Amadeus Project

Amadeus Installation and Maintainence Guide

Distributed Systems Group

Trinity College Dublin

Abstract

The installation and maintainence of an Amadeus system is described.

Document Identifier	Release Doc:Orange:1
Document Status	DRAFT
Created	1/Nov/1991
Revised	1/Nov/1991
Distribution	Public

© 1992 TCD DSG

Permission to copy without fee all or part of this material is granted provided that the TCD copyright notice and the title of the document appear. To otherwise copy or republish requires explicit permission in writing from TCD.

Contents

Preface

The Amadeus v2.0 release is the second release from the Amadeus project by the Distributed Systems Group of the Department of Computer Science, Trinity College Dublin (TCD). It extends C++ for distribution and persistence; the extended language is called C**.

Amadeus v2.0 is itself implemented in C and C++, on top of Digital Equipment Corporation's Ultrix 4.2 on both MicroVAXes and DECstations, and on Sun's SunOs 4.0.1 on Sun-3 workstations. A small amount of assembly language is included to support a lightweight process package. The system relies on the use of NFS for remote file access. The implementation also uses Unix sockets, signals, shared memory, and semaphores. The C** compiler in Amadeus v2.0 is a modified version of the Free Software Foundation's g++ 1.37 compiler.

Future releases, both planned and underway, will extend the functionality, supported languages and host operating systems.

The Amadeus Project has been influenced by the Esprit Comandos-1 and Comandos-2 projects, in which TCD has been a partner. The project is also being influenced by the Esprit Harness and Ithaca projects. TCD acknowledges the fruitful interactions with all of the participating institutions and in particular with Inesc in Lisbon; Bull and IMAG in Grenoble; GMD in Bonn; and the University of Glasgow.

Information Set

The information set for the Amadeus v2.0 release includes the following three documents:

- *Overview of the Amadeus Project* presents an overview of both the Comandos and Amadeus projects. This document also includes an introduction to distributed systems and explains the terminology used in the Amadeus Project.

- *C** Programmers' Guide* contains both tutorial and reference information on programming in the C** language. The text assumes a knowledge of object-oriented programming and, in particular, the C++ language. This guide lists sample C** programs which are supplied with Amadeus v2.0, and which can be compiled and linked after installing the Amadeus environment.

- *Amadeus Installation and Maintainence Guide* (THIS DOCUMENT) describes the installation procedure together with instructions for system startup, and configuration.

Organisation of This Document

The intended audience for *The Amadeus Installation and Maintence Guide* is systems programmers or others responsible for the installation and maintainence of an Amadeus system. The document is arranged in five chapters:

Chapter 1 provides an overview of the Amadeus v2.0 distribution kit and specifies the requirements for running this version of Amadeus.

Chapter 2 describes the installation and use of the Amadeus controller in detail.

Chapter 3 describes the installation and use of the Amadeus server.

Chapter 4 describes the installation of the C** compiler and some details of running an Amadeus application.

Chapter 5 describes the procedures for bug reporting and obtaining and applying updates to this Amadeus distribution.

Appenidx A presents a useful installation checklist.

Trademarks

The following are trademarks:

- Chorus – Chorus Systèmes
- DECstation – Digital Equipment Corporation
- microVAX-II – Digital Equipment Corporation
- Network File System – Sun Microsystems, Inc
- NFS – Sun Microsystems, Inc
- OSF/1 – Open Software Foundation

- SunOS – Sun Microsystems, Inc
- Ultrix – Digital Equipment Corporation
- UNIX - UNIX Systems Laboratories, Inc
- VAX – Digital Equipment Corporation

Chapter 1

Release kit overview and pre-requisites

The v2.0 distribution of Amadeus runs on Digital Equipment Corporation's DECstation and microVAX II workstations running Ultrix 4.2, and on Sun-3 workstations running SunOs 4.0.1. A distributed Amadeus system may consist of up to `MAXNODES` [1] workstations of either type.[2] An Amadeus system can simultaneously run distributed and persistent C** applications from several users.

1.1 Release kit contents

The Amadeus release kit comes on tape in *tar* format. There are several [3] tar files as detailed below.

It is suggested that the installer of the Amadeus system creates a top-level [4] directory in which the required components of the Amadeus release kit are placed.

