Zeitschrift: IABSE congress report = Rapport du congrès AIPC = IVBH Kongressbericht

Band: 11 (1980)

Rubrik: VII. Computer analysis and structural engineering

Nutzungsbedingungen

Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften auf E-Periodica. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Das Veröffentlichen von Bildern in Print- und Online-Publikationen sowie auf Social Media-Kanälen oder Webseiten ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Mehr erfahren

Conditions d'utilisation

L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. La reproduction d'images dans des publications imprimées ou en ligne ainsi que sur des canaux de médias sociaux ou des sites web n'est autorisée qu'avec l'accord préalable des détenteurs des droits. En savoir plus

Terms of use

The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. Publishing images in print and online publications, as well as on social media channels or websites, is only permitted with the prior consent of the rights holders. Find out more

Download PDF: 16.01.2026

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

VII

Computer Analysis and Structural Engineering

Calcul électronique et constructions de génie civil

Elektronische Berechnung im konstruktiven Ingenieurbau

Leere Seite
Blank page
Page vide

VII

Introduction to the Theme

Introduction au thème

Einführung zum Thema

M. FANELLI
Prof. Dr.
ENEL
Milan, Italy

Two half day sessions have been scheduled for the theme "Computer Design and Structural Analysis: Synthesis or Antithesis?"

In addition to the Introductory Reports VIIa and VIIb, it seems appropriate to describe the themes for both sessions:

A. Hardware and software systems evaluation and qualification problems

Choice and organization of hardware and software in relation to structural design objectives and requirements; "quality control" of programs; standardization of input, output; software classification, indexing, information retrieval and maintenance. Danger of diffusion of uncontrolled-quality programs, especially with dissemination of minis.

B. Compatibility of computer applications (in structural analysis) with design practice

- Adaptation of educative programmes (teaching how to use correctly c.s.a., both in engineering schools and in recycling courses for professionals)
- Adaptation of communication structures (dissemination of information about hardware and software possibilities and limits, communication with and between computers (computer networks); telephone mass service to users; communication with people: graphics, intermediate and final written documents, etc.
- Adaptation of legislative structure, e.g. interaction with building codes; legal liabilities for errors or misuse of programs; proprietory rights etc.
- Adaptation of economic structure: costs, cost/benefit analysis, rates and tariff structure for users etc.
- Adaptation (integration) with earlier and subsequent phases of "design": e.g. with planning and fabrication.

Intending authors are invited to submit papers along this line.

Leere Seite
Blank page
Page vide

VIIa

Computers in Structural Design: Some General Thoughts

Emploi de l'ordinateur dans le dimensionnement des structures: quelques considérations générales

EDV-Anlagen in der Bemessung von Bauwerken: einige allgemeine Bemerkungen

DONALD ALCOCK
Alcock Shearing and Partners
Redhill, Surrey, England

SUMMARY
This paper deals with general problems facing structural designers who would use computers; the opinions expressed are subjective ones. The structural designer should be able to program simple calculations himself. To make the task agreeable he should be able to type; to make it rewarding he should reject facilities that are not Standard. When considering programs developed by others, the designer should refuse to use systems that would usurp his engineering judgement unless he is perfectly clear by what criteria the decisions are made. The possibility of publishing descriptions of such programs in an intelligible form is discussed.

RESUME
Cette communication traite des problèmes généraux qu'affrontent les ingénieurs projeteurs qui voudraient se servir des ordinateurs; les opinions exprimés sont subjectives! L'ingénieur projeteur devrait être capable de programmer lui-même des calculs simples. Il devrait aussi savoir écrire à la machine; il devrait enfin renoncer à tout équipement qui ne soit pas standard. En considérant les programmes préparés par d'autres, le projeteur devrait refuser de se servir de tout programme qui lui enlèverait son indépendance de jugement comme ingénieur – à moins qu'il ne connaisse exactement les critères selon lesquels les décisions importantes ont été prises. La possibilité est également envisagée de publier des descriptions compréhensibles de tels programmes.

ZUSAMMENFASSUNG
Dieser Artikel behandelt allgemeine Probleme, welche bei der Benutzung von EDV-Anlagen durch den entwerfenden Ingenieur entstehen. Die vertretenen Ansichten sind subjektiv. Der entwerfende Ingenieur sollte fähig sein, einfachere Berechnungen selber zu programmieren. Um sich diese Aufgabe zu erleichtern, ist es von Vorteil, wenn er die Maschinenschrift beherrscht. Damit die gestellten Aufgaben wirtschaftlich gelöst werden können, sollten nur standardisierte Anlagen benutzt werden. Um eine fachmännische Beurteilung zu gewährleisten, sollten vom entwerfenden Ingenieur nicht selber entwickelte Programme nur verwendet werden, wenn er gründliche Kenntnisse über die dabei getroffenen Annahmen hat. Die Möglichkeit, Beschreibungen solcher Programme in einer verständlichen Form zu veröffentlichen, wird erörtert.

1. PROBLEMS FACING DESIGNERS

It is difficult to classify computer applications in structural design because of their interdependence. For example one category might be "Small ad-hoc programs" and another "Large general purpose systems", but in such a classification a typical BASIC program could belong in either category, being an entity in itself as well as a set of data for a large BASIC system. Nevertheless a crude classification of some problems of structural design amenable to solution by computer is attempted below to serve as a framework for the paper.

- simple calculations such as can be programmed ad-hoc by a structural designer who is not an expert programmer

- analytical calculations of a kind so frequently encountered by structural designers that it is worth employing programs designed for general use and developed by professional programmers

- complicated analytical problems needing numerical formulation by applied mathematicians and which may or may not be amenable to solution by existing systems such as those for finite-element analysis

- problems of sizing and synthesis often referred to by structural designers as "design"

- specialized tools for automated drafting, word-processing, estimating, scheduling, information retrieval, and other tasks for which proprietory systems are available, some of which include both hardware and software as a package.

This classification excludes the many applications of computers common to all professions such as job costing, payroll, invoicing and so on.

2. SIMPLE CALCULATIONS

Not long ago every structural designer was expected to be proficient in using a slide-rule and electro-mechanical calculator. Nowadays an electronic calculator is cheaper than a good slide-rule and more powerful than a mechanical calculator. Furthermore the cost of the electro-mechanical calculator was higher (in real terms) than that of today's "personal computer" which is able to compile and execute programs written in a popular programming language. In other words the slide-rule is dead and there is no economic reason to prevent a structural designer using a computer.

This discussion refers to "personal computers" but the arguments apply as well to the use of a terminal connected to a large computer.

There are many and varied personal computers on the market but most have two things in common: the ability to process programs written in BASIC (or a language similar to it) and a typewriter keyboard by which to convey such programs to the machine. There is universal enthusiasm for teaching children and older students how to write programs in BASIC but apparent reluctance (at least in the author's country) to face the problem of communicating with the machine physically.

