[*Machine Translation Today: Translating and the Computer 15*. Papers at a conference…18-19 November 1993 (London: Aslib)]

TERMINOLOGY AND THE COMPUTER - ATTENTION SHIFTS TO THE MICRO

Roger Bennett
Terminology Unit, Commission of the European Communities, Brussels

Following an outline of the organizational context of terminology work in the Commission, this paper considers the various types of computer application used, ranging from mainframe terminology databases (Eurodicautom) through documentary databases and wordprocessing software to specialized PC applications, with particular attention to current developments. An attempt is made to explain the factors leading to the development of an in-house terminology management tool for a networked PC environment and the features of the package developed are reviewed. Consideration is also given to the lessons which others might draw from the Commission's experience.

INTRODUCTORY HEALTH WARNING

Several words of warning should be given at the outset. Firstly, this paper is not intended as a balanced presentation of terminology and the computer in the Commission of the European Communities. Rather, attention is centred on the author's own special area of interest, namely **terminology and the microcomputer,** which it is to be hoped will also correspond to the interests of conference participants. Secondly, those hoping for an academic discourse on the topic will hope in vain. It is sixteen years since the author could, by any stretch of the imagination, have been described as an academic, and the topic is at all events one which should benefit from a pragmatic approach. Finally, and perhaps almost in contradiction to the last point, no attempt will be made to recommend, or criticize, specific products available on the market. Aside from the need to maintain the position of impartiality incumbent on a public servant, it would quite simply be unwise to make qualitative assertions in the context of a rapidly evolving market in which yesterday's leading contender may well be today's lamebrain and vice versa. What will, however, be attempted is a general guide to potential gains and pitfalls.

THE REAL THING

Organizational Context

All that follows must be seen against the background of a translation operation with a unique combination of characteristics. The Commission's Translation Service is the largest such service in the world, with over a thousand staff. It is also unusual in its need to cover nine languages as both source and target, with a substantial proportion of the texts translated acquiring direct legal force in the Member States.

The impression of a monolithic organization which this might suggest is, however, misleading, since the Service is broken down for operational purposes into seven subject-based departments, (each serving a number of Commission Directorates-General), each of these in turn subdivided into nine units of around twenty staff (translators plus support staff). The correlation of nine units and nine official languages is, of course, not coincidental. Each unit is linguistically homogeneous and has responsibility in principle for translating all texts in its subject area into the mother tongue of its translators. Thus, there is a Danish translation unit for external relations, customs and development, and an English translation unit for agriculture, fisheries and the structural funds - and so forth.

A further significant division exists in geographical terms, with part of the Service located in Luxembourg - for a combination of political and operational reasons. As an aside, be it said that this geographical separation has its impact on questions of integration of computer systems.

Finally, in addition to its seven translation departments, the Translation Service has a number of "horizontal" or staff units, most of them grouped together in a Directorate for General and Language Matters. The author's own unit - Terminology - falls into this category, being additionally split more or less evenly between Brussels and Luxembourg (36 staff in Brussels and 39 in Luxembourg).

Terminology - Pragmatically Speaking

This first mention of "terminology" - after the title - brings us belatedly to the question of what we actually mean by the word. Since this topic could, of itself, easily occupy several papers of high soporific potential, the definition adopted here will be purely pragmatic (and - please! - not open to discussion). Terminology will be taken as the identification of equivalents in other languages for problematic words or phrases (in all fields - not just the conventionally "technical") and the utilization of such equivalences as aids to the work of translators and others. Note that this includes, but is by no means confined to, normative terminology (the production of terminology "standards"), which in any case only represents a relatively small proportion of the work of the Commission's Terminology Unit.

Eurodicautom - The Mainframe in its Element

In a translation operation of the scale indicated above, it is not surprising that the best known and most important dimension of terminology and computers should be the Commission's mainframe terminology databank Eurodicautom, which will already be familiar to many conference participants. In spite of its software having for the most part been devised as far back as the 'seventies, this system continues to demonstrate some of the strengths of the mainframe at its best. In particular, the speed of response (at least when using the Commission's own interrogation software) is impressive bearing in mind the volume of information (this remains the largest terminology databank in the world) and the fact that the computer holding the information is located physically in Luxembourg. A further plus point is the relative simplicity of the user interface, which is sufficiently unintimidating for even the computer illiterate. These two points of response time and interface are crucial to the implementation of almost any system which is to be used by more than one person and will come up again in our consideration of microcomputer applications.

At this point it is worth emphasizing the great variety of the texts to be translated by the Commission. It used to be said that we could be expected to encounter all fields of human endeavour except defence - now even this exclusion can no longer be taken for granted. This situation contrasts with that of many other relatively large-scale translation operations (e.g. those in multinational companies, which often concentrate heavily on technical manuals and technical marketing material for a limited range of products) and means that for the foreseeable future a high-performance, broad-coverage tool for on-line terminology queries is going to remain vital. Also significant in this context is the availability of Eurodicautom on line to users outside the Commission, including translators working for the Commission on a freelance basis.

Eurodicautom is certainly not ideal as it stands. Weaknesses exist as regards user interface and interrogation capabilities, speed and flexibility of updating, batch processing capabilities and quality of content in certain areas.

User interface and interrogation capabilities. In spite of Eurodicautom's already considerable ease of use, some improvements have been requested, notably the possibility of establishing user profiles which will avoid the need to specify such parameters as source and target languages at system startup. Similarly, some fields which cannot currently be interrogated under normal circumstances should be made available to searches.

Speed and flexibility of updating. As matters stand, updates do not always occur as fast as users would wish, although progress has been made in the last year or two. There has also been criticism of the lack of scope for on-line updating of information by ordinary users. This, however, is more an indication of one of the areas where a central system does not provide the whole answer than a criticism to be answered within such a system. Consider the implications of allowing all (or many) of the thousands of users of such a system to modify entries directly - a recipe for confusion, and certainly not a path to improving the consistency of the quality of entries (already a sensitive point). The answer would rather seem to lie in the addition of a further, local layer to the system - of which more anon.

Batch processing capabilities. Limited facilities for batch processing (i.e. the processing off line of a series of queries submitted to the computer together, as distinct from conventional interactive query processing) already exist, but this is certainly an area where more needs to be done in order to integrate Eurodicautom with current efforts to promote labour saving in the translation process.

Quality of content. There has been some dissatisfaction as regards the quality, and particularly the consistency of the contents of Eurodicautom. This can partly be explained by the system's underlying philosophy, which has from the outset been to provide a maximum of information, leaving ultimate responsibility for judging the information to the user. Clearly, such an approach has disadvantages for the non-translator user in particular. It does, however, have the advantage of having enabled the databank to grow sufficiently rapidly to reach "critical mass" before users lost patience with the system. This refers to the generally accepted principle that a mass-user system will only gain a degree of popularity amongst ordinary users if it provides viable answers to a significant proportion of questions. This is clearly a problem in the early days of a system, when insistence on very strict quality-control can damage long-term credibility by slowing down the pace of data entry to a crawl. The other option - to leave the system off line until it reaches critical mass - is also unlikely to be an answer, since the impatience of both senior management and the prospective users, aware that an information tool has been under development for some time, will prevent this situation being maintained for any significant time. It is consequently not surprising that Eurodicautom, like other large-scale translator-oriented terminology databanks, initially gave priority to volume of data input.

Now, however, the early days are past and quality control must occupy a greater place in the scheme of things. This is, indeed, the policy now being followed as regards both new entries and existing material, at least for the public version of Eurodicautom (though some material of a more questionable quality may continue to be added for in-house use only). The methodology of efforts to improve quality is a question of considerable delicacy, influenced also by cost factors and considerations of copyright attaching to certain collections in the databank, and falls outside the scope of this paper.

At all events, there is no reason why Eurodicautom, in either the more refined version of its present incarnation currently under development or possibly a radically restructured "reincarnation" incorporating other resources and meeting rather wider goals, should not continue to provide the central, high-performance base required to answer the majority of Commission translators' terminology queries.

Who Needs Anything Else?

The idea that a central terminology databank can be **all things to all men** is ridiculous, however. Even at the mainframe level, terminologists both require access to other resources and may assist in the feeding of other systems.

As far as feeding other systems is concerned, mention should be made of machine translation systems (specifically, the Systran system which has been operational, for limited purposes, within the Commission for some years). There is scope for input to system dictionaries from the Terminology Unit. This possibility has, however, yet to be fully exploited, partly due to the operational context in which machine translation has developed and partly because of constraints which prevent the transfer of information "en masse" (e.g. the need to specify parts of speech or the like so that grammar rules can operate correctly). Similar possibilities for data input by terminologists (not subject to the same constraints and hence more fully exploited) exist in relation to machine-aided translation systems (currently taking the form of text preprocessors).

As far as other resources used by terminologists are concerned, the most obvious are full-text documentary databases such as those maintained by the Commission, which provide a means of rapid access to a vast body of textual sources (most of them already available in multilingual form). Terminologists also find access to a variety of other databases helpful in their problem-solving work. These can range from terminology databases produced by other organizations, such as the Canadian base Termium, through scientific abstract bases and technical standards bases to magazine and newspaper databases. This is the first point at which the microcomputer truly comes into the picture, since some of these bases are simply not available on line and can only be consulted in CD-ROM form, thus effectively necessitating the use of a microcomputer (and, indeed, an IBM PC-compatible for access to the maximum range of material). At the "sharp end", terminology work will always have an open-ended, potentially insatiable requirement for such information sources. Thus far, however, there is little to distinguish the terminologist's requirements from those of almost any information officer - albeit the linguistic orientation of his searches often require databases to be used in ways for which they were never intended and are ill-suited.

Local Terminology Manipulation

Now we come to the meat of this story, namely the need for computer systems to cope with terminology management tasks not well suited to the powerful but rigid mainframe environment.

As has already been indicated, mainframe systems allow only limited scope for on-line updating. This, and the undesirability of exposing "unripe" material to the public gaze too early in its life, are already powerful arguments in favour of some kind of local tool to manipulate terminology collections prior to uploading to the central databank. Additionally, the continued (and considerable) demand for paper products on the part of many customers (yet another disproof of the notion that computers were going to lead overnight - or even within the foreseeable future - to the paperless office) demands desktop publishing capabilities not readily available in a mainframe environment.

Minicomputers - Neither Fish nor Fowl

Attempts to meet the need for local tools within the context of the Unix-based minicomputer network which has until very recently been the Translation Service's primary working environment suffer from some of the shortcomings of the mainframe whilst lacking its strengths. Thus, aside from the data manipulation weaknesses inherent in wordprocessing systems (considered below), the sophistication of the wordprocessing packages available within the minicomputer environment proved unequal to the demands of serious desktop publishing and dependence on central system administrators constituted a severe handicap in the development of solutions specific to the Terminology Unit (whose needs almost inevitably took a back seat to those shared by 90% of the Translation Service and therefore perceived as offering more scope for productivity gains).

More potential at one time appeared to be offered by a minicomputer database system (developed using Oracle, for the technically minded) designed to provide a tool for terminology management to the individual translator and translation unit. Unfortunately, this failed to meet (and could not readily be adapted to meet) several of the Terminology Unit's cardinal requirements. Specifically, it was unable to cope with Greek at the same time as providing access to the full range of accented characters in the other Community languages and offered little in the way of assistance towards our desktop publishing goals. Furthermore, the system was initially cumbersome for data entry and searches were slow. Improvements have since been made in these last two respects, though the new search routines achieve speed at the penalty of a lack of sophistication.

Salvation in the PC

In these circumstances, the Terminology Unit in Brussels stumbled along as best it could for some time using the tools at its disposal (primarily the Unix wordprocessing system and a certain number of old dedicated wordprocessors - which were in fact more effective for the production of paper glossaries). The advent of a limited number of IBM PC-compatible microcomputers (and the prospect of a PC network in due course) together with the arrival in the Unit of a computer freak with some programming knowledge (guess who) combined to direct attention to the scope for achieving rather more, given the potential flexibility of PC applications and the possibility of at least to some extent controlling development from within the Unit. With hindsight it is clear that the outcome could equally well have been a total flop, and the lesson to be drawn from this paper is certainly **not** that in-house development is the panacea for all ills!

No Commercial Package Ideal

Over the past couple of years, consideration has been given to the possibility of using a variety of commercially available software packages to meet the Unit's specific needs. So far, no package or combination of packages has seemed to meet the bill. To see why, and also to highlight for others in a similar situation the importance of considering all requirements from the outset, a brief summary (not necessarily in order of importance) of our key needs is called for:

<u>Full multilingual capabilities.</u> As mentioned earlier, any system chosen would have to be able to display Greek correctly at the same time as all other Community languages. It should also permit easy data entry in all Community languages. More than this, it should sort correctly in each language (including the difficult cases such as Danish, the Spanish ll and ñ, and Latin characters occurring in Greek texts), preferably also allowing for "filler" words such as "de" and "à" in French to be ignored, as many linguists would expect in a dictionary.

