University of Southern California
Department of Contracts and Grants
Los Angeles, CA 90089-1147

DIstributed VIRtual System (DIVIRS) Project

formerly

Center for Experimental Research in Parallel Algorithms, Software, and Systems

Final Report
December 1996

Principal Investigator:
Herbert Schorr
Co-principal Investigator
B. Clifford Neuman
Stockton R. Gaines
David Mizell
USC/Information Sciences Institute

Prepared under NASA Cooperative Agreement NCC 2-539
for Henry Lum, Technical Officer
NASA Information Sciences Division 244-7

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Aeronautics and Space Administration, the Defense Advanced Research Projects Agency, or the U.S. Government.

Final Report
Covers period 1 May 1988 through 30 May 1996

As outlined in the original proposal and continuation proposals on NASA cooperative agreement NCC 2-539, ISI has conducted research in three areas: operating system development for multiprