SOFTWARE TECHNOLOGY

A FUTURE REQUIREMENT OR CURRENT NECESSITY ?

Dr. Harry Blask

Dr. H. Blask
Ministery of Research and Technology
Bonn
West-Germany

"SOFTWARE TECHNOLOGY

-

A FUTURE REQUIREMENT OR CURRENT NECESSITY?"

DR. HARRY BLASK

Ladies and Gentlemen,

1. Introduction

Every time a new technical achievement is made there are a few resourceful minds who are quick to make use of it to the disadvantage and detriment of their fellowmen. So the development and propagation of information technology has "blessed" mankind with so-called computer crime. I would like to cite one of the cases reported to date which I think is of particular interest to the subject of my paper:
A few years ago, an audit was announced unexpectedly in an English bank. The result was that some employees of the data processing division disappeared overnight (and probably left the country). This practically confirmed the suspicion that irregularities had occured. The software was then examined with special care; but in spite of an intensive search, the manipulations which had most probably been made could not be discovered.
This example throws special light on the present software situation: Even specialists fail to detect existing errors or manipulations. I would like to examine this situation further in a somewhat generalized manner before I eventually embark upon the actual subject namely software technology.

2. Situation and impact of information technology

In modern industrial societies, the generation, processing and transfer of information are playing an increasingly important role. Together with microelectronics, data processing and technical communication, information technology represents a key technology which none of the major industrialized nations can afford to neglect. Consider, for instance, the increasing integration, and the further decline in prices, of electronic components: here, future development will lead to fundamental changes for manufacturers and users of information technology as well as for private users. Since we are still at the beginning of this process, we can by no means predict all the consequences which will be brought about by information technology. Some effects, however, are already emerging today with a sometimes depressing clarity:

- The institutions using information technology and, in the final analysis, society as a whole is becoming more and more dependent on this technology. Important sectors of production engineering, the control of complex technical plants, the administration and handling of large stocks of information, all these are practically no longer possible without data processing. If data processing ever fails, this may have disastrous consequences, including even the destruction of companies. One of the main elements of the vulnerability of those who rely largely on information technology is the increasing complexity of technical systems and their applications. They make use of electronic information processing because, above all, this allows numerous links and interconnections leading to new information. The resulting trend towards increasing complexity makes the systems less transparent, less

controllable, and capable of being mastered by only a few specialists as was illustrated by the example I quoted at the beginning.

- Information technology also penetrates increasingly into sensitive areas. It controls railway and tram signalling installations; air traffic control is more and more computer-assisted. Presumable nuclear power stations, too, will soon be monitored and controled by computers; this, however, is not yet permitted in the Federal Republic of Germany. Errors or failures occuring in these areas would result in serious accidents involving loss of life. We have still far to go before finding a solution to the problem of ensuring the reliability of information systems.

If we now consider information technology from the hardware and software aspects, we will find that the software has nothing comparable to set against the technical progress of hardware. This is probably due to the fact that hardware is the result of engineering to a much greater extent than software and that engineering uses sophisticated working techniques and theoretical elements with a long tradition. They were already known and practised long before data processing was developed. Programming, on the other hand, is relatively new. It takes place at the boundary between technology, organization and the humanities and to this day is considered an "art" and not a science. When a system is implemented the cost share of software todey is already in the region of 80% and shows a tendency to increase.

3. The software crisis

Let us first of all consider the manifestations of the so-called software crisis. When comparing the software development of 20 years ago of that of today we find that little has changed. The scene is still dominated by the "free lance artist" who produces software without observing too many rules, giving free rein to his intuition. For him, it is not important that

- software be structured in a transparent and intelligible way so that later on it can be understood and, if necessary, modified or supplemented also by the other users,

- adequate documentation be provided for programmes; this applies even more to considerations of, and decisions on, design, which today are put down in writing in very few cases only,

- an extensive phase of problem analysis and definition precede the design and implementation phase,

- programme protability is achieved.

As a result, software is prone to many errors; in the case of major operating systems, for example, between 5000 and 10000 errors per year are reported to the manufacturer. It is often found that software is no longer usable soon after its developer has left the company or if development capacity is no longer accessible for other reasons.

Other symptoms of the crisis are:

- Users and developers have difficulty in communicating with each other. Having different engineering backgrounds, they do not speak the same language. Communication is further complicated by the fact that the colloquial language we normally use is informal and imprecise.

