are self-evident:

- The records will be easier to maintain and the current status of any paper will be easier to determine.
- It will be easy to obtain a list of papers which are at a particular stage of the refereeing process.
- It will be possible to automate the sending of reminders, both to referees and authors, for the return of reports or revised papers.
- The more mundane tasks, such as producing status reports and standard form letters, could also be automated.

4.2. The Computerised Refereeing System

As an undergraduate Computer Science team project, four undergraduates chose to implement a system for aiding the editor in the refereeing process. The main aims of the system were to store all necessary details of each paper, and its referees, in a database of some description. This could then be used to automate some of the processes listed in the previous section.

One restriction imposed on the project group was that the system should run on the UNIX system so that the editor-in-chief would be able to access it easily, and to make it easy to interface the system to electronic mail. It also needed to be terminal independent, and written in a modular fashion, in order to allow enhancements to be made in the future.

The design of the refereeing system, called '*Wizepp*' (**Wiz**ard **E**lectronic **P**ublishing **P**roject!), was based on the use of a 'database' (actually a UNIX file with a fixed-record structure imposed on it), and a set of programs that could modify it. Figure 1 shows how the different programs interact with the database in

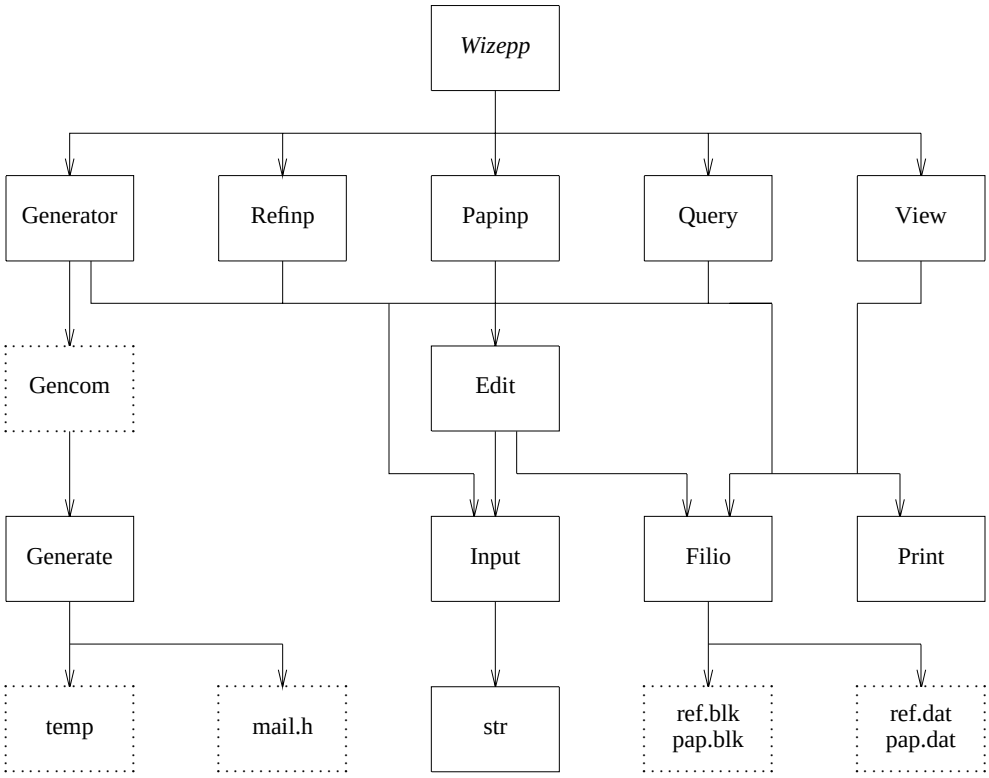

Figure 1: Program and database interaction for *Wizepp*

Wizepp. Each of these programs import standard modules that provide useful routines to access the database. The dotted boxes in figure 1 indicate data files, with the `pap.dat` and `ref.dat` files being the paper and referees databases respectively. The files `pap.blk` and `ref.blk` are blank records used as templates during the editing of some other record. This helps to maintain the integrity of the database by updating a record only when it is complete.

Wizepp was designed so that the user entering data could either use *Wizepp* itself to be presented with menus of commands to be carried out (edit or update paper information, add new paper, edit referees database etc), or the commands could be called directly from the command line (see the second row of figure 1).

The whole system is menu driven using the *curses* terminal-independent library routines which makes it accessible from a wide variety of dumb terminals.