Typewriter keyboards have had the traditional QWERTY layout since about 1890. For nearly a century millions of people have found such a keyboard efficiently suited to transcribing thoughts to paper using ten figures acting under the sense of touch. Nowadays the same keyboard is used to convey programs to

computers. This would suggest that anyone who wanted to communicate with a computer would first learn touch typing, for although journalists may pick up amazing speed and accuracy using only three fingers, no secretarial college would impose such acrobatics on trainee typists. Yet the obvious advantage of touch typing seems to be ignored among computer users. Worse! Computer codes are devised which permit abbreviations of words *so as to save on typing!* No trained typist would find LIS DAT, PRG FIL easier or quicker to type than LIST DATA, PURGE FILE, nor would anyone trying to read and understand what had been typed enjoy the cryptic version.

The universal language of the personal computer is BASIC, originally devised as a simple code for traching the elements of programming to beginners. It proved so popular that it has been imitated and adapted in various ways by different computer manufacturers and others. Now the response to the statement "My program is written in BASIC" is likely to be "*What* BASIC?". Extensions (some of them wild) are being made at a rate outstripping the formulation of Standards.

It is useless to bemoan the inadequacies of BASIC: the impossibility of structured forms, dependence on a fixed set of global variables, absence of integers, *etc.* because BASIC is in popular demand and there is no way (in some countries at least) to impose a "better" language on the populace.

Faced with programming in BASIC the structural designer would do well to apply self discipline. He should read the relevant Standard (ECMA, ANSI, or whatever) and go through the BASIC Manual for his own computer, ruthlessly crossing out non-standard facilities however convenient or exciting they may seem. His reward comes when he finds he can sell his programs to someone who uses a different make of computer, or when he has to use a different computer himself.

At the time of writing, the approach advocated above is not fully realizable because no Standard has yet covered such facilities in BASIC as the MAT statements, yet these are implemented with reasonable consistency in many versions of the language. The author has tried elsewhere to summarize such "de-facto" standards so as to encourage "portable" programming [1].

3. ANALYTICAL CALCULATIONS

A structural designer need no longer *be* a computer to determine the elastic distribution of moments in a frame. To initiate such an analysis he has only to write or type data in the form described by a user's manual, then let a standard program do the rest. Because of the demand for skeletal and finite-element analysis, programs and suites of programs have been developed by teams of computer-minded engineers and professional programmers, and offered to designers on a leasing or royalty basis.

Some of these programs are sound and reasonably free of errors; others are notoriously unpredictable in behaviour especially when transferred to new computers or when there are changes to an operating system.

A structural designer who uses such a program is seldom permitted to study its internal documentation, let alone make changes or extensions. The supplier argues that a complicated piece of software has to be "maintained" by a team of specialists who must be the only people allowed to look inside. Yet the conditions (which a potential user has to accept) stipulate that the supplier of the program takes no responsibility for damages caused as a consequence of wrong results.

This apparent dilemma is not as bad as it might seem as long as a program is *analytical*, in which case its results are essentially right or wrong and may be compared with those produced by a competitor's program. Confidence is thus gained or lost as the case may be. Nevertheless it would be better if structural designers were permitted to know more about the programs they use, and in the author's opinion it becomes vital when programs are used for synthesis. This is discussed later.

4. COMPLICATED ANALYTICAL PROBLEMS

Among academics the advent of the computer stimulated the formulation of engineering problems in matrix and other numerical terms; the necessary "number crunching" being no longer impractical. Many such problems are of interest to structural designers, but few structural designers have the mathematical expertise to formulate solutions themselves. Accordingly they consult experts at Universities and research establishments where large computer installations are often used.

If a problem is difficult to formulate mathematically it does not follow that it must be difficult to program. An abstruse problem resolved as a sequence of matrix operations might be programmed very easily. On the other hand a more simple formulation may demand the expertise of professional programmers before a practical program can be realized. A solution formulated mathematically becomes a *computing* problem when the required volume of input data or intermediate data is large, when iterative techniques are employed, when numerical accuracy is critical, and so on.

There is no reason to suppose that an academic who is an expert at formulating engineering problems in mathematical and numerical terms is *ipso facto* capable of writing foolproof computer programs. The sad experiences of computer bureaux when trying to adapt "University" programs to practical use testify to an unfortunate gulf between the disciplines of mathematical formulation and sound programming.

Computer science has not long been recognised in academic circles as a worthy discipline in itself, and many structural designers are not aware that it has much to offer their profession. In the present state of the art the structural designer should appreciate there is more to the numerical solution of structural problems than mathematics and self-taught fluency in Fortran. But there is nothing he can do to ensure that a program originating from a University or research establishment will be subjected to any form of quality control. Some programs are developed with the very highest degree of professional competence, but some are not.

5. SIZING AND SYNTHESIS

Analysis by computers is an essential part of structural design, but sizing and synthesis by computer is having more impact on the profession. It is now commonplace, for example, to let a computer program choose certain dimensions of reinforced concrete structures.

This practice poses the problem of responsibility for disaster. If such a program is developed and used by a structural designer then responsibility for a subsequent structural failure is obviously his. But if he accepts the results of a program developed outside his own organization he is allowing other people to design his structure for him. And it would seem that ever more structural designers are *willing* to take responsibility for other peoples' designs in this way.

Because these programs have to work according to Codes of Practice it might be supposed they are easy to test and evaluate. An evaluation of this kind was indeed attempted by the Design Office Consortium [2] revealing enormous differences of interpretation of a single Code of Practice among the programs tested. From users' manuals it was not possible to know in advance how each program would interpret the Code of Practice. How, then, is a structural designer to know what program to trust? At present he cannot.

There are two ways in which the situation could be improved:

- by trial and evaluation of proprietory programs carried out by some agency with power to award "Seals of Approval"

- by owners of proprietory programs disclosing in comprehensible form the internal descriptions of their programs.

Whereas the first approach can be made to work in specialized areas (for example by the Highway Engineering Computer Branch of the Division of Transport in the United Kingdom) the second approach may have more general potential and is considered in more detail below.

Program development is often undertaken piecemeal; an original version being augmented as demands for extra facilities arise and as new ideas occur to the developer. This style of development is *encouraged* by computer manufacturers and bureaux who provide software tools for interactive editing, selective tracing of execution, and so on. Although such facilities are difficult for a programmer to resist, the problem with the approach is that a program matures without documentation, and its "listing" might be unintelligible to a potential user even if he were allowed to see it. By adopting the more responsible approach advocated below, documentation automatically precedes testing and is, furthermore, intelligible to a potential user.

To be intelligible a program should be described in a notation less cryptic and less specialized than a computer language but more concise than a natural language. It is not obvious exactly where between these extremes the ideal notation should be pitched but an attempt has been made to define such a notation. It is called 3R and has been used to describe a program for which the internal description has been published [3]. From the published description "realizations" of the program have been written in Fortran, APL, and BASIC. The largest system so far described in 3R has a Fortran realization of some 20,000 statements [4].

Experience with 3R convinces the author that it is both possible and desirable to use such a notation for describing any program for technical application before encoding it in computer language. This approach avoids the problem of programs being developed without documentation - hence avoids the likelihood of their algorithms being intelligible only to their authors (and then only for the duration of the development period).

Even when thorough documentation does exist it may be argued that internal descriptions should not be disclosed outside the author's organization because the secrets are valuable capital to be protected. But against this it can be said that the potential user or buyer is more likely to adopt a program for which the internal description is made available than one for which it is not. Jealously guarding internal documentation may (it is hoped) become a short-sighted marketing policy for program developers. "Open" documentation leads to wider use of a program, so developers stand to recover investment from wider sales at cheaper prices - despite the occasional breach of Copyright that such openness might invite.

