Zeitschrift: IABSE reports of the working commissions = Rapports des commissions de travail AIPC = IVBH Berichte der Arbeitskommissionen

Band: 31 (1978)

Artikel: Discussion

Autor: Blauwendraad

DOI: https://doi.org/10.5169/seals-24927

III SESSION

DISCUSSION

September 1, 1978, Morning.

Chairman: BLAUWENDRAAD (Netherlands)

BLAUWENDRAAD - I think now we can open the discussion to your contribu_ tions and comments.

KLEMENT - When you have made programs, you know that not only the tech_ nical aspect of the programs, but also the work which you have to do until the program is tested, gives you the right to say: "this program is mine". If you document the way in which the results are made in a mathematical form, you give what the user needs. You do not need to give full information about the inside working of a program to let a user be able to use it in a perfect way, and not always there is some progress in making programs again.

BLAUWENDRAAD - I think this is for Mr Alcock.

ALCOCK - Of course, I understand this, because my organization lives by sel_ ling software systems. For example, we have a system that is a version of STRESS, and I think there are about thirty copies already sold but we do not protect the internal documentation; we are quite happy for it to go out. To the best of our knowledge, nobody has stolen it yet, and it is just more convenient to pay the knowledge, nobody has stolen it yet, and it i s just more convenient to pay the price of this, for the person who knows its inside and can support it, but at least when they make mistakes, they can see what it is that the pro_ gram is trying to do. Furthermore, they can adapt it to their own purposes and add to it, and so on. One more point on this : I am not, in fact, necessarily pro_ posing that if you write in FORTRAN you will provide the FORTRAN. What I try to present or just to suggest is a notation which is half way between the mathematical description and, say, the FORTRAN realization. We have done it for this program FORPAR, and what you buy when you buy FORPAR is not necessarily the FORTRAN. If we have that, it will cost another sum of money, but what you get is an exact description of what the program attempts to do, and one buyer of this can provide to make his own realization of it. Someone has adapted it for interactive use; some have put it into back use, but the point is that there are many installations on many computers, and this has been done by means of this intermediate notation and not FORTRAN. Thank you.

PFAFFINGER - I would like to make two brief comments about the responsibi_ lity. Today, for almost all the programs, you have an agreement with the de- veloper of the program that is not liable in any respect for anything that might

happen with the program or the results of the program, and the same is true for the data center. If you work on a data center, you expressly have to accept the fact that the data center will not be liable for any program they use there, or any result which might be wrong. So, I would like to stress this point : even if from a little point of view, you are forced to check your results. The second comment I would like to make is this: I strongly support the opinion of Prof. Klement, that it is really not necessary to know every detailed documentation and every detail that are in the program. As a matter of fact, we see the development of something which I would like to call: "The Fortran Industry", and there are people making their living on developing software. Software is much protected from a legal point of view. We have to know the basics of the procedures; we do not have to know all the details, and we are just glad if we are in a position to check what is coming out of the program.

ALCOCK - To explain shortly on the legal side, I do not know the laws: I think that in all the countries they are different, but I know that in Britain I am not allowed to put on the front of my car: "The driver of this car is not responsible for anyone standing in his way".

HAAS - I would like to comment on the contribution of Mr Alcock, concerning documentation of programs. I think we have to distinguish between the part which deals with the stress analysis and the part which handles with design forces. I think that the first must not be so well documented and it can remain a black box, whereas the second, which handles with design forces, must be clearer. The design forces are open to interpretation, and it must be said very clearly how to interpret them and how we handle the results, how we get them.

DEPREZ - I would like to comment on Mr Uherkovich' s paper about professional needs. I think the main problem is that the computer permits to all designers to all consulting engineers to think thay can solve any problem.
We can solve it in two ways: first, we can ask the user to have a licence for this, but I think we have another possibility. We must know exactly what we can do, what we can get like guarantee. Now, to precise correctly the responsibility of the computing center or the consulting engineer. Everyone can choose a specialist in the structure he wants to compute, or a consulting engineer specialized in the use of computer in this particular field. For instance, we can say in dynamic problems of large structures, there are no many organizations in Western Europe which are able to do it, and we know that somebody has this kind of problems and goes to the computing center to try to have a solution. If there is no previous experience in this field, I think there are a few chances to reach a good solution. In the ethic of the profession, in my opinion, it is important that the consulting engineer knows that he is unable to solve this problem himself. He must not try to obtain from the computing center the information to solve it, but he should have to go to consulting engineers who are specialized in this field.