The current Amadeus distribution Tape consists the following tar archives in the following order, users need only install those archives which they require:

- **Documentation**

  This tar file contains the release documentation for Amadeus. This archive should be placed in /amadeus/doc. This file contains the information set for Amadeus as detailed above.

- **Amadeus system sources and binaries**

[1] Defined in `/amadeus/sys/src/kernel/consts.h`.

[2] Mixed systems are not currently supported.

[3] This is a major difference from the v1.0 release of amadeus. The previous version contained only one (very large) tar file in which all components of the Amadeus kit were placed. From experience with users we found that most users wanted only wanted access to some components of the kit, therefore we have now split the kit into seperate logical components

[4] e.g. /amadeus

This tar file contains the sources and binaries for the Amadeus kernel. This archive should be placed in a directory /amadeus/sys. The Amadeus sources are found in /amadeus/sys/src/{kernel,grt,utils}. The Amadeus kernel binaries (server, library and controller) are found in /amadeus/sys/bin/{mips,sun3,sun4}/{amadeus,libpga.a,amc}.

- **C** binaries**

  This tar file contains the binaries for the C** compiler. This archive should be placed in /amadeus/cstarstar/bin. The file *css* is the cstarstar compiler, and is found in /amadeus/cstarstar/bin/{mips,sun3,sun4}. Other files are associated programs.

- **C** sources**

  This tar file contains the sources for the C** compiler. This archive should be placed in /amadeus/cstarstar/src.

- **C** demonstration programs**

  This tar file contains a set of demonstration C** programs. This archive should be placed in /amadeus/demos.

- **GNU binaries (mips)** This tar file contains the pre-requisite gnu G++ and Gmake binaries required to compile and run C** programs, on a mips platform. This should be place in /usr/local/gnu. Contents of this archive are gnu executables, libraries and headers.

- **GNU binaries (sun3)** Same as above, for a sun-3 machine.

- **GNU binaries (sun4)** Same as above, for a sun-4 machine.

- **g++ sources**

  This tar file contains the sources for gnu G++ and Gmake. Installation guide for gnu found in this archive.

1.2 The Amadeus server, library and controller

Each host involved in an Amadeus system must run the Amadeus server. A new host joins the system when a server is first started on that host. A host leaves the system when its server terminates. Hosts may join or leave the system at any time [5].

When started, a server informs other servers (if present) about the node's existence and initialises local Amadeus state information. This includes various shared tables used by the Amadeus communications and location services. The server also initialises the local storage system, mounting any local containers.

A controller program is provided to exercise control over the set of servers making up a single system. Using the controller program it is possible to launch a server on the local

[5] In the present version this will result in the premature termination of any on-going distributed applications that were present at the host.

or a remote node, discover which hoats are involved in a system and to terminate one or all servers in the system. These operations can of course be performed manually.

The Amadeus library must be linked with each C** application as described in the C** Programmers' Guide.

1.3 The C** compiler

The C** compiler is used to compile all C** applications as described in the C** Programmers' Guide.

1.4 Pre-requisites

As noted previously, this distribution of Amadeus runs *only* on DECstation and microVAX II workstations running Ultrix 4.2, as well as on a Sun-3 workstation running SunOs 4.0.1.

Moreover every host participating in an Amadeus system must use NFS to import certain directories which must be shared by all hosts. These directories include the SS root directory and *all* the directories which will be used to store persistent objects [6]. On every participating host, each such directory should appear with the same path in the local directory hierarchy. This is necessary since Amadeus applications always read/write the files used to store objects as if they were stored locally, relying on NFS to implement remote read and write.

The distribution of Amadeus was compiled and linked using the following utilities:

- Host cc compiler.
- Gnu g++-1.37.1
- Gnu libg++-1.37.0
- Gnu make-1.34

The C** compiler is a modified version of the Free Software Foundations's gnu g++ 1.37 compiler. in order to run the C** compiler, the user must previously have the g++ 1.37 compiler installed on the local system. Installing this compiler, and the associated libraries, is a tedious task. The user may wish to avail of the gnu binaries as provided with the Amadeus release. Copying these to /usr/local/gnu will complete the installation process for this software. Otherwise the user may do a full installation of the g++ compiler using the sources provided in the release kit.

Documentation was prepared using LaTex version 2.09, and is available in *dvi* and *ps* formats.

[6] i.e. those representing containers.