<u>Full multiuser functions.</u> Even if initially implemented on a standalone basis, the system should be fully network ready. In other words, it should allow files to be opened and modified simultaneously by several users, with full file and record locking facilities to ensure data integrity. The primary reason for this requirement is to rationalize and speed up the entry of multilingual terminology lists produced under pressure of time - a frequent requirement. Very significant gains are possible here if several terminologists and/or secretaries can work on the file at the same time, entering or correcting information in different languages from different machines.

<u>Sophisticated and fast search functions.</u> Information currently being worked on must nevertheless be available at least to local searches, preferably on both a single file basis and across multiple files. Such searches should be structured so as to work rapidly even in a networked environment, since the average terminologist is not much more patient than the average translator.

<u>Direct export to the central databank without manual reformatting.</u> Since the central databank is the ultimate destination of most material worked on in the Unit, logic demands that information be exportable with an absolute minimum of effort.

<u>Easy import of information from other data formats.</u> Given the amount of information that the Terminology Unit receives from outside sources, any system adopted should be as open as possible to data in commonly encountered formats.

<u>Extensive desktop publishing capabilities.</u> In view of the demand for printed glossaries and terminology lists, desktop publishing capabilities (achieved either on a standalone basis or by dose integration with a mainstream wordprocessing or DTP package) are a must. Perhaps most difficult of all, this should include at least partial automation of the production of cross-referenced indexes, which are both vital to effective use of a multilingual glossary and one of the most tiresome and time-consuming components to produce by hand.

Wordprocessing Packages Out of the Running

Looking at the above shopping list it is clear that even the most powerful wordprocessing packages fall short by a wide margin. Aside from other considerations, few of them have sorting routines which handle all Community languages correctly, even at the most basic level of absolute alphabetical order. They also have to be ruled out for any system which is to run in a multiuser environment, since even the mailmerge facilities of the most advanced wordprocessors fail to provide the proper facilities for file sharing. Practically speaking, this should rule them out not only for the Commission's purposes but for any terminology use other than **possibly** an individual translator keeping a very simple wordlist for personal use.

General "Out-of-the-Box" Database Packages Also Fail

General database packages which are supposed to be usable straight out of the box (in other words without added programming by the user) also fail on several of the above counts. The multilingual requirement in particular is a problem, with some packages unable even to display Greek, whilst others have sorting routines which are inadequate and cannot be adjusted. Search routines are another area of difficulty. Whilst many packages at first sight have the necessary capabilities, their **fast** search routines are often unsatisfactory because they work on the basis of indexes and will only locate an element at the **beginning** of an entry. More flexible search strategies are usually also available, but operate unacceptably slowly on anything but a minimal volume of data (a problem exacerbated in a network environment). Indeed, speed of operation tends to be a general problem with PC database packages.

Those database packages which offer a sophisticated programming language have more potential. Here again, however, caution is required on several counts - one of speed. A distinction should be made in this connection between the majority - "interpreted" database languages, which are slow because each instruction has to be translated into machine language every single time it is executed - and the minority of "compiled" languages which are faster by several orders of magnitude because the instructions are compiled **once** into machine language and the whole program can then be executed directly, without the intervention of an "interpreter". Technically speaking, this is an oversimplification since even "compiled" programs contain some interpreter-like features. Nevertheless, the basic point remains true.

A further, and perhaps more crucial problem, lies in the extent of the programming effort required to meet a sophisticated set of requirements. When this is to involve such enterprises as customizing sorting routines and automating the creation of cross-referenced indexes, the task should certainly not be undertaken lightly, or without substantial prior experience. Nevertheless, compiled database languages do have potential under certain circumstances and we will return to them later on.

Full-text Search Tools - Impressive But...

A number of software packages have appeared in recent years whose purpose is to locate text rapidly anywhere in a large body of files. Clearly, such tools can be of benefit to the translator, e.g. in permitting searches of prior translations for problem phrases, and they mostly meet the requirement for sophisticated and rapid search routines. Unfortunately, they fail the requirements for a terminology-oriented application on almost all other counts.

Specialized Terminology Packages and Translation Workbenches

Specialized PC software packages aimed specifically at the translation market are now available. Some of them are designed as terminology management tools only, whilst others combine machine-translation capabilities with some terminology features and yet others claim to provide a complete toolkit ("workbench") for the translator including machine-aided translation on the basis of existing terminology collections or bodies of parallel texts.

It might be thought that one or more of these would provide the answer to all our prayers, but alas no. This is not to say that they would not provide the ideal solution for individual translators, or for organizations with requirements different to ours. The following comments should therefore be taken merely as illustrating a series of potential problems. In any case, no one product suffers from all the weaknesses cited.

Multilingual capabilities. Surprising as it may seem, many even of the specialist products are unable to meet our Greek "litmus test". Though our situation is perhaps more demanding than that of other organizations and individuals, it is clear that a key question for any potential buyer should be: "Does the system cope properly with all the languages I am likely to need?" Beware in particular of statements such as: "That language is not yet supported but soon will be" - believe it when you see it. For that matter, even seeing should not always mean believing - sometimes a particular language may be displayed correctly but not sorted correctly or (and this can be a problem in the Windows environment) you may not be able to use certain characters in a search because their display has in fact been achieved by screen font trickery without proper underlying coding.

Flexibility. Some products were originally designed for very specific needs and suffer from rigidities partially consequent on their origin. This may mean that certain key functions are absent and cannot be added. (In our case, no system came, as supplied, with routines to automate the creation of cross-referenced indexes.) Beware of giving up too easily on your most fondly cherished hopes (though some of them **may** ultimately prove unrealistic).

At least as annoying, and in some ways more so, is rigid file structure. Many systems work with fixed-length fields and records - for what were, at least originally, sound software design reasons. At its extreme, this may mean that the user has no choice whatsoever in relation to record structure, and is obliged to work with exactly the fields (and amount of space) provided by the system designer. A particularly irritating variant of this occurs where the designer has also, in his infinite wisdom, decided that a whole series of fields **must** be completed before the system will accept a record **and** there is little or no scope for the automatic entry of default values. This is in direct contravention of the principle that computers should **save** labour and a sure way to demotivate all but the most dedicated users.

Less irksome, but still more of a problem than might be imagined, are the limitations on maximum field length (often around the level of either 100 or 250 characters) and/or maximum record length. At first, 250 characters may seem like a lot, but we, at least, have found many instances where it simply does not suffice. If the user is lucky, systems with this type of file structure may allow one (or possibly more) variable-length ("memo" or "note") fields but usually subject to severe constraints (e.g. exclusion from searches) and possibly hidden disadvantages (see below - "File bloat").

