# Final Report on Research in Parallel Computing: Very Large Scale Operating Systems 1990 - 1993

6 December 1995

**Carnegie Mellon University**
**Computer Science Department**

## Abstract

This report documents the progress Carnegie Mellon's Mach Group made in research in very large scale operating systems, for the period 1990-1993. Parts of the Mach Operating system have been incorporated in a number of commercial operating systems, including Encore's Multimax, NeXT OS, MachTen for the Macintoshes, Omron's Luna, OSF/1 for the DEC Alpha, and IBM's OS/2 for the RS6000 based machines.

This research was sponsored by the Advanced Research Projects Agency, DoD, through ARPA Order 7330, and monitored by the Space and Naval Warfare Systems Command under contract MDA972-90-C-0035. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the Advanced Research Projects Agency or the United States Government.

# Introduction

Work on the Mach operating system began in 1985, and throughout the project, research goals have included:

- Providing interprocess communication functionality at the kernel level and using it as a building block for the rest of the system.
- Providing virtual memory support in the kernel and in user level servers. Providing kernel-level support for lightweight threads.
- Supporting closely and loosely coupled multiprocessors and various commercially available workstations.
- Designing a microkernel architecture that limits the functions supported by the microkernel and enables multiple user level servers to support various Application and Programming Interfaces.
- Maintaining at least one UNIX-style API, enabling the Mach system to support the everyday needs of project members and other researchers.
- Distributing Mach technology to other research and commercial sites to use as a basis for further development or products.

At this time the project has come to a close at Carnegie Mellon. Parts of the Mach operating system have been incorporated in a number of commercial operating systems including:

- Encore's Multimax
- NeXT OS
- MachTen for the Macintoshes
- Omron's Luna
- OSF/1 for the DEC Alpha
- IBM's OS/2 for the RS6000 based machines.

## 1 Kernel development

Our work on Mach has produced a highly portable OS kernel that can operate on a variety of multiprocessor architectures, flexible enough to support a variety of operating system environments, and fast enough to be speed competitive with traditionally implemented timesharing systems.

In Mach 3.0, primitive abstractions such as address spaces, task creation and destruction, and communication are implemented in the kernel, with higher-level services implemented in a separate protection domain as a server.

Mach-US is an OS developed as part of the Mach project. It is comprised of a set of servers, each of which supports orthogonal system services. For example, instead of one server supplying all of the system services as under the Mach single server (UX), the Mach Multiserver (Mach-US) has several servers: a task server, a file server, a tty server, an authentication server, a network server, etc. It also has and emulation library that is mapped dynamically into each user process, and uses the system servers to support the API of the UNIX operating system.

Specific enhancements to Mach during this contract are detailed below.

### Realtime scheduler

The objective of Real-Time Mach (RT-Mach) has been to develop a real-time version of Mach that can support a predictable real-time computing environment together with a real-time toolset. To complement Mach's high portability, we wanted RT-Mach to provide a common real-time computing environment in various machine architectures, including single-board, computer-based targets.

Toward these goals, we:

- Integrated the Realtime Mach scheduler with the pure kernel and demonstrated it at the OFS Mach 3.0 Collaborators meeting.
- Rewrote large portions of the standard Mach scheduler to incorporate the features of the Realtime Mach support for distributed systems and to improve the scheduler's performance on multiprocessors. [Savage and Tokuda 93]

### Using continuations to improve the performance of IPC and context switching

The initial Mach 3.0 IPC facility efficiently supported many different styles of communication, including server-client remote procedure calls, distributed object-oriented programming, and streams. The Mach IPC interface allowed intermediaries to extend the IPC facility transparently, and a user-level server extended Mach IPC across networks.