Chapter 2

The Amadeus controller

The Amadeus controller is a tool which is provided to facilitate management of the servers making up an Amadeus system. The controller consists of a simple interactive program, which is found in the /amadeus/sys/bin/{sun3,mips} directory.

2.1 Running the Amadeus controller

Essentially the controller provides four operations:

- start a server on a specified host;
- terminate a server on a specified host;
- terminate an Amadeus system;
- list all the nodes currently involved in an Amadeus system.

2.1.1 Command syntax

To get help on available commands the following command must be run:

```
amc -h
```

To start a server on a particular host with a given server number, the following command must be run:[1]

```
amc -i -n<hostname> -s<server number>
```

[2]

To terminate a server on a particular host, the following command must be run:

[1] This command executes a *rsh* command. Also available in the /amadeus/sys/bin/{sun3,mips} directory is a sample file Startup, which can be used to startup Amadeus server on all nodes.

[2] For this command to work the caller's default directory must be /amadeus/sys/bin/{sun3,mips}

```
amc -k -n<hostname> -p<port number>
```

Where the port number is the port which amadeus servers are using.

To terminate all existing servers the following command is run:

```
amc -k -p<port number>
```

To list all the servers which are currently running in the system:

```
amc -l -p<port number>
```

2.1.2 Installation

The Amadeus Controller requires that all hosts where Amadeus servers may be run, have corresponding entries in the user's ".rhosts" file.

Chapter 3

The Amadeus server

The Amadeus server is fundamental to the operation of the Amadeus system and must be active on each participating host.

3.1 Starting an Amadeus server

An Amadeus server can be started via the Amadeus controller, as described previously, or interactively with the command:

```
/amadeus/sys/bin/{mips,sun3}/amadeus <ann> [log]
```

on a DECstation, where the required parameter `ann` gives the Amadeus node number by which the current host is to be known and is a number in the range 1 to `MAXNODES`. Obviously no two hosts should use the same `ann` [1] - otherwise the choice of `ann` for a particular host is arbitrary.

3.2 Terminating a server

A server can be terminated using the Amadeus controller or interactively by issuing a Ctrl-C.

3.3 Server configuration

Once started the server will first read its so-called *configuration file* with the name `amadeus.par` from the same directory as the server image.

The configuration file gives values for the dynamically alterable parameters of the server including the location of the SS root directory, the internet port to be used for server-server and application-server network communication and debugging options.

[1] Amadeus will detect such attempts.

The format of the server configuration file is fixed. A sample is given below:

```
amdebuglevel = 1
csdebuglevel = 1
vomdebuglevel = 1
rlsdebuglevel = 1
ssdebuglevel = 1
debuglevel = 1
shmkey = 9
ssroot = /amadeus/demos/ssroot
```

The `debuglevel` parameters specify the level of debugging information that will be output by the server. The value of each of these parameters is a number in the range 0 to 9. For normal operation a value of 1 should be chosen so that only error messages will be printed. Level 9 provides extremely detailed information about the operation of the server.

The `shmkey` specifies the key to be used to identify the shared memory segment being used by the server and local applications.

The `ssroot` parameter gives the path name for the root of the Amadeus storage system i.e. the directory in which the system mount file is stored. Note that this must be an NFS exported directory available at the same path name on all participating hosts.

Care should be taken to ensure that all servers participating in an Amadeus system should have the same value for their `ssroot`, `shmkey` and `serverport` parameters. [2]

[2] A good way to do this to run the server from the same NFS directory on all hosts, so that all servers see the same configuration file.

3.4 Storage system initialisation

Once configured the server will begin by initialising the local storage system i.e. mounting local containers. To do this it first reads the system mount file `amadeus.mount` which gives the control node at which each container is to be mounted in the system. The `amadeus.mount` file is read from the `ssroot` directory Since this directory is shared by all servers, there is a single `amadeus.mount` file for the entire Amadeus system.

An example of an `amadeus.mount` file is given below:

```
====================================================================
This file contains the following information:

1) Amadeus System Mount table
The token "START_TABLE" denotes
the start of the Mount table.
A mount table entry contains
an Lc Number, a Node number and
an Lc path.

2) Next Free Container Number.
This is found after the token NEXT_FREE.


====================================================================

NEXT_FREE       9

START_TABLE

 1    1  /user/john/myclusters
 2    1  /cds/clusters/objectsA
 3    2  /cds/clusters/objectsB
 4    2  /work/amadeus/ss/lca
 5    2  /work/amadeus/ss/lcb
 6    2  /kernel/NewClusters/lc6
 7    3  /kernel/NewClusters/lc7
 8    3  /users/vinny/amadeus/clusters/lc8
```

This system mount file is for a 3 node system where node 1 is the control node for containers 1 and 2; node 2 for containers 3 to 6 and node 3 for containers 7 and 8.