Even in their most innocuous form (leaving the user more or less total freedom to specify field size), systems with fixed field lengths require very careful design. The length of each field normally has to be specified at the time of file creation and cannot be changed with safety thereafter. Despite the best laid plans, it seems almost inevitable that Murphy's law should operate here, with the discovery at a critical stage that one field is too short for the information you want to enter!

File bloat (disk guzzlers'). Another feature often ignored until too late is "file bloat". This is particularly characteristic of fixed-field length systems (which typically have large amounts of "white", aka wasted, space since every entry must be as long as the longest you are going to require). It may be imagined that ever greater hard disk capacities make this a matter of little concern. Unfortunately, the space demands of software (particularly Windows software) have also grown, and space-intensive text databases such as the typical terminology database can consume the remaining megabytes in no time. A simple calculation, multiplying the length of all the fields required by the number of terms you expect to enter during the life of your existing hardware configuration (not forgetting to add the overhead normally required by the control information for each record), can be a chastening exercise.

Even systems with variable-length fields are not immune from the phenomenon of file bloat. At their worst (exemplified by one very common file format), they can actually be more wasteful than fixed-length fields. Even when the worst excesses of bad design have been avoided, it is unwise to assume that space requirements will be anything like as little as the number of characters the user actually types into the system. Significant additional amounts of space are almost always consumed by control information necessary to rapid movement through the file and reliable data retrieval, though the performance gains from this are usually enough to compensate for the disk space lost.

Network performance. Aside from the straightforward problem of disk guzzling, file bloat can also contribute to poor performance in a networked environment. At least in the case of systems where the software is run on each individual PC, fetching data from and writing it to a central file server, the volume of data to be moved between the remote disk and the local machine can be a crucial brake on performance, bearing in mind that even with modern networking systems this type of retrieval is still significantly slower than accessing a local hard disk (i.e. one physically forming part of the same PC). Systems which manage to minimize data movements will thus tend to enjoy an advantage, although the extent of this will often only become clear when an application is actually implemented.

Other difficulties can arise in a network context, notably as regards record and file locking (essential for a multiuser system, as mentioned earlier). Many of the systems currently on the market have developed from an essentially single-user background and still reveal evidence of this in fragile network implementations. Of course, for the self-employed translator, or even very small workgroups, this handicap may be of limited significance.

Absolute speed/power/simplicity mix. There is often a tradeoff between absolute speed, power and ease of use. Speed may be achieved at the expense of flexibility and power (in the sense of the range of tasks which a system can perform), since it is easier to program a system to perform a relatively rigid and limited range of tasks at high speed than it is to devise a package which will be both open-ended and fast. A compromise solution chosen by at least one package (following the pattern of many mainstream software packages) is to provide an extensive scripting or macro language which the end user or intermediate developer is supposed to use to customize the system. Unfortunately, such languages generally suffer from the speed shortcomings of "interpreted" computer languages outlined earlier. At the same time, they are non-standard and thus require even those with previous experience to relearn how to program (with a language which can only be used in this one context). They will also tend to lack some at least of the elements of full-blown programming languages, making certain customization tasks either impossible or, at the least, fiendishly difficult. Once again, the automation of cross-referenced index production proved a case in point, giving the author a severe case of intellectual indigestion.

User friendliness. Ease of use is also impeded by the need to use scripting languages to implement more than the most basic of functions. Of course, the reverse may be true if an enthusiast can be found to undertake all the script language programming or this work can be contracted out (though the narrowness of the market means that the package's authors will probably enjoy a near monopoly of such contracting services).

Lack of user friendliness is a problem in other respects too. Many packages require a considerable effort at the outset in terms of designing file structure and/or expect the end user to master a relatively complex set of manipulations even for basic data entry and searches. As a generality, some potentially powerful systems appear offputting for all but the enthusiast, for (and often by) whom they may well originally have been designed. Such a shortcoming may be of little significance to an individual computer-aware translator or lexicographer, but constitutes a serious impediment to implementation in even relatively small organizations where the majority of staff see computers at best as no more than a tool and at worst as something approaching the devil incarnate. Few software developers and buyers have ruined their careers by overestimating user resistance to apparently complex systems.

Desktop publishing capabilities. Though this may be of lesser significance to some user groups, lexicographers and terminologists require advanced desktop publishing capabilities. In addition to data manipulation questions such as the already cited problems of sorting and index production, this also implies sophisticated text formatting functions, since the users of "paper" terminology products increasingly expect output of near-book quality even in limited-distribution documents. Specialized products can rarely offer all the facilities necessary on an internal basis.

Integration with other software. A potentially powerful solution is to opt for integration with a mainstream software package, perhaps by producing files in a format compatible with that package, ready for final tidying up (if necessary) and printing. Some terminology management packages at least claim to implement such a solution. Unfortunately, a measure of caution needs to be exercised here because integration is usually with one of the simpler file formats - perhaps Ventura Publisher (which uses "logical markers" rather than special codes for formatting and text attribute information) or Microsoft RTF (Rich Text Format) - a file format designed in principle to permit the exchange of information between a number of packages with minimal loss of formatting and attributes. This **may** be all the user needs in certain circumstances, but inadequacies are likely to show up when attempts are made to operate in a translation workbench mode, using terminological information as an aid in text processing.

Conversion and reconversion - a recipe for frustration and wasted effort. Ideally, a translation workbench should operate directly on files in the "native" format of the user's customary wordprocessing system. Few systems actually do so - be it noted that neither Ventura Publisher format nor Microsoft RTF are truly native to any major wordprocessing package (though a point might be stretched in the case of RTF and Microsoft Word). To take a concrete example of the problems which can result, the Commission's Translation Service is in the process of converting to Wordperfect as its main wordprocessing package. Wordperfect can read Ventura Publisher files with no problem, but it cannot translate the formatting and attribute markers into the corresponding codes, so these markers will be printed as ordinary text! The apparent alternative of using Ventura Publisher itself for wordprocessing is simply not realistic, firstly because the rest of the Commission would not follow suit (thus guaranteeing that texts would have to be "converted" at some stage) and secondly because using Ventura for ordinary wordprocessing would be rather like trying to type wearing woollen mittens. Similarly, though Wordperfect claims compatibility with RTF, the implementation is imperfect at least in the versions used by the author and results not only in the incorrect conversion of some formatting information but also the loss of **all** accented characters. Yet worse is the situation with a package which will only use "ASCII" or "generic wordprocessing" files. This effectively loses all formatting and attribute information and the resulting files will require extensive reworking before they can be delivered to any customer requiring other than the simplest of output. The initial instinct may well be to shrug and accept the additional work of file conversion (particularly if there seems to be no better alternative), but experience of working over a period of several years in a context of repeated file conversions has taught the author to mistrust any systematic requirement for conversions as a source of frustration for all but the most dedicated enthusiasts.