6. SPECIALIZED TOOLS

Dependence of structural designers on large installations operated by computer bureaux diminishes steadily as small machines become available. The price of a typical desk-top computer is about the same as that of a family car, and on such a machine a structural designer may run programs covering many (if not most) technical applications in his field.

There are, of course, practical limits to running design-office programs on a small computer; limits on size of program, volume of data, speed of processing. In some cases the designer may still need access to a big computer, perhaps using his small one as an intelligent terminal.

But because this modern electronic machinery is so small and cheap it may economically be dedicated to particular tasks of which a few were named earlier.

An example of a task of particular interest to structural designers is automated drafting. A typical drafting system has a dedicated computer controlling a digitizer for input and flat-bed or drum plotter for output. It may also have tape drives from which to read data generated by a general-purpose computer.

A system of this kind is sold as a package comprising both hardware and software. And, as was predicted a decade ago, the value of the software content is beginning to overtake the value of the hardware.

The structural designer may look forward to having ever more specialized tools at his disposal, each comprising a dedicated computer running dedicated programs. As long as the specialized tool is not designed to take over any of the tasks to which he should apply skilled judgement as a professional engineer he need not be concerned about the details of the software which controls the specialized tool. If the tool does not work properly it can be sent back to the supplier. But the structural designer would be ill advised to adopt the same attitude towards any specialized tool that might usurp his judgement as an engineer. Before using such a system he should know precisely what criteria and algorithms the machine uses to "design" structures.

7. DETAILING

A task that consumes a lot of time both of the structural designer and his draftsman is "detailing". In reinforced concrete work this includes both the choice of concrete dimensions and the selection of bar sizes and their placement; in steelwork it includes the choice of section sizes and details of connections. Detailing falls into the last two of the author's categories and is therefore discussed separately.

Detailing is heavily prescribed by Codes of Practice and might therefore seem to be a task that could be automated completely, making possible a specialized tool comprising items of hardware such as those used for automated drafting systems. The fact that such a tool does not yet exist seems to indicate that detailing is not as simple as it may at first appear. Certainly the evaluations undertaken by the Design Office Consortium [1] suggest this is so because every program chose a different pattern of reinforcement for the same beam. Codes of Practice - however restrictive they may seem - allow enormous variations in interpretation.

Perhaps our "built environment" will retain more interest if detailing is allowed to remain a minor art, but for those preferring to strive for efficiency

the problem could be resolved if future Codes of Practice were written to include *algorithms* as well as constraints. The algorithms could then be transcribed into computer languages, thus encouraging automated detailing.

The problem of detailing can be circumvented altogether by the use of "system building". No such system is complete today without a computer program with which to plan a building on a grid displayed on a screen. All schedules, drawings, documents necessary for construction emerge untouched by human hand. But it pleases the author to reflect that the slightest unusual requirement such as a skew gable or a step in the level of the ground makes many such a system collapse.

8. IN CONCLUSION

The author sees a widening gulf in the use of computers by structural designers; on one side the development of ever more complicated systems: on the other a phenomenal increase of "do it yourself" programming. Developments on both sides of the gulf have one thing in common; how the computer reaches its results is becoming increasingly obscure to all but those who wrote the programs - and often to them as well.

Structural designers can have little influence on developments in computer hardware or general-purpose software because structural design represents such a small market. But it should be possible for them to keep their house in order by adopting policies and attitudes some of which are discussed in this paper. At risk of being judged both trite and arrogant the author dares to summarize his subjective conclusions as follows:

- a structural designer who intends to use a computer frequently should develop some skill in communicating with the machine via its keyboard

- a designer who writes programs in BASIC should take care to use only the forms and facilities of that language that are Standard (and this applies similarly to Fortran programming)

- a designer who uses a proprietory system for analytical calculations should satisfy himself (by comparisons if necessary) that the chosen system is sound

- a designer should not assume a program is sound simply because it is based on a demonstrably sound methematical model

- a designer should refuse to take responsibility for "designs" generated by a computer program unless he fully understands the processes by which such designs are synthesized.

Finally the author believes it is both possible and desirable to make the working of computer programs comprehensible to potential users; the ideals expressed above are attainable.

REFERENCES

1. Donald Alcock: "Illustrating BASIC", Cambridge University Press, 1977.

2. Design Office Concortium: "Computer programs for continuous beams - CP110", DOC, Guildhall Place, Cambridge, 1978.

3. DOE PSA/LAMSAC/DOC: "FORPA Computer Program", Design Office Consortium, Guildhall Place, Cambridge, 1978.

4. S.I.A. Limited: "NUSTRESS", SIA Ltd., Ebury Gate, London, SW1. (To be published).

VIIb

Evolving Design Practice in the Computer Era

Evolution de l'art du projet à l'époque de l'ordinateur

Projektentwicklung im Zeitalter der EDV-Anlagen

JOHAN BLAAUWENDRAAD
Dr Eng. Sc.
Rijkswaterstaat – DIV
Rijswijk, Holland

SUMMARY
A survey is given of the design society as it evolves in the computer era. The designer's task will be adapted gradually. The dissemination of software is discussed, as well as the adaptation of the education curriculum. Broad spectrum integration of programs and economic matters are considered. Finally the question of responsibility is raised.

RESUME
Une vue d'ensemble de la société des projeteurs est présentée à l'époque de l'ordinateur. La tâche du projeteur évoluera constamment. La dissémination de programmes est discutée ainsi que l'adaptation dans la formation des utilisateurs futurs. Une intégration très large des programmes est considérée et les aspects économiques sont pris en compte. Finalement le problème de la responsabilité est soulevé.

ZUSAMMENFASSUNG
In diesem Artikel wird ein Überblick über die Entwurfspraxis, wie sie sich im Zeitalter der Rechenanlagen entwickelt, gegeben. Die Aufgaben des entwerfenden Ingenieurs sollen allmählich angepasst werden. Die Verbreitung der Programme, sowie die entsprechenden Anpassungen des Lehrplanes wird behandelt. Die Integration der Programme auf breiter Basis und die wirtschaftlichen Aspekte werden erörtert. Zum Schluss wird die Frage der Verantwortung aufgegriffen.

1. SURVEY AND PROBLEM AREAS

It is a commonplace to state today that the computer has become a powerfool tool in structural design and analysis. This tool was gradually developed in a twenty year period in which a number of computer generations have been used and in which several types of programmes could be proved. In the early days from the late 1950's till the mid 1960's, structural designers quickly attempted usage of the first type of computers and the corresponding software. This period may be characterized by small stand-alone computers operating in batch--mode only. In the proceeding time sharing and minicomputer period the application area has been widely broaded and a new impuls from the graphics-oriented point of view is seen after the mid 1970's.
Other recent developments are the network concepts and the revolution in hardware miniaturization techniques which caused a drastic fall of prices and led to a new market of micro-computers.

