Tripreport E.W.Dijkstra: NATO Summer School Marktoberdorf 1975.

It was a hot summer school and, therefore, and exhausting one. I left Eindhoven on a Monday evening by train in the pleasant company of Braam, Feijen and Rem of the THE. After a few glasses of wine in each other's company, each of us went into his compartment to sleep. We left the train at seven thirty next morning in Munich, two hours later we were in Marktoberdorf with the rest of the day available for adjustment and final preparations. The breakfast that "Wagons Lits" served us between Augsburg and Munich was not exciting, but apart from that, it is a way of travel that I can heartily recommend.

The audience was a very mixed lot: nearly one hundred participants from twenty different countries. (The speakers --nine from about six different countries-- would turn out to be an equally mixed lot!) And as the course progressed, I realized that such a great variety in origin was responsible for a lot more problems than just language barriers.

Different participants had been subjected in their home countries to quite different educational systems, and, as a result, they had come with widely diverging expectations to Marktoberdorf. At the one end of the educational spectrum is the teaching and training that is primarily oriented towards the transmission of knowledge and very concrete abilities, at the other we have the educational tradition that concentrates on the transmission of insights and understanding. The enumeration of facts that meets the expectations of the students of the former tradition, bores those from the latter; the considerations that the students of the latter tradition find illuminating, exasperate those of the former. I found it impossible to satisfy both types with the same talk.

Besides those general cultural differences, there is the technical problem that in different nations computing science in general, and programming in particular, is quite differently appreciated.

Programming is one of the most difficult branches of applied mathematics because it is also one of the most difficult branches of engineering, and vice versa. To explain this, for instance, to a Frenchman --even if he understand English!-- is close to impossible: the impossibility --let alone the obligation!-- of having a foot firmly planted in both fields of human endeavour is for such a victim of Bourbaki just way beyong his powers of imagination. For quite different reasons it is equally difficult to understand for the pragmatic bit-pusher from the American Mid-West.

A separate word or two must be devoted to German computing science, because there were so many Germans --about one third-- among the participants. German computing science is still very unbalanced. With the intention of catching up and promoting the field, the German federal government has performed a magic act of high-speed foundation of departments of computing science at only God knows how many German universities. The problem left the universities was how to fill all those newly created chairs. The expected happened: most of them were filled either by specialists in automata theory and the like, or by experiecned pragmatists, and new life was breathed into the barren controversy between "theory" and "practice". In between (and hopefully above) those two extremes a number of "true" computing scientist should hold the field together. The political decision that those true computing scientists should present (and represent) a unified view of the subject --and should do so in a unified, German terminology-- became disastrous, when the technical mistake was made of

choosing the concepts of ALGOL 68 as the fundamental ones on which the subject should be further developed. But one cannot base a scientific discipline on confusion, and, consequently, we have seen preciously little German development of computing science since then. (The German adoptation of ALGOL 68 has had a similarly paralyzing effect upon the Germans, as the Russian decision in the late sixties to develop as their next national computer series a bit-compatible copy of the IBM 360 --the greatest American victory in the cold war!--.) And with the mental blinkers that only allow operational definitions of programming languages, there is not really much that one can do. Either forget about programming languages and return to the more familiar subject of recursive function theory, or focus upon the difficulties of managing complicated implementations of complicated languages on complicating machines; and that is what seems to happen. The separation between theory not needed in practice and practice without supporting theory is again complete, and I hardly dare to ask a German colleague about the progress of his work, afraid that the question will be too painful.

A third problem was caused by the great difference in "professional level" (as could have occurred between participants from a single country). There were participants that identified the problems of computing science with the problems encountered within the four walls of their own institute. I was already familiar with the phenomenon of young computing scientists regarding the existing hardware as a God-given constraint within which computing science should be developed. Now I encountered a few that had gone a step further: also the commercially provided operating systems and the generally supported programming languages were accepted as unquestionable confinement. An alarming development.....

I gave eight lectures on eight (of the nine) working days of the summer school, and they were not an unqualified success. During the first one --on reasoning and pondering-- the sound systems did not function properly. Besides that I felt the a majority of the attendant wanted more concrete material. So I switched quickly to formal semantics for the next three lectures; it was only in the fifth lecture, when I showed and discussed four little programs, that that section of the audience felt more satisfied: at last I was showing programs and actually "saying something". The last three lectures were devoted to the on-the-fly garbage collection; I restricted myself to rather coarse-grained interleaving. Some welcomed the confrontation with a new type of problems, but certainly not all of them.

Tony Hoare gave also eight lectures (on each of the first four days two). He used an old collection of transparencies, the lectures were perfect and he was practically the only other speaker who tried to transmit developed theory. (David Gries would do so on the last day as well.) On the whole he reached his audience well.

