# Another misguided effort

Computing science has had very little influence on computing practice, which is mostly performed in a highly unscientific manner. I have been -and still am- particularly worried about the ever-widening gap between computer science and the practices within the computer industry, which -I think- should be our high-technology counterpart "in the real world". It has greatly saddened me to see that industry on the whole persist in a behaviour as if common sense suffices where I know that the techniques of scientific thought are required. My many crusades to remedy the situation, I am sorry to observe but not ashamed to mention, have not been particularly effective (so ineffective, in fact, that one of my Eindhoven colleagues called me a Don Quichote). Obsessed as I am I have frequently raised the topic in industrial environments; the average reaction admits the gap but then argues that computing science has very little to offer to their business. The latter I have never been able to believe, a disbelief I found most embarrassingly confirmed when I heard of the following industrial project.

Inspired by the soaring cost of software development they wanted to do something about the (undeniable) fact that it is often so hard to understand a program written by someone else. To remedy the situation they intended to develop some sort of "algorithm animator" that, as an aid to understanding, would graphically display how the computation evolved in time. Needless to say, the animator would have some sort of conceptual zoom lens that would enable the computation to be viewed at different levels of abstraction.

There are at least three reasons why this project will fail.

(i) Program texts are highly compact deposits of our intellectual labour. In my case the essays I write about programs are about ten times as long as the raw code in which they culminate, a factor that cannot be "explained away" by the verbosity of my prose (which in these essays tends to be terse). In short: the program text is only the top of the iceberg. And it is vain to reconstruct mechanically the iceberg, given only the top.

(ii) Despite what herds of educationists want us to believe, pictures are a most inadequate vehicle for representing the relationships one has to capture when reasoning about programs. (Are pictures ever adequate? To quote Morris Kline: "But the pictures are not the subject matter of geometry (!) and we are not permitted to reason from them. It is true that most people including mathematicians, lean upon these pictures as a crutch and find themselves unable to walk when the crutch is removed.")

You can take the experiment when lecturing. Draw a few pictures and the dumber students, who a moment ago had no idea what you were talking about suddenly nod "understandingly". In fact they still have not the foggiest notion, but they nod "understandingly", recognizing the picture —boxes with arrows—, the visual familiarity of which gives them a superficial but false sense of security. (Furthermore pictures are usually overspecific, even in geometry: you cannot draw a triangle without making it obtuse or not, even if the distinction is irrelevant for the argument.)

(iii) You can only animate individual computations, whereas the program is about the set of all computations possible under control of it. It is impossible to come to grips with a huge set by looking at some of its (hopefully!) representative elements. The project has all the well-documented and well-understood intrinsic inadequacy of operational reasoning as built-in feature.

* * *

Is it really true that computing science has hardly anything to offer to the computer industry? I shared my exposure to this project with a colleague, and we both saw instantaneously that this project would fail. Only God knows how many million dollars computing science could have saved for them.

Such exposures are always very embarrassing, for how does one explain politely and without hurting how utterly foolish the project is? I always want to know where people undertaking such projects have been educated as the answer could enable me to strike on or more Universities from my list. But I have become very hesitant to ask that question, for too often the people had not been educated at all. The exposure left my colleague wondering whether such people really believe that such projects make sense. He found it hard to imagine and began to assume that only during office hours they think and communicate in the slogan-ridden Doublespeak we heard, because everybody does it and they have been made to believe that it is part of the job. There seems no way out: they only have the choice between incompetence and dishonesty. It is terrible.

But, please, don't blame us when the computer industry collapses.

Austin, 21 April 1985

prof.dr. Edsger W. Dijkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712-1188
United States of America