THE PROJECT SEQUOIA 2000 OPERATING SYSTEM SOURCE POOL

MANAGEMENT AND DISTRIBUTION OF NEW OPERATING SYSTEMS

SOURCE CODE FOR SEQUOIA 2000

February 18, 1993

Fred L. Templin
University of California
Electronics Research Laboratory
473 Evans Hall
Berkeley, CA 94720
(510) 642-6160
templin@postgres.berkeley.edu

Abstract

Project Sequoia 2000 [STON91] is a large-scale research collaboration between computer scientists and global change researchers including several University of California campuses and industrial sponsors. Computer equipment for the project consists primarily of Digital Equipment Corporation (DEC) file servers, workstations, and dedicated network routers which run the DEC ULTRIX V4.2a operating system. Nearing the end of its second year, the project has begun to spawn new computer science technologies in the areas of networks, storage systems, filesystems, databases, and visualization. As these technologies continue to emerge, it is clear that a coherent management scheme for the integration and distribution of Sequoia 2000 software will be a key to the success of the project. It is also clear that, as new commercial operating systems and computer architectures evolve, portability must be a primary consideration. The following document focuses on a subset of this larger problem; the management of operating system software, both in the kernel and in operating system-related user applications, created for Sequoia 2000. We introduce the *Sequoia Source Pool* which will be the vehicle for management and distribution of new operating system software for Sequoia 2000 academic researchers and industrial partners. This is a working document which gives a roadmap for those wishing to add new code, build Sequoia operating system kernels and user applications, or simply wander around out of curiosity. It is not expected that this document will specify a rigid framework for all current and future Sequoia 2000 software management, but rather will continue to evolve along with the project.

1. INTRODUCTION

Sequoia 2000 presents an inherent management challenge due to the decentralized nature of the project. Overall project vision is made possible only through the occasional *retreats* and *swat team* visits which bring researchers of diverse backgrounds together in a common forum. This same challenge exists for the establishment and management of a Sequoia Source Pool. While it is expected that individual researchers and research groups will continue to design, develop and test new operating system software in their own private environments, some software will eventually reach a state of maturity at which time they will wish to distribute it on a project-wide basis. It is for this reason that we propose a source pool for operating system software integration and distribution.

As the project continues to mature, there will be a continued and increasing producer/consumer relationship between those researchers designing and developing new operating system software and those wishing to make use of the new technologies. In many cases, researchers' activities will overlap in both of these areas. In this document, we describe how the Sequoia Source Pool will satisfy the needs of researchers from both of these categories for the current ULTRIX V4.2a operating system. The following sections give the structure of the Sequoia Source Pool, instructions for building kernels and user applications, guidelines for adding new code, and portability considerations for future operating systems migration.

2. STRUCTURE OF THE SOURCE POOL

The Sequoia Source Pool resides on a DECSystem 5900 located at U.C. Berkeley which is part of the Project Sequoia *Bigfoot* fileserver. [STON91] This machine can be accessed over the Internet as *toe.CS.Berkeley.EDU* or over the Sequoia 2000 Network (S2Knet) [PASQ92] as *toe.s2k.Berkeley.EDU*. Since the S2Knet employs private high-speed leased lines expressedly for Sequoia 2000 activity, we recommend that researchers whose workstations are directly connected to the S2Knet use the S2Knet name if possible. To access the source pool, you must be a member of the *s2k-src* group on *toe*. You may access the pool by logging into *toe* directly or by mounting the pool via NFS. Please direct requests for *s2k-src* group privileges and/or NFS mount privileges to the *toe* system manager.

The Sequoia Source Pool is rooted in the directory */usr/src/sequoia* and contains source code for both operating system-specific and other user-contributed Sequoia software. While the source pool is intended to be a generic area for any code Sequoia researchers may wish to contribute, no guidelines are established for non-operating system-specific code in this writing. The ULTRIX V4.2a code is located under */usr/src/sequoia/ultrix-4.2a*. A file called *FILES* in the top level directory gives a list of the source files included in the hierarchy. If you'd like to modify or examine source files, you can search for strings in the file *FILES* to learn the location of the source files in question. For example, to locate the source code for *ifconfig*:

```
% grep -i ifconfig FILES
./bin/ifconfig.d
./bin/ifconfig.d/_vax.b
./bin/ifconfig.d/_mips.b
./bin/ifconfig.d/Makelocal.mk
./bin/ifconfig.d/ifconfig.c
./bin/ifconfig.d/Makefile
./usr/man/_vax.d/man8/ifconfig.8c
./usr/man/_mips.d/man8/ifconfig.8c
```