Bill Wulf (Carnegy Mellon) and Per Brinch Hansen (Cal.Tech.) reported both on their development project (the Hydra system and a pilot model to try out the applicability of Concurrent Pascal, respectively). Both gave eight lectures, and it was a pity that their subject were so similar: sometimes all the details became rather boring and the relative importance of operating system design became overstressed.

Gerhard Seegmüller (Munich) and Andrei Ershov (Novosibirsk) gave together about eight lectures, the first on a Municj project on a machine independent system implementation language, the second on a language/machine independent translation project. I found both projects depressing, the anomalies of PL/I and the IBM/360 having pervaded both of them.

Mike Griffiths (Rennes) and Fritz Bauer (Munich) also talked very much about the same subject: recursive formulations and the massageing thereof in an effort to avoid variables and assignments to them. After having hear Rod Burstall a few times on that subject I could not distinguish any new material, and I am still of the opinion that this line of attack on the programming problem was better justified in the early sixties than it is now.

On the last Saturday morning David Gries (Cornell University) caused a surprise in a one how lecture, when he showed how newly developed techniques for proving the correctness of parallel programs --developed by his Ph.D-student Susan Speer Owicki-- could be applied to the on-the-fly garbage collector. That was nice, and his presentation was very much appreciated (David himself was very grateful for the opportunity of applying his new technique to a more complicated parallel program than he had ever envisaged himself.)

There were a few incidents. On about the third day Koster (Berlin) could stand it no longer that none of the speakers cared to stick to the ALGOL 68 dogmata, worse even: dared to challenge them, and during the discussion he read aloud a sort of manifesto in defense of ALGOL 68, that was amazingly aggressive, full of arguments "ad hominem" --Tony and me in particular--. I was baffled: is this the style of "scientific" discussions as is practiced nowadays at German universities? (His outburst was all the more surprising because none of the speakers had mentioned ALGOL 68 at all! Even politely ignoring it is apparently already offensive for that dogmatic environment.) On the last day, a Germand staff member from Munich asked all the speakers -- because the subject had not been mentioned-- their opinio about "interactive programming". Did he have vested interests? When nearly all the speakers had given the verdict "insignificant" in one way or another, he got so annoyed that he started a voting procedure among the participants in the hope of getting a more favourable answer!

Thanks to Tony, my visit to Marktoberdorf has personally been very rewarding. On Sunday --our first free day-- we have worked together in the garden of Hotel Sepp. Upon my request he has explained to me the SIMULA "class" --something I could never understand-- and, in the course of that Sunday, we have moulded the notion into something that looks very promising. On the next Thursday --the day of the excursion-- we continued our work and at the end of the afternoon we had designed what had started as the design of a recursive data structure, but suddenly admitted the interpretation of an elephant built from mosquitoes as well. A surprising discovery, the depth of which is --as far as I am concerned-- still unfathomed.

One discussion during a dinner should not remain unmentioned- Bill Wulf raised the question: "If there were a Nobel prize for computing science, what would be the next achievement in our field, worthy of it?" In order to come into the mood, we went through the past and up till 1970 we had no difficulties; the significance of things done since then was much less clear, and, eventually, Bill Wulf's question remained unanswered.... How do we interpret this fact? One possible interpretation is that in computing science all essential discoveries have been made, and that the subject approaches completion, etc., an interpretation that is compatible with the scientific stagnation that various groups seem to display. But I feel, that this interpretation is totally wrong. It may be true that certain lines of research seem to have got stuck, but a number of dead alleys do not imply a dead subject! The next achievement Bill Wulf was asking for might very well take the form of successfully challenging on of our common "tacit assumptions". Von Neumann's "instruction counter" --and the notion

of "a sequential process"-- seems a most likely victim: any workable conceptual framework in which "parallel programming" becomes as meaningless a term as "sequential programming" could be a worthy condidate for computing science's Nobel prize!

* * *

Eight days ago we returned, again by train but this time by daylight. I had told my compatriots how beautiful the ride through the Rhine valley would be; later they told me, that they had enjoyed it very much: I myself had slept! I was very tired when I came home, and my wife ordered that I would take two weeks off, a prescription that I followed gratefully, as the heat wave continued in full vigour. As my spectacles were beginning to hurt me on warm days I ordered new, very light ones ten days before my departure to Marktoberdorf. To my great regret they were not ready yet when we left. I finally got them, nearly a week after my return, two days before the heat wave ended....

Plataanstraat 5
NL-4565 NUENEN
The Netherlands

prof.dr.Edsger W.Dijkstra
Burroughs Research Fellow