1.1 Penetration and impact

It is worth considering to which extent the development of computer technology and software engineering did penetrate into the structural engineer's environment and which impact and consequences may be shown. Going back for (say) ten years, the expectations had been put very high and the readiness to invest money in this area was reasonable. Today it is not to be denied however that sometimes higher criticism regards computer usage can be heard and in any case a more waiting attitude is adopted. Nevertheless it is certain that the computer has got its role in the structural design proces and it is expected that this role may expand, be it perhaps with another speed than assumed in earlier days.

Until now the computer is used rather in structural analysis then in structural design. It is not difficult to show that the majority of applications is in statics and dynamics, specially using general purpose finite element programs. Such advances in computerized analysis have been prompted initially by leading events like jet aircraft design, space flight programs and now the energy crisis, but the achievements are useful in civil engineering as well. Nevertheless we have a disappointingly small number of programs for design and dimensioning structural members. Granted, a huge effort has been made in what is said to be interactive computer-aided design (CAD), but until now the hardware needed was too expensive, and the software, if anyhow manufactured, too high-brow. Above that, the major developments have been tuned to car-industry and electric logic circuit design. Perhaps much will change with the introduction of the microcomputers.

Also an immense progress has been made on the theoretical side of automated optimal design, but its implimentation today is only a negligible aspect of computerized structural practice and it has fallen far short of expectations that many held out for it. Some people credit the practitioners for this, stating that they are still too unfamiliar with the underlying mathematical concepts. There are however good grounds to think that the real reason will be a different one. Optimal design is in fact automated analysis and the analysis is only a small part of the whole design activity. Above that not every method of analysis in structural mechanics can be applied with the same ease and it is not clear up to now which technique for linear programming will be the best one in the long run.

Everybody who had to design himself or who had the opportunity to watch designers, knows that the *principal* decisions are made on the basis of a series of considerations among which cost of maintenance, cost of insurance, cost of operation, cost of construction and irrational matters like esthetics. After that, the application of an optimal design program may just be working in the margin.

Apart from some exceptions most computer usage has been of the analysis type if necessary applied iteratively during the design process. The large volume of today calculations is not due to the availability of the computer only. Outer circumstances also make the use of computers obligatory, among which better knowledge of the material properties and perfectionistic design codes. Another reason which forces the use of computers is the introduction of more complicated construction methods. Even the most pertinent representative of the 'old school' will be glad to examine in advance all different construction phases which occur during the erection of (say) a major cable-stayed bridge, be it steel or concrete, to control the deformations and stresses.

1.2 Needs and consequences

It has been mentioned above that a large volume of calculations is made today. This does not mean however that all design offices are mixed up with these calculations and those who are, will not be so to the same extent. According to an estimate of a couple of years ago one out of three structural engineers (chiefs of offices included) is involved. Most of this one third will be found in the bigger offices or departments with ten or more potential users which can afford to have an in-house computer plant, either a main frame or a mini. However these offices and departments cover only 20 percent of all of them. The other 80 percent consists of small design offices which may use a computer incidentally at a service bureau, but will mainly do their job with top-desk calculators and hand-held computers. One can hear of new expectations held out for these smaller firms, considering the cheap microcomputers now arriving on the market, but also of severe concern that a wave of poor and 'grey market' programs are being set up without appropriate documentation.

The undertone of the presentation so far tends to the conclusion that much attention has been paid to sophisticated number crunching programs which are powerful analysis tools but are limited in scope for actual average design practice. The intellectual father of one of the leading structural analysis systems even heaved the sigh that 'we solved the wrong problem'. At several occasions a plea has already been made to manufacture simpler tools for simpler problems which belong to the everyday job of the design offices. The 20 percent bigger offices, mentioned above, did manufacture such software themselves to some extent, but the 80 percent of smaller offices could not.
A major problem is how to finance this. The practice of structural engineering is incredibly fragmented and each small design office cannot manufacture its own software. Above that most users have difficulties to adopt other peoples programs, due to their claim that their own problem solving process is far superior. This symptom has been said to be our NIH-syndrome - 'not invented here'.

To end with in this survey, the computer usage period made some social, legislative and educational consequences manifest. The availability of programs now allows people to analyse a structure without mastering the theoretical basis for such an analysis. It should be examined how to restructure curriculae to cover this unacceptable situation.
Furthermore we face specialisation of the engineering profession. Will a devision occur now between designers and program specialists comparable with what did occur earlier between architects and designers? How is the relation between designers and program specialists? Does computerization of analysis and design give rise to reduction of engineering manpower requirements? What about responsibility?

The next parts of this introductory paper will enter into some problem areas which have been touched directly or in an implicit way in the comments above and which are intrinsic matters of the design practice in the computer era. Attention will be given in succession to:

- relation between designer and program specialist
- communication structures
- educative curriculae
- integration of planning, design and construction
- economic structures
- legislative matters

It cannot even be tried of course to be complete for each subject or to produce answers to each possible question. At the best it is an initial attempt to locate areas which need adaptations and an indication in which directions we should move.

2. STRUCTURAL DESIGNER VERSUS PROGRAM EXPERT

The adoption of software as a tool in the design process and structural analysis is not an innocent decision without engagement. The structural engineer will notice that he is confronted with a new relation to a type of colleague he did not know before in the profession, the program expert. This colleague not only manufactures the program, but is also supposed to support the use of the software and to provide a proper documentation. The cooperation between designers and program experts give rise to comments of different nature. In this section we want to discuss the consequences for the profession of the designer, that is to say, how does it hit his task. In other sections we will touch other aspects concerning education and responsibility.

2.1 Cooperation between designers and program experts

Good communication between the designer and the program expert does not guarantee that a computation is correct, but the reverse certainly holds that poor communication runs the great risk of a bad result. The background and orientation of the designer and program expert are totally different and you may not expect to convert the designer to a highly qualified computer expert and the computer expert cannot be induced to become a designer. The designer is interested in structural engineering matters, economics and esthetics, whereas the program expert by nature orients to mathematics and software engineering. But even in regards of the mutual interest, the application program, there is a marked difference in the way users and programmers interpret is .

The users are interested in the availability and accessibility of useful programs and have special interest in an understandable way of input preparation and an output which is interpreted in an easy way. On the other hand the software men have more attention for standardization matters and have more other typical software engineering problems.

Watching the profession, one must say that several types of communication can be seen. Part of the designers show a pronounced readiness to cooperate with the program expert and succeed in maintaining a fruitful conversation. This part will be found in bigger offices handling more advanced programs and these designers may have had an academic training. However, a far bigger part, generally speaking performing less advanced design jobs, did not build such a communication bridge and even sometimes suspects the program expert of impertinent penetrating into his cherished profession. On the rebound the program experts fall short of respect as regards these designers accusing them to have too less knowledge of computers and computer science. This description may be too black and white and may be overruled by new facts when time goes on, today it still holds for too many of them. It will be clear that education and responsibility is highly connected with this subject.

As will be discussed later, the designer should autonomously make the final decisions. This does however not exclude the role of computing centres and their programmers to be ready to contribute to the interpretation and understanding of the interface steps and of the limits of applicability of the algorithms used. In bigger offices and departments with their own computer plant and staff such a support task can be realized in a rather easy way, but problems arise for the huge amount of small offices which depend on service bureaux. Practice reveals difficulties to guarantee continuity of such sustaining help for the broad spectrum of all offered programs. Here we touch a serious concern which is strongly connected with the dissemination of software. Incidental attempts to construct a national solution just stayed in the planning stage or lead a poorish existence.