We have made the following modifications to the kernel to improve performance:

- Modified the pure kernel to improve the IPC facility and to reduce physical memory requirements for large, multithreaded applications. An essential part of this work was eliminating from Mach threads the requirement to have kernel stacks. [Draves 91a]
- Rewrote the pure kernel's context switching code to use continuations *explicitly* when transferring control between contexts. This change reduced memory requirements for kernel thread stacks by 80%; in addition, it improved the performance of IPC facilities by 15% and of exception handling by 300%.
- Modified the IPC interface to improve network message sequencing.
- Adapted Mach's **C**-threads implementation to employ the ''continuation'' technology used in the kernel. [Dean 93]

### Default pager

- Moved the Mach 3.0 default pager to the user level [Golub and Draves 91].
- Revised the external pager interface to support dynamically sized and numbered paging files.

## 2 Ports

One principle objective for the Mach operating system has always been portability. In keeping with this goal, we ported Mach 3.0 to the following new architectures as they arrived:

### i386 multiprocessors: Sequent and Corollary

- Jointly with the Open Software Foundation, ported the Mach 3.0 kernel to the i386 Corollary multiprocessor.
- Completed porting Mach 3.0 to the 20-CPU Sequent multiprocessor.
- Improved the performance of multithreaded applications on a Sequent Symmetry multiprocessor by modifying the **C**-threads library.

### Motorola-based machines

- Successfully booted Mach 3.0 and Carnegie Mellon's UNIX single-server on the Omron (Motorola 88K) shared-memory multiprocessor. The new system can now run simple user programs.

### Port to virtually indexed cache architectures

- Jointly with the University of Utah, ported Mach 3.0 to the 64-bit HP-800 (Snake) workstation.
- Designed virtual-memory structures that can handle HP800's large address space and merged them with Mach's machine-dependent code for managing the HP-800 virtually indexed cache.
- Devised a consistency model for managing virtually indexed caches and implemented it in the HP-720 machine. We demonstrated that, compared to other virtual-memory management strategies, this model manages aliases, mappings, and direct memory access cleanly and efficiently. [Wheeler and Bershad 92]

### DEC platforms

- Upgraded the Mach 3.0 Pmax port to the DS5000/120 architecture.
- Built a simulator for DEC's shared-memory Alpha multiprocessor, a high-performance, 64-bit RISC architecture, and ported the Mach 3.0 kernel and the UNIX single server to it. When we received Alpha-based machines, we developed Mach 3.0 on them and evaluated their performance.

### NORMA architectures

- We ported the 3.0 kernel to the Intel 860 Hypercube.

## 3 Single servers

### The POE bootstrap server

- Designed and implemented a bootstrap server that acts as initial software environment atop the pure kernel and allows the user to run UNIX programs.
- Upgraded the bootstrap server by including Berkeley networking support, such as TCP and sockets, and standalone booting of the multiserver UNIX. Released this unrestricted software version to all sites through SUP, Carnegie Mellon's electronic distribution facility.

### BSD 4.4 single server

- Implemented a Mach/BSD-4.4 server that incorporates a 0-free networking protocol.
- Successfully booted Mach 3.0 and BSD-4.4 (Berkeley Networking Tape) on an i386 machine.

### Mach-UX single server

- Analyzed and improved Mach 3.0/UNIX single server's networking (TCP/IP) performance twofold by modifying device driver, kernel, and server code. This improvement makes the networking component competitive with commercial implementations.
- Incorporated in the UNIX server a VFS interface that allows running NFS code and the new ''in-server'' AFS 3.1.
- Jointly with OSF we completed OSF/1, a commercial UNIX server for Mach 3.0.
- Integrated Coda, Carnegie Mellon's distributed file system, with the UNIX server.
- Implemented user-space UDP sockets, which provide a tenfold performance improvement over sockets running in the UNIX server.
- Improved the UNIX server by identifying and removing several major bottlenecks between it and the kernel. We also incorporated performance debugging tools in the UNIX development tree.

### DOS server

- Continued developing our DOS application server on the i386 platform. We tested it on large applications, including Microsoft 3.0 Windows, enhanced it by providing access from DOS applications to the Mach/UNIX file system, and released the server to several alpha test sites [Malan et al 91].
- We also upgraded Mach to support DOS 5.0 and distributed the MS-DOS/Mach-microkernel environment to the academic computing community [Forin and Malan 94].