Operating environment constraints. The above problem is actually only one facet of a wider question of operating environment constraints. Such constraints exist for all users, but are perhaps most acute not for the self-employed translator working alone (who may have limited funds available but can at least decide what to spend the funds on with a reasonable degree of freedom) but rather for those operating in the context of a large organization. In the latter case, it is likely that any system acquired will have to fit in with an existing computer environment rather than vice versa. Typically, this will be a network of IBM PC-compatible machines running Windows 3.1 and DOS, with either Wordperfect or MS Word (DOS or Windows versions) as the main wordprocessing software. In this situation, opting for a system which requires OS2 (as is the case with at least one market contender) or one of the various PC flavours of Unix is a non-starter. Even a DOS package may be unacceptable if it will not run reliably under Windows or requires more base (as opposed to total) memory than is available in the network environment. The reverse side of this coin is that a system specifically designed for Windows should be looked at carefully to check that its speed has not suffered significantly because of the greater effort it needs to put into display management and other "housekeeping" tasks. For all of the above reasons, beware of demonstrations using limited volumes of data in a computer environment different from the one in which a system will have to operate (even if you are assured that this should "make no difference"). This is all the more true if the cost of the system is high. Here, too, comparisons need to be made with care, since costs can vary widely and the value-for-money calculation can look radically different depending on number of users and pricing structures - one system which is expensive for a single copy but requires only one licence for the whole organization may, for example, turn out much cheaper than another which has a lower cost per copy but requires the purchase of a licence for each user in the organization.

AND NOW FOR THE GOOD NEWS

From the tenor of this paper so far, the reader might be forgiven for thinking that he should go home and hide in a corner hoping that the computer age will go away, or that the boss will select some other poor fool to find the "ideal system". In reality, the intention is merely to introduce a degree of healthy scepticism before going on to outline, in the light of the Commission's own incomplete experience, some of the things the user can aspire to do with PC-based systems in the field of terminology work.

Genesis of a PC-based System

A good deal has already been said about the whys and wherefores of not choosing a commercial system. To look at the development of our in-house system in a rather more historical perspective, it originated (essentially in the author's mind) as a response to two prime needs. First came the need for a system to replace and improve on the functionality of the ageing dedicated wordprocessor system used for the production of glossaries when the author first joined the Terminology Unit some four years ago. Interestingly, however, the second impulse behind the system came from a requirement which partook of both terminology and computer-assisted translation, namely the need for a system to compile and manage the information required for the processing of a regularly recurring (monthly) list-type document to be translated into all Community languages, and actually produce (re-sorted in all the target languages) the actual translation.

These relatively modest beginnings defined a number of the essential features of the system, but others have been added at regular intervals. Even now, the pace of development has slowed only a little, and hardly a week goes by without functions being added to the package or significant changes being made in existing functions. This, indeed, is one of the major benefits of not just in-house but in-unit development - the ability to respond very rapidly to new demands (or the discovery of shortcomings!). Of course, there is a downside too - potentially dangerous dependence on a single programmer (the "And what if he walks under a bus?" syndrome). Partly to allow for both of these considerations, the development tool chosen was Clipper, a popular database development language descended from DBase with a strong admixture of C and the latest trend - "object orientation". This combines the advantages of short development time (because it is a very high-level and modular language) with relatively good performance (since it is compiled not interpreted). Moreover, add-on modules in C or Assembly language can be acquired (at generally modest cost and from a vast range of third-party products spawned by the language's popularity as a development tool) and incorporated to perform speed-critical tasks. It is, in fact, only thanks to such modules that the system now has fast and flexible text search routines and an efficient, space-saving file format.

As it now stands, the local system - known within the Unit as TMan (supposedly as an acronym for "Terminology Management System" but really for no better reason than the author's delusions of grandeur) - can be summarized as follows:

Prime function: To store and manage multilingual (in practice usually 9-language but theoretically unlimited) terminology lists in a networked PC environment whilst allowing easy transfer to the central databank. As already mentioned, Eurodicautom itself is not really suited to the preliminary stages of work on terminology collections, added to which some short-term terminology support projects (tied directly to a very specific text or group of texts) may only incidentally give rise to Eurodicautom input.

Operating environment: Currently, the Terminology Unit in Brussels operates with a mixture of UNIX terminals (with no local processing capability) and 8 PCs (486 33 MHz machines with 210 MB hard disks) linked up to a PCNFS network (not one of the most common systems but not particularly unusual in the features offered at local level) providing printer and archiving services, access to central Commission (and external) databases via UNIX servers and a shared DOS-format hard disk on one server to hold data for the local terminology management package. The prime working environment is Windows 3.1 (with MS DOS 5), using a standardized (and simplified) menu-driven interface. As mentioned earlier, the main applications software is Wordperfect for Windows (version 5.2). TMan is not a native Windows application, but rather a Windows-aware DOS application which can be run from the Translation Service's standard menuing system, if necessary remaining open alongside Wordperfect or other applications. In this context, it is important to note that TMan is equally happy running full-screen or in a reduced-size window. Display and entry of all Community languages is achieved by means of special DOS keyboard and display drivers (not in-house produced) requiring at least an EGA screen adapter. As the display drivers will not work in a "windowed" DOS session (though the keyboard driver does!), correct display of Greek in windows has been achieved by (in-house) adaptation of the fonts used by Windows 3.1 for windowed DOS display (a relatively simple task using readily available software tools).

<u>Full multi-user and network performance features:</u> One of the main impulses behind the development of TMan was the need for a system which would radically simplify the process of managing collections to which a number of people (terminologists/secretaries for different languages) need potentially simultaneous write access. Previously, the best that had been achieved was wordprocessor files archived in a UMX network location accessible to all in the Unit, but this did not permit **parallel** updating by several users, and the wordprocessing option had a number of other serious weaknesses (notably the difficulty of restructuring file output and the inherent fragility of arrangements imitating a database using wordprocessor "forms" (where the deletion of a single character can all too easily ruin record structure - and sometimes even a whole file). Even the standalone version of TMan was preferable to this in terms of file structure, but much time was wasted in exchanging copies of files and version tracking (to ensure that the most recent versions of all languages were to be found in the same copy of the file!).

