Streamlining Publishing Procedures

E. van Herwijnen[1] and J.C. Sens[2]

(1: CERN, 2: National Institute for Nuclear and High Energy Physics, Amsterdam)

Big changes have taken place in the ways physicists prepare their scientific papers and new opportunities have arisen to limit retyping and to speed communication in the phases prior to publication. The EPS Publications Committee is exploring possible methods of standardization in the light of its survey of the word processors currently in use and the merits of the SGML generalized mark-up language.

The world of science publishing appears to suffer from indigestion in several key areas and the road from logbook to journal is long and winding. Submitting a paper to an editor resembles staring down a long tunnel; there is often no end in sight.

Attempts at improving this situation centre on increased automation in the submission and publication of papers through the use of inter-connected computers. While daily communication between some scientists has improved to the point where the absorptive capacity of the human brain is the limit, little use has been made of these facilities for reducing delays in publishing papers. Although computers are used heavily by both publishers and authors, there is as yet no link connecting the two ends of the process.

Need for a Standard Format

In a zero-order world, there would be one type of computer, functioning with one operating system, housing one package of word processing software, equipped with software for handling mathematics and figures, connected to a single network, and available to all potential authors, editors and publishers. An author would prepare a manuscript, issue a preprint for local consumption in the one and only standard format, and send the manuscript through the network to the editor. The editor corrects the manuscript and forwards it, again through the network, to referees who return it with comments. The editor then communicates with the author, and after a second round, sends the manuscript to the publisher, who stores the article in his database, generates abstracts, arranges the page layout and prints the journal, without the need for rekeying the text.

In the real world, there is a bewildering variety of computers, word processors and networks, serving some of the potential authors, editors, referees and publishers *(q.v.)*. The most obvious solution *i.e.* abandoning all but one of the existing systems, is obviously out of the question, in view of the cost and effort that have been invested in today's systems.

The only viable solution is to accept a truly machine independent standard that is compatible with existing computer hardware, word processor software, and the requirements of transmission across heterogeneous networks.

In the traditional system it is the copy-editor who annotates manuscripts with detailed instructions to the typesetter as to layout, fonts, spacings, indentation, page numbering, *etc.* This "mark-up" is the first step in bridging the gap between the different MS formats and that of the final publication. In electronic processing, a "mark-up language" may be defined as a framework into which are embedded commands which spell out the exact action to be taken when such a command is encountered. For example — \bf — would imply that the text following must be printed in bold-face type style.

There is effectively mark-up in every word processor: the string — \bf text \rm — placed in the middle of a line of text tells the system to put the piece [text] in bold-face type style and to continue in roman style from there on. This type of mark-up is called "specific mark-up".

Mark-up of this type can take on many forms: it may be applied at the point at which it is required (as in the example), by means of a special character at the beginning of a line (such as a dot '.'), by means of a "style sheet" at the beginning of a paragraph, or by means of macro's at the top of the manuscript, or by any combination of the above. All these forms are restricted to a single word processor language. Their development has revealed an increased need for a clear separation between the content and the form of a manuscript: while writing, the author should not have to worry about questions of presentation.

What is therefore needed is a system of "generalised mark-up", *i.e.* a system which permits the author to indicate the logical purpose of a given textual element without considering its physical appearance. For example, instead of entering — \bf text \rm —, the author would say — \highlight text — and leave it up to the system to decide how the text would be highlighted. Identifiers like — \highlight — are called "tags". Such "generic mark-up languages" remove the need for the author to bother about questions of presentation, and make the document less device dependent since the tag — \highlight — may produce boldface text on one system and italicised text on another. They are the natural extension of the mark-up procedures in today's word processors. Unfortunately however, mark-up codes are still heavily dependent on the word processing package which is used.

A meta-language is needed, which carries the notion of generic mark-up one step further: the author should not have to worry about which word processor will ultimately deal with the manuscript at the time of printing the journal. This meta-language should be able to define a mark-up language which is independent of any word processor.

In order to "connect" such a mark-up language to a word processor language another ingredient is required namely an "application program" which translates mark-up symbols into symbols that are recognized by the language of a particular word processor. Fig. 1 indicates the relationship between a candidate for a meta-language, SGML, and the different word processor mark-up languages, as well as the rôle of application programs. The traffic is one-way: having generated a marked-up manuscript an author can "reach" one or more word processor languages by invoking the corresponding application programs.

[1] Although there is a difference between word- and text processing systems, we shall use one term to indicate both.