### Shared-memory server for NORMA architectures

- Developed a new, ''distributed shared-memory'' server for no-remote-memory-multiprocessor architectures (NORMA) and tested it on the i386 Hypercube and the Nectar network system.
- Jointly with Intel, ported the 3.0 kernel to the Intel 860 Hypercube (in preparation to porting to the Intel Touchstone) and tested it with simple message-passing programs.
- Implemented a more sophisticated network shared-virtual-memory server for no-remote-memory-access NORMA architectures [Barrera 91].
- Jointly with Intel we created transparently-shared libraries and ported the UNIX single-server to the iPSC/860 Hypercube/Mach 3.0 kernel.
- Ported Mach's NORMA code to the MIPS R3000-based DECstation and brought it up on a cluster of these workstations.
- Constructed tracing software for monitoring and analyzing UNIX system-call traffic on a NORMA platform.

## 4 Mach-US multiserver

The goal of the Mach-US effort was to develop a new operating-system architecture with a set of servers that supply generic operating-system functionality atop the Mach 3.0 microkernel. A further goal was to support application programs written to industry-standard OS interfaces, e.g., UNIX, using application space libraries, which call the generic servers as needed [Guedes and Julin 91, Julin et al. 91].

- Reviewed the multiserver UNIX environment at the OFS Mach 3.0 Collaborators meeting.
- Extended the multiserver environment by providing support for UNIX shells, editors, and compilers. Demonstrated these programs for DARPA.
- Jointly with the Open Software Foundation, converted the key components of the multiserver UNIX to **C**$^{++}$ to improve their performance and language accessibility.
- Extended the UNIX multiserver by providing self-hosting capability.
- Booted our multiserver UNIX on a multiprocessor architecture.
- Accessed the UNIX multiserver via telnet by utilizing the X kernel for protocol support.
- Distributed the prerelease Mach-US multiserver to several research sites.

## 5 Networking support

Performance measurements of network protocols in microkernel systems have in the past been discouraging, typically 2 to 5 times slower than comparable macrokernel systems [Maeda and Bershad 92]. Toward improving network support in the Mach microkernel system, we:

- Analyzed the Mach 3.0 networking layers and demonstrated that network latency is due to the UNIX server implementation and not the microkernel.
- Evaluated the UNIX socket interface on a DECstation 5000/200 and found that a mapped strategy provides 15% faster throughput for packets of 4KB and larger. Integrating this improvement brings us closer to evaluating the microkernel's networking support.
- Implemented initial drivers for point-to-point (non-switch-based) communication as a step toward providing microkernel support for low-latency, high-bandwidth networking.
- Completed the microkernel's ATM networking driver and integrated it with higher-level IP software. Resulting system performance was within a factor of two of that obtained with the raw networking hardware. The discrepancy is due to the architecture's lack of DMA capabilities.
- Finished evaluating the ATM software and identified modifications that will improve the drivers' performance: The reliable protocols need low-level support; for NORMA usage the ATM drivers need to be more tightly integrated with the kernel; and the ATM software requires multiplex routing to support IP communication over transient virtual circuits.
- Integrated new ATM protocols in the Mach distribution.
- Completed a user-level TCP/IP emulation that enables full UNIX-style networking without operating-system interaction.
- Designed a new packet filter interface to support large numbers of communication endpoints.
- Integrated user-level IP protocols in the Mach distribution.

## 6 The Midway distributed shared-memory system

Midway is a distributed-shared-memory programming system supporting multiple memory consistency models within a single parallel program. Midway is intended for use on medium-scale multicomputers. Milestones in the development of Midway include:

- Ran distributed-shared-memory programs on top of our distributed-shared-memory system, which is now stable on two nodes.
- Modified the compiler for our distributed-shared-memory environment so that it allows executing non-annotated programs and using the runtime system as a guide to where performance annotations are required.
- Imported Stanford University's SPLASH suite for our distributed-shared-memory system.
- Compared several of the Midway environment's memory-consistency models for different data items within single tasks. Results indicate that, in properly written parallel programs, Midway's support for entry consistency greatly reduces the number of transferred messages and data.
- Successfully executed code from C-PARA-OPS-5, a large-scale parallel production system, in Midway.
- Made the ATM protocols more effective for the Midway environment.
- Reduced the runtime overhead of Midway's memory-consistency implementation.