(Note: in the above example, the source code for *ifconfig* is located under a directory called *./bin/ifconfig.d*. This is a standard naming convention for most user application source code in the Sequoia source pool. In general, the source code for a user application named *xxx* will be located under a directory named *pathname/xxx.d*).

Sequoia operating system source code can be categorized as either kernel code or user application code. The Sequoia kernel code is located under */usr/src/sequoia/ultrix-4.2a/sys* and is the directory hierarchy where researchers will make all changes to Sequoia kernel code after requesting and receiving approval from the *submit review committee* . (See section 4.) To

maintain the integrity of the pool, we require that researchers place new kernel source files and changes to existing files under RCS control. (See *rcsintro(1)* for an introduction to RCS commands.) Researchers should build Sequoia kernels from a separate directory tree called */usr/src/sequoia/ultrix-4.2a/READONLY_sys* which is a read-only *snapshot* of the most recently-submitted kernel code. The *READONLY_sys* directory is rebuilt by the source pool administrators on an as-needed basis and represents the most recent *release* of Sequoia kernel code.

The other directories in the *usr/src/sequoia/ultrix-4.2a* hierarchy include user application source files which are part of the ULTRIX V4.2a operating system release. Once again, researchers are required to place new source files and changes to existing files in these areas under RCS control. Finally, the original "vanilla" ULTRIX V4.2a source code is available under the directory: */usr/src/ultrix-4.2a*. This directory can be used as a point of reference to check against code in the Sequoia source pool. The following sections give instructions for building kernels and user applications as well as adding new code to the pool.

3. BUILDING SEQUOIA KERNELS AND USER APPLICATIONS

3.1. Building Kernels

Building kernels from the most recent Sequoia ULTRIX V4.2a kernel pool can be accomplished by using the *lndir(1)* utility to create a shadow of the *READONLY_sys* directory. To do this, create a directory called *PRIVATE* (where *PRIVATE* is an arbitrary pathname on your local disk), then type:

```
% lndir /usr/src/sequoia/ultrix-4.2a/READONLY_sys PRIVATE
```

This will take on the order of 8 minutes and occupy approximately 3 Mbytes. However, we recommend that you reserve at least 10-20 Mbytes since builds will typically create several Mbytes worth of binaries and kernel images. The hierarchy under *PRIVATE* will consist of symbolic links to the files found in the real source pool. To build a kernel, you must first create a system configuration file for the target machine. To do this, change directory to *PRIVATE/conf/mips* and type the command:

```
# /etc/sizer -n SYSNAME
```

where *SYSNAME* is the name of your machine. This will create a file called *PRIVATE/conf/mips/SYSNAME*. Next, you will need to add the Sequoia-specific systems options and pseudo-devices for any Sequoia functionality you'd like to build into the kernel. As a minimum, you must add the lines:

```
options SEQUOIA
pseudo-device   sequoia
```

to the *SYSNAME* file to enable the Sequoia *basic kernel functionality*. (See the file *GENERIC* as a model of how to add these two lines.) Examples of basic kernel functionality include new device drivers, kernel code which must run on all Sequoia machines, changes to global header files, etc. Next, decide which other Sequoia kernel options you would like to enable by scanning the *README* files located in the directory *PRIVATE/conf/mips/README* and adding the appropriate lines to the *SYSNAME* file as directed by the *README* files. (See section 4 for a discussion of adding new code to the kernel pool.) Finally, execute the following command sequence:

```
% mkdir ../../MIPS/SYSNAME
% ./config SYSNAME
% cd ../../MIPS/SYSNAME
% make depend; make
```

The above command sequence will create an executable *vmunix* kernel image. To boot the system with this new kernel image, type:

```
# mv /vmunix /vmunix.good
```

```
# mv vmunix /vmunix.new
      # ln /vmunix.new /vmunix
```

(Note: Running the */etc/sizer* command and booting the new kernel require super-user privileges. If you do not have super-user privileges, you must work with your local system administrator to perform these operations. Refer to [FORR92] for a discussion on super-user password access for Sequoia Standard ULTRIX machines.)