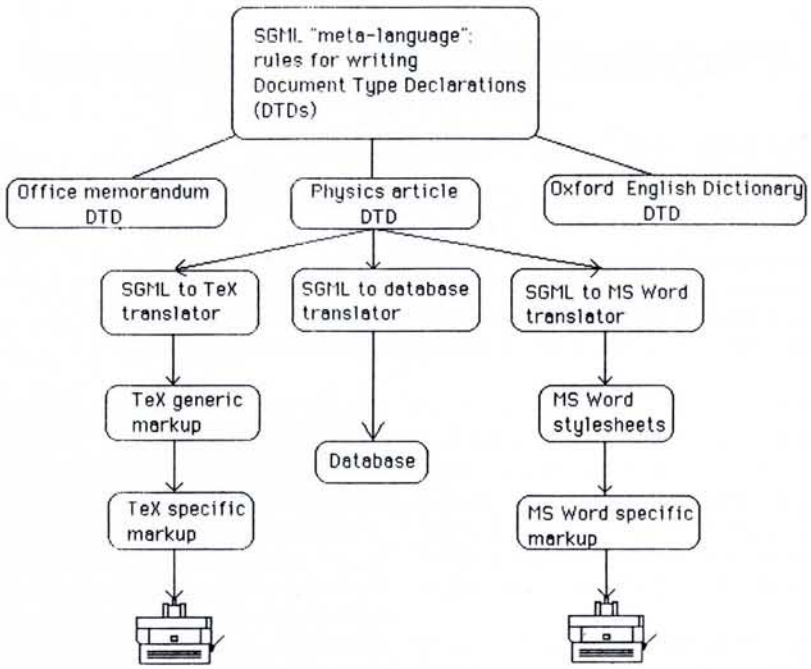


Fig. 1 — SGML and its applications.

A scenario for using mark-up in the publication of scientific papers is then the following. The author prepares his manuscript to his satisfaction on his own word processor and once he has finished the text, he modifies it in the context of a universally accepted generalized mark-up language. Fig. 2 shows an example of a tagged manuscript. It is then processed and forwarded, either by mail on a diskette or by transmission through a network, to the editor of the journal in question. Preprints for limited distribution in advance of publication are obtained by invoking the application software of the word processor in use in the author's Institute or University.

The editor's computer reads the manuscript and uses the application software relevant to *his* processor, to obtain a readable, *i.e.* a tag-free, copy. He may then add his comments to the marked-up version and send the revised manuscript on to one or several referees, who use their own application programs to obtain readable copies for their own use. Their comments are added to the original marked-up manuscript (or a copy), and returned *via* the editor to the author if revisions are required.

Once the editing process is complete, the final marked-up manuscript is sent to the publisher. The publisher uses the tree structure of Fig. 1 for more than one purpose. One application program leads to an "archive" where the manuscript is classified and stored in a permanent fashion. Another copies the abstract of the paper, if any, on to a file containing abstracts. A third translates the manuscript into a locally used word processor language to obtain a printed version, while a fourth generates camera-ready copy appropriate to the journal. In the case of publishing a book, other programs could handle the generation of an index, *etc.*

SGML

A meta-language capable of describing any (software and hardware independent) generalised mark-up language is the "Standard Generalized Mark-up Language (SGML)" developed by the International Standards Organisation (ISO). This system does not itself describe the structure of a document, but provides the framework for a mark-up language that does.

SGML has been an international standard for document interchange since December 1986 (ISO 8879) and it has attracted considerable attention from software developers (it is the second most sold ISO standard). The underlying assumption is that the text consists of logical components called "elements"; SGML then specifies the manner in which the elements of a document should be indicated in the text. Marking-up is done *via* the "tags" mentioned above. A "Document Type Definition (DTD)" defines the structure

Fig. 2 — Example of a physics article marked-up with SGML.

```
<!DOCTYPE ARTICLE SYSTEM "ARTICLE DTD A">
<ARTICLE>
<TITLEP>
<TITLE>Experimental obeservation of lepton pairs of invariant
mass around 95 GeV/c&sup2; at the CERN SPS collider
</TITLE>
<COLLAB>UA1 Collaboration, CERN, Geneva, Switzerland
<DATE>3 june 1983
<AUTHLIST>
<AUTHOR ID=RAL>G. Arnison
...
<AUTHOR ID=CERN>C. Rubbia
...
<ADDRESS ID=RAL>Rutherford Appleton Laborarory
...
<ADDRESS ID=CERN>CERN
</AUTHLIST>
<ABSTRACT>Abstract
<P>We report the observation of four electron--positron pairs
and one muon pair which have the signature of a twobody
decay of a particle of mass &approx; 95 GeV/c&sup2;. These
events fit well the hypothesis that they are produced by the
process &pbar; + p &rarrow; Z&sup0; + X (with
Z&sup0; &rarrow; &ell;&sup+; + &ell;&sup-;), where Z&sup0; is
the Intermediate Vector Boson postulated by the electroweak
theories as the mediator of weak neutral currents.
</ABSTRACT>
</TITLEP>
<BODY>
<H1>Introduction
<P>We have recently reported the observation of large
invariant mass electron--neutrino pairs...
</BODY>
</ARTICLE>
```

for a given "class" of documents; scientific articles could be defined to be such a class (see Fig. 1),

The DTD contains *inter alia* the following information:

- The names and definitions of the tags (and the end-tags) that may occur in the document.
- The order, permissible frequency of appearance, and the nesting of the tags.
- The attributes of the tags (*e.g.* < p id = A > permitting a paragraph to be referred to elsewhere in the manuscript as < refpar refid = A >).
- The reference symbols used in the text.

