

Alsys Ada

SYNOPSIS

This chapter describes how to compile, bind, and execute Alsys Ada programs as well as how to maintain Alsys Ada families and libraries under the AdaWorld environment on a variety of UNIX systems.

SPECIAL SITE DEPENDENT NOTE On most systems, Alsys Ada is installed as the command "*ada*". However, on the IBM RISC/System 6000 sites that make up the campus wide AIX cluster, Alsys Ada is installed as "*aada*". The reason for this difference is the presence of a second *ada* compiler on the RISC systems with the same command name.

SPECIAL USAGE NOTES This document describes how to customize your account and maintain Alsys Ada in your account. The Computer Science department also has some procedures which install Alsys Ada specific files in your account on many of the UNIX systems on campus; those instructions are available from the El Corral Bookstore. Please consult your instructor about the proper procedure for using Alsys Ada if you are going to be using it for a Computer Science course.

This document will not address the specific instructions used in the Computer Science handout.

INSTALLING ADA IN YOUR ACCOUNT You will need to do some customization of your account for Ada to run smoothly.

A. CHECK YOUR SHELL VARIABLES

Be sure the following line is in the ". *login*" file in your home directory.

```
setenv ALSYCOMP_DIR /usr/local/lib/alsycomp
```

If it is not, you must add the line. (Right after the "*stty*" line that sets control functions is a good place.)

Be sure that "/usr/local/lib/alsycomp" directory is included in your path. Find the line in the ". *cshrc*" file in your home directory that begins with the words "*set path*". A line that sets the path in a ". *cshrc*" file might look like the following:

```
set path=( /bin /usr/bin /usr/ucb /usr/local/bin /usr/sbin  
/usr/local/lib/alsycomp /etc $home/bin . )
```

Look for the *alsycomp* directory in your own ". *cshrc*" file. If you do not see the directory in the path, add it.

As an alternative, you may copy the system default ". *login*" and ". *cshrc*" files from "/usr/local/skel" into your home directory. You will have to log out and log in again for the change to take place, or you can enter the following commands:

```
% source .login<CR>  
% source .cshrc<CR>
```

If your account is very old (created prior to June 1991) or you are unsure of how to do this, you may want to replace all your dot files with the command

"*new.dots*". Old accounts may have to issue the command as
"/usr/local/bin/new.dots".

B. SET UP THE DEFAULTS FOR ADA WITH DOT FILES

Ada dot files are not required, but they will simplify using Alsys Ada. These files are used primarily to avoid typing the same commands before each Ada session. You can put sequences of Ada commands in the Ada dot files, and they will be invoked upon start-up. This documentation assumes that the default Ada dot files are installed in your home directory. The Ada dot files are:

. adarc	for the Ada world level
. fammgrrc	for the Family Manager
. libmgrrc	for the Library Manager
. unitmgrrc	for the Unit Manager
. proberc	for Ada Probe

The "Default" commands are by far the most useful commands to place in the Ada dot files. You can find out more about what defaults you can set in the Ada dot files by reading on-line help about the Default command. (See the section about Ada on-line Help.)

Sample dot files are available in /usr/local/skel/ada. There is also a command called "*ada.dots*" that will install these default Ada dot files in your account for you. Move to your home directory with the "*cd*" command and then use the following command:

```
% ada.dots<CR>
```

NOTE: These sample Alsys Ada dot files assume that the Ada family that will be created in your account defaults to the name "*ada_family*" and is located in your home directory. They also assume that your libraries will be named "*adali b*" and they will be accessed from the current working directory.

C. CREATE AN ADA FAMILY

The following commands place an Ada family named "*ada_family*" in your home directory and give the family the proper permissions.

<u>Most Systems</u>	<u>RISC/6000 Sites</u>
% <i>cd</i> <CR>	% <i>cd</i> <CR>
% <i>ada fam.new</i> <CR>	% <i>aada fam.new</i> <CR>
% <i>chmod 700 ada_family</i> <CR>	% <i>chmod 700 ada_family</i> <CR>

Unless you explicitly specify otherwise, the system defaults will place all your Ada libraries in this family.

D. CREATE AN ADA LIBRARY

The following command creates an Ada library named "*adali b*" in the current working directory.

<u>Most Systems</u>	<u>RISC/6000 Sites</u>
% <i>ada lib.new</i> <CR>	% <i>aada lib.new</i> <CR>

NOTE: All Ada commands can be abbreviated. Ada requires only that you type enough letters of the commands to uniquely identify them. All commands can be abbreviated to their first three letters. In the

example above, "lib" is the abbreviation for "lib_manager". This document abbreviates all commands to their three letter form.

COMPILING ADA SOURCE CODE INTO EXECUTABLE PROGRAMS

There are two steps to creating an executable program from Ada source code: You must first compile it, and then you must link it.

Most Systems

```
% ada c filename.ada<CR>
```

RISC/6000 Sites

```
% aada c filename.ada<CR>
```

to compile a source file called "filename" and

Most Systems

```
% ada b runme<CR>
```

RISC/6000 Sites

```
% aada b runme<CR>
```

to bind to an executable named "runme".