3.2. Building User Applications

For the short term, we ask that you build user applications by going directly to the appropriate directories in */usr/src/sequoia/ultrix-4.2a* on *toe*. (See the previous section on the structure of the pool). A more robust public access policy for building user applications will be set in the very near future. For now, however, it is vital that those wishing to build user applications from the source pool coordinate with others who may be attempting to build the same applications concurrently. This can be accomplished by checking the ownership of the source files which comprise the application in question and/or checking for active files using the RCS commands. (See *rcsintro(1)*). The *makefile's* for most user applications make use of a global makefile in the */usr/src/sequoia/ultrix-4.2a/genmake.d* directory which defines several global macros for make(1). To build an application, you must first set the *SRCROOT* environment variable to point to the generic makefile. (In the case of the Sequoia source pool, set SRCROOT to */usr/src/sequoia/ultrix-4.2a*.) Below is an example of how to build the *ifconfig* utility:

```
% cd /usr/src/sequoia/ultrix-4.2a/bin/ifconfig.d
% setenv SRCROOT /usr/src/sequoia/ultrix-4.2a
% make clean
% make
```

(Note: Not all system utilities will have consistent makefile conventions. Some will require that *make depend* be run, while others will require that other special make operations be run. It is the responsibility of the person wishing to build the utilities to read and understand the appropriate makefiles and take appropriate actions.)

4. ADDING NEW CODE TO THE POOL

We require that individual researchers first build and THOROUGHLY TEST new code in their own local environments before requesting permission to submit it to the Sequoia source pool. We are in the process of establishing a *submit review committee* which will be responsible for reviewing requests for source code submission. The actual mechanism for reviewing and granting requests is TBD, and will be documented soon. For the short term, send mail to:

```
templin@postgres.berkeley.edu
```

if you would like to make a request to submit code. Please review the section on "Portability Considerations" before adding new code.

To build and test new kernel code in a local environment, first follow the steps listed in the previous section on building kernels, then add the new code to your local kernel build area. To modify EXISTING source files, first remove the symbolic link(s) created by the *lndir(1)* operation which correspond to the files you wish to edit, then copy the textfiles directly from the source pool into your local area. Next, modify the files:

```
PRIVATE/conf/files
PRIVATE/conf/mips/files.mips
```

to change the designators for the files you intend to modify from "Binary" to "Notbinary". In order to differentiate the new code from existing ULTRIX V4.2a code, we ask that all new code added to existing kernel source files be delimited with C-preprocessor directives with descriptive names beginning with the string *SEQUOIA*. If the code you are adding must be built into all Sequoia-related kernels, then it is considered Sequoia *basic kernel functionality*. (See the

preceding section for a discussion of code considered basic kernel functionality). Delimit any new basic kernel functionality code with the directives:

```
#ifdef  SEQUOIA
#else   SEQUOIA
#endif  SEQUOIA
```

However, if the code you are adding to the pool is to be optionally selectable on a per-machine basis, you must choose a descriptive C-preprocessor directive name to delineate the new code. For example:

```
#ifdef  SEQUOIA_RTIP
#else   SEQUOIA_RTIP
#else   SEQUOIA_RTIP
```

Additionally, we recommended that all code which is closely related but which requires more than one new C-preprocessor directive be delimited with directives beginning with the same prefix. For example:

```
#ifdef  SEQUOIA_RTIP_PERFORMANCE
#ifdef  SEQUOIA_RTIP_NOINTR
```

To add NEW source files to the kernel, first decide which directory(s) the new files belong in and choose a new C-preprocessor directive name as was discussed in the previous paragraph on modifying existing code. The new directive name will be used to enable the new kernel source files as well as delimit any related modifications to existing code. Next, if the new C-preprocessor directive name you chose is *SEQUOIA_XXX*, create a file called:

```
PRIVATE/conf/mips/files.sequoia_xxx
```

