Mach Bibliography

MT XINU, Inc.
2560 Ninth Street
Berkeley, CA 94710

September 1, 1989
Version 0.0

This bibliography suggests background reading about Mach for the forthcoming MT XINU/CMU operating system release.[1] The release will contain the Mach kernel and libraries, based on CMU Release 2.5; both the Network File System (NFS) and the Andrew File System (AFS); the X Window System, Version 11, Release 3 (including the Andrew Toolkit and Message System); Camelot; the 4.3BSD-tahoe libraries and utilities; and various user-contributed software.

We have tried to include enough information about unpublished works so that you will be able to obtain copies without too much trouble. Whenever possible, we have included where to get these documents by anonymous FTP.

[1] Mike Accetta, Robert Baron, David Golub, Richard Rashid, Avadis Tevanian, and Michael Young. *Mach: A New Kernel Foundation for UNIX Development*. Technical Report, School of Computer Science, Carnegie Mellon University, Pittsburgh, August 1986. Also in *Proceedings of the Summer 1986 USENIX Conference*, pp. 93–112, July 1986.

This paper describes Mach and the motivations which led to its design. Also described are some of the details of its implementation and status at the time of writing.

[2] Robert V. Baron, David Black, William Bolosky, Jonathan Chew, Richard P. Draves, David B. Golub, Richard F. Rashid, Avadis Tevanian, Jr., and

[1] This material was developed under a subcontract sponsored by the Defense Advanced Research Projects Agency (DoD), ARPA order 4864, monitored by the Space and Naval Warfare Systems Command under Contract Number N00039-87-C-0251. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing offical policies, either expressed or implied, of the Defense Advanced Research Projects Agency or Department of the Navy, Space and Naval Warfare Systems Command, or Carnegie Mellon University, unless designated by other documentation.

Michael Wayne Young. Mach Kernel Interface Manual. October 1988. Unpublished manuscript from the School of Computer Science, Carnegie Mellon University.

This manual describes the interface to the Mach kernel in detail for a wide variety of uniprocessor and multiprocessor architectures.

Copies are available on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/manual.ps through anonymous FTP. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by sending a message to mach@cs.cmu.edu or by calling 412/268-7665.

[3] David L. Black. Mach Interface Proposals—Priorities, Handoff, Wiring. August 1989. Unpublished manuscript from the School of Computer Science, Carnegie Mellon University.

This paper proposes three sets of changes: 1) export priorities on individaul threads to users, introducing the notion of scheduling policies and support for fixed-priority threads, 2) handoff scheduling, which specifies the next thread to run and related mechanisms, and 3) wiring, which locks data and threads into memory to prevent paging and swapping.

Copies are available through anonymous FTP on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/rfc/priority.ps.

[4] David L. Black. Mach Processor Allocation Interface. August 1989. Unpublished manuscript from the School of Computer Science, Carnegie Mellon University.

This document is a basic proposal for a processor allocation and control kernel interface which supports control over processors by user-mode programs. Application areas include gang scheduling and a method for speeding up benchmarks; both require allocating a specific number of processors to an application for the exclusive use of that application.

Copies are available on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/rfc/cpu.ps through anonymous FTP.

[5] David L. Black. The Mach cpu_server: An Implementation of Processor Allocation. August 1989. Unpublished manuscript from the School of Computer Science, Carnegie Mellon University.

This document describes the cpu_server, a user-mode server which performs processor allocation for the Mach operating system.

Copies are available through anonymous FTP on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/rfc/cpu_server.ps.

[6] David L. Black, David B. Golub, Richard F. Rashid, Avadis Tevanian, Jr., and Michael W. Young. *The Mach Exception Handling Facility*. Technical

Report CMU-CS-88-129, School of Computer Science, Carnegie Mellon University, April 1988.

Exceptions are caused by the occurrence of unusual conditions during program execution; raising an exception invokes the operating system to manage recovery from the unusual condition. Although many exceptions can be handled and dismissed transparently to the user (e.g., page faults), some must be exported in some form (invoke handler, core dump, etc.). This paper concerns the design and implementation of exception handling facilities that perform this exporting for general purpose operating systems. It discusses both the overall design principles for these facilities and the specific design and implementation of an exception handling facility for the Mach operating system. It also discusses the support provided to applications that use exception (e.g., debuggers) by this facility and other Mach kernel facilities.