Then to run your program, type:

```
% runme<CR>
```

To execute your program

ACQUIRING ADA UNITS FROM OTHER LIBRARIES

Ada units can import other units that are in the same library as well as units that are in the Alsys standard library. This is accomplished by referencing units with the "with" statement in the acquiring unit's source code. The units in the standard library (/usr/local/lib/alsycomp/predef) are linked to your local library by default. To list the units as linked members of a local library, type

Most Systems

```
% ada uni.list link = yes<CR>
```

RISC/6000 Sites

```
% aada uni.list link = yes<CR>
```

You can also import units from other libraries with the Ada acquire command.

Most Systems

```
% ada uni.acq unit_name from = library_name [by = link]<CR>
```

RISC/6000 Sites

```
% aada uni.acq unit_name from = library_name [by = link]<CR>
```

If you specify "by = link", the unit will not actually be copied into your library, but instead, a link will be created in your library to the unit in the library you are acquiring from. If you do not specify "by = link", a copy of the unit will be copied into your library and occupy valuable disk space.

For example, to acquire the unit "queue" from the Computer Science Ada library, by creating a link to that unit, type:

Most Systems

```
% ada uni.acq queue from = /ulib/csc/lib/csclib by = link<CR>
```

RISC/6000 Sites

```
% aada uni.acq queue from = /ulib/csc/lib/csclib by = link<CR>
```

For example, to acquire an actual copy of the unit "queue" from the Computer Science Ada library, type:

Most Systems

```
% ada uni.acq queue from = /ulib/csc/lib/csclib<CR>
```

RISC/6000 Sites

```
% aada uni.acq queue from = /ulib/csc/lib/csclib<CR>
```

When you compile and bind units, Ada adds the compiled units to your Ada library. This space is in addition to the space required by the source and the executable program. This additional space can grow to be very large. You may begin to experience problems with your account as you approach your disk quota. To avoid problems with space, you should maintain your Ada libraries. You can keep your disk usage to a minimum without removing your source code, but you will have to follow these guidelines faithfully and regularly.

A. REMOVING UNITS FROM YOUR LIBRARY

It is very important to remove units when you no longer need them. If you find you no longer need a compiled unit, remove it from your library. You can erase units with the following command:

Most Systems

```
% ada uni . erase uni t_name<CR>
```

RISC/6000 Sites

```
% aada uni . erase uni t_name<CR>
```

You can keep the source code for units so you can re-compile them if you find you need them later. The source files are generally small compared to the space consumed by compiled versions.

B. COMPRESSING YOUR LIBRARY

When you erase a unit, the space freed is reserved by Ada for future use. This means that you do not increase the amount of available space when you erase units. Ada will try to use the space the next time a unit is compiled.

If your Ada library has gotten very large due to this "feature" and you want to reduce its size, you can use a command written at Cal Poly to compact the library. Change directories to the directory that holds the library you want to compress. Then type the command.

```
% adacompress<CR>
```

This procedure can take several minutes. **Do not interrupt it.** "*adacompress*" will prompt you for any needed information and will inform you of its progress as it runs.

NOTE: "*adacompress*" may not be available on all systems and at the writing of this document, does not support libraries created with a "*parent=*" parameter.

C. REMOVING YOUR LIBRARY

When you no longer need an Ada Library, you can remove it from your account by entering

Most Systems

```
% ada lib . erase confi rm=no<CR>
```

RISC/6000 Sites

```
% aada lib . erase confi rm=no<CR>
```

If you receive an error message which indicates that Alsys Ada was unable to delete the "adalib" in the current directory, it may have become corrupt. You must delete the library by using the following UNIX command

```
% rm -fr adalib<CR>
```

D. UNLOCKING YOUR LIBRARY

During the course of using Alsys Ada, you may receive a message which indicates that your library is locked or busy. You may unlock the questionable library by entering

Most Systems

```
% ada lib.unlock confirm=no<CR>
```

RISC/6000 Sites

```
% aada lib.unlock confirm=no<CR>
```

MAINTAINING YOUR ADA FAMILY

In most cases, your Alsys Ada family needs little maintenance. There are basically two operations that may be required. These are

A. REMOVING YOUR FAMILY

When you no longer need an Ada family, you can remove it from your account by entering

Most Systems

```
% ada fam.erase confirm=no<CR>
```

RISC/6000 Sites

```
% aada fam.erase confirm=no<CR>
```

B. UNLOCKING YOUR FAMILY

During the course of using Alsys Ada, you may receive a message which indicates that your family is locked or busy. You may unlock the questionable family by entering

Most Systems

```
% ada fam.unlock confirm=no<CR>
```

RISC/6000 Sites

```
% aada fam.unlock confirm=no<CR>
```

C. WHAT TO DO WHEN THE INSTALLATION FAMILY IS LOCKED

The installation family (common to all Ada compilations) occasionally becomes locked or busy for extended periods of time. When this happens, you should: Stop the compile if it has been waiting for several minutes by using *Ctrl - <C>*; try again later. On several of the systems, there are automated jobs which check the installation family periodically and unlock it if necessary.