The multi-user features of TMan are consequently fairly highly developed even though the PC network has only been operational for a relatively limited period. In fact, the system was more or less network-ready before the network was more than a gleam in the Computer Department's eyes, thanks at least in part to the fact that Clipper is specifically designed to work safely in such a context. In particular, both file-wide and single-record locks are catered for to arbitrate potentially simultaneous updating by multiple users (with most changes being committed to disk and becoming visible to other users immediately following modification of a **single field**). Optional password protection is available at both file and individual record level, though with provision for "superuser" and system administrator overrides. Also contributing to data integrity are backup and archiving procedures and, arguably, a recent switch away from the traditional "DBF" format as the primary storage vehicle (though it remains available as a secondary vehicle). The main reason for this switch was to make better use of disk space, since the new format is an efficient variable-length fields format, whereas the DBF format is a fixed-field length disk gobbler. Two secondary networking benefits come with this switch, however. Firstly, the resulting smaller files allow faster access since less data has to be moved over the network. Secondly, data integrity safeguards are no longer vulnerable to the intervention of an unauthorized user accessing the data with one of the many tools for viewing "DBF" files to be found on the market (some of which can damage a Clipper file by writing to it just once even without changing data, since Clipper allows fields longer than is permissible in a "standard" DBF).

Quick start: One of the problems of some of the simpler-to-use systems is that they impose a predefined file structure with little scope for user modification (and very possibly incorporating fields which the user may regard as unnecessary but will have to complete every time he makes an entry (or at the least leave in the file, wasting space and causing general irritation). At the other extreme, some of the more flexible systems show their "enthusiast" background by forcing the new user to go through a complex process of file design before he can get started. On the one hand, the Terminology Unit requires files varying significantly in structure for different projects, whilst on the other most of its staff are **not** computer enthusiasts simply dying for the opportunity to engage in a bit of file design (and a limited range of file structures will in any case suffice for most purposes). TMan therefore offers three basic file creation options: a Eurodicautom-compatible file structure in which the user has only to specify the languages required; creation of a new file modeled on an existing file but with the opportunity to change characteristics; full custom file creation (for the brave). For the truly lazy user, a default Eurodicautom-structure file with fields for all official languages is provided.

Rapid, guided data entry: Users expect fast data entry with all the labour-saving devices to which they are accustomed in wordprocessing systems. In the case of TMan, this means: cut and paste - including direct cut and paste with WordPerfect (via internal procedures, not the Windows clipboard, which does not work well with DOS applications even at its best); keystroke macros, various forms of global replacement (field by field, file-wide, with or without confirmation, with or without "wildcard" characters, etc.); default entries for certain fields; record date and time stamps; deletion and reinstatement of records with a single keypress; both full-screen ("Page") and table (spreadsheet-like) viewing modes; optional checking to prevent duplication of entries (either as they are being made or as a batch operation). On-screen help is also available, not only in the shape of context-sensitive help accessed by pressing the F1 key (as in most PC applications), but also in the shape of reminders remaining on screen during normal operation. The main data entry screen includes a summary of key functions, for instance. This latter type of assistance is specially appreciated, perhaps because the majority of TMan users rarely think to press the Help key, having switched from Unix applications in which on-line help was very limited.

Fast record location: Also important in our system are fast record location facilities. It is relatively simple to implement "absolute movement commands" (such as movement to the first or last entry, or to a specific record number, or *x* entries forward or back - assuming a system in which records are basically ordered sequentially by date and time of entry). These, however, are primarily of use in data entry and modification - they do not answer the problems of searching for terms.

Even in a package originally intended for the management of terminology collections prior to uploading to a larger-scale databank, there is a substantial need for search tools, accentuated by the almost inevitable development of secondary uses for the system. A conventional database package answers these requirements (as indicated earlier) by searches which seek entries starting with the search term in a pre-established alphabetical index. This operation can be very fast. Unfortunately, it is seriously flawed for a text-storage system such as a terminology application. All too often, the search term will be found not at the beginning of an entry, but somewhere within an entry. Conventional database do have mechanisms for this type of search in the shape of "filter" or "locate" functions. In general, however, these mechanisms have to scan every entry in the base using unwieldy sequential access routines. As a result, they work too slowly to satisfy users (especially in a network environment, where they often entail "fetching" the whole of the database to the local machine). Additionally, their use is frequently less than intuitive (at least to those who think Boolean operators might be a kind of telephonist). Some systems (including the Commission's Unix-based local system mentioned earlier) endeavour to answer the problem by maintaining a secondary "keywords" file listing each significant word in every entry in the primary file, together with pointers to the entry in the primary file, this secondary file being indexed in alphabetical order. Searches then operate on the secondary file, which returns references to the primary file. Unfortunately, there are several disadvantages to this approach. Firstly, some systems using the approach will only allow single-word searches, which is obviously a severe limitation for a terminology application. Secondly, it is very cumbersome to manage, either imposing a significant overhead in processing time if the secondary file is updated simultaneously with the primary file or compromising functionality if the secondary file is updated off-line (e.g. overnight - implying that searches will only operate correctly the day after data entry). Application maintenance also becomes more complex because of the link with secondary files. Finally, the keyword files are significant additional consumers of disk space.

To solve these problems, TMan now uses an implementation of a bought-in fast text search module, which allows terms to be found on the basis of each element in the question (separated from other elements simply by a space) occurring at some point in an entry. This logic is intuitively satisfying even to the most unsophisticated users. In the event of a search failing, a partial matches option is also available, which will find any entries containing one or more of the query elements.

Without going into technical details, this result is achieved by maintaining a very compact index of text signatures which acts as a filter, returning a list of the entries whose signatures indicate that they probably include the search term. A verification routine then eliminates any entries which do not actually match. Because this mechanism keeps to a minimum the volume of data fetched from the network, it works nearly as fast in a networked environment as on a standalone basis. It also imposes very little processing overhead for maintenance of the signatures file simultaneously with the primary data file and consumes little disk space. For absolute maximum speed, one option simply takes the user to the first entry matching the search criteria (without bothering to check how many other entries match), and will then move on to the next matching entry if the first proves unsatisfactory. In practice, however, most users appear to prefer the slightly slower "hitlist" search, which verifies all entries and presents a view of the database restricted to those entries matching the search criteria.

