PROGRAM CONTROL OF OPERATING SYSTEMS

SØREN LAUSEN

Abstract.

The traditional job control language becomes superfluous if the existing programming languages are extended slightly. Such extensions also allow drivers and operating systems to be programmed entirely in high level languages. Ultimately, we may see machine independent operating systems. A framework is presented for an extendable operating system which allows a simple, uniform implementation of these language extensions.

1. Introduction.

This paper is concerned with what shall be called the control language, meaning the complete interface between the user and the operating system of a computer. This control language includes both statements of the job control language and statements of the programming language proper. The purpose of the paper is to show the advantages of an approach which eliminates the job control language by augmenting the ordinary programming languages with a few inter-process communication primitives.

In more detail, in present day systems the full interface between user and operating system has two parts. One part consists of external control statements of the job control language. They treat the compilers and object programs as procedures, and leave little or no possibility for the user program to modify the flow of control. The other part consists of the statements of the ordinary programming language for opening files and for calling input and output. In the approach proposed, the second part is extended so as to make the first part superfluous. This requires that the ordinary programming languages are extended with inter-process communication primitives and more complicated procedures for file definition, file creation, calling of translators, etc. External control statements as used to-day would then have to be expressed by an ordinary user program. Loops and conditions are then easily expressed, and we may think of the user program as controlling the operating system.

With an appropriate design of the extended programming language it will also become possible to handle peripheral devices in ways not anti-
It seems feasible to introduce the communication primitives in all the major programming languages in a simple and uniform way, and so that mutual job security is maintained.

The advantages of this approach are, first, that the ordinary programming languages will become useful over a wider range of applications, second that the user control of the operating system will become more natural, and third that the control language will become as clearly defined as present day programming languages and will admit a similar degree of standardization.

Solutions to many of the technical problems are outlined in the paper. A set of communication primitives and a set of basic operating system features are proposed in Section 5. One major technical problem remains as discussed in Section 5.5: The design of a strategy for working store allocation which is efficient, flexible, and allows the sharing needed for the communication primitives.

The incentive for this paper was the Nordic Working Conference on Basic Software, held in Copenhagen on October 1971 [3]. A main theme of this conference was the possibility for constructing a machine independent control language for operating systems. Starting from the existing major operating systems it seems tempting to propose a common job control language, including conditions, loops, and procedures (macros), and with an Algol-like appearance. In the author's opinion this approach would hide the fundamental problems of the field (besides adding to the Tower of Babel). I therefore submitted the present proposal, in a preliminary form, to the conference. Later I have found that the CODASYL Data Base Task Group follows a similar approach in another field [1, 2].

2. Interface between user and operating system.

The control language is defined as the complete interface between user and operating system. The commands of a control language may be classified roughly as follows:

External control commands specify data files, calls of translators, resources for the job step, etc. They are in many systems called "control cards" and they may be the only control commands of which the ordinary user is aware. The external control commands are interpreted by the operating system and form the external control language (sometimes called the command language or the job control language).