In this context it may be worthwile to distinguish several user levels. There is a range from the humblest user to the most experienced and most computeroriented user: the programmer. Each level requires a different amount of help and help of different nature. Some papers in literature even make a plea for a new type of profession in between the designer and the software manifacturer. This new support functionary is supposed to be a better medium than documentation.

2.2 Evolution of design profession

We have also to enter into the question if a shift may occur in the task of a designer. Observations make clear that two opposite trends reveal. On the one hand we place new possibilities at the disposal of lower level analysts, who have not been trained at school conventionally to understand the behaviour of the complex stress states which can be calculated with the new capabilities. In such cases the designer/analyst has a lack of knowledge of his limitations, but he does use the software today, no user certification being introduced. It is therefore much more difficult to know the real limits of theoretical capacity of an engineer. All or almost all will analyse structural problems in the same way: with the help of a computer program independant of the fact whether without computer they would be able to solve the problem or not.

To quote a speaker at the 1978-colloquium on 'Interface between Computing and Design in Structural Engineering' who said it a bit exalted: 'with a computer program we have created a means which will allow also the incompetent to give the impression of being competent'. Here it is sufficient to diagnose that at least the educational curriculum of such analysts should be subject of re-examination.

The counter-trend is the danger of specialization between designers and program experts, already mentioned in the first surveying section. Both designers and program experts often are engineers by training, but they specialize on different matters. There are design engineers who propose the total concept of a structure and take final decisions on the basis of the stress results passed to them by others. They have control on the utilization of the results, but the mastering of the forces flowing in a structure will slowly escape, as well as the forming of the structure adequately for this force flow. On the other hand there are analyzing engineers who produce the stress results and force flow. They know how and what to calculate, but have no influence at all on the initial concept and only weak influence (if any) on the utilization of the results.

There are well founded doubts about the acceptibility of the two opposite trends described above. However, the question is not if we regret such a development but if we can stop it. It seems hardly possible to change this development drastically but we can canalize it. To quote again: 'It is the duty of schools, engineering societies and organizations to consider these facts in order to remove the negative effects of the computer revolution and in order not to let the engineering profession drop to a narrow-minded level'. Undeniably the design profession will be adapted, but no reasons exist to get up set. The changes will evolve gradually, be it only for reasons of inertia in society and education.

Until here we discussed the evolution in the designers task in regard of the computer programmers task. We could even go one step further and raise the relation of the designer and the computer itself. In the 1960's expectations were held out for electronic brains capable of solving any task reserved for the human brain. Now we judge that high-brow talk and know better the limitations of computers in terms of creativity and ideas. The recent advances in micro-processor technology may have a new impact but this will be rather in 'personal electronics' and text processing. In the design environment this new technology will provide a new set of CAD-tools, but an associative ability is not to be expected in the foreseeable future. For the moment computers are not capable of synthesis in the way designers are and therefore it is an abuse to request for thoughts or creative ideas.

3 COMMUNICATION STRUCTURES

Communication is a problem which did not arise for the first time when computers were introduced, but new needs for communication appear due to computer usage. When we bear in mind that existing communication in the world of structural engineering springs from long time experience, we may understand that new communication types in the computer era come into being by the method of trial and error. Which needs exist for communication?

First we think of the dissemination of information about hardware and software possibilities and limits. Second we may discuss the communication with and between computers. Third the possibilities for support and service to the users belong to it and finaly the communication via the documentation.

3.1 Dissemination of information

Many motives exist to justify effort as regards dissemination of information on existing software and possibilities of hardware. As has been said above, the design practice is enormously fragmented while the cost of software manufacturing increases continuously. Above that specially the big number of smallest offices do not use computers intensively. The NICE-survey in the United States of America indicates that much of the existing software is of questionable value in terms of universal transferability. Information concerning the documentation, program size applicable hardware, languages and availability is sparse. Further, the existing information, although highly variable, shows much of the software to be specialized as to application and/or not amenable to easy adaptation to a variety of computer systems.

Another aspect is the point of competition. Economic cooperation between consulting engineers, contractors and governmental institutions will be a new and extraneous feature, for them that traditionally adopt a rather conservative attitude. Especially consulting engineers tend to assume that software development constitutes in fact an engineering effort, and therefore, the development of programs is considered to be a feature of competition. This makes it more difficult to set up dissemination structures.

Despite some disencouraging motives there have been meager attempts to share program libraries, e.g. CEPA in the USA and CIAD in the Netherlands. Other poor means to share libraries are the so-called integrated systems like ICES, GENESYS and IST of which ICES has a lot of users, using however many different versions, and GENESYS adopted a strict centralized concept, resulting in few users. It proves to be very laborious to initiate centres to promote the use of computers and software in civil engineering. The group which should be interested hesitates to give his blessing to such plans and even government only avows by mouth that such initiatives are worth wile, not being prepared to make funding available.

Such centres could provide information on the software to the user. The service of such a centre could be extended to a clearing house, and/or broker for information on sources where software is available. Specially included are search, survey and collation of all sources of software information. Also the publication of a current and comprehensive catalogue of existing software which will contain detailed information that will allow the user to make his own evaluation and selection. Such a catalogue has existed in The Netherlands and does now exist locally in a number of countries, e.g. Australie (ACADS) in Sweden, Germany and Italy. It is mostly distributed by means of an engineering periodical or magazine which offers space for this purpose. The IABSE-taskgroup 'On the Use of Computers in Structural Engineering' proposed a model which could be used internationally. Another attempt is the EURONET activity.

The succes of such a dissemination structure stands or falls with the want the users feel for it, and this want seems not to be a very urgent one.

Sometimes it is believed that the only reason to receive and read such a catalogue is to know the state of the art of the colleagues and this is not enough encouragement to keep it in the air. Above that it is not easy to judge from the catalogue the usefulness of a program, taking in account the experience that rather long time is spent in finding your way with an unfamiliar program.

While dissemination within one country is not an easy job, the dissemination between different countries is even more difficult. The problem of code of practice dependency then arises but even more important that not each country has the same organizational degree in the engineering society to realize the wanted transfer of technology. Some countries may not have any institution whilest other countries enjoy the existence of foundations or associations to house such activities, e.g. ACADS in Australia, Design Office Consortium in England and CIAD in The Netherlands.

Recently it is heard that government is supposed to spread the software in future (German view). Observing the increasing government interference in general this might be a trend in the long run if the profession fails to settle its affairs itself. And the profession will most probably fail, being not interested in general standards and a global optimum but on the contrary heavily concentrated at its individual local optimum. It is a concern how far the government should go in establishing and maintaining public domain programs. There are no ready answers yet, but it is clear that such decisions will have strong influence on future progress in computerized structural analysis.

3.2 Structures of user support

When a program has been developed a stage starts in which the program must be implemented in design practice. This is a stage which requires a lot of support to the user. But also after the initial implementation such support is continuously required.

The most self-evident mean for support is the documentation of the program but it will not meet many objections to state that most documentation is poor. It is not recognized much in what way the documentation should be written. If the programmer does it, as is the normal situation, there is a danger to give information which is not relevant and otherwise to omit parts which should have been inserted. It has therefore been heard that infact only the users are able to write the documentation properly. In any case it is not easy to make good documentation in one shot. But who is willing to write (and fund!) a new one after say one year, feeding in all comments and experience got then?