At each stage where the user is asked to enter data he is presented with the present value for that field (which may be empty) and a series of dots indicating how much information may be held in the field. For example, an author's name may be up to 28 characters long and so the field might appear as:

```
Mr F J Bloggs..............
```

It is then possible to move backwards and forwards within the field using the cursor keys, and to insert text in the middle as well as appending to the end. Some elementary checks are carried out to ensure, for example, that digits are not allowed in names and alphabetics are not allowed in numbers.

Once the paper details have been input for the first time the referees' names will be prompted for, and these checked for in the separate referees database. If the referee is not already known to the system he or she can be added as a new entry at this point. The system will also ask for a date by which the referees' reports are due back so that automatic reminders can be generated.

So far *Wizepp* allows the paper details to be input into its database, permits easier modification of the paper details than the old paper-based records, and can display the status of the papers in the system. The following are planned as enhancements in the near future:

- Improve the letter generator to align form letters correctly on the *EP-odd* stationery.
- Provide another interface for automatically sending status reports to the editor on a weekly basis.
- Automatically remind the referees and authors when reports and amended papers are due back.

5 Final production stages

The main job carried out at Chichester is the final copy-editing and pagination. Each paper is processed using the same text processing system that was used for authoring. The computer used at Chichester is a SUN 386i system, which makes UNIX and MS-DOS available on the same workstation; this system is connected via a LocalTalk network to other Macintosh and IBM PC computers. The connection of the SUN machine to the wide-area computer network, and thus to the UK and US editors, is via dial-up line and modem. This permits short e-mail messages to be exchanged with editors and authors but the communications software, and the modem itself, both need upgrading before it will be possible to transfer papers from Nottingham to Chichester over the network. For this reason the source text of the papers is usually sent down from Nottingham on either floppy discs or on SUN magnetic tape cartridges.

Once the material is available for a particular issue it is processed through the appropriate text formatting system to produce laser-printed proofs, which are sent to authors and editors for approval. When the final amendments have been made the altered source documents are processed again and the PostScript output thus obtained is stored as MS-DOS files on an IBM PC machine. These files are sent off, on IBM 5¼ inch floppy discs, to a typesetting bureau, which produces the bromide. The bromides are then returned to the production department at Chichester and final pasting-up is carried out for any illustrations that were not included in the original source file. The completed camera-ready copy is then sent to the printers.

5.1. Practice and Experience to date

The authoring schemes we support — *troff*, TEX and LATEX — all have some sort of default style for items such as tables, lists and footnotes and some kind of control over where these appear on a page. However, these crude mechanisms easily break down and it is not uncommon to find figure 6 appearing before figure 5, or table 2 before table 1. Much of the effort in Chichester centres around the fine-tuning of these details and this requires a carefully planned production schedule. The detailed adjustments to spacing and positioning were easy to effect in TEX because of the accumulated expertise with that system at Chichester. The same process in *troff* can be much more frustrating on those occasions where the requested changes do not seem to work, but most problems can be solved by requesting help from Nottingham.

Without doubt the most noticeable change in *EP-odd* papers over recent months has been the desire to use LATEX. This system is very ‘author friendly’ because its highly structured nature makes it very easy to generate section

numbers, cross references, new sections, itemised lists and so on. However, the commands to do all this are more concerned with the *logical* structure of the document than with its *physical* appearance. Provided one is happy with the default layouts produced by LaTeX everything is easy, but if any layout detail needs to be changed it can be appallingly difficult to make this happen — not the least because it is hard to apportion the problems that occur between LaTeX itself and the underlying TeX codes that it produces. We have not made any great use, as yet, of the BibTeX pre-processor for LaTeX but we hope that this can be tailored, along the same lines as the *refer* pre-processor for *troff*, to enforce a uniform 'house style' for references.

It has already been noted that several authors have submitted diagrams in the form of PostScript inserts. The *psfig* package can usually handle these with no problem, provided that the supplied PostScript truly conforms to Adobe's rules for Encapsulated PostScript. Unfortunately on the occasions where these rules are not followed it can be very time-consuming to analyse the PostScript code to determine the correct `translate` and `BoundingBox` commands to align and resize the diagram correctly. Another problem concerns the sheer size of the PostScript files for those illustrations which are dumps of workstation screens. Although *EP-odd* illustrations are normally produced in black and white only, a screen dump from a colour workstation can amount to more than 2 Mbytes of uncompressed PostScript. Fortunately, there is a facility in *psfig* which allows a diagram to be placed as a blank box, at the proofing stage, to cut down on the computer processing overheads. One memorable paper, recently, had no less than **seven** such screen dumps, each of which took 30 minutes to image on an Apple LaserWriter II. In principle, these same screen dumps should have been sent off for imaging, on bromide, at the PostScript typesetting service (with each dump occupying two high-density floppy discs) but the costs of 3½ hours of typesetter time would have been prohibitive. In this particular case the laser-printed screen-dumps were pasted into the final copy.