TAFFS - Sometimes you treat the computer just like you were treating an engineer or an assistant. When we talk about this possibility, to delegate someone to carry out a part of the design, we do not feel we are giving a part of responsibility for that design to another person. We automatically say: "The subordinate will make a mistake without our guidance and he is perhaps the judge". We can give this subordinate some guidelines and we can ask him to follow them, but we will expect that the subordinate automatically is able to follow to the letter or we will check. If we look upon the computer like a subordinate, I think it can help very much in clarifying which the areas of responsibility are. Another important point of discussion could be the use of a particular system, when you are forced to assume it by the client.

TOMINO - If we are talking about the responsibility, whose is the responsibility in an engineering society ? We are responsible and the designer too, this is obvious. But if we are talking about the computer programs, any computer center has in contact a limited responsibility. It is always said in a contract that if someone uses a program, the computer center is unresponsible at any degree about results. For example, if we give some program to some user, we say in our contract: "We are responsible for the maintenance, but we are not responsible to any extent about results coming from that kind of program". That is our current practice. Then the question which comes up to my mind is again: "Are we talking about avoiding such kind of catastrophe which may occur to the practical engineers ?"

MILSTON - I am very interested in Dr Gallico speaking about a failure which has occurred with some computer design on dam operations. I would like him to amplify about this failure, presumably due to a computer program.

BLAUWENDRAAD - Mr Milston wants to have a 'scandaleuse rubrique', so you are cordially entitled to contribute.

GALLICO - This failure ... is a typical "case history" and very serious. In fact, it involved about 140 millions of dollars of loss. Luckily, no people was injured, but anyway, as far as money is concerned, it was and still is a big affair. This happened in South America, and it was reported in the various papers. It was a system of a hydroelectric power plant. The entire operation of the system was designed by engineers and the input was given according to meteorology, hydrology, operation energy conception, in order to optimize the use of water.
This was translated into a program and centralized in a despatching center which was completely responsible and had the best trust in model idealization of operation. Everything was arranged for several years, as far as I know, because we were called - and we are still called - to give our opinion.
It happened that the local operators were unable , and hadn't the legal authorization or the technical skill, to discuss the orders given by the dispatching center. Nearly one year ago, the inflow was a little bit different from what

they had forecast, and the local operators advised - informed by telephone the dispatching center, that there was something wrong, not matching with the anticipation of the operation. The dispatching center did not discuss that at all. They only said: "You keep silent, obey and do not discuss". But the local operators insisted and it was a matter of three days, two or three days, they insisted saying that the anticipation of the program and the model were not covering this same case. Then, the responsible personnel was convinced that there was something wrong in the model, but it was to late to operate. So, they decided to close a power station and open a discharging canal. The water was too much and overflowed the arch dam, destroying it. At the same time, power went out of service, the flood came to a second dam and destroyed it; then it entered into the power station. Now, this case is dealt on the trial and I cannot give at the present time what will be the result of the analysis, or what happens. I would not like to give the impression that the mistake was on the computer, nor in the program. This case is an example of insufficiency and blind trust in the results which, sometimes, are not covering all possibilities.

TAFFS - Often it is not the computing of the program, of course, which is at fault, but the designer, the human element. One experience we have had, which was very painful, where there was an error in a program used very seldom at that time. If the failure is embodied within the program, some social trouble can result from it. The program we used was well accepted, many of the normal structures for which the program was designed have run successfully through. Something wrong happened in one case: if the program had been well tested, the fault would be detected at the time, but testing a program is something we cannot cope with. The cost for testing a program is out of proportion to the cost of developing the program and the cost of development, we know, is already exorbitant. We have gone up to the point of spending up to 2/3 additional cost. We had another case where we are analyzing a primary structure, a very important structure, and we could not understand the answers coming from the computer. We sent the information to three researh centers, for their advice: in each case we had a reply which did not completely explain how was such a complex inter-relationship within the various parts of the building. Luckily, the project engineer insisted on this investigation and we discovered there was a great mistake in the program.

DUTERTRE - Just about program testing, if you have had a program which you have used for one year or two years, and if you think you have tested when you try to commercialize it, then you have a big surprise, because it is tested in your firm, with people thinking and doing the same way. When the program goes outside and you have fifty persons, fifty different firms doing things in fifty different ways, only in this way we can say the program begins to be really tested, because the best testing procedure is: "The more people use a program, the better the program becomes."