If a program in the structural engineering field becomes more sophisticated and covers various features and complex functions, some kinds of back up user support becomes necessary. Too many programs are developed indeed but not effectively used in the organization because of poor usage promotion and support. For naturally, engineers are reluctant to use what they can not fully understand. This is specially a problem for a practical engineer if he has to study the usage of certain program which he uses once in a while. As a matter of fact, even for a small program, usage consulting support is undoubtedly needed for the effective use of such recources in the organization.

This problem not only plays a role in private firms but as much in public time sharing services. It can prove to be a limitation in the succes of a program if no proper support can be offered.

In a Japanese proposal a plea is made for a new type of engineer, the application consultant for structural analysis and design programs. For some complex programs the data preparation is too time consuming even if the program features and data input are reasonably understood. The man power needed for this is also heavily dependent of the engineer's skill of the program usage and his capability for the problem solving of the particular problem to be analyzed. According to the Japanese expectation will the availability of program usage consulting support soon become the key factor for deciding what particular program usage environment engineers choose to use.

Such a development may be unavoidable, but it is not without some concern. Both the design engineer and the consulting engineer are in danger of a degrading profession. The designer will find himself to have evolved after time to a new type of architect (section 2) and the consultant risks to be labelled as a arithmetician who is needed for a while but is not essential in the decision-making situations, with all consequences for his salary and respect in the engineering society. The social implications of such an evolution should not be underrated.

3.3 Communication between users and computers

The practical engineer expects that the computer system is an excellent assistant who is able to respond to our engineer's questions properly and quickly. At the same time the engineer wants to stick to his own profession. There are many possibilities. We have begun with stand-alone single run batch machines, followed by multiprocessing batch machines and timesharing environments. Today the mini's have entered into competition with the main frames and we are waiting, as has been said in section 1, for a new wave of changes due to the micro-computer technology.

The communication with the machines has been improved by introduction of free-format readers, problem oriented languages (POL's) and user friendly operating systems which, for instance, allow conversationally, directing the system to do processing in either foreground or background. In more advanced environments like POLO the user even can select at run time the mode of desired operation. Yet there is still a lot to be desired and to clarify this we quote a UK-author: 'It has been said that the invention of the self-starters for cars did more to liberate women than any militant movement. Today engineers are in a similar position to the ladies of yesteryear. We too have a mighty machine - the computer - but it requires a lot of specialised cranking to get it working'. Recent improvements have been found in the cheap storage tube and the micro-technology based work-station. The tube specially proves itself an interactive visualizing tool which is easy in use and which is appreciated by engineers, more advanced and exciting refresh displays being too expensive for our purpose.

The next stage in communication improvement is expected in replacing the filing system by the database. Up to now general purpose databases are used, but first attempts are being reported to manufacture databases which have been specially designed for engineering projects.

A question which also has been raised with respect to the man-machine communication is whether to use a centralized hardware arrangement or a decentralized arrangement. If much complex software is used which requires a powerful computer it was decided that a central arrangement is preferred. Another reason has been found in the fact that the same code is used by many offices and that manning each office for the management of a larger computer, even if it could be afforded economically, is not an easy task.
However increasingly often the advantages of decentralized processing using minis is underlined. The speed in performance is greater and real time assistence can be rendered by software specialists. Above that these specialists acquire better understanding of the requirements of a design office. Of course the maximum availability on the spot is a very important feature as well.

In future we foresee a mixture of the two extreme modes, mentioned above. The rapidly evolving technology in the field of small- and medium-capacity computers makes a decentralized computer network an achievable prospect. Such a structure allows to tune to the engineer on the spot (workstations), keeping a back up for number crunching runs at a main frame. For the moment is it believed, however, that the techniques for creation are still in a premature stage, hampered as yet by both technical and commercial difficulties of different nature. But the situation is supposed to change revolutionary.

4. EDUCATION

The gradual adaptation of the designer's task and the probable introduction of a new type of consulting engineer have consequences for the educative program of engineering schools and recycling course for professionals. It can not even be tried to cover in this introductory report all aspects which could be shown. We have to refer the interested reader to author's paper 'CAD and the Educational System' on the 1977 international conference on computer aided design education at Teesside, England. Here we will touch a couple of relevant items.

4.1 Education for who?

Most contributions in literature concentrate on the education of users and of course this is the main group of interest. However managers and software engineers need education too. It is the manager, the head of a design office, who has to approve the use of software. It may be appropriate to instruct him as regards cost consciousness. A lot of chiefs still are fascinated by the computer bills, not being aware that (reductions of) other cost places are equally important.

The software engineer manufactures the programs. Apart from skill which is typically connected with his own branch, he has to have knowledge of the user. Otherwise he is in danger of producing software which is not recognized as useful by the user. We see in practice continuously sinning against this extremely important facet. Another facet, closely connected with the first one, respects the drafting of specifications. At this territory the user meets the writer of a program in joined action. It is of particular interest to speak the same language for a good communication. The possible new type of usage consulting engineer intervenes here too.

Now coming to the actual user, the designer, we have to distinguish carefully, between the several levels of engineering training. They all have the common interest that education has to be user oriented and not program oriented, but for the rest a practical oriented engineer in a day round design position has another educational want then a somewhat acadimic engineer. The humblest user may ask for an education in which computer science is hidden behind some user-friendly screen using the simplest input language imaginable, while the specialist user (e.g. the usage consulting engineer) might need a computer science oriented education in its own and allow for a high--level input language.

4.2 Education. What and how

The use of programs is becoming a professional skill and we must decide how far the engineer has to master this skill. And anyhow we have to offer him the opportunity to grow gradually from the humblest level to the most advanced level. The way we teached the usage of computers in the past ten years is disputable. Did we not lecture too much how the tool was composed instead of how the tool should be used? A man who buys an electrical do-it-yourself set for sawing and drilling will not study the internal electrical and mechanical layout of the apparatus, but is only interested in its functions. This indicates the way of thinking for the training of engineers.

Unavoidably we hit the question to which extent we can afford to use software as a black box. The existing engineering curriculum often is already overburdened, making it necessary to drop well established parts before you can put new subjects on the list. This is one more reason to consider if we can accept a measure of canned knowledge. An examination in depth of the existing curriculae shows today that some parts are no longer relevant at all and that other parts can be lectured in a different way. Where we spent a lot of time for the mathematical procedure in the past we get time now to increase the understanding of the physical phenomenon. If we spent hours to solve a differential equation in earlier days, we might concentrate today on the fysics of our materials, that's to say what stresses and displacements will occur and what constitutive laws apply. You do not always need in-depth mathematics to show the essence of a structure. We can use with great succes well-established simplifying models. For instance, rigid bars connected by rotational springs are pleasant aids to explain buckling and vibrations in an easy way. Each teacher could develop in his area such educational tools, to understand the essence without tiring by superfluous mathematics.

