

The Compile and Execution Process

SYNOPSIS

This chapter describes generically, the compile and execution process that can be applied to almost any of the programming languages described in this document.

INTRODUCTION

For most languages on UNIX systems, the compile and execution process remains the same, only the programming language changes. The following diagram that process.

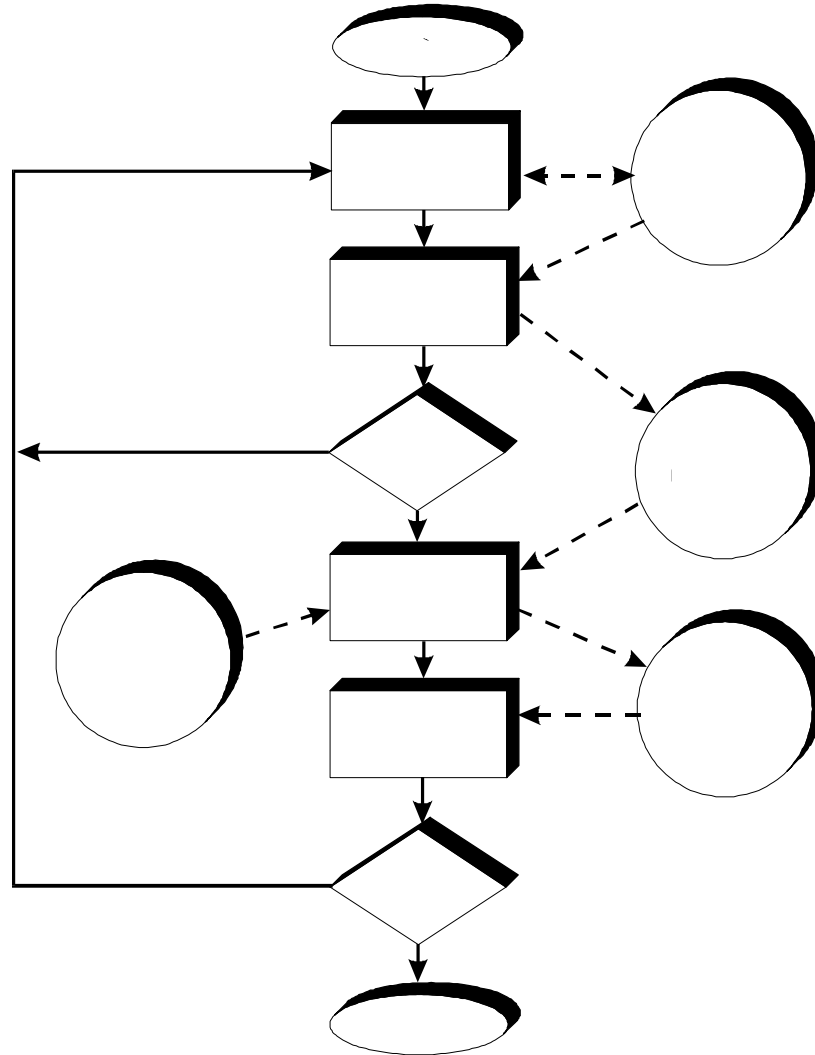


Figure General-1: The Compile and Execution Process

Subsequent chapters within this document describe the process for particular languages. With the exceptions of Alsys Ada and IBM Ada/6000, the process is almost identical for the various compilers and other utilities. Alsys Ada and IBM Ada/6000, on the other hand, require the use of their own utilities to perform the various tasks required for program development.

Additional chapters describe utilities which can be used with the non-Ada languages. These utilities include:

- Debugging tools for interactive and batch debugging methods. Debuggers can help a programmer determine the location of an error in the source code after it generates an error message that would otherwise be meaningless.
- Programming maintenance tools for the development of programs consisting of multiple source modules. Such utilities allow multiple modules to be modified and recompiled without having to recompile all of the modules. Only those which have been modified or are dependent upon the modified routines need be recompiled.
- Source code management tools which allow one or more programmers to maintain software and keep a record of the changes made. These utilities allow a group of users to maintain software and control who has a portion of the code checked out to them at any given point in time. They also provide a good history of changes when tracking a problem in a program.
- Object library management tools which allow a group of object modules to be bundled together and used later by the linker when creating an executable program. These tools allow users to create and maintain libraries of subroutines that may be used over and over again with a large number of programs. This provides a savings to both the programmer and the system in not having to redevelop or recompile the same routine for another use.

The following sections describe the process in a little more detail.

EDITING YOUR SOURCE CODE You should always edit your source code carefully, making changes to obvious errors as you spot them while editing. Errors fixed at this stage will save time and effort later. The source code may be edited in any available text editor on the target system (usually vi on UNIX).

COMPILE THE SOURCE CODE Using the appropriate compiler for the language you are programming in, compile your source code. There are additional utilities on the systems that can help save time by performing a syntax pass over the code in an effort to save CPU time.

EXAMINE ANY ERRORS PRODUCED BY THE COMPILE PROCESS If you receive errors from the compile, examine them carefully. Sometimes one error in the code will generate several error messages. Go back to the "Editing Your Source Code" step and make the appropriate changes in your source code. Try to address as many of the errors as possible before the next compile attempt. Doing so will save valuable CPU time as well as free resources for other users to use.

THE LINKER STEP As a result of the linker step, which is usually part of the compile, you may receive error messages.

The linker is responsible for producing an executable file after linking your object module, produced by the compile step, with those modules in the system libraries and other libraries which you have specified.

EXAMINE ANY ERRORS PRODUCED BY THE LINKER STEP If you receive any errors from the linker step, go back to the "Editing Your Source Code" section and make the appropriate changes in your source code. Then continue from that point as you did earlier.

TRY TO EXECUTE YOUR PROGRAM

Once you have had a successful linker step, you may wish to try to execute your program. As a result of the execution attempt, you may get some execution errors. If not, you are finished with the process.

If you do receive any execution errors, examine them and see if you can determine their causes. This may require recompiling your program with special debugging flags and using a debugger during a trial execution. Once you have identified the causes of the execution errors, go back to the "Editing Your Source Code" step and make the appropriate changes in your source code. Then continue from that point.

SUMMARY

The upcoming chapters will give you more detail on several of these steps, depending upon the language you are programming in.

DOCUMENT CODE: UNIX-40001B

DATE REVISED: September 7, 1995

NOTES