You may order printed copies from the School of Computer Science, Carnegie Mellon University by sending a message to mach@cs.cmu.edu or by calling 412/268-7665.

[7] Eric C. Cooper and Richard P. Draves. *C Threads*. Technical Report CMU-CS-88-154, School of Computer Science, Carnegie Mellon University, February 1988.

The C Threads package allows parallel programming in C under the Mach operating system. The package provides multiple threads of control within a single shared address space, mutual exclusion locks for protection of critical regions, and condition variables for thread synchronization.

Copies are available on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/threads.ps through anonymous FTP. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by calling 412/268-7665.

[8] Richard P. Draves, Michael B. Jones, and Mary R. Thompson. MIG—The MACH Interface Generator. July 1989. Unpublished manuscript from the School of Computer Science, Carnegie Mellon University.

Matchmaker is a language for specifying and automating the generation of multilingual interprocess communication interfaces. MIG is an interim implementation of a subset of the Matchmaker language that generates C and C++ remote procedure call interfaces for interprocess communication between Mach tasks.

Copies are available on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/mig.ps through anonymous FTP. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by calling 412/268-7665.

[9] R. F. Rashid. From RIG to Accent to Mach: The Evolution of a Network Operating System. In *Proceedings of the ACM/IEEE Computer Society, Fall Joint Computer Conference,* ACM, November 1986.

This paper describes experiences gained during the design, implementation and use of the CMU Accent Network Operating System, its predecessor, the University of Rochester RIG system and its successor CMU's Mach multiprocessor operating system. It outlines the major design decisions on which the Accent kernel was based, how those decisions evolved from the RIG experiences and how they had to be modified to properly handle general purpose multiprocessors in Mach. Also discussed are some the the major issues in the implementation of message-based systems, the usage patterns observed with Accent over a three year period of extensive use at CMU and a timing analysis of various Accent functions.

[10] Richard Rashid, Avadis Tevanian, Jr., Michael Young, David Golub, Robert Baron, David Black, William Bolosky, and Jonathan Chew. *Machine-Independent Virtual Memory Management for Paged Uniprocessor and Multiprocessor Architectures*. Technical Report CMU-CS-87-140, School of Computer Science, Carnegie Mellon University, July 1987. Also in *Proceedings of the 2nd Symposium on Architectural Support for Programming Languages and Operating Systems*, ACM, October, 1987.

This paper describes the design and implementation of virtual memory management within the CMU Mach Operating System and the experiences gained by the Mach kernel group in porting that system to a variety of architectures. As of this writing, Mach runs on more than half a dozen uniprocessors and multiprocessors including the VAX family of uniprocessors and multiprocessors, the IBM RT PC, the SUN 3, the Encore MultiMax, the Sequent Balance 21000 and several experimental computers. Although these systems vary considerably in the kind of hardware support for memory management they provide, the machine-dependent portion of Mach virtual memory consists of a single code module and its related header file. This separation of software memory management from hardware support has been accomplished without sacrificing system performance. In addition to improving portability, it makes possible a relatively unbiased examination of the pros and cons of various hardware memory management schemes, especially as they apply to the support of multiprocessors.

[11] Avadis Tevanian, Jr. *Architecture-Independent Virtual Memory Management for Parallel and Distributed Environments*. PhD thesis, School of Computer Science, Carnegie Mellon University, December 1987. Also as Technical Report CMU-CS-88-106 from the School of Computer Science, Carnegie Mellon University, 1987.

This thesis describes the design and implementation of a new, portable, memory managment system and evaluates it by direct comparison with commercially developed memory management systems. It is in production use at CMU, a number of industrial organizations and research laboratories, and is a central component of the Mach operating system, a distributed and parallel operating system developed at Carnegie Mellon University. At the time of this writing, it runs on more than half a dozen uniprocessors and multiprocessors including the DEC VAX family of uniprocessors and multiprocessors, the IBM RT PC, the SUN 3, the Encore MultiMax, the Sequent Balance 21000 and several experimental computers.