5.2. Production problems

The novel factors with *EP-odd* from the viewpoint of the production staff are:

- The production of the journal is 'in-house' at Chichester. Copy for other journals would normally be sent to outside agencies for keyboarding.
- The papers arrive on disc rather than as typescript. Their appearance is much closer to the 'final form' than would normally be the case.
- The editors and many authors use e-mail daily and expect the production staff to be equally assiduous in checking e-mail and responding to it.

The last point is an important one. It takes quite some time to acquaint production staff with new technologies and, though we would not want to read any significance into this, *EP-odd* has now had three Production Editors, each of whom has had to be initiated into the delights and mysteries of e-mail. This training and administrative overhead was certainly not taken into account when the journal was first mooted. The benefit of having the staff at Chichester on e-mail is that day-to-day problems with *EP-odd* can be put right very quickly. The main drawback is the tendency to rely too heavily on this high-speed communication for informing the production staff of last-minute alterations, or in the editors assuming that a deadline is met if the required copy is sent out by e-mail at the stroke of midnight on the day in question.

5.3. Other benefits

There is no doubt that experience with *EP-odd* has given John Wiley Ltd. an invaluable introduction to a future where authors, editors, referees and production staff may be drawn together, ever more tightly, in the origination, dissemination and design of a journal. Indeed, since *EP-odd* was started a new Wiley journal called *Concurrency—Practice and Experience* has been founded and it has the stated policy of encouraging all authors to use LaTeX. The experiences with *EP-odd* have greatly facilitated the setting up of a LaTeX style for this journal and in forewarning the production staff of what they were letting themselves in for!

ACKNOWLEDGEMENTS

We would like to thank SUN Microsystems for providing us with a SUN 3/160 for the support of *EP-odd*, and also Rupert Weare, Geoff Dennis, Simon Gooch, and Wendy Chui for their work on the *Wizepp* refereeing system.

References

[1] David F. Brailsford, *Electronic Publishing—Origination, Dissemination and Design*, Editorial, Vol. 2, no. 4, December 1989,

[2] James Sutton and Alan Bartram, *An Atlas of Typeforms*, Lund Humphries, 1968.

[3] D. F. Brailsford and R. J. Beach, "*Electronic Publishing*—a Journal and its Production", *Computer Journal*, Vol. 32, no. 6, 1989, pp. 482–493.

[4] J. F. Ossanna, "NROFF/TROFF User's Manual", Bell Laboratories: Computing Science Technical Report No. 54 (April 1977).

[5] D. E. Knuth, *TeX and METAFONT: New Directions in Typesetting*, Digital Press and the American Mathematical Society, Bedford Mass. and Providence R.I., 1979.

[6] Leslie Lamport, *LaTeX: A Document Preparation System*, Addison-Wesley, Reading, Mass., 1986.

[7] J. André, R. Furuta, and V. Quint (eds.), *Structured Documents*, Cambridge University Press, 1989.

[8] Zeev Becker and Daniel Berry, "triroff, an adaptation of the device-independent troff for formatting tri-directional text", *Electronic Publishing — Origination, Dissemination and Design*, Vol. 2, no. 3, October 1989, pp. 119–142.

[9] Ned Batchelder and Trevor Darrell, *Psfig — A Ditroff Preprocessor for PostScript files*, Computer and Information Science Dept., University of Pennsylvania. Internal Report 1988.

[10] Adobe Systems Inc, *Encapsulated PostScript File Format*, (EPSF Version 1.2) March 1987.

[11] Andrew S. Tanenbaum, *Computer Networks*, Prentice-Hall International, Englewood Cliffs, New Jersey, 1981.

[12] M. Gien, "A File Transfer Protocol (FTP)", *Computer Networks*, Vol. 2, September 1978, pp. 312-319.

[13] S. R. Bourne, "An Introduction to the UNIX Shell", Computing Science Technical Report, Bell Laboratories (12th November, 1978).

[14] Alfred V. Aho, Brian W. Kernighan, and Peter J. Weinberger, "Awk — A Pattern Scanning and Processing Language, Programmer's Manual", Computing Science Technical Report No. 118, AT&T Bell Laboratories, Murray Hill, New Jersey 07974 (June 1985).

[15] S. I. Feldman, "Make — A Program for Maintaining Computer Programs", Computing Science Technical Report, AT&T Bell Laboratories, Murray Hill, New Jersey 07974 (August 1978).