## 7 Related studies and services

- Built and evaluated, on DECstation 3100 and 5000 architectures, user-state drivers for Ethernet and disk devices that handle device-related processing inside Mach tasks.
- Completed three, functionally equivalent implementations of the X11-R4 windowing protocols and compared their performance. The results showed that the Mach-3.0 IPC implementation is competitive with one based on UNIX sockets, while the client/server shared-memory implementation increased windowing operations by 25% [Ginsberg et al. 93].
- For debugging i386-based machines over a network we implemented basic, source-level, Mach-kernel capability that uses a modified Gnu debugger.
- Designed and evaluated microkernel cache-management policies to prevent caching side effects. Wrote guidelines for running microbenchmark tests on high-performance RISC processors. [Bershad et al. 92a]
- Converted the first set of Mach course notes into distributable form.
- Simulated memory-consistency models (such as sequential, weak, release, lazy) for various processor and network configurations and compared their performance.
- Compared Mach 3.0 and Ultrix and found that, for similar functions, Mach executes more instructions, and each incurs a higher memory cost due to caching effects. [Chen 94a]
- Developed WAX, a failure-resilient programming environment for parallel/distributed systems, and used it to implement several applications.

# 8 Technology Transfer

Another goal of the Mach project was to enable other research sites to make use of our work as quickly as possible. This goal contributed to the decision to provide binary compatibility with a standard UNIX environment, i.e. BSD 4.3. This compatibility was obtained at the cost of using BSD licensed code, which in turn complicated the distribution procedures. This goal also required that we support timely distribution of new releases, facilities for Mach users to trade information, and an easily accessible repository of Mach information.

During the contract period we maintained the following mechanisms:

### Code distribution

After 1990 most code was distributed over the network using the Software Distribution Protocol (SUP) program developed by the Carnegie Mellon's facilities support group. This program allowed client sites to obtain automatically the latest versions of the distribution and required minimal support from Carnegie Mellon.

Our distributions include:

- In 1990, tape distributions of UNIX-compatible, licensed Mach 2.5 systems (including sources and binaries) for Sun-3, Vax, and IBM PC/RT platforms. Subsequently we made similar distributions via SUP for the Sun-3, VAX, i386/486, and IBM RS6000 platforms.
- Unlicensed FTP distributions of Mach 3.0 micro-kernel sources for i386, DECstations 2100, 3100, and 5000, and Omron Luna44k platforms.
- Unlicensed network distributions of the POE single server, MDOS server, and OSF tools to build our sources.
- Unlicensed SUP distribution of kernel sources for some DEC Alpha platforms.
- Some limited distribution of a prerelease version of the Mach Multiserver code to sites that requested it.

Approximately 300 sites received some version of the complete Mach systems, and many other sites anonymously FTP'ed the kernel sources.

### Mach information repository

In addition to the unlicensed code, the Mach public FTP directory contained:

- An annotated bibliography of all published Mach papers
- On-line Postscript and ASCII versions of most of the published papers, unpublished notes, reference manuals, and code examples
- A set of answers to frequently asked questions, either composed by Mach project members or saved from the comp.os.mach bulletin board.

Recently this whole area has been set up for World Wide Web browsers as well as standard FTP access.

### Support Services

Our project personnel support two subscription mailing lists, each with a membership of about

250. One of these lists is devoted to general questions about Mach 3.0, the other to questions about the Mach i386 release and its installation procedure.

Another mailing list was monitored by a project member who answered questions about how to receive Mach information and distributions and provided help to sites first installing Mach.

Furthermore, project members have been answering questions posed on the comp.os.mach Internet bulletin board.

# I. Bibliography

[Barrera 91] Barrera, J.S.
A fast Mach network IPC implementation.
In *Proceedings of the Second USENIX Mach Symposium*. USENIX, November, 1991.