- The means of expression and description available for software development are inadequate. This applies in particular to graphical means. The large variety of programming languages available complicates standardization.

- Progress and productivity in software production are particularly difficult to control. While between 1958 and 1978 the processing productivity of computers increased by a factor of 1000, the productivity of programmers increased only by a factor of 2! Projects are managed and organized in a shirt-sleeve manner and consequently yield unsatisfactory results. A study carried out by the University of Karlsruhe involving 100 automation projects showed that, on average, software had additional time requirement of about 50%, while the costs were about 75% higher than estimated.

In addition to these qualitive characteristics of the "software crisis", there are some important economic aspects:

a) In the Federal Republic of Germany alone, about DM 10 thousand million are spent annually on software. About 80% of the costs of a data processing application are accounted for by software. On the one hand, this is due to the fact that software is designed to fulfil

increasingly complex tasks - thus becoming increasingly complex and expensive itself -, on the other hand, microelectronics brings out price cuts for hardware, while this can hardly be said about the soft- ware side.

b) Competition between data processing suppliers has been shifting to the software sector. This applies without reservation to operating systems, which determine computer characteristics much more then hardware does. If you just remember how many computers - especially smaller ones - use the standard components from Intel, Texas Instruments, Motorola and Fairchild, you will recognize that the net value added to a product and its identity and hence its competitve ability come from the software, including application software.

c) Available data processing solutions are inadequately portable or not portable at all. The same problem definitions are therefore handled again and again - a practice which constitutes an enormous economic waste.

This is why we have to look for new approaches in order to reduce the cost of software development and maintenance. Just as we use high-precision tools in production engineering, we will have to use the computer itself for software production. Neither our own capabilities nor our own ability to analyse and integrate are sufficiently powerful to cope with the software crisis.

4. The software technology scene

Approaches to software production technology have developed since the late 60's and appear promising: at the present time, however, the term "technology" is rather flattering.

Today, software development and maintenance are generally based on the following phases:

- requirement analysis and specification
- rough design
- final design
- coding and implementation
- maintenance.

To include testing in this list as a phase of its own would, I think, not be correct, since testing activities occur in all the phases cited above. For all phases, there are isolated solutions with quite different capabilities. Computer-aided tools and methods for the requirement (analysis and specification) phase have so far been the least developed elements. One of the main reasons for this is probably the fact that there is a contradiction between the necessarily application-oriented character of this activity and the need for highly flexible tools for varying applications. For the rough and final design phases, the situation is better: several methods (e.g. the Jackson method, structured programming) are at least known to users by name, although they are actually used only by a few advanced software developers. In the coding phase, which is the traditional field of activity of programmers, methods and tools of software technology are more wide-spread, of course, whereas the maintenance phase is still characterized by a very small inventory of methods and tools.

The methods, techniques and tools used are often quite sophisticated and hence can be applied only by experts. In addition, the value of software technology is frequently not easy to understand. It is absolutely impossible for the normal user to estimate the value of such technological tools. As a consequence, the barriers preventing their introduction are tremendous: there is a great reluctance to introduce such tools on the part of mangement boards, who hesitate to invest the large funds required for the necessary training effort and for investments, as well as on the part of software developers, who are expected to give up their old familiar working methods and to aquaint themselves with new sophisticated techniques. In view of this situation it is no wonder that tools of software technology are not yet widely used either in the Federal Republic of Germany or, probaly, in other countries.

5. Measures initiated by the BMFT to support software technology

In order to prepare the ground for support measures, the Federal Ministry for Research and Technology entrusted the Society for Mathematics and Data Processing (GMD) with the undertaking of a study to analyse measures for the improvement of software production and to identify research areas which, in the medium term, are, or will be, of importance for users and software producers. The study also included a detailed analysis of the market of software suppliers in the Federal Republic of Germany. Let me cite a few results of the study:

> There are about 3.000 suppliers of software in the Federal Republic. About 85% of them have a capital of less than DM 250.000, which means that most of them are not able to raise enough funds of investment in tools of software technology. But also the more powerful suppliers who would be able to invest have not, as a rule, reached a satisfactory level of know-how concerning software technology. Owing to this situation, the software industry will, in the medium term, not be able to make use of the existing growth opportunities, since its technological know-how and also its economic potential are limited.

