SOFTWARE REVIEWS

Editor: Paul Oman
Computer Science Dept.
University of Idaho
c/o 1350 NW 19th St.
Corvallis, OR 97330
Compmail+: p.oman

"New compilers and environments and favorable early experience presage an Ada boom," wrote Ware Myers in the March Computer *("Ada: First Users Pleased; Prospective Users Still Hesitant," pp. 68-73). Because of Ada's potential boom, we thought readers would like a closer look at two of these new Ada systems. We review Telesoft's TeleGen2, for the VAX line of computers, and Alsys's Ada World, for the IBM PC AT. While versions of Telesoft Ada have been around for a few years, the Alsys product is a recent entry. And Alsys's president, Jean Ichbiah, led the Ada development team.*

Telesoft Ada revisited

Paul Oman,
Software Reviews Editor

In May 1985, we ran a review by David Rudd on Telesoft Ada Version 1.3. Since then, Telesoft's Ada programming environment has undergone multiple revisions and improved substantially. In fact, the revised version even carries a new name: TeleGen2.

Early versions of Telesoft's Ada were inordinately slow; for example, the welcome program, which consists of one Put_Line operation, took over a minute to compile and 32 seconds to execute! However, upgrades to version 3.*xx* dramatically improved both compilation and execution times to such an extent that another look at the product is worthwhile.

Product overview. The company touts the product as its second-generation development system for Ada; it is indeed a new generation, distinct from the company's earlier Ada system. It complies fully with the ANSI/MIL-STD-1815A specifications and is validated under ACVC test suite Version 1.7.

The TeleGen2 Ada compilation system includes

- the primary compilation system: compiler, binder, library management system, TeleQuiz system, and set of manuals;
- a collection of Ada language tools: cross-reference lister, source-dependency lister, and source-code formatter;
- an Ada source-level debugger;
- a global optimizer; and
- an Ada profiler for monitoring programs and subprograms during execution.

The language tools, source debugger, optimizer, and profiler are development tools now included in the price of the host compilation system. The manual set includes an installation guide, compiler users manual, TeleQuiz manual, and a TeleGen2 users guide.

System structure. The compilation system's major components are the three-pass compiler, the library management system, and the binder. A source-code editor is not included in the package. Output from the compiler is a VMS-formatted object module and, optionally, an assembly listing. The compiler generates VMS-compatible object modules so you use the VMS linker and loader to generate VAX native code.

All object code, intermediate code, dependency information, and the run-time support code is stored in Ada program libraries maintained by the TeleGen2 library manager. We found the library manager awkward to use, mostly because this portion of the user guide reads like a reference manual rather than a guide.

Also, the primitives for library manipulation are restricted to the /Create, /Show, /Copy, /Move, and /Delete options on the library manager call. Contrast this with the Alsys Ada library-management routines discussed in the accompanying review.

The TeleGen2 binder is used to create executable images from object-code files by linking them with other compilable units residing in the same or different libraries. The binder works by creating a linker command file and invoking the VMS linker with that command file. The resulting file is a VMS-executable image that can be passed to the VMS loader for execution.

Evaluation. Aside from creating and manipulating user-defined Ada sublibraries, we had virtually no problems using TeleGen2. The installation procedure is trivial if VMS has the appropriate page allocations for the working-set extent size (4500) and enough global pages (8000). If not, a Sysgen is necessary to reset the VMS parameters. Com-

IN BRIEF

Telesoft Ada has been significantly improved. The system, now called TeleGen2, varies in price from $1200 for the VAXStation II to $53,000 for the industrial version on the VAX 8800. The university discount price for the VAX 11/780 compilation system running under VMS is $3900. Cross-compilation systems are available for the entire line of VAX computers as well as for MC680*x*0-based maxhines. VMS should be set up so you have 4500 pages for the working-set extent size and 8000 global pages. We tested Version 3.10 under VMS Version 4.5 on a VAX 11/780. **RS 11**

Alsys Ada World is a programming environment for the IBM PC AT. The system costs $3000 and includes the compiler and a Profit Systems AT Power 4M-byte expansion board. We reviewed Version 1.2 of the compiler on an IBM PC AT with 640K bytes of RAM plus 4M bytes of extended RAM and a 30M-byte hard disk. **RS 12** piling and binding Ada programs with the existing predefined Ada library was simple.

We used the benchmark programs Sieve and Calculations (from the January 1983 and May 1985 issues of *Byte*) and the Count word-counting program (similar to the Unix WC program) as simple measures of the systems efficiency.