[Beguelin and Seligman 93]
Beguelin, A., and E. Seligman.
*Causality-preserving timestamps in distributed programs*.
Technical Report CMU-CS-93-167, School of Computer Science, Carnegie Mellon University, June, 1993.

[Bershad 91a] Bershad, B.N.
*Mutual exclusion for uniprocessors*.
Technical Report CMU-CS-91-116, School of Computer Science, Carnegie Mellon University, April, 1991.

[Bershad 91b] Bershad, B.N.
*Practical considerations for lock-free concurrent objects*.
Technical Report CMU-CS-91-183, School of Computer Science, Carnegie Mellon University, September, 1991.

[Bershad 92] Bershad, B.N.
The Increasing Irrelevance of IPC Performance for Microkernel-Based Operating Systems.
In *Proceedings of the 1992 USENIX Workshops on Microkernels*. USENIX, , 1992.

[Bershad and Zekauskas 91]
Bershad, B.N., and M.J. Zekauskas.
*Midway: Shared-memory parallel programming with entry consistency for distributed memory multiprocessors*.
Technical Report CMU-CS-91-170, School of Computer Science, Carnegie Mellon University, September, 1991.

[Bershad et al. 92a]
Bershad, B., R.P. Draves, and A. Forin.
Using Microbenchmarks to Evaluate System Performance.
In *Proceedings of the Third Workshop on Workstation Operating Systems*. WWOS-3, April, 1992.

[Bershad et al. 92b]
Bershad, B.N., D.D. Redell, and J.R. Ellis .
Fast Mutual Exclusion for Uniprocessors.
In *Fifth Symposium on Architectural Supports for Programming Languages and Operating Systems*. ASPLOS V, October, 1992.

[Bershad et al. 93] Bershad, B.N., M.J. Zekauskas, and W.A. Sawdon.
*The Midway Distributed Shared Memory System*.
Technical Report CMU-CS-93-119, School of Computer Science, Carnegie Mellon University, March, 1993.

[Brustoloni and Bershad 92]
Brustoloni, J.C., and B.N. Bershad.
*Simple Protocol Processing for High-Bandwith Low-Latency Networks*.
Technical Report CMU-CS-93-132, School of Computer Science, Carnegie Mellon University, March, 1992.

[Chen 93] Chen, J.B.
Software Methods for System Address Tracing.
In *Proceedings of The Fourth Workshop on Workstation Operating Systems*. WWOS, October, 1993.

[Chen 94a] Chen, J.B.
*The Impact of Software Structure and Policy on CPU and Memory System Performance*.
PhD thesis, School of Computer Science, Carnegie Mellon University, March, 1994.

[Chen 94b] Chen, J.B.
Memory Behavior of an X11 Window System.
In *Proceedings of the Winter USENIX Conference*. USENIX, January, 1994.

[Dean 93] Dean, R.
Using Continuations to Build a User-Level Threads Library.
In *Proceedings of the Third USENIX Mach Conference*. USENIX, April, 1993.

[Draves 91a] Draves, R.P., B.N. Bershad, R.F. Rashid, and R.W. Dean.
*Continuations: Unifying thread management and communication in operating systems*.
Technical Report CMU-CS-91-115, School of Computer Science, Carnegie Mellon University, March, 1991.

[Draves 91b] Draves, R.P.
Page replacement and reference bit emulation in Mach.
In *Proceedings of the Second USENIX Mach Symposium*. USENIX, November, 1991.

[Forin 91] Forin, A., D. Golub, and B.N. Bershad.
*An I/O system for Mach 3.0*.
Technical Report CMU-CS-91-191, School of Computer Science, Carnegie Mellon University, October, 1991.

[Forin and Malan 94]
Forin, A., and G.R. Malan .
An MS-DOS File System for UNIX.
In *Proceedings of the Winter USENIX Conference*. USENIX, January, 1994.