A country like the Federal Republic which - because of its dependence on imports - must be able to offer technologically advanced products cannot affort to neglect important key industries. The situation I have just described almost inevitably calls for the government to initiate supporting measures.

Under 3 successive data processing programmes from 1967 to 1979, the Federal Government gave support to research and development activities for hardware and software. A total of about 3.5 thousand million was provided to support research and development in this field. The first programme focused on hardware, the second and third on software, with an increasing emphasis on small computers. Applications of data processing in many different areas were supported (e.g. in administration, medicine, process control, computer-aided design or education). By promoting the field of data processing during the years up to 1979/1980, many of the objectives of the data processing programmes, which were designed as a pump-priming measure, have been achieved. As a consequence, the financing in particular of future products and standard applications will now be left to industry.

Present support measures concentrate on existing deficiency areas in information technology. To give you an idea of these measures, let me cite - apart from software technology - a few other important areas in which support is given predominantly to basic research work and systems know-how:

- data security
- information technology for office and administration tasks
- very-large-scale integration of electronic systems
- fundamental principles of systems engineering
- pattern recognition
- communications technology.

I do not want to go into greater detail. For those who are interested, let me point out that the support measures proposed are always announced in the Bundesanzeiger, the official organ of the Federal Government.

From the above remarks on the software crisis it is immediately obvious that much remains to be done in the field of software technology. This need is certainly not controversial. It is also, I think generally accepted that the quality of software can be improved, above all, if a systematic approach is developed and introduced for the problem definition and the design phases. But opinions differ when it comes to deciding which methods are the most efficient, since this quantitative efficiency is analysed or proved only very rarely; which means often beliefs are stated in the absence of evidence.

It would be just fine if the whole software life cycle could be covered by a few methods. But this is wishful thinking and hence unrealistic. One of the conclusions of the abovementioned study by the **GMD** is that the development of methods cannot be separated from the individual products to be developed.

The means of expression for design and testing certainly are - other methods may be - dependent on the kind of product to be developed (e.g. micro-systems, compilers). Nevertheless, there are certainly also a number of methods which lend themselves to general use. The variety of techniques, methods and tools required is and will be large - at least judging by the present state of the art and by present know-how. A policy aiming at the advancement of software technology and its use in the Federal Republic of Germany has to take this fact into account and consequently has to support a relatively large number of means of software technology before strategies can emerge which it would be promising to pursue with emphasis and in cooperation between several parties. We are focusing our support measures on areas where we believe the standard of know-how is particularly low. These areas are:

+ Methodology

Existing methods and tools are at present often used for only one or several software development phases. As development work proceeds to the next phase, extensive modification operations and manual adaption work are usually necessary, which would not be required if we had a coordinated and integrated system of methods. Whether eventually a system can indeed be developed which would be able to support the whole software life cycle from the requirement analysis to the maintenance phase without intermediate manual work is, I think, open to doubt. It is clear, however, that methodology is at present an area with quite considerable deficiencies. And since even relatively little progress can be expected to yield significant growth in productivity and improvement of quality, about 60 per cent of the funds available for software technology are earmarked for this area.

Individual methods in deficiency areas

- Description of requirements / Software design

Although some methods do exist, they are not all particu- larly satisfactory. In this area, we give specific support to graphical methods, because they seem to us especially promising for an engineering approach. The Petri networks may play an important role one day in this connection.

- Quality assurance

in particular testing. In addition to testing the code, testing (also by machines) during the first phases of the software life cycle will be of increasing importance, in particular because errors in the description of re- quirements and in software design are particalarly expen- sive errors, since they are usually detected at a late stage and following extensive searches.

It is hoped that two projects which we support and which are dedicated to programme verification, i.e. proving that the code contains no errors, will be successful. If a real breakthrough is achieved here, we shall feel far less concern than before at the thought of using software also in sensitive areas.

- Maintenance

50% to 80% of the cost of a software product during the entire period of its use are caused by maintenance. Improved maintenance will be achieved automatically when better methods are used for the initial phase of the software development. But apart from that, improved methods of maintenance must also be developed. We are giving special support to activities for the development of remote diagnosis and maintenance.

Evaluation of methods

As I said before, this is an issue of special interest to the user. The trouble is that the situation with regard to methods evaluation is particularly gloomy. There are practically no measuring methods and techniques which could yield quantifiable and meaningful evaluative results. The projects we are supporting do not at present include any such activities.