NOTE: It is normal for the installation family to become locked or busy during every user's compile. Usually this is very temporary. When you see such messages and you have not waited for more than a few moments, **DO NOT** use *Ctrl - <C>* to get out, as you may cause the installation family, your own family, or your own library to become locked.

If the installation family does not become unlocked after an extended period of time (about an hour), contact your system administrators and report the problem.

DEBUGGING ALSYS ADA PROGRAMS

There are two basic places that a user may have problems debugging an Alsys Ada program. The first is during the compile phase, and the second is during execution.

A. ERRORS DURING A COMPILE OR BIND

If the compiler aborts during the compile with an obscure message, the user can usually perform several steps to isolate the problem. Usually, these errors will also lock your Ada library. Please refer to section 5 on how to unlock your Ada library.

1. Check your account for free disk space.

Alsys Ada requires a fairly large amount of free disk space within your account in order to build and maintain your library and executables. Information Technology Services recommends that you try to maintain approximately 600 K bytes of free disk space.

If you are getting an error during either the compile or bind process that isn't clear, check your disk space by entering

```
% quota -v<CR>
```

The command will return how much disk space is in use, the total amount available for you to use, and, if any of the limits are exceeded, what the remaining warnings or grace period is. (On AIX, you need a report of disk space on your home site where your files are located; for example "*on cymbal quota -v*<CR>" would provide you with information on your files if they are located on the AIX site named cymbal.)

If you are over quota or are very close to being over (within 250 K bytes), you should consider deleting files from your account which you no longer need.

2. There may be an error in your source code that a phase of the compiler can't handle. This can be checked quite simply by providing a parameter to the compiler to reduce the amount of levels it checks to the most basic. To do this enter

Most Systems

```
% ada c '(source=>filename, options=>(level=>parse))' <CR>
```

RISC/6000 Systems

```
% aada c '(source=>filename, options=>(level=>parse))' <CR>
```

This will produce a listing file containing the errors which may have caused the problem. Correct the errors, then try to compile again.

B. ERRORS OCCUR DURING EXECUTION OF YOUR PROGRAM

When an error occurs during execution which generates an obscure error message, Alsys AdaProbe can be used to determine the problem. To compile, bind, and execute your program under Alsys AdaProbe enter

Most Systems

```
% ada c '(source=>filename, keep=>(debug=>yes, copy=>yes))' <CR>
```

RISC/6000 Systems

```
% aada c '(source=>filename, keep=>(debug=>yes, copy=>yes))' <CR>
```

This compiles the program with debugging turned on such that both necessary tables and a source image are saved in the library. Then enter

Most Systems

```
% ada b '(program=>unit_name, keep=>(debug=>yes))' <CR>
```

RISC/6000 Systems

```
% aada b '(program=>unit_name, keep=>(debug=>yes))' <CR>
```

This binds the unit and saves necessary information that AdaProbe requires in a file with the unit name and an extension of ".cui". The program is then executed by entering

Most Systems

```
% ada workbench.probe '(program=>program_name)' <CR>
```

RISC/6000 Systems

```
% aada workbench.probe '(program=>program_name)' <CR>
```

Output from AdaProbe is sent to a file with the unit name and an extension of ".log". In most cases, this should produce the information needed to solve the problem.

ADA ON-LINE DOCUMENTATION

On-line Documentation is available with the Ada Help command. To access the information, enter the Ada environment by typing "ada" at the system shell prompt. You will then get an Ada prompt. At the Ada shell prompt, type "help" as in the following example.

Most Systems

```
% ada<CR>  
Alsys Ada Version 5.3  
Copyright (C) Alsys, 1985, 1990. All rights reserved.  
Ada. help<CR>
```

RISC/6000 Sites

```
% aada<CR>  
Alsys Ada Version 5.3  
Copyright (C) Alsys, 1985, 1990. All rights reserved.  
Ada. help<CR>
```

The Ada help shell is organized in hierarchies. When you enter the HELP sub-shell, you will be prompted for topics and sub-topics. The topmost level of HELP contains information about tools, and you will be asked to choose from topics in the next lower level. Top level topics correspond to commands at the top level of Ada, (i.e. *Compile*, *Bind*, *Lib_Manager*, *Unit_Manager*, *Family_Manager*, *Default*, *Global*, *Help*, *Invoke*, *Quit*, and *System*)

You can exit from any level of the HELP sub-shell by hitting the "Return" key, and you can move up through topic levels with the "<" key. Find more documentation about navigating through Ada's help facility by typing:

Most Systems

% *ada*<CR>

Ada. *help*<CR>

Topic: *help*<CR>

HELP Subtopic: *Navigating_characters*<CR>

RISC/6000 Sites

% *ada*<CR>

Ada. *help*<CR>

Topic: *help*<CR>

HELP Subtopic: *Navigating_characters*<CR>

Exit the Ada shell by typing "q".

DOCUMENT CODE: UNIX-40101C

DATE REVISED: September 7, 1995

NOTES