[Ginsberg et al. 93]
Ginsberg, M., R.V. Baron, and B. N. Bershad .
Using the Mach Communication Primitives in X11.
In *Proceedings of the Third USENIX Mach Conference*. USENIX, April, 1993.

[Golub and Draves 91]
Golub, D.B., and R. Draves.
Moving the Default Memory Manager out of the Mach Kernel.
In *Proceedings of the USENIX Mach Symposium*. USENIX, November, 1991.

[Guedes and Julin 91]
Guedes, P., and D. Julin.
Object-Oriented Interfaces in the Mach 3.0 Multi-Server System.
In *Proceedings of the Second International Workshop on Object Orientation in Operating Systems*. IEEE, October, 1991.

[Jones 91] Jones, M.B.
Bringing the C libraries with us into a multi-thread future .
In *Proceedings of 1991 Winter USENIX Conference*. USENIX, January, 1991.

[Jones 92] Jones, M.B.
Transparently Interposing User Code at the System Interface.
In *Proceedings of the Third Workshop on Workstation Operating Systems*. WWOS-3, April, 1992.

[Julin et al. 91] Julin, D., J.J. Chew, J.M. Stevenson, P. Guedes, P. Neves, and P. Roy.
Generalized Emulation Services for Mach 3.0: Overview, Experiences and Current Status.
In *Proceedings of the USENIX Mach Symposium*. USENIX, November, 1991.

[Maeda and Bershad 92]
Maeda, C., and B.N. Bershad.
Networking Performances for Microkernels.
In *Proceedings of the Third Workshop on Workstation Operating Systems*. WWOS-3, April, 1992.

[Maeda and Bershad 93a]
Maeda, C., and B.N. Bershad.
Protocol Service Decomposition for High-Peformance Networking.
In *Proceedings of the 14th ACM Symposium on Operating Principles*. ACM, December, 1993.

[Maeda and Bershad 93b]
Maeda, C., and B.N. Bershad.
*Service without servers*.
Technical Report CMU-CS-93-144, School of Computer Science, Carnegie Mellon University, August, 1993.

[Malan et al 91] Malan, G., R. Rashid, D. Golub, and R. Baron.
DOS as a Mach 3.0 application.
In *Proceedings of the Second USENIX Mach Symposium*. USENIX, November, 1991.

[Savage and Tokuda 93]
Savage, S., and H. Tokuda.
Realtime Mach: Exporting time to the user.
In *Proceedings of the Third USENIX Mach Symposium*. USENIX, April, 1993.

[Stodolsky et al. 93]
Stodolsky, D., J.B. Chen, and B.N. Bershad.
*Fast Interrupt Priority Management in Operating Systems Kernels*.
Technical Report CMU-CS-93-152, School of Computer Science, Carnegie Mellon University, May, 1993.

[Subramanian 91] Subramanian, I.
Managing discardable pages with an external pager.
In *Proceedings of the Second USENIX Mach Symposium*. USENIX, November, 1991.

[Tokuda 91] Tokuda, H., and T. Nakajima.
Evaluation of realtime synchronization in realtime Mach.
In *Proceedings of the Second USENIX Mach Symposium*. USENIX, November, 1991.

[Wheeler and Bershad 92]
Wheeler, R., and B.N. Bershad.
Consistency Management for Virtually Indexed Caches.
In *Fifth Symposium on Architectural Support for Programming Languages and Operating Systems*. ASPLOS V, October, 1992.

[Yuhara et al. 94] Yuhara, M., B.N. Bershad, C. Maeda, J. Eliot, and B. Moss.
Efficient Packet Demultiplexing for Multiple Endpoints and Large Messages.
In *Proceedings of the 1994 Winter USENIX Conference*. USENIX, January, 1994.

# Table of Contents

**Introduction** **1**
- **1 Kernel development** **1**
- **2 Ports** **2**
- **3 Single servers** **3**
- **4 Mach-US multiserver** **5**
- **5 Networking support** **5**
- **6 The Midway distributed shared-memory system** **6**
- **7 Related studies and services** **6**
- **8 Technology Transfer** **7**

**I. Bibliography** **9**