Organisation of software

Autor(en): **Sriskandan, K.**

Objekttyp: **Article**

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

Band (Jahr): **11 (1980)**

PDF erstellt am: **13.07.2024**

Persistenter Link: https://doi.org/10.5169/seals-11330

VII

Organisation of Software

Organisation du logiciel

Organisation von Software

K. SRISKANDAN
Deputy Chief Highway Engineer
Department of Transport
London, England

SUMMARY
Computer hardware costs are decreasing all the time. On the other hand the production and maintenance of applications software is still a very expensive item and needs to be organised in such a way as to maximise the benefits from the investment in it. This paper discusses how to get the best out of applications software.

RESUME
Les coûts d'ordinateur sont en baisse constante. Cependant la création et le maintien du logiciel d'application sont toujours très chers; il faut donc organiser celui-ci de telle manière qu'on en tire au maximum les avantages de son investissement. Cet exposé discute les possibilités de mieux utiliser le logiciel d'application.

ZUSAMMENFASSUNG
EDV-Kosten sinken ständig. Anderseits sind die Erzeugung und der Unterhalt von Anwendungs-Software noch sehr teurer und sollen so organisiert werden, dass die Investitionen einen grösstmöglichen Nutzen bringen. Es wird erörtert, wie man solche Anwendungs-Software am besten benützen kann.

1. INTRODUCTION

Computer costs have reduced drastically in the last few years and this trend seems likely to continue. The production and maintenance of applications software is nevertheless a very expensive item and needs to be organised in such a way as to maximise the benefits from the investment in them. The Highways Engineering Computer Branch was set up in the Department of Transport in 1969 as a focal point for all Highways and Bridges Computing matters, including the provision of programs. In the light of experience in the Department of Transport, this paper discusses how to get the best out of applications software.

2. Structural programs and systems, whether they be for analysis or for optimised design, tend to be large (4,000 statements and upwards). The determination of the engineering requirements, parameters, the development and testing, and the subsequent documentation, is a very time-consuming process. At roughly 200 statements per man-month, these programs do represent a considerable investment and it is essential therefore that they are

- User orientated.
- Used by a large number of users without the massive duplication of effort that will take place if each organisation had to write its own programs.
- Maintained adequately and can be used during their working life without the need for significant alterations to match changes in hardware - even with advent of microcomputers.

3. USER REQUIREMENTS

3.1 Computer programs are written for the benefit of users who should therefore be consulted. The larger general purpose programs require considerable amount of data preparation for input and also unravelling of the output before it can be eventually used by the designer. Significant improvements can be made in the usefulness of programs by writing special input and output routines. The first of the input routines can be specific to a loading code, reducing even further the work of the designer. This will be of particular benefit with the new generation of codes which require the consideration of a variety of load combinations.

3.2 As for any development project, the cost-benefit from a program should be evaluated before the decision is taken to write it. In most civil engineering applications the designer has been able to use methods which would have been impractical without a computer. The benefits come partly from savings in design time but more from savings in material in the final design.

4. PORTABILITY

4.1 Clearly, the benefits from any one program will increase with the number of users. However, user organisations will not all have the same type of computer. Therefore programs which are written for a wide audience must be capable of running on any machine with

the minimum of change. In my Department programs have been written in a special sub set of ANSIFORTRAN which has enabled the programs to be run on any computer which offered FORTRAN, ie all makes of main frame and mini computers that have so far been encountered. A total of 8,124 copies of 166 different programs, some of which are grouped into systems, have been issued.

4.2 The writing of programs is so expensive, that the need for this kind of portability will apply not only to a central organisation which acts as a focal point, but also to individual organisations writing their own programs. The larger general purpose programs have a useful life of about 10 years before they become obsolete. Within this time the company computer is likely to have changed at least once - maybe even twice. Re-writing a program is a costly exercise. All of this assumes, of course, that the same high level language will be available in successive generations of computers. Even as I am writing this, FORTRAN is becoming available on some MICROs. Computer manufacturers should be made aware of the tremendous investment in the existing stock of software and unless there are over-riding reasons, should be "encouraged" to provide the capabilities to continue using these programs.

5. THE USER AND THE PROGRAM WRITER

5.1 Apart from the very small programs which a designer may write himself, the others are written by those who are employed full-time on programming. However, it is the designer who uses the program and he must have complete documentation, in the form of a user manual, on how to use the program. Any self respecting designer will also want to know the basic theory underlying the program, the suitability of it to his particular problem, and advice on idealisation and selection of parameters. In my Department we also prepare a program manual which describes both the theory and the program itself. In addition to helping the designer it also serves as a record which can be referred to if the program has to be amended or translated for some reason. Following discussions with designers on what further advice they would like, we have started work on the preparation of a series of guides giving advice on how to analyse given types of structures. This consists of the selection of type of computer program, idealisation of the structure and determination of the properties of its constituent parts, all to give a sufficient and sensible degree of accuracy at a reasonable cost. Some parts of the first of these guides dealing with slabs and pseudo-slab bridge decks [17] were completed in 1978.

5.2 A designer who uses a program (whether it be from the computer section of his own firm, or one from a central organisation, or one at a commercial bureau) will want to have confidence in its correctness. The user can check the overall goodness of a program by carrying out his own sensitivity checks and comparisons with exact results or he can rely on such checks done by a central organisation. However much a programmer may check his own program or check someone else's on behalf of the designer, it is the latter who prepares the data, unscrambles the output and finally uses it in the design of the structure. It is therefore the designer who must take full professional and legal responsibility for his design. There should never be any question of divided responsibility between the designer and the program writer.

6. MICROS

With the advent of the micro-computer and its direct response to the user, it should be possible to provide a library of short micro functions, ie individual processes of about 100 statements which can be assembled, preferably by the engineer, to carry out a number of different functions. Such micro-functions will be easy to write, test and document and easy to understand. This will make designers more conversant with the computer and narrow the gap between programmer and designer. However the one great problem with this is that it requires open documentation and the user could corrupt the coding. In the interest of progress it seems to be a risk worth taking and my Department is looking into the feasibility of this proposal.

7. FOCAL POINT

Duplication in program writing is a waste of highly skilled man-power. Apart from program writing and associated documentation, there is also the need for program maintenance, specialist advice, and even having research carried out in special cases for program verification. It is very desirable that all these activities are centralised in one focal point which provides a service to the users. In UK, the Highways Engineering Computer Branch of the Department of Transport acts as the focal point for all computing related to road and bridge design. On the building side, there are cases of some Consulting Engineers forming a group of their own for this very purpose. Organisations vary in each country - but it should be possible to have focal points in software. At this stage such focal points can only be national - not international.

8. CONCLUSION

- Programs must reflect user requirements and therefore have special input and output facilities.
- Programs must be written in such a way that they can be used on a variety of computers - ie Portable.
- Manufacturers must be encouraged to provide capabilities in their new machines to continue using existing stock of programs.
- Users must have adequate documentation, both on the program and on how it is to be used.
- Designers must take full responsibility for the results as used in a design.
- The advent of Micros could result in a new approach to computing with the Engineer getting closer to the Computer
- A Focal Point for Computing will help to achieve some of the aims set out above.

REFERENCE

1. HECB/B1/7: User Guide for Slab and Pseudo Slab Bridge Decks, Department of Transport, London.