A further important feature in a system which allows a multiplicity of separate term files - in principle, one per terminology project - is the possibility of searching either a angle file, several specified files or, indeed, all available files. The last option is particularly important for "occasional" users, who may well have little or no idea where to search for the information they require. It also largely answers one of the main criticisms of maintaining multiple files in a local system, an approach which in our experience suits local systems much better than the single, monolithic file (which is, however, in its turn more appropriate to the central mainframe system).

Provision also exists for batch query sessions, allowing the user to pose a series of questions one after the other without waiting for the answers, which he will receive later as a printout or a hitlist file. In practice, however, this facility is of more importance in a large-scale system, or one where on-line response times are very poor. Of more interest for a local, project-oriented system are the text analysis routines outlined below.

Import and export of files in various formats: Given that TMan's primary function is to manage terminology collections prior to uploading to Eurodicautom, it is obvious that it must be able to produce files which can be uploaded without further manipulation. In addition, it is desirable that the system should be able to read and write files in other common formats, since the Commission's Terminology Unit endeavours to exchange data files with other organizations working in related fields. In this context, the system is able to access directly files in the formats used by well-known general-purpose packages such as DBaseIII and IV, FoxPro and Paradox, as well as importing and exporting files in two common data interchange formats (SDF and comma-delimited). More advanced facilities exist for interaction with Wordperfect files, but these will be discussed below.

File sorting facilities: In addition to the viewing of files in chronological order of entry (which is often the most useful order for data entry and updating), there are cases where it is desirable to be able to view files in alphabetical order according to any of the languages in the file. This option is also available in TMan.

Desktop publishing: TMan is required to produce paper output in a variety of forms, from apparently straightforward bilingual lists through translations of list-type documents to major nine-language glossaries with full indexes in all languages. To do this, it uses two quite separate sets of functions. For the production of the body of glossaries and the translation of list-type documents, routines not dissimilar to the "report-writing" functions of general-purpose databases are used to generate fully formatted output (including a range of text attributes - bold, italics, etc. - and block protection to avoid an entry being split by a page or column break) in either raw order of input or numerically or alphabetically sorted order according to any field. The main difference from conventional reporting routines lies in the fact that the output is not normally sent directly to printers but rather to files in Wordperfect format. This has the advantage of permitting fine adjustments to be made within Wordperfect, which is designed precisely for the purpose of delicate text manipulation, rather than attempting at great expense (and probably only with limited success!) to provide the same functionality within TMan. It also avoids the listing-like "look" still evident in the output of even quite sophisticated report-writers, notably through such relatively rare features as the ability to switch text attributes on and off **within** fields (e.g. just one word in bold or italics).

The second set of functions is unlike those provided by report writers, which normally expect each record to be output once only (though multiple output is conceivable, particularly in the case of relational designs maintaining a secondary, keywords file). The purpose of these functions is to generate fully crossreferenced output (i.e. lists containing an entry in the appropriate alphabetical location for each significant word in each primary file entry). Several variants are available to suit different purposes and/or user preferences. User-defined lists of "empty words" for each language reduce the amount of "noise" (pointless entries) produced and can also (at the user's choice) be ignored in alphabetical sorting (which results in a dictionary-like order generally preferred by customers). For many purposes, particularly where speed is of the essence, these lists (once again produced in the form of WordPerfect files) can be used without further processing, subject to the inclusion of a "health warning" along the lines of "Index generated by computer" at the bottom of each page. In the case of documents which are to be more widely distributed and therefore require more careful finishing, the raw index can be reworked in Wordperfect (which mostly involves the deletion of superfluous items plus the restructuring of some entries) and then re-sorted in TMan (whose sorting routines are more sophisticated and flexible than those of the wordprocessing package) before final printing.

<u>Text analysis routines:</u> This covers two groups of routines, the first endeavouring to at least partially automate the process of exploiting source texts for useful terminology or phraseology and the second using terminology collections as an aid in the preprocessing of texts before delivery to the translator. In both these areas, TMan clearly overlaps the functionality of workbench packages, though in ways affected by the specific features of the Commission's translation workload.

The most ambitious of the text exploitation routines is specifically aimed at identifying repetitive elements in groups of texts (available in parallel language versions as Wordperfect files) and adding these to a TMan file for later use in preprocessing of texts in the same, or a similar class. It should be noted that the repetitive elements may be of any length from several words upwards, thus distinguishing the approach from that adopted by certain sentence-based commercial packages, which tend to be more useful in the context of technical manuals and the like. In Commission texts, unfortunately, there are numerous classes of document with standard bodies of phraseology running into the hundreds or thousands of phrases, but the standard elements are commonly less than full sentences, thus making packages which use unprocessed bodies of parallel texts as their source ineffective because they are rarely able to operate with units of less than sentence length (except for discounting simple variable elements such as figures and dates). It should also be stressed that, at present anyway, these routines are not used for the constitution of vocabularies for full machine translation systems, in part because such systems have hitherto tended to impose the manual entry of such vocabularies, with additional lexical information required for their grammar rules.

Less ambitious but useful in a wider range of contexts is the routine which enables dements of a Wordperfect file to be added to terminology files simply by marking them within Wordperfect. Marking may take the form either of a specified text attribute or attributes (e.g. underlining - so as to extract all section headings from a document) or of

Wordperfect's list-marking feature. This latter option enables a terminologist to reduce the initial stage of exploiting a document to simply highlighting on paper the phrases to be extracted, which a secretary will then list-mark in Wordperfect (in one or two languages) and automatically import into a term file, thereby minimizing both the effort of retyping and the risk of typing errors (which is increased in our environment by the fact that secretarial staffing constraints in the Terminology Unit imply some languages being entered by non-native speakers). Those terminologists who prefer working on screen can even reduce the marking process to a single stage.

The text preprocessing routines relate particularly to the types of text mentioned above - those which are neither sufficiently standardized to be coped with by the use of models in which only a few words need to be changed nor susceptible to reduction to an entirely standardized vocabulary and restricted grammar (as in the case of the AVIMA project) but do contain a substantial body of standard elements. It might at first sight seem that these texts would be prime candidates for full machine translation. Whilst this might be true in a few cases, it is not in many others, precisely because the purpose of full machine translation is to attempt to translate every word in the source document. The weakness in this for an organization such as the Commission is best illustrated by a concrete example - the monthly Bulletin of Community activities (a very real example since the Bulletin has been the subject of preprocessing for some time now - using Unix software tools but with data at least in part generated by TMan). In addition to its standard phrases (some three thousand of which have been identified for English alone), this publication largely consists of summaries of (or extracts from) material already published elsewhere in official (and frequently legally binding) versions in all Community languages. Even supposing that the machine translation system were to produce valid renderings for these parts of the text, they would be unlikely to correspond to the official versions published elsewhere (and unlikely at any time in the near future to themselves be the subject of full machine translation!). It would, for instance, be impolitic at the least for the Commission to translate the summary of a European Parliament resolution in ways at variance with the Parliament's own translation of the full resolution. This in turn means that translators would be obliged to spend much of their time undoing the work of the machine translation system - and inexperienced translators in particular might well be led into a false sense of security, accepting more of the machine version than was advisable. All of this leaves aside the fact that machine-translation systems have not so far proved especially good at coping with the type of non-controlled language prevalent throughout much of the Bulletin.