In order to let the DTD interact with the document some sort of framework that will hold both the document and the DTD is needed. This framework is called the *SGML input system*. It first task is to submit the DTD to a compiler called the *parser*. The next task is to have the compiled DTD check that the document, as marked-up by the author, contains no mistakes (*e.g.* no end-tag where there should have been one). The final tasks are to submit the document to an application program which translates the tags *etc.* into a form suitable for processing by a word processor such as T_EX, Microsoft Word, Script, *etc.* and to transport the document to a file transfer utility for transmission across a network.

The principal limitation of SGML as sketched above is its user-unfriendliness in coding the manuscript with tags. Replacing the tags (or some of them) by any context sensitive properties of the text (*e.g.* defining a blank line followed by a line indented by 3 spaces as the equivalent of <p>, a new paragrah) would violate the principle of separation of structure and appearance, which is at the very basis of the system.

A second drawback is that there are only a small number of WYSIWYG (What You See is What You Get) word processors that understand SGML, and those that exist require a fair amount of programming to set up an application.

A third drawback is that authors take pride in creating aesthetically pleasing documents; imposing a structure on a document by using a generalised mark-up scheme defined by SGML is sometimes seen as a restriction on their creativity.

Some well known SGML systems are Datalogics "Writerstation" (for IBM compatible PCs); SoftQuad "Author/Editor" (for the Macintosh); the IBM SGML/DCF (Document Composition Facility) translator products for VM/CMS and the Sobemap SGML parser for VAX VMS and for other systems.

Current Use of Processors: a Survey

During 1988, the EPS Publications Committee formulated a plan of action comprising the following:

1. Conduct a survey on the use of word processors for the preparation of physics papers.
2. Initiate a pilot project, to be carried out by experts in SGML and develop an SGML input system, a DTD, a parser, and application programs.

Work on point 2 is in progress and will not be discussed here.

For the survey, questionnaires were distributed to publishers, who mailed them to authors *e.g.* along with proofs of papers. The questions concerned the types of computer, word processor packages, the processing of mathematics, storage media and access to networks.

An estimated 2800 questionnaires were sent out to authors worldwide who had submitted papers to a range of European physics journals published by EPS, Academia (Czechoslovakian Acad. of Sci.), Edition de Physique, the Institute of Physics Publishing Ltd., North-Holland, Nuovo Cimento, the Royal Swedish Academy of Sciences and Springer Verlag.

Of these, 1089 or 39%, were returned out of which 932 authors, or 85%, used a word processor in preparing their manuscript.

The replies were subsequently divided into those of users of large, multi-user systems and those of small, single-user systems, personal computers or work stations. Version numbers were ignored. Of the 932 authors using a word processor, 246, or 26%, used a large system, and 686, or 74% used a small system. This confirms the trend towards distributed computing and the suitability of small systems for word processing.

For each of these two categories the sample was further sorted according to the type of system, the type of word processor, and the capability for handling mathematics.

Fig. 3 shows the fraction of cases where the types of computer and word processor were known to the author ("unknowns" comprise the categories "not mentioned", work done by secretaries, *etc.*).

