DEVELOPMENT OF EDUCATIONAL SOFTWARE USING THE DEC PDP-11

D.D. Cowan, P.H. Dirksen, J.W. Graham, J.W. Welch
Department of Computer Science
University of Waterloo

ABSTRACT

During the past two years, the Waterloo Foundation for the Advancement of Computing (WATFAC) and the University of Waterloo have been developing educational software in a number of projects using DEC PDP-11 mini-computers. These projects include the development of WATFOR-11 (a load-and-go FORTRAN compiler), WATBOL-11 (a load-and-go COBOL compiler), and WIDJET (a student editing and job-entry system). This paper presents a survey of these projects and discusses several problems encountered during their implementation.

INTRODUCTION

Many high schools, community colleges, junior colleges and similar types of educational institutions are developing and teaching computer-related courses which require the student to do programming exercises. In order to facilitate use of the computer and to allow large numbers of students to write and successfully execute programs, an appropriate computing environment should be available.

Programs written by students usually have a short lifetime and normally are not used to produce more than four or five separate calculations. Since the programs are so short-lived, it is important that the period between conception and actual operation of the program be minimized. Specifically, the computer system available should have a number of properties which are directed to student use. The software and hardware should detect common errors and diagnose them so that the student may quickly make corrections and put the program into production. It should be easy to enter program text and put the programs into execution, and an attempt should be made to minimize the amount of program text to be entered by a student.

The provision of an adequate computer system to support these courses has generally been expensive and often is beyond the limited budget of many educational institutions. The development of educational software based on mini-computers should provide a solution to this problem.

During the past two years the Waterloo Foundation for the Advancement of Computing (WATFAC) and the University of Waterloo jointly have been developing educational software which forms a solid base for an inexpensive educational computer system based on the DEC PDP-11 series of computers. These software systems include:
(i) WATFOR-11: a load-and-go FORTRAN compiler (Cowan, 1975a) which has a high compilation rate and excellent error diagnostics both at compile- and run-time.
(ii) WATBOL-11: a COBOL compiler (Cowan, 1975b) with characteristics similar to WATFOR-11.
(iii)WIDJET: an editing and job-entry system (Graham, 1975) which is tailored to student use.

In this paper a description of these software systems and a discussion of some implementation problems are presented.

LOAD-AND-GO COMPILATION

A load-and-go compiler normally translates a source language program in a language such as FORTRAN or COBOL into executable machine instructions or equivalent forms which are stored directly in the computer memory. Once stored in the memory the translated program is usually placed in execution. Consequently, load-and-go compilers have substantially less overhead compared to the more traditional method of compilation in which the object code is written as a file to be subsequently linked with other files of object code. The size of a program which can be compiled and executed by load-and-go compilers is bounded by the amount of memory available to store the translated program and other relevant data such as symbol tables.

Load-and-go compilers are often used in environments where the programs are generally small, and where a large number of such programs are run. These conditions are generally found in educational and research environments. Historically, load-and-go compilers have been designed to run on large-scale computing systems (Cress, 1969; Rosen, 1965; Shantz, 1967; WATBOL, 1972). It has been only recently that load-and-go compilers have generally been implemented on mini-computers. Two such compilers are WATFOR-11 and WATBOL-11.

WATFOR-11 (FORTRAN) and WATBOL-11 (COBOL) are modelled after the highly successful WATFOR/WATFIV (Cress, 1969) and WATBOL (WATBOL, 1972) compilers, which were developed at the University of Waterloo for the IBM 7040, 360 and 370 series of computers. WATFOR-11 can compile programs at a rate of over 1000 statements per minute (PDP 11/45). At this rate, there is still a substantial amount of processing capacity available for multi-tasking. In addition to being able to compile at a fast rate, these compilers provide excellent error diagnostics. Considerable effort has been expended so that appropriate error messages can be printed during either the execution or compilation of a program. Because of their diagnostic capability, many people have termed these compilers 'debugging compilers'. WATFOR-11, for example, can issue over a hundred different error messages. WATBOL-11 has at least double that number of error messages.

There are a number of differences in the way these load-and-go compilers are designed for mini-computers rather than for larger systems. While main memory is a limited resource on both kinds of systems, it is usually more restricted on mini-computers. This shortage of memory has influenced our design in a number of major ways.

Both WATFOR-11 and WATBOL-11 make extensive use of overlay facilities. The PDP-11 architecture permits addressing over only 32K words. This is insufficient space in which to store either the WATFOR-11 or WATBOL-11 compilers. The original WATFOR/WATFIV compilers do not use overlay facilities. The original WATBOL compiler has an overlay facility as an integral part of its operation. Both WATFOR-11 and WATBOL-11 can be generated in a number of overlay configurations. As the compilers are configured to use less memory the speed naturally decreases because of the increased number of direct-access operations required to load more overlay segments into main memory.

Both WATFOR and WATBOL have facilities whereby error messages are printed as text. The size of a message is approximately fifty characters. In WATFOR-11 and WATBOL-11, error-message codes are used. For example, the code "UV-00 XXX" indicates that a variable XXX was used with an undefined value during the execution of a program. The elimination of textual error messages occurred because of the limited amount of memory.

