Letter on Parnas' View of Dijkstra vs. deMillo, Lipton and Perlis

H. Kilov

I disagree with Dr. Parnas' view in the October 1978 SEN (3 4, pp. 20) that there are two kinds of programs -- useful ones (the result of software engineering) and, let us say, textbook ones (intended to be understood by someone else). That the latter may be useful is shown by Dijkstra (updating a sequential file, string pattern matching, ...) and others. Of course, often good textbook programs seem too simplified -- maybe, because we are used to the need to adapt to and sometimes to fight with the existing software systems. But then the software systems are to blame!

I think the correct analogy is not between software engineering and mechanical engineering, but rather between software engineering and applied mathematics. (The essence of the latter is very well described in the book "Applied Mathematics: Subject, Method, Approaches" by Blekhman, Myshkis, and Panovko, published [in Russian] in Kiev in 1976.)

Finally, I think sometimes it is quite possible to "test" theorems in mathematics -- i.e., to find counterexamples to them. (In fact, journal mathematics proofs can contain errors as well!)

Karl Marx Str. 75-13
Riga 11 USSR

H. Kilov

An Alternative to Structured Programming: Syndicate Programming

Recent work [1,2,3] has pointed out the need for the organization of programming teams and for efforts to improve software quality and reliability.

The idea of Chief Programmer Teams, although well thought out, tries to induce an artificial and unproven organizational structure in the programming effort. It seems, however, that a structure should indeed be used to curb the amount of poor quality being produced nowadays. Careful sociological and anthropological studies show that two social groups have reached exceptional organizational efficiency, the Papuan tribes of New Guinea [4] and East-coast organized underground illegal organizations [5]. In this paper we examine only the second, as the high efficiency of the first seems to be somehow related to the low demands of their prehistoric life -- a situation not at all similar to that of the world of software (except for the use of prehistoric tools).

(An Alternative to Structured Programming, Continued)

Programmer Teams

The head of the programmer team will be called the Godfather, or simply, the Don. He is assisted by a backup programmer called the consigliere and by two secretary programmers, each called "capo di programazione". Unlike the IBM scheme, the Godfather makes essentially all important decisions. He makes the appropriate offers of help in return for code to the programming team through the capi, using careful wording so that these offers of guidance cannot be refused. Sitting between the programming staff and the programming team is a group of strong programmers called the enforcers. They help maintain the team discipline and help the programmers (also known as soldiers) to accomplish their tasks. Also, unlike the IBM situation, the documentation is simply carried by one of the capi who keeps only the good runs and promptly destroys the rest. Programmers are supposed to know as little as possible about other programmers' jobs, and silence is considered to be a golden rule. Breaking the silence rule and collaboration with other programmer teams are both capital offenses, causing the enforcers to initiate proceedings for immediate termination of the culprits from the group. Should the Godfather not be able to carry out his duties, the consigliere will usually take his place unless the capi can convince him otherwise, using appropriate arguments.

When a system to be constructed is too large for this approach, a team of teams may be assembled. Conflicts are solved in a meeting of Godfathers (usually called a family meeting).

It is expected that the present setup will help clean up programming messes such as that of the off-track betting system in New York state. (Additionally, it is expected that the betting itself will increase heavily.)

Further concepts of syndicate programming are known, but as yet undocumented. These include the Family Theorem, showing that the instructions FORMAT, EQUIVALENCE, and CALL are really unnecessary, and proposing the new concept of sideways programming.

It is important to remark, however, that syndicate teams provide a strictly top-down organization.

(With apologies to E. W. Dijkstra, IBM, and M. Puzo)

References:

1. Mills
2. Baker
3. Mills and Baker
4. Margaret Mead, Coming of Age in Papua
5. M. Puzo, The Godfather