Moreover, the directory implementing container 1 has pathname /user/john/myclusters on *all* hosts, while the directory implementing container 8 has pathname /users/vinny/amadeus/clusters/lc8 on *all* hosts.

It is the responsibility of the system manager to create and export or import (as appropriate) the required directories via NFS. The server will detect the absence of any of the container directories.

3.4.1 Protection considerations

The storage system root directory and all container directories must be both readable and writeable by the Amadeus server and by all applications. Thus the user `amadeus` and all users who will use the Amadeus system should be members of a common group. All the relevant directories should then be protected with `rw` protection for that group.

Chapter 4

The compiler and applications

This chapter gives some information concerning the installation and operation of the compiler and C** applications.

4.1 The C** compiler

The current C** compiler is derived from version 1.37 of the Free Software Foundation's g++ compiler.

4.2 Applications

Once linked as described in the C** Programmers' Guide, an application can be run on any participating node. The application can be run in the same way as any other Ultrix application e.g. from the shell or as a result of an `exec` call. When running an application, there must be a directory called `clusters`[1] located in the same directory as the application.

If the application diffuses to a new node, the same image will be invoked on that node. Thus all applications should be stored in NFS directories with the same path name on each participating host.

Anyone running a C** application should be a member of the Amadeus group in order to be able to access both the storage system root and container directories.

4.2.1 Initialisation

Each application begins by reading configuration information from a configuration file with the same name as the application image and the extension `.par` from the same directory as the application image. For example, the image `/users/vinny/css/phil` reads the configuration file `/users/vinny/css/phil.par` when run.

[1]This directory must have world read, write and execute permissions

The configuration file is read on the starting node and on any other node to which the application subsequently diffuses.

An example configuration file is given below.

```
amdebuglevel = 1
csdebuglevel = 1
vomdebuglevel = 1
rlsdebuglevel = 1
ssdebuglevel = 1
rtdebuglevel = 1
debuglevel = 1
serverport = 2222
stacksize = 10
maxprocs = 20
shmkey = 9
classdict = .class-dictionary
clstore = clusters/aon_clstore
```

The format is similar to that for the server described previously 3.3. The meaning of the `debuglevel`, and `shmkey` parameters is the same as for the server. Note that there is one extra debug level parameter - `rtdebuglevel` and that all applications should use the same `shmkey` as the servers.

The `maxprocs` parameter specifies the maximum number of lightweight processes that can be created on a single node by the application, while the `stacksize` parameter specifies how big the stack for each lightweight process will be. `stacksize` is measured in kilobytes.

The `classdict` and `clstore` parameters are temporary fields required for Amadeus v2.0 and the above values should not be changed.

Each application has its own configuration file and may specify different values for each of the parameters [2] independently of other applications.

[2] i.e. except `shmkey`.

Chapter 5

Support and maintainence

5.1 Support

DSG encourages users to provide feedback on all aspects of the system including bug reports, functionality, ease of use, performance etc In particular suggestions for modifications and improvements are welcome. In the first instance please send mail to `amadeus@cs.tcd.ie` and we will endeavour to respond as quickly as possible.

5.2 Maintainence

Until the next full distribution of Amadeus, upgrades will be generated using `rcsdiff` so that only differences between the last upgrade and the latest version will be distributed. These can then be incorporated locally using `rcsmerge`.

Appendix A

This appendix presents a checklist for those responsible for installing Amadeus at a new site.

- Install the Amadeus sources from archive;
- decide which hosts will participate in the local Amadeus system (either DECstations *or* microVAX IIs, or Sun-3s);
- create the storage system root directory with appropriate permissions;
- NFS import the storage system root directory on each host;
- create a directory for each container required with appropriate permissions;
- NFS import each container directory on each host;
- edit the server configuration file to identify the storage system root directory;
- edit the `amadeus.mount` file in the storage system root directory to identify each container directory;
- install the C** compiler;
- install the demonstration suite;
- Start amadeus servers on each node in the system;
- Compile and run C** programs;