In the design office a structure never consists of the idealized members and components which they have been the subject of at school training. The engineer first has to translate the structure to a scheme which is composed of such idealized members to allow for a calculation. This was already necessary in old days when applying calculations by hand. In the computer era a new stage is added however,making an input model of the schematized structure in order to do a computerized analysis. The computer output is converted to quantities which hold for the scheme of the structure and these quantities in their turn are interpreted for the actual structure under consideration. This is the responsibility of the design engineer or, if he passes it to him, the usage consulting engineer, and therefore we have to stress in their training the skilfulness to translate the physical reality to a scheme and to model this scheme in a proper input model, at the cost of knowledge of the program itself. It has no sense to know all details of the thin plate theory when the most important decision should be to use a thick plate element accounting for shear deformation!

4.3 Education. Where and by who

Above we did not distinguish between regular engineering schools and recycling courses for professionals. The manager education seems to be a matter of recycling courses only whereas the usage consulting engineer and software manufacturing engineers can be trained at engineering schools. The adaptation of the design engineer's training at schools should start readily now, while the designers already in profession may join recycling courses.

Of major importance is the question where to find the wanted teachers. Basic principles of computer science, not explaining in depth how the computers work but rather which functions it can execute, can be teached by mathematicians or people specially trained in computer science matters. However, for the integration of the computer usage in the design office task we prefer teachers with an engineering background. The education in this phase, though connected with computers, is particularly focussed on design. It it therefore highly important to recrute teachers who are willing to prove their power in design knowledge rather than in computer knowledge.

To end with, we just note the special ability of the teacher to react flexibly on the changes in this field. Generally speaking it is felt that we live in a dynamic era in which the training at the start of the career of an engineer will not be sufficient for whole life. If anywhere, the call for permanent education will be heard in the area of computer aided design education. This will impose a continuous heavy load on the teaching staff. One of the main problems of the future might appear to be how to teach the teachers themselves continuously.

5. INTEGRATION OF PLANNING, DESIGN AND CONSTRUCTION

It appears to be very difficult to create a chain or suite of programs which can be used from conception until realization. The bulk of software just tackles design problems. Yet the design and analysis is only a minor part in all tasks to be done. Granted, there are programs for planning and to make automatic drawings, but a real horizontal line of integrated programs does not exist in spite of all pretentious so-called integrated systems or integrating systems.

The reason for this failure can most probably not be found in the computer science matters. It is most likely a matter of bad understanding in what way communication was realized in old day. For also without computers we did and still do communicate. We first have to analyse the mechanism of information transfer between architects and planners, planners and designers, designers and contractors, and so on, for nobody seems to know exactly how they succeed in transmitting data to each other, or -if they fail- why they do so. It may be a hotchpotch when done by hand and the need for rationalization may be great, it does operate nevertheless.

One thing is clear. It will be a mass of data which must be handled and the members which operate on the data belong to multidisciplinary teams. There must be found a methodic approach to structure all deliberation in the several stages. The information which is now carried by drawings and tender and construction documents will consist of datasets on a computer in the future. The shared engineering database is a prerequisite for this purpose and the several programs will operate on this common project datapool.

There is a cost problem here as well. Broad system integration can only be justified if the benefits are evident. Such benefits are more easily shown at the contractor's side than at the architect's side. When scheduling of equipment, maintenance control of equipment and rental of equipment is improved by computers, the gains are remarkable and can be shown. The problems of architects however are less demanding from a computational viewpoint and more from the information processing viewpoint and benefits are therefore not easily seen. It is not believed that revolutionary developments will appear in the near future. The nature of data processing in the other stages than design is so different from the nature of computing in the design stage that a slower evolution must be foreseen. This is a field where it is important to standardize and agree on formats and procedures before software manufacturing comes in picture and such matter needs lots of time.

6. ECONOMIC STRUCTURES

Speaking of economics such matters like costs and cost/benefit analysis should be discussed and rates and tariff policies to earn the money. The software development costs are well mastered today, since a lot of consulting engineers and software houses have experience in this area. It is however less understood which huge costs it will take to maintain software and hardly any information is available as regards the costs of instruction and support. But these last costs are not to be underrated and may be determining in future whether or not some program can be kept in a public domain area.

6.1 Cost-benefit analysis

The most troublesome exercise has been and still is the cost-benefit analysis. Practice shows that most firms have good recordings of all costs but no registration at all of the benefits, which may result in the quick decision that computer usage just increases the costs of the design stage. This may be the case indeed and we could enjoy such an opportunity for better accomplishment of structural engineers professional responsibility. But if it is not a benefit for the designer, it still can be a benefit for the whole project. Here we hit the major barrier that according to existing regulations, agreed in old days, the design costs are allowed to be a fixed percentage of the total construction costs. This excludes the possibility to spend more cost in the design stage in order to save money in the construction stage. Thus the justification of computer usage must be found in the design stage itself and will actually not be found in cost-saving.

Observations make clear that no change to computer usage is made if one has no relevant applications or not sufficient tasks to justify it, but on the other hand, the real reason to change indeed is time-shaving, improved quality and new possibilities and in no manner cost savings. A cost-benefit analysis in its strict meaning can actually only be made for real automization where routine labor is substituted by computers. This is not the case with the introduction of design programs, for they are not just adopted for labor savings but to improve the ability of the designer. Now the benefit is found, among others, in the possibility to examine more alternatives in a given restricted time and to analyse more advanced structures, which could not be handled before. Nobody can reasonably be supposed to quantify such benefits and fortunately increasingly more managing people start to appreciate this.

Managers will in future ask for a long term plan for computer usage rather than for a cost-benefit analysis for each separate project.

Such a long term plan clarifies the interest of the computer and its costs during a certain time span, which enables management to judge its merits in relation to all other activities of the firm or company. If done in this global way one even can accept some desirable projects which may appear less profitable if on the other hand enough beneficial projects occur as well. A strict piece-wise applied cost-benefit analysis is really killing for the development of a broad spectrum 'toolkit' of design programs. It is a delighting sign that selfconfident managers view this realistically and so give free way for a modern procedure that most probably will be of much importance in the future.

6.2 Rates and tariffs for users

The user must pay the data processing center, be it in his own office or externally for all resources that have been used: the computer, peripherals, input preparation and analysis of output. Each data processing division has its own strategy to compose its tariff and can use it as a policy tool to promote or moderate the load of the components of his plant. It is common practice that such costs only depend on the resources involved, without any relation to the cost of the structure. Normally the same applies for the return rates that are paid to cover the development costs of the software. As a rule such royalties are a fixed percentage of the computer bill, although some attempts have been made to relate these royalties to the extent of the structure and the considered load cases. This latter trend can be wellcomed and will unquestionable proceed in future.

A peculiar symptom can be observed in organizations that did decide to install their own computer. Such a decision is made when the bill of external centres has reached a level which justifies a change to in-house usage, but after the installation of the own plant rather soon users will incidentally complain that external centres offer cheaper service. This indeed can be the case, notwithstanding a careful cost analysis carried out to underline the decision and two apparent reasons can be pointed out, which are briefly discussed here. The first reason is to be found in the greater appeal that is imperceptibly made on the support of usage consulting people when such people are callable in house. But in one way or another such support people have to be paid for, be it via an overhead in the cost of the resources or the rates for the used programs. The firm can easily find grounds to cover these costs in another way and to separate them from the tariffs to eliminate undesirable 'outwards driving forces'.