HAAS - Sometimes there is the trend of treating programs like a human being. But people cannot say: "That is your program, that is your baby and you have to look and bother that it works well." Then we say: "Yes, in our opinion the program works well, but if the program does not do completely what you want and makes some mistakes, having or buying it, you must check and take the responsability that it works well".

GALLICO - I am a little worried at this time because I heard there may be an error in the program even more than one time. I don' t agree : so, again, there is a misunderstanding. When in our office our young engineers use the Hewlett Packard, a little computer, I agree completely, because it is manageable, little, let us say they can face quick problems and that is all. When people have a big problem - I mentioned before Kalayaan power station in the Philippines, something very complicate static structure, or the structure shape, underground construction, etc. rock characteristics not well known - we rely completely on the program made by specialists . We keep our responsibility, we take the responsibility of the design because we know that these gentlemen probably can explain better than I what is the responsibility of the designer or the consulting engineer - but we assume that the computing center gives us and develops results using 100% tested programs with no mistakes.

SHIMADA - Luckily you keep your possibility to do a lot of equilibrium checks and you always can do some more check calculations to find the order of the displacements, and so and so.

KRUISMAN - I would like to make a comment on errors. I think it is good to start from the statement that all is right, what has not been proved to be wrong. The only way you can approach what is wrong is to get something like a "common opinion" and that again is a play sitting together and exchanging experience on programs and so on, and then you get from all the new users who check and find the errors in the program; then you get an idea of what Mr Alcock also mentioned: the evaluation of several programs that has, for instance, been done in the Dutch Association. There are several groups, say groups of programs, that evaluate and exchange information about them. I think it is the only way to come to the statement that something wrong is in the program. You never can prove that the program is right, because we do not know what is right.

PFAFFINGER - This philosophical statement is challenging, I think.
I put a thesis and this thesis is: unverified calculation is wrong until it is verified. To my understanding, it is the best approach, because we know by experience that our tools often are insufficient or inadequate and we do not know what is right in many of the date problems that we are solving. For instance, let us consider linear elastic analysis: the basic of this analysis is well-established, and we have enough data to verify a problem. It is a different story if you do highly sophisticated dynamic analyses of something that no-

body has done before. There I agree: we do not know what is right, but in the other cases, we know what is right and I think we have to verify our data assuming they are wrong, the result are wrong, and we have to verify and prove that they are right.

KRUISMAN - Well, you say that the point is that a linear elastic analysis is proved to be right. It was about 15 years ago or 20 years ago that a lot of English planes came down and that was the start of a deep research on mechanics; up to then we did not know that the problem existed. There is a lot of things we do not know, because we never experienced them. This is the reason why I say that unless it is proved to be wrong, it is right. Of course, you can contradict that. Is was a large contradiction, I think, in the country of Prof. Klement at the beginnging of the century, whether you approach the problems from the positive side or the negative side.

DUTERTRE - About right and wrong, I would like to quote a statement from Prof. Newmark in 1965, in a Soil Mechanics symposium. He said : "All the formulas are wrong; however, they are wrong in a consistent manner". The problem with computer is: "they are wrong inconsistently, with no coherence in the way they are wrong".

VOS - The serious face of the question I think that is the point of view of the designer. I should really have looked at this way. Every program, even the most universal programs are written by a first committee who decided on them, considering a particular constructuion in their head. Therefore, I think you should not talk in terms of 'right' and 'wrong', but of a program being fit for certain constructions or for certain problems; and then at a certain moment, when you put a new type of construction in such a program, you can say: the program was not fit for that construction, or the model was not fit for that problem, and then - of course - it was wrong.
I think it is better not to speack about right or wrong.

KLEMENT - You see, we are speaking here about statics, and static is a very reliable thing if you compare it with other parts of the technical calculation. When I was in charge of a computer center, I had a lot to do with tunnel calculations and designing boilers of a size which have never been built before. The mathematical models you do are much more away than models in static problems, but by means of a mathematical model, you get at least a rule from which you can, afterwards, find in what way results are away. Before, people had calculated boilers with two days manual calculation. But, if this grows up to very high dimensions, they found that you must take mathematical models, so the computer gives a possibility to have at least a scale for comparing the results afterwards and to make better mathematic models in the future. Therefore, I think we cannot say "results of a calculation are wrong" if they are not related to their building. I say: "all our calculations are only calculations to get the correct dimensions", but nobody believe at the stresses you have calculated. You will always find that the stresses are very different from your calculated stress.