[12] Avadis Tevanian, Jr. and Richard F. Rashid. *Mach: A Basis for Future UNIX Development*. Technical Report CMU-CS-87-139, School of Computer Science, Carnegie Mellon University, Pittsburgh, June 1987.

Computing in the future will be supported by distributed computing environments. These environments will consist of a wide range of hardware architectures in both the uniprocessor and multiprocessor domain. This paper discusses Mach, an operating system under development at Carnegie Mellon University, that has been designed with the intent to integrate both distributed and multiprocessor functionality. In addition, Mach provides the foundation upon which future Unix development may take place in these new environments.

[13] Avadis Tevanian, Jr., Richard F. Rashid, David B. Golub, David L. Black, Eric Cooper, and Michael W. Young. *Mach Threads and the Unix Kernel: The Battle for Control*. Technical Report CMU-CS-87-149, School of Computer Science, Carnegie Mellon University, August 1987.

This paper examines a kernel implemented lightweight process mechanism built for the Mach operating system. The pros and cons of such a mechanism are discussed along with the problems encountered during its implementation.

[14] Avadis Tevanian, Jr., Richard F. Rashid, Michael Young, David B. Golub, Mary R. Thompson, William Bolosky, and Richard Sanzi. *A Unix Interface for Shared Memory and Memory Mapped Files Under Mach*. Technical Report, School of Computer Science, Carnegie Mellon University, Pittsburgh, July 1987.

This paper describes an approach to Unix shared memory and memory mapped files currently in use at CMU under the Mach Operating System. It describes the rationale for Mach's memory sharing and file mapping primitives as well as their impact on other system components and on overall performance.

[15] Mary R. Thompson. Mach Environment Manager. July 1988. Unpublished manuscript from School of Computer Science, Carnegie Mellon University.

The Environment Manager is a Mach server which facilitates the sharing of named variables between tasks. This paper discusses types, primitives, and integration with the Unix environment.

Copies are available on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/envmgr.ps through anonymous FTP. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by calling 412/268-7665.

[16] Linda R. Walmer and Mary R. Thompson. A Programmer's Guide to the Mach System Calls. February 1988. School of Computer Science, Carnegie Mellon University.

This document is one of two tutorials designed to teach basic Mach programming skills. This manual explains the use of the Mach kernel calls. The companion document to this one, *A Programmer's Guide to the Mach User Environment* explains the use of higher level methods for implementing multi-threaded programs and interprocess communication. Before writing programs that use the system calls directly, the user should be aware that the methods outlined in the other document may be used to solve his problem more simply.

Copies are available through anonymous FTP on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/machsys.ps. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by calling 412/268-7665.

[17] Linda R. Walmer and Mary R. Thompson. A Programmer's Guide to the Mach User Environment. February 1988. School of Computer Science, Carnegie Mellon University.

This document is one of two tutorials designed to teach basic Mach programming skills. This manual demonstrates the use of the C Threads library primitives in writing a multithreaded program and the use of the Mach Interface Generator (MIG) to generate remote procedure calls for interprocess communication. The reader should be familiar with the basic Mach abstractions of ports, messages, virtual memory, tasks and threads which the introduction to the companion document, *A Programmer's Guide to the Mach System Calls*, explains.

Copies are available through anonymous FTP on host *wb1.cs.cmu.edu* in /usr/mach/public/doc/machuses.ps. You may also order printed copies from the School of Computer Science, Carnegie Mellon University by calling 412/268-7665.

[18] Michael Young, Avadis Tevanian, Jr., Richard Rashid, David Golub, Jeffrey Eppinger, Jonathan Chew, William Bolosky, David Black, and Robert Baron. *The Duality of Memory and Communication in the Implementation of a Multiprocessor Operating System*. Technical Report CMU-CS-87-155, School of Computer Science, Carnegie Mellon University, August 1987. Also in *Proceedings of the 11th Symposium on Operating Systems Principles*, November, 1987.

An important component of the Mach design is the use of memory objects which can be managed either by the kernel or by user programs through a message interface. This feature allows applications such as transaction management systems to participate in decisions regarding secondary storage management and page replacement. This paper explores the goals, design and implementation of Mach and its external memory management facility. The relationship between

memory and communication in Mach is examined as it relates to overall performance, applicability of Mach to new multiprocessor architectures, and the sturcture of application programs.