Times for the Sieve of Eratosthenes program represent one iteration to find all the primes less than 7000. The floating-point calculations program performs 10,000 successive multiplications and divisions. The word-counting program was run on its own source code, which contained 98 lines, 315 words, and 2954 characters. Table 1 summarizes the test results. Times were obtained through the VMS system monitor.

Two striking characteristics are apparent from the benchmark data. First, the execution times are incredibly fast. In general, small programs usually run in less than one second, whether they contain floating-point, integer, or character operations. Second, the size of the object files is incredibly large, thanks to the runtime-support library. (Version 3.10 binds private copies of the entire Ada runtime code with each executable image. Version 3.13 addresses this problem with a binder switch, Shared/Noshared, that permits executable images to share the runtime support code.)

Table 1.
Telesoft TeleGen2 benchmark data.

Program	Source code (bytes/blocks)	Compile time (seconds)	Bind time (seconds)	Runtime (seconds)	Object file (bytes/blocks)
Sieve	608/2	31.34	34.49	00.35	107K/214
Calculations	375/1	32.22	23.62	00.51	107K/214
Count	3150/7	56.88	30.29	00.49	110K/220

(1 VMS block = 512 bytes)

Comments and conclusions. Many of the criticisms raised in the initial review of Telesoft Ada have been addressed in subsequent versions: Separate compilation is supported, constraint errors are raised, the calendar package is available, variant records and the delay statement have been implemented, the use of generics has been partially implemented, and compilation subunits are now permitted.

However, there are still some issues that need to be addressed. For example, generic specifications and bodies must be compiled together, compiling a separate package body containing a task is not permitted, and signed literals are not fully supported. Most of these problems can be worked around and are addressed in the Version 3.13 system release notes.

Telesoft's TeleGen2 system is a good system for experienced Ada programmers using the VAX/VMS system. It is a fairly complete implementation, straightforward in its approach and use, reliable, and generates very efficient VAX object code. Compile and binding times are not comparable to the fast Pascal environments now available, but they are comparable to many of the Modula-2 systems on the market.

The manuals tend to be difficult to understand and should contain more examples; and it would be nice if the distribution tape contained useful example Ada programs to make it easy to get started. But these are minor issues that are easily corrected.

TeleGen2 is an Ada programming environment worthy of consideration.

Thanks are due to Darren Curtis, Steve Furgason, and Troy Pearse, all at the University of Idaho, for helping test TeleGen2.

Wonderful world of Alsys Ada

Troy Pearse and Steve Furgason, University of Idaho, and Youfeng Wu and Naresh Gupta, Oregon State University

Alsys Ada World is an environment for the Alsys Ada compiler that transforms your IBM PC AT (or compatible) into a Defense Dept.-approved Ada programming environment. The Alsys Ada compiler is a well-designed product that provides a very comfortable environment for Ada programming. It is well-documented, reasonably easy to learn (assuming that you already know Ada), and contains extensive help facilities.

The system supports Ada capabilities with few limitations. The compiler's speed, although slow compared to Fortran and Pascal compilers, is good considering the language and machine. The Alsys Ada compiler is a quality compiler and a powerful tool for the serious Ada programmer. Its $3000 cost is not a lot to pay for a desktop Ada system that includes a 4M-byte memory expansion board.

Using the system. In Alsys Ada World, you can compile and bind programs, control many features of the compiler, and manage program libraries. Commands are organized into a hierarchical structure with three command levels: Ada, Lib_Manager, and Unit_Manager. One nice feature is that Ada World commands have the same syntax as Ada procedure calls. Four timesavers make the system easy to use:

- Command parameters can be specified by name, position, or a combination of the two.
- Command names, parameter names, component names, and enumeration literals may be abbreviated.
- Parameters can be omitted, defaulting to system values.
- Batch files can be called with the Invoke command, avoiding laborious typing and retyping.

At the Ada level, four commands are available: Compile, Bind, Lib_Manager, and Unit_Manager. The Compile command compiles each source unit to produce a corresponding object unit in the Ada library, and the Bind command binds the units to produce an object file. The Lib_Manager and Unit_Manager commands transfer control to their respective levels. Both Lib_Manager and Unit_Manager are utility programs that have their own specific commands to maintain and manipulate Ada program libraries. Ada program libraries are created and maintained by the Lib_Manager and manipulated and inspected by the Unit_Manager. Lib_Manager can also be invoked directly from the DOS level to maintain program libraries.

```
C> ADA                              enter Alsys Ada World from DOS
Ada.LIB_MANAGER;                    enter the library manager
Lib_manager.NEW                     create a program library
(LIBRARY\DEMOLIB);
Lib_manager.QUIT                    return to the Ada context
Ada.COMPILE                         compile source file DEMO.ADA
(SOURCE = DEMO.ADA,                 into demolib
LIBRARY = \DEMOLIB);

Ada.BIND (PROGRAM = DEMO,           bind program units from demolib
LIBRARY = \DEMOLIB);
Ada.SYSTEM.DEMO                     execute demo
```

Figure 1. Example Ada World programming session.