A second cause of user complaints is found in a difference in the tariff strategy of the in house data processing division and an external centre. In both centres the total costs have to be covered but the distribution of the total costs over the several activities and programs may differ. The single user has normally no global view on all facets but is magnetized by his own problem area and therefore inevitably pursues his own local optimum. This attitude is, sad to say, sometimes encouraged by management in the case that a design office has been bounded to so strict commercial goals, that the trials of users to escape from the own data processing centre can be appreciated indeed. Sound managerial judgement of all aspects will solve such problems and create the climate that is wanted to have the own centre flourish optimally. Possible additional costs can be kept apart from costs which are charged and are to be considered as the premium paid for the new possibilities that undoubtedly attend the installation of an in-house computer.

7 LEGISLATIVE STRUCTURE

Computer usage could lead to adaptations in the responsibility scheme as it was established in old day. Further one might discuss the interaction with building codes and enter in the matter of propriety rights. For reasons of shortness we restrict here to the legal liability for errors or misuse of programs.

A realistic and unremitting presentation of risks which are involved in a computer run results in a long list of potentialities. To start with, limits imposed by the present state of science, syntheses capability of human mind and hardware prevents the writing of a general purpose program to compute any structure while taking into account all possible types of behaviour. One allways has to choose from a library of programs only computing some types of behaviour for some parts of the structures. Here the designer has to make a major decision. Having made a choice, one further risks errors in hardware and software. Fortunately the hardware rarely fails and once it happens, it will be clearly visible. However, other sources of errors are numerous, ranging from syntax errors, bugs in the code, wrong interpretations of the results, used approximations in the numerical method and others.

The problem with most programs is that they are 'top secrets'. The list is either not published for economical reasons, or if published it cannot be read due to lack of proper comments and documentation. However, nothing else remains than to use such programmes for which nobody takes any responsibility. In many countries one tried to set guidelines or regulations for liability, and all of them have the common undertone that final responsibility always rests with the designer. He selects the program to be used, the data center to be called for and is responsible for the input data and for the output, no matter whether or not he engages usage consulting support. If needed he can refuse the results.

It is not difficult of course to list in the same way a number of responsibilities which are attached to the mission of the data centers, but it is difficult to connect legal consequences with it. Everybody, having experience in this field, knows and appreciates that a well-tested program which seemed to be correct can show a bad result after years when data input is in a special constellation. And it is agreed praxis not to make the computer centre too much responsible for the consequences. The centre is just supposed in such a case to correct the mistake and to rerun without additional costs.

In fact the only discussion can be how to facilitate the designer to execute his responsable task and not how to share this responsibility with the programmer. This has an impact on the education of the design engineer but may also result in directives how to write manuals and what to include in output. Therefore it is necessary to let the futural users have a big say in the specification of program outputs.

8 CONCLUSIONS

The presented survey and the more or less detailed discussion of several special items allow for some conclusions, listed below.

- A gradual evolution of the design profession cannot be denied. Adaptations take place in time due to the growing role of computer technology.
- Big parts of the designers still do not use computers intensively. Only a minor part has bridged the gap with computer science people.

- Designers are in danger to move away from their original ability to analyse structures. In future they may increasingly more become decision makers who are able to judge all interacting activities at a global level, getting stress results passed from others. But perhaps, this is just the proper definition of a designer!
- The dissemination of software and/of information on software still is a source of anxiety, for a number of reasons.
- Education is needed for chiefs, designers and computer people. The curriculum of design engineers has to be adapted to fit the evolution in the design practice.
- Broad spectrum integration of all stages of the building process meets with difficulties in practice. Probably it is necessary to analyse the existing communication structure before we can propose new computerized approaches.
- Cost-benefit analyses are not always useful for separate design programs. Managers should rather ask long term plans to judge globally the total computer effort in relation to all other efforts of the company.
- The final responsibility cannot rest but with the designer. The discussion therefore should be directed to the means how we can facilitate things for him to prove this responsibility.

REFERENCES

1. FENVES, S.J.: Scenario for a Third Computer Revolution in Structural Engineering, Journal of the Structural Division, ASCE, jan. 1971.

2. BOYER, L.T. and FENVES, S.J.: Environment and Characteristics of Civil Engineering Applications Programs, ONLINE 72 Conference Proceedings.

3. LOGCHER,R.D.: The Impact of Computers by Engineering Practice, ICES-journal, July 1974.

4. CEPA: Proposal for a National Institute for Computers in Engineering (NICE). Rockville, Maryland, USA, 1975.

5. KRAUS, H.: Attitudes towards Computer Software and its Exchange in the Pressure Vessel Industry, Journal of Pressure Vessel Technology, Febr. 1975.

6. BLAAUWENDRAAD,J.: Succes Criteria for the (International) Exchange of Program Systems, Proceedings CIB-W52 Symposium by Correspondence on Computer Languages in Building, Budapest, Hungary, 1975.

7. DOE: Main Study on Use of Computers in the Construction Industry, Department of the Environment, England, August 1976.

8. GALLAGHER, R.H.: Computerized Structural Analysis and Design - The Next 20 Years, Keynote Lecture, Second National Symposium on Computerized Structural Analysis and Design, Washington, March 1976. Comp. and Str., Vol 7, 1977.

9. CIAD: Policy report 1977 - The Way Ahead, Zoetermeer, The Netherlands.

10. LAWRENCE, D.J. and SHEPPARD, D.J.: CAD-Education - The more the Education, the less the Use, Proceedings CAD-ED conference, Teesside, England, 1977.

11. BLAAUWENDRAAD, J.: CAD and the Educational System, Invited paper CAD-ED conference, Teesside, England, 1977.

12. SCHAEFFER, H.G.: A Review of the International Symposium on Structural Mechanics Software, Comp. and Structures, Vol. 8, 1978.

13. DOLCETTA, M.: Trends in Computer Management for Structural Engineering, ENEL-paper for IABSE-Colloquium at ISMES, Bergamo, Italy, 1978.

14. JONES, M.V.: Computers in Engineering, ACADS-paper for IABSE-Colloquium at ISMES, Bergamo, Italy, 1978.

15. LANG-LENDORFF: CAD-Promotional Funds by the Federal Government, IABSE-Colloquium at ISMES, Bergamo, Italy, 1978.

16. TOMINO, H.: The Computer Usage Environment for Structural Engineers, IABSE-Colloquium at ISMES, Bergamo, Italy 1978.

17. KLEMENT,P.: The Responsibility for Electronic Calculations, IABSE-colloquium at ISMES, Bergamo, Italy, 1978.

18. UHERKOVICH,I.: The Influence of the Computer on the Professional Ethics, IABSE-colloquium at ISMES, Bergamo, Italy 1978.

19. DEPREZ,G.: Professional Responsibility of Engineers using Computers, IABSE-colloquium at ISMES, Bergamo, Italy, 1978.

20. KRUISMAN, G.: The Unknown Triangle, IABSE-colloquium at ISMES, Bergamo, Italy, 1978.

21. PEANA, A. and MAIER, G.: Educational and Professional Implications of Reliability Assessment in Computerized Structural Analysis, IABSE-colloquium at ISMES, Bergamo, Italy, 1978.

Leere Seite
Blank page
Page vide