(lower case is significant) and add the names of the new files. The names of the new files must be entered AS THOUGH they were being added to the *PRIVATE/conf/files* or *PRIVATE/conf/mips/files.mips* files. For example:

```
% cat PRIVATE/conf/mips/files.sequoia_rtip
net/netinet/rtip_output.c           optional inet Notbinary
net/netinet/rtip_input.c            optional inet Notbinary
net/netinet/rtip_queue.c            optional inet Notbinary
net/netinet/rmtp_output.c           optional inet Notbinary
net/netinet/rmtp_input.c            optional inet Notbinary
net/netinet/rmtp_usrreq.c           optional inet Notbinary
net/netinet/rt_subr.c               optional inet Notbinary
```

Finally, whether you are modifying existing code, adding new source files, or both, you must either create a new *README* file in the directory: *PRIVATE/conf/mips/README* or modify an existing *README* file there. (For example, if you have added new source files and created a new C-preprocessor directive called *SEQUOIA_XXX*, you must create a *README* file called *README.sequoia_xxx*.) The file *PRIVATE/conf/mips/README/README.sequoia* gives an example of the information which should be included in the new or modified *README* file.

5. PORTABILITY CONSIDERATIONS

While reviewing existing ULTRIX V4.2a source code, it is expected that researchers may encounter obvious bugs, performance bottlenecks, or simply poor coding practices. Although it may seem tempting, we HIGHLY discourage researchers from engaging in modification of existing ULTRIX V4.2a code unless absolutely necessary. Since we will soon be faced with porting Sequoia operating system code to new systems, such as OSF/1 or NT, it is essential that changes

to existing source files be kept to a bare minimum and limited to code that is specific to Sequoia. Ideally, running *diff(1)* against the original source files in */usr/src/ultrix-4.2a* and the modified source files should show up only code bracketed in C-preprocessor directives beginning with the string *SEQUOIA*.

Modifying kernel *global* header files, both in the kernel and in system utilities, should be done sparingly, if at all. (Here, "global" refers to files that are included by many kernel source files and/or system utilities; not those that are specific to only a few source files.) Modifying global header files can greatly affect the integrity of the source pool due to the volumes of source files which would need to be recompiled. In general, there should be no reason to change existing code in an ULTRIX V4.2a header file; the only changes to a header file should be the addition of new code which cannot simply be added to individual kernel or user application source files. Any such code must be delimited by the C-preprocessor directives:

```
#ifdef   SEQUOIA
#endif   SEQUOIA
#else    SEQUOIA
```

since any header file changes will be considered as Sequoia *basic kernel functionality*.

When adding new source files to the pool, we respect the fact that different researchers may write code using different styles and conventions. However, the style of coding should be clearly readable and generally consistent with existing code in the pool. Documents on BSD's *Kernel Normal Form* and the Postgres Coding Style guide suggest common coding styles and are available in the directory */usr/src/sequoia/ultrix-4.2a/DOC*

6. CONCLUSION

In the preceding sections, we present the structure of the Sequoia source pool, methods for building kernels and utilities from the pool, guidelines for adding new code to the pool, and portability considerations. Although this document lists several policies and procedures, we rely heavily on responsible behavior from the Sequoia researchs to ensure the integrity of the pool. As new operating systems are adopted for the Sequoia 2000 environment, we expect that this document will be expanded to include new policies and procedures for them as well.

Acknowledgements

I would like to thank Jon Forrest for his many significant comments as the principal editor for this document. I would also like to acknowledge Kevin Fall for establishing the S2Knet router source pool which was the forerunner of the Sequoia source pool. Finally, I would like to acknowledge the entire Sequoia 2000 system management team for maintaining and upgrading the Sequoia infrastructure in a professional manner.

REFERENCES

[FORR92] Forrest, Jon, ‘‘Sequoia Standard Ultrix,’’ *Sequoia 2000 Technical Report (Unpublished),* (1992)

[PASQ92] Pasquale, Joseph, ‘‘Sequoia 2000 Network (S2Knet) Handbook,’’ *Sequoia 2000 Technical Report 92/14,* (August, 1992)

[STON91] Stonebraker, Michael and Dozier, Jeff ‘‘Sequoia 2000: Large Capacity Object Servers to Support Global Change Research,’’ *Sequoia 2000 Technical Report 91/1,* (July, 1991)