Default, Invoke, System, Help, and Quit are grouped into a category called general commands. General commands are available at all times, whether you are at the Ada, Lib_Manager, or Unit_Manager level. The Default command displays, sets, or clears default parameter values for other commands. Invoke lets you execute a batch of commands collectively, and the System command executes a DOS command without leaving Alsys Ada World. Help requests on-line information about the use of Ada command language. Quit exits from the current command level and returns to the level from which it was entered. For example, using the Quit command inside the Lib_Manager or Unit_Manager level returns you to the Ada level.

After entering the source code, there are four steps in the programming process: compiling, binding, linking, and executing. Figure 1 shows the steps to create an executable file, DEMO.EXE, from the Ada source code, DEMO.ADA, and then execute it. Since no external units are needed to produce an executable code file, linking is not required for this example.

Memory management. RAM memory for Ada World is divided into two segments: user memory (which resides from 0 to 640K bytes) and extended memory (which is located above 1024K bytes). You can add expansion boards to bring the extended memory up to 16M bytes. Extended memory is organized as a virtual disk so the memory can be viewed as files. Extended memory is managed by a subcomponent of Alsys Ada World called Adadrive. Adadrive is not a general utility that you can access. Instead, Adadrive is invoked automatically to handle references to extended memory. Object code from programs that do not use extended memory can be run on IBM PC and PC XT machines.

One problem with memory management in Ada World is that the dynamic memory used by tasks cannot be extended beyond user memory. This may limit concurrent Ada programming applications. For example, at the default memory setting in Ada World, the Dining Philosophers problem can be solved for a maximum of only eight philosophers. Ada World lets you specify memory use at bind time or at runtime. This lets you side-step the dynamic memory problem by rearranging the memory distribution to specify optimum sizes of the main program and test stacks and the size of the basic heap. After optimizing memory, the same Dining Philosophers problem can be solved for as many as 75 philosophers.

A potential storage problem is that the hard disk becomes cluttered with many duplications of commonly used packages, since each program may have its own library. For example, the generic package Integer_IO is used by many programs, and therefore shows up in many program libraries. A good solution to this problem would be to arrange the program libraries into a hierarchical structure that would let units be shared. A less desirable solution is to interface to external modules in each library.

Documentation. The Ada compiler documentation is very good. The installation guide is easy to follow and includes instructions for upgrading Version 1.0 files to Version 1.2.

The user's guide is complete, has a well-organized table of contents, and fully describes the Ada compilation system. Its three appendixes contain example sessions, a comprehensive command summary, and a software trouble report. It includes a full description of file handling and error control.

The Ada sampler is a collection of programs designed to help users become more fluent in Ada and gain experience with the Alsys Ada compiler. For each program, the Ada sampler includes a description of what the program is supposed to do, an example of the program's execution, and the source listings.

Benchmarks. Table 2 shows compile and bind times, sizes, and special features of some Ada programs. The average size of a program's source file was 4.6K bytes, and average size of the object code was 56K bytes. The average compile time was 1 minute and 40 seconds, and average binding time was 1 minute and 20 seconds. Although the compiler does extensive memory management and goes through eight passes to process one unit, these times are comparable to other Ada systems.

Table 2.
Alsys Ada World benchmark data.

Ada programs	Compile time (minutes)	Bind time (minutes)	Source code (K bytes)	Object code (K bytes)	Special features
Dining Philosophers	1:27	1:03	5.7	65	Tasks
Producer/Consumer	1:03	1:03	1.6	63	Generic buffer unit, producer and consumer tasks
Hospital management*	1:20	0:47	4.6	46	No tasking, no generic units
Sorting algorithm*	1:21	0:44	4.2	46	No tasking, no generic units
Weather program*	2:38	1:05	6.0	70	Five modules
Rational I/O program*	1:50	0:44	6.7	46	No tasking, five modules, four specification packages

*included in Alsys sampler program