E.W. DIJKSTRA

Copenhagen, 20th March, 1970
EWD284-0

As I came to the conference with the explicit intention to submit my considerations, as laid down in EWD249, to the judgement of the other members, I feel it my duty to summarize, what "messsages" I received.

To start with I would like to express my appreciation for the care with which, on the whole, my "Notes on Structured Programming" - despite their length - have been read (at least by those who received them in time). Trying to describe the overall reaction to the document, I feel tempted to say something like "Well, all you have written down strikes us as perfectly sound, we are glad that you have taken the care to write all this down, that you have taken the trouble to sort out a number of programmer's habits we are all so used to that we are hardly aware of them any longer, but what do you do with it?" In other words, what I have written down seems more a set of boundary conditions to be imposed upon programmer's behaviour, than a "pattern of behaviour" with the degree of completeness such that it could be followed. This lack of completeness does not bother me, it just means that I still may have a far way to go; I interpret the general agreement as an encouragement, as an indication that I am heading in good direction. I am bothered, however, by the fact that I got rather little explicit indication of the nature of my story's incompletion; from the remarks made I have not deduced (at least up till now) very clear indications as to into which direction I am to proceed. To avoid misunderstanding: for my being bothered, no committee member could ever be blamed except myself.

Now, what have I learned in these days? Well, quite a few things.

1) I was thrilled by Tony Hoare's remark that quite a lot of the imperative descriptions occurring in my programs could be re-phrased as assertions valid after the execution of the action thus described. This certainly applies to all idempotent actions ("S" being equivalent to "S; S"), what might give a handle to recovery problems.

2) Christopher Strachey told me about his efforts to prove the equivalence of the two programs, which could be derived from each other by inverting a repetition clause with a conditional clause or vice versa. It was nice to hear that

a) in that proof the concept of "idempotency" played a role (see above)

b) that the proof was far from trivial.

His remarks have confirmed my opinion that my attention to "sequencing" was fully justified from the point of view of "how do we understand a program?"

3) Any reader of the Notes, who is familiar with my past experience, must have wondered why recursion had been left out of the pearl-picture. The reason was, that I could not fit it in. Now, thanks to Tony Hoare (and an excellent lunch provided by Regnecentralen), I see the cause: I tried to fit it in in the wrong way. I associated the algorithmic refinement as given in lower pearls and the mechanism of recursion both with the subroutine mechanism, and as a result I was wondering whether I should try to give a manageable meaning to some sort of dynamic necklaces, a recursive call being represented by a some sort of pearl generating concept (an "oyster" say), generating under itself as many copies as the dynamic depth of the recursion. These efforts have been abandoned, as they resulted into a most unconvincing mess. But the effort itself was wrong: I had forgotten that the pearls are associated with levels of abstraction and the different incarnations of a recursive routine are most definitely to be understood at the same semantic level. My conclusion is that recursion should be treated inside the pearl, rather like the repetition clause.

4) Before I came to this conference I had the vague feeling that sequencing patterns (such as recursion, co-routines etc.) and "outlaying" patterns (such as concatenation of data structures) deserve more detailed attention. This feeling is no longer vague.

5) I have been defending a less static view of what should be considered as "the system". Mainly thanks to Doug. Ross I now see that I have been somewhat of a coward, a victim of my past experience and that what I envisaged was only timid first step - and what is more: an unsufficient one. I had envisaged "a translator" which would be called with "the source context" as an implicit parameter, but the envisaged form in which this parameter could influence the translator's activity was a rather rigid one, somewhat like a name list of library procedures. I now see that lower pearls in the context should be able to influence the translator's activity much deeper, such that lower pearls can introduce the repetition clauses and outlaying clauses. In a dynamically growing system my view of the translator

was still as static as ever!

6) I should mention the many remarks related to my "over linearization" as reflected in the linear necklace structure. They have had hardly any influence on my thinking. The decision to restrict myself (at least for sequential programs) to linear necklaces was taken after long pondering; if I then had taken the trouble to make these ponderings explicit and had this been included in the Notes, the remarks would probably not have been made. Personally, I now feel this as an omission in the manuscript; I do not know whether I still can repair it.

7) I have asked very explicitly about the members' opinions as to feasibility of such an approach in the case of very large programs. Two members have given a reasonably strong affirmative answer, viz. Doug. McIlroy and Doug. Ross. The latter did so, as he recognized it in his own work (or the other way round), the first did so conditionally, in the line of "if method, tool and target, although the gap between the latter two may be wide and the method may be elaborate, are reasonably well established". He considered operating systems as outside this class; for me personally, this was a curious example, as the Notes are a direct consequence of my latest large experience, viz. the design of an operating system. Apparently he has been exposed to rather different operating systems.

Doubts have been expressed as well, but they have not disturbed my faith - and me being far from thick-skinned, this is no empty observation. They were

a) many different problems have been attacked which did not admit such a nicely structured solution. My answer to this is "Was the messy solution well feasible?" Looking at OS360, I have my doubts.

b) we have made a series of translators and had always to redo 90% of the job. Answer "the contents of the family of programs is closely related to the structuring and, apparently, in this case the structuring has been insufficiently tailored to the desired degrees of freedom. Doug. Ross comes with figures like 80% constant".

c) I don't work that way. Answer "Neither did I before I trained myself to do so."