Preprocessing using standard bodies of terminology from a TMan file avoids both of these dangers, allowing the machine to do what it does best - recognize standardized elements and translate them with total consistency - whilst leaving the rest to the still much more flexible tool of the human brain. In addition to the possibility mentioned above of using TMan-generated material in an external preprocessing package, TMan itself possesses three internal preprocessing options, each designed to process either Wordperfect files (maintaining formatting and attribute information) or straight text files. The first of these options produces, on file or paper, a text-related glossary listing all the terms from one or more TMan files found in the target text. The second produces essentially the same information, but this time intercalated at appropriate points in a copy of the target file. The final (and most ambitious) option produces a copy of the file with all standard phrases replaced by the appropriate phrases in the target language (thus yielding a text in, for example, mixed English and French). In particularly productive cases, whole paragraphs or pages of this preprocessed output file may be used in the final translation with little or no change, thereby saving both translation and typing time.

There are, of course, weaknesses with such preprocessing options. In the first place, the full replacement routine is currently limited to using a single TMan file because it (unlike the other options) requires replacement of the longest matching phrases first (a process which would be substantially complicated by the use of multiple files). A second weakness lies in the limited extent to which morphological reduction (e.g. correct recognition of both singular and plural noun forms) and sentence reordering are implemented. Some rules are applied to improve performance in both these areas. Nevertheless, it is arguable that more sophisticated rules might bring significant benefits in, for example, the case of the German/English language pair, where sentence order differs systematically and one of the languages is highly inflected.

A caveat should be entered here, however. Because of the challenge inherent in devising rules to cope with these situations and the satisfaction to be achieved by overcoming the difficulties, there is a danger that system developers will lose sight of the harsh realities of cost-benefit calculations. Particularly where the purpose of a routine is to replace standard phrase elements, the number of variants actually encountered in practice may be quite small (often no more than two or three). In this case, providing an efficient file format is used and copy routines are available to minimize manual re-entering of data, it may actually be more rational (though less elegant and intellectually satisfying) to maintain separate term file entries for the common variants rather than expending substantial effort on the development of sophisticated grammatical rules. A few simple, but high-productivity rules (which may not even be "correct" in the strict sense) are probably all that is worthwhile in the early stages. Beware especially of implementing any rules which will require users entering terms to specify parts of speech, gender or other grammar-related information. Hard experience suggests that, faced with such requirements, most users will simply abandon (or radically cut back) their data entry efforts. Even terminologists will complain vociferously.

THE BOTTOM LINE

So much for TMan as it now stands. Some of the prospects for future development can be inferred from what has already been said. For the remainder, matters are very much in the lap of the Gods. Precisely because of the options chosen, however, we can view a situation of uncertainty with some equanimity. Even if the Unit were obliged to abandon the system at relatively short notice (an unlikely prospect), all data could be recuperated without difficulty, no expensive hardware or software would have been wasted, and the savings already achieved would be sufficient justification for the (limited) resources expended. But what of the lessons for others?

In-house development not for everyone. By reason of the particular combination of requirements facing the Commission's Terminology Unit and the confluence of a number of fortunate circumstances (most notably the availability of on-the-spot programming expertise in a computer language permitting rapid application development combined with a reasonable level of performance), in-house development has proved a viable option for the production of a highly customized system. It is also notable that a number of other international organizations (the FAO for example) have followed a similar path, even to the extent of using the same application development language. Specialized dictionary publishers (who may well have index-production needs similar to those met by the Commission's application) could probably also benefit from this approach - unless and until a commercial package appears which really brings an acceptable level of functionality to the lexicography facet of terminology work. For the average organization, however, it may be wiser to regard our experience more as an indication of what might be demanded of a commercial package (though almost inevitably with some compromises) or of a commissioned systems developer. If the in-house or commissioned development options **are** chosen, several key points must be borne in mind.

As far as in-house work is concerned, choice of computer language and development tools are critical. Management would be unwise to forget the computer enthusiast's tendency (familiar to the author as precisely such an enthusiast himself!) to underestimate the difficulty of a project whilst simultaneously overestimating both his own programming knowledge and the power of his favourite language and development tools. As a rule of thumb, suggestions that high-performance multiuser applications can be built with any version of Basic (even the latest Windows incarnations) should be treated with the greatest caution. Performance problems also exist, as mentioned earlier, with most "interpreted" database programming languages, which may in addition straightforwardly lack the means for implementing certain functions at all (and not be open to the addition of such functions through "bolt-on" modules). At the opposite extreme, application development using the most popular "professional" languages - C or C++ - is both complex and highly time-consuming even for the most experienced programmer, not only making initial development costly (often unpredictably so) but also rendering ongoing development cumbersome at best. It is not unknown for the experienced Clipper programmer to be able to add in half an hour a function which will take a C programmer days (albeit the result in C will normally run faster - though even this may be imperceptible given the possibility of bolting C modules onto Clipper applications for speed-critical system components).

Not dissimilar considerations apply when commissioning outside developers. In this context, project specification and performance guarantees may be critical if the customer is not to end up with a system which will fly supersonic fighter missions with all the speed and grace of a Graf Zeppelin. Computer professionals too can be guilty of underestimating the problems involved in language industry applications (especially on their first venture into the field and with the prospect of a juicy contract dangling in front of them).

Whichever path is chosen (and whatever their shortcomings one of the specialized commercial packages is probably at present the best option for most individuals and small translation, terminology or lexicography operations), there is little doubt that language industry professionals should be expecting great things of the microcomputer over the next few years. In particular, it seems to the author that if an in-house package developed by a single part-time programmer can, for instance, be made to operate directly on files in the most complex of wordprocessing file formats, the customer should begin to demand this of commercial products. Also expect more in the way of text preprocessors, text analysis tools and perhaps dictionary publishing software (though this last market niche may be too small for rapid progress). Increased user pressure for developments in these directions would at all events help to promote the much needed shift in emphasis from the computer as a replacement for the language professional to the computer as an *aid* to language work.