To be presented at the International
Federation of Information Processing
Working Conference on Command
Languages, Lund, Sweden,
July 29-August 2, 1974

SOME THOUGHTS ON STANDARDIZATION

David F. Stevens

January 25, 1974

Prepared for the U. S. Atomic Energy Commission
under Contract W-7405-ENG-48

32

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

LBL -2499

SOME THOUGHTS ON STANDARDIZATION *

David F. Stevens

Lawrence Berkeley Laboratory
University of California
Berkeley, California

January 25, 1974

ABSTRACT

A position paper against early standardization of Command Languages.

* Work done under the auspices of the U.S. Atomic Energy Commission

The term "standardization", as it is currently being applied in the computing community, means "the abolition of variants to promote the portability of programs". Pressure for standardization is an understandable consequence of the growing investment in working programs and data bases. The desire for "standardization" (which promotes stability) thus finds itself opposed by the desire for "development" (which entails change). The major philosophical question which must be resolved when considering whether an area is ripe for standarization or no is: Which is more important for the time being: stability or further development?

There is thus a heavy burden on the proponents of standardization to demonstrate that the present level of achievement is so beautiful as to warrant preservation, or that the present flurry of activity is so chaotic as to warrant suppression. (The burden rests with the standardizers, of course, because all of the natural pressures work for change.)

Unfortunately, there are relatively few situations where one might argue for the preservation of the status quo. Only if everything works, and works as one would expect it to, even when one is trying something new, is such an argument at all tempting. In other words, unless the present version (of whatever it is) is natural, consistent, convenient, and accomplishes that purpose at hand, there is constant outcry for improvement and adjustment.

Let us consider the case of the Command Language. There may, indeed, be some command languages which accomplish their designed purposes, but the very existence of this conference suggests that the precise nature of those purposes is in some doubt. Is there any command language which is natural, consistent, and convenient? Can there be a single language which seems natural to all of its various classes of users? Constructs which seem natural to a scientist may seem less so to a banker; conventions which a libraian finds convenient may confuse a sales clerk.

The accepted understanding of the proper scope of the command language was broadened appreciably with the introduction of JCL; is this new understanding a workable one? Does its very breadth make it an unsuitable language? Such highly respected IBMers as F.T. Baker and H.D. Mills can say (in the December 73 Datamation) "the functions of (JCL) are impressive, but they are called into play by language forms that require much study, experience, and sustained mental effort to use effectively".

Or, conversely, is the present scope of command languages still too narrow? Should the possible influence of radically different hardware be considered? Can current languages be extended naturally to handle multiple processes, increasingly complex memory heirarchies, more sophisticated data base organization?

Until these, and the other questions discussed at this conference relating to the proper scope and function of command languages, can be settled, standardization efforts are premature. Most end users would probably agree that the standardization of Fortran was a good and useful effort. But that was a different situation: A common base had spawned a Babel of dialects, all claiming the name of Fortran. Although different computers do have different command languages, they differ in obvious ways. There is not the confusion caused by identical syntax with different semantics, or by similar semantical extensions leading to different syntax. The situation is in a state of flux, but it is healthy ferment, aimed at defining and refining the concept of Command Languages. To standardize now would be to settle for funny tasting grape juice instead of producing decent wine.

Before attempting to develop a standard Command Language we need first, to define those aspects of job control which can, or should be standardized, and second, to establish human-oriented schemata for expressing such a language. I use the plural advisedly, for fluency with the command language is essential to effective use of the computer; the users of computers are coming more and more frequently from the ranks of nonprogrammers; and non-programmers can deal with the computer best in their own (natural) languages.

The computer is the subject of a great deal of bad press these days. As long as the mysteries of the computer are protected by the assembled high-priests of the great gods Fortran, JCL, APL, BAL, etc., this bad press will continue to grow. The only way out is to let the common man speak to the computer in his own language. We must not begin to standardize Comman Languages until we can begin to approach this goal.

LEGAL NOTICE

TECHNICAL INFORMATION DIVISION
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720