DUTERTRE - I perfectly agree. What I menat when I got the first sentence by Prof. Newmark can be applied to any model in computing, providing that the program does work right. Now, when I meant the "erratic error" is from a program error, but sure, when you build a model, the model is wrong, it is always wrong but it is consistent, it is a way to move.

BLAUWENDRAAD - There is time for other questions, if you like, Prof. Werner.

WERNER - Something about the errors : I think we are talking now about the program errors - errors in the source code - but I think there are many other possibilities of errors. First, not the errors between the user ' s han book and the code. If you change the code, it is often forgotten to change the corresponding item in the user's handbook; for instance, the description input is often not in direct relation with the code itself. The second, we detected some errors in the standard itself. The standards often are proved by normal applica tions. When you are writing a program, you must take into account a broader area, and you have to program up these standards and often in these standards it is not thought about the various applications, and sometimes there are errors in the standard itself.

BLAUWENDRAAD - I think you are raising a problem on which it is worthwhile to have a colloquium as we had here for three days. You can feel that the recent building codes of practice have not been written with the use of computer in mind. So, you even have to restructure them at all, taking use of the modern possibilities of decision tables and things like that.
I think it is a rather big problem.
Is there any other in the audience who likes to comment on this subject or on another one ?

ADLER - I am busy in Switzerland in an office of consulting engineers. I had to point this, so that you see clearly that I am just a user of software and not a producer. It is 100% clear for me that we must check all programs we get, and it is on our own responsibility, somehow I do not feel it so good. Software men say in a strict way they are not responsible for what they give out. I do not know. I would like to here someone else about this point. I would like to know whether there are some other profession men who dispense themselves from their job or work they sell or not.

UHERKOVICH - I think in many countries this label is not valid, legally not va lid. You can make ten labels " I am not responsible"; in the real case, you will be.

PFAFFINGER - In the contract, you can exclude your liability. If it comes to a case and the lawyers can prove your gross neglecture, you will be liable, that is the case, but usually the lawyers try to exclude everything.

HAAS - I have a question which has something to do with money, because we cannot check the program so that we weed all mistakes out. Such program would become very expensive and almost nobody could afford its use, because

we have not so many applications of our standards that we could consider, let me say, 100 times, than we can put such an effort in our programs and we can check them to a higher degree as we do now.

DEPREZ - I think there are two kinds of responsibilities: one kind of responsibility is that of means and the other is that of results. The consulting engineer has the responsability for results; he must provide a good design. The responsibility for data processing center is that of means; it must be careful using the program and it must supply the engineer with good documentation as well. In the lawyer's courts, there is difference from country to country. In some countries, you can limit your responsibility to results, in others - like in France and Belgium - you can never limit your responsibility to results, but for your responsibility in means you can always do it, because this kind of responsibility does not engage the results themselves. You are responsible only for negligence or not good work. And this is why I think it is important to make engineers and software centre know correctly what is their own responsibility.

BLAUWENDRAAD - If I may interrupt, you have said before that you see no responsibility for the data processing center about the results. But when the user has no knowledge of your program, you said he should go to some advanced consulting engineer. Who tells the way he should do it ? For the everyday practice, I can understand that the user will have knowledge enough and so the data processing center needs no responsibility, but what about a very small number of advanced programs we use ?

ALCOCK - Well, what about the small beam or something ? The problem is almost completely automated. You have to put the overall dimensions of the building and the computer output and the drawing out come . That situation is with us now and it means that we, engineers, are actually using that. He is legally responsible, because he is the engineer. But he may not have witness; his bureau has probably no witness. There is a third part involved and clearly the responsibility concerns people who actually produce software. For this reason, the only hope of ever clearing the matter up is that the inside is exposed to those who are able to read the signs.

VOS - We really must distinguish between liability, which can come to court, and responsibility, which is used in many senses here, also in the sense of engineering ethics, moral responsibility. Thinking of design practice and what does the designer : when you are really in doubt of using a certain program, you only have to use a computer program and just look if the problems you have now, have been solved. I think this may be a good practice.

HAAS - One word on separating the responsibility in legal and moral responsibility. Of course, each software producer should feel responsible in this moral kind of responsibility and should give correct programs as quickly as possible.

BLAUWENDRAAD - Thank you. We would close now.
In private discussions, you can go on.
Thank you very much for your contributions . You made my job very easy .