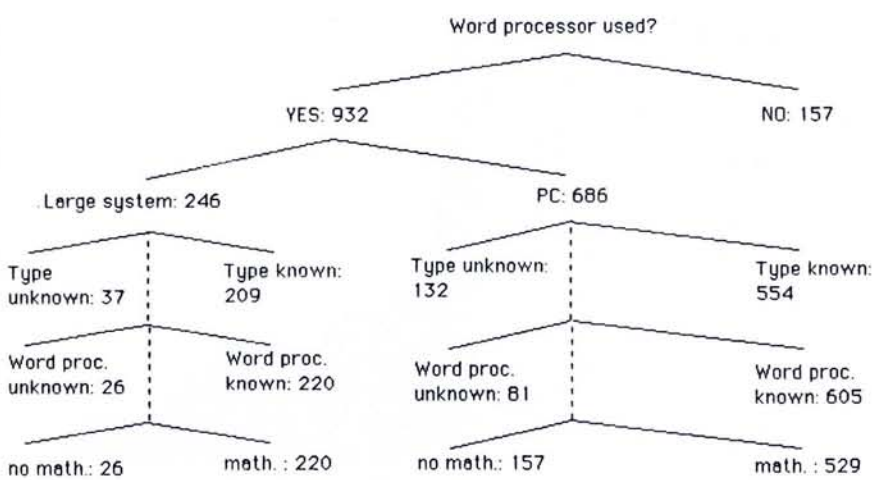
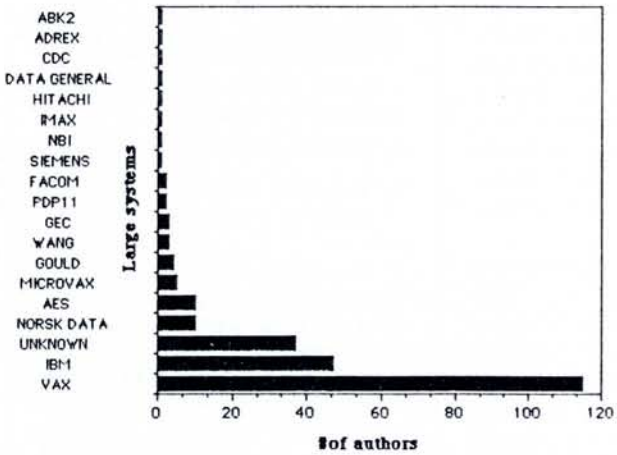
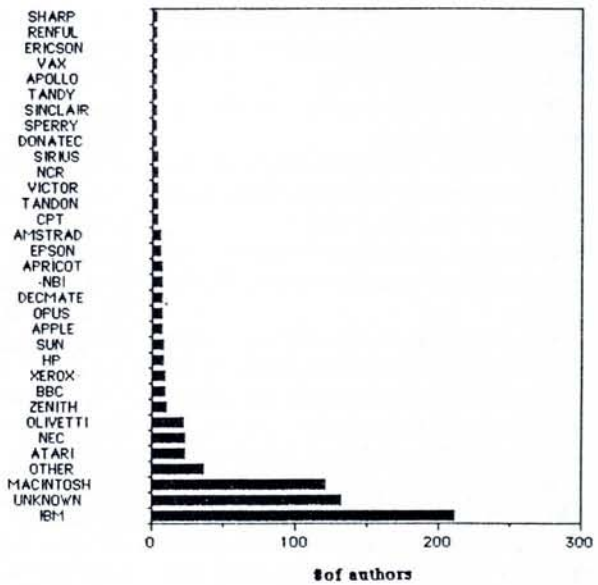


Fig. 3 — Summary of the 1089 replies analysed, 39% of the total sent out.

Fig. 4 — (Below) Large systems used by physicists for document production.

Fig. 5 — (Right) Small systems.





Figs. 4, 5, 6 and 7 show, respectively the large systems, the small systems, the word processors on the large systems and the word processors on the small systems.

For the large systems, a total of 18 different types were reported; of these eight were reported by one author each. For the small systems, there are a total of 77 different types of computers, of which 34 were reported by one author each.

In the large systems 20 different word processors were reported. Of these, 10 were reported by one author each, and 15 have mathematics. In the small systems, 85 different word processors were reported. Of these, 46 were reported by one author each, and 71 have mathematics.

Among the large systems, VAX was the predominant computer: 46% of all authors used it. The next largest category was IBM, with 20% of all users. Among the small systems IBM dominate with 30% of all users, while 16% used Macintosh. If one adds IBM compatible machines to the category IBM, this group increases by a further 10%. The number of these systems connected to networks (*via* mainframes) is unknown.

Among the word processors T_EX dominated on both the large and the small systems followed closely by Wordstar on the small systems. In these totals, no distinction has been made between the various T_EX macro packages ($LA T_EX$ PHYZZX, AMS-T_EX *etc.*). Although T_EX was used most frequently it still accounts for only 22% of the users.

Mathematics appears to be a general requirement: more than 80% of the users handled mathematical formulae with word processors.

Concluding Remarks

The high cost of journals and the long delays in the appearance of papers in journals on the one hand, and the ready availability of word processing software, installed in desk top work stations on the other, are pointing the way towards new, speedier ways of communicating scientific results among scientists.

The major challenge lies in defining a system that can act as an umbrella over the system currently in use. The Standard Generalized Mark-up Language is a contender for this position. It has been adopted by organizations as diverse as the Commission of the European Communities, the U.S. Department of Defence, and Oxford University Press. The authors believe that the community of physicists working in physics institutes and Universities should also adopt this standard.

Fig. 6 — (Below) Text processing systems on large computers. The category "other" refers to systems which were cited in one reply only.

Fig. 7 — (Right) Text processing systems on small computers. The category "other" refers to systems which were cited in one reply only.