One of the requirements we have fixed for all the projects we promote is that the processes and methods can be used with computer support. If they cannot be used in this way we are afraid they will mostly not be a long-term success in practice.

A second reason for this requirement is the fact that a method is not mature, logical and consistent until it can be programmed. The work we support is intended to achieve progress in software technology. The development results must be largely portable so as to facilitate more general application. Developments which are of value first and foremost to one interested party only, will not receive support. In the case of private enterprises it is out general policy to bear only 50% of the project costs.

6. Supportive measures by the European Communities

I shall now say a few words about support measures by the EC.

In 1979, The Council of the European Communities adapted a programme of supportive measures for the field of data processing covering the period from 1979 to 1983. Over these four years, the programme will provide 25 million European Accounting Units. The programme is subdivided into two parts:

a) General action

Dealing with standardization, public procurement, cooperation between research centers and organizations which advance the use of data processing, studies on employment issues; data security and data protection as well as the legal protection of computer programmes.

b) software and applications

In the field of general software, the EC Commission gives support to projects aiming at:

- the establishment and dissemination of standards
- improved portability
- improved conversion conditions
- improved efficiency of data processing systems.

The supportive measures in the field of applications are not limited to certain fields, but they have to comply with specific criteria, which I do not, however, wish to enumerate here. The programme's terms of reference are so general that a relatively wide range of tasks can be included. More specifically, software technology projects are also eligible for promotion. In order not to raise false hopes, let me point out to you that all projects to be supported by the EC must comply with the requirements of advancing cooperation within the EC, which means that they must be carried out jointly by institutions in at least 2 EC countries. The measures supported by the Commission are aanounced annually in the official Gazette of the European Communities.

In addition to the steps taken by the EC Commission, other European countries, besides the Federal Republic of Germany, also have initiated action for the support of information technology. These measures differ very much from each other so that it would take too much time to go into further detail. Generally speaking, it can be said, I think, that small EC countries (e.g. Denmark, the Netherlands, Belgium) do not give financial support to industrial information technology activities, whereas larger countries like the United Kingdom and France provide quite considerable funds for this purpose. The resposability for national support measures is assigned to different government departments in different countries, often to the department of industry, research or education.

7. Considerations on a future coordinated system technology

Before - in conclusion - trying to answer the question asked in the title of my paper, let me draw your attention to a technology of which the software technology is only one aspect. Hardware and software (by software I mean both operating systems and user software) were poorly coordinated from the first years of the great increase of computer use. As hardware development raced ahead, leaving software a long way behind, the gap between the two grew wider and will even increase as new hardware architectures are developed. The existing programming languages and techniques would make only very effective use of the possibilities offered by hardware. Consequently, it is an increasingly urgent task to make hardware and software development converge into a co-ordinated system technology.

8. Conclusions

To summarize, let me emphasize that the software crisis must be overcome as soon as possible. As I said before, some DM 10 thousand million are spent annually on software in the Federal Republic of Germany. Successful rationalization in software production which reduces in particular the amount of maintenance required will soon result in major economies. If the software crisis is not overcome, the majority of staff engaged in software production would spend all their time on maintenance work and therefore be unable to tackle urgent new tasks. It is frightening to think that a very large portion of the qualified manpower engaged in data processing - which is in short supply anyway - is not available for innovation work which is urgently required by the economy. The need to reduce the great dependence and vulnerability caused by information technology by improving the structure and thus the managability of software is another aspect which is of interest to the society as a whole.

The software producer is at present in a rather favourable situation compared with other producers. This is due on the one hand, to the extremely high demand for software, which can hardly be met, and, on the other hand, to the fact that for many customers software is still a book with seven seals, so that they have to fully rely on the producers. I am sure that this situation will, however, change during the next few years. Users will become increasingly aware of the problems involved in information technology and will endeavour to aquire at least some basic knowledge. They will then look at software with a more critical eye and will - more often than in the past - demand better quality. Software producers who are not able to meet such requirements will not be able to hold their market position.

The availability of the tools of software technology will then be crucial prerequisite for the survival of a company. Since software technology cannot, however, be introduced overnight, but probably only in the course of several years, software producers would be wise to begin immediately to tackle this task. Difficult as this task may seem in the face of such a multitude of methods, it should not be postponed, since any delay jeopardizes future opportunities.

Thank you for your attention.