The speed of the mini-computer versions is comparable to the speed of larger systems. Typically, the rate of job throughput is governed by the slower of the input (card reader) or the output (printer) devices. Both systems have been operated in "cafeteria style" situations where students queue for a card reader, read the programs into the computer, and proceed directly to a printer to tear their own listings from the printer. In the larger systems, card readers and printers process at approximately 1000 cards or lines per minute. Although peripherals with these speeds exist for mini-computers, it is more usual to configure them with devices whose speeds are slower (i.e., about 300 lines or cards per minute). In cases such as this, the load-and-go compilers can be configured so that the minimum amount of memory is used. Even when operating with the maximum number of overlays, the compilers are able to operate the slower devices at their maximum speed.

Our experience with the PDP-11 has convinced us that load-and-go compilers are very practical on mini-computers. Two such compilers have been produced and are being successfully used. The major problem to overcome when implementing these kinds of compilers is the management of limited memory. This has been accomplished in WATFOR-11 and WATBOL-11 by using overlay facilities without a serious degradation in performance.

EDITING

Another area of research has been student-editing and job-entry systems. A simple editor, the WIDJET system, was implemented and augmented with commands which allow submission of jobs for execution. Two basic configurations were used. In the first, jobs were passed to the WATFOR-11 compiler for execution and the WATFOR-11 compiler returned a listing which could be inspected at the appropriate terminal. In the second configuration, jobs are passed to another computer (an IBM 370/158) over a bisynchronous communications line in order to be executed and the listings are returned to the PDP-11.

The motivation for developing the editing system was provided by the rising cost of cards, paper, and the unit-record equipment which processes these media. We are interested in investigating the economics of providing equivalent service to a relatively large student population using cathode-ray terminals. The students would normally be first-year computer science students with little or no computer experience.

One of the prime considerations was that students could use the editor to write their first computer program after their first lecture. We feel strongly that novice students should run programs immediately rather than possibly losing interest during several "theoretical" lectures. The command language was designed to be very simple to learn. Basically, the editor is a context editor whose commands apply to a current line. The current line may be changed by repositioning commands. The original editor could only move the current line pointer forward in a file; in order to move backward the pointer had to be repositioned at the beginning of the file. In response to student suggestions, the editor was modified so the pointer could be moved either forward or backward.

A public file facility was also implemented in WIDJET. This allowed the instructors to leave messages about the conduct of the programming laboratory and to even distribute the problem sets on-line. It was also possible to place portions of a program in the public file so that students could move this module into their own program. For example, this technique often saves the student from having to copy large portions of a COBOL DIVISION.

The editor was implemented as a re-entrant program to process commands from a single terminal. When a number of editors is run simultaneously, a single copy of the executable code is present. Each editor has approximately 1.5K bytes of data allocated to it. In the fall term of 1975, 32 editors were being run simultaneously on a PDP 11/50 with 80K words of memory.

Over 600 students used this system to run their assignments. Our plans are to expand this to 64 editors. A PDP 11/10 has been configured to support 8 editors. This system, however, has not been run in a production environment.

Although a number of 'teething' problems were encountered, the system has been successfully run during two school terms. Two of the future applications of the system are:
(i) To present mobile "computer-science days" at a number of high schools in Ontario. Computer-science days have been used at the University of Waterloo to introduce high school students and teachers to computing. By using a mini-computer configuration, we hope to be able to educate small groups of users at a reasonable cost in their own location.
(ii) To teach structured-programming techniques to business programmers.

We are presently developing some of the packages required by these groups. Using our load-and-go compilers, students are able to develop programs quickly since they are provided with fast turnaround and excellent error diagnostics. Textual materials are currently being prepared for courses using these compilers. The WIDJET system provides a method to use these compilers without using card readers or printers.

In summary, it is hoped that these developments will introduce computing to a large audience. Of course, this can only occur if the cost of this type of computing is reasonable.

FUTURE PLANS

Our overall goal is to develop a comprehensive package of program materials for a number of educational groups. Increasingly, high schools, community colleges, and junior colleges are developing computer-related courses and educational programs and are initiating more and more computer-based applications. All groups are similar in that they normally cannot afford large computer installations or the manpower to support them. Instead, they require low-cost hardware configurations and the software packages to support their operation.

Relatively inexpensive computer installations can be configured using mini-computers. We are in the process of developing software to run on PDP-11 hardware. To date, the major developments have been two load-and-go compilers and an editor. We are in the process of developing textual material for courses which use these systems. We are also developing a portable mini-computer configuration. This portable computer could be shared by a number of remote users or it could be used for demonstration purposes.

More information regarding these and other software may be obtained by writing WATFAC at the following address: WATFAC, Box 803, Waterloo, Ontario, Canada.

REFERENCES

Cowan 1975a — Cowan, D.D., et al; WATFOR-11 and WATFOR-11S; 1975 Spring DECUS Symposium.

Cowan 1975b — Cowan, D.D., et al; WATBOL-11; 1975 Fall DECUS Symposium.

Cress 1969 — Cress, P.H., Dirksen, P.H., Ward, S.J.; WATFIV Implementation Guide; Computing Centre, University of Waterloo, 1969.

Graham 1975 — Graham, J.W., et al; WIDJET: A Student Editing System; 1975 Fall DECUS Symposium.

Rosen 1965 — Rosen, S., et al; PUFFT - The Purdue University Fast FORTRAN Translator; CACM 8, 11 (Nov. 1965), pp. 661-666.

Shantz 1967 — Shantz, P.W., et al; WATFOR - The University of Waterloo FORTRAN IV Compiler; CACM 10,1, pp. 41-44 (January 1967).

WATBOL 1972 — WATBOL Implementation Guide; Computing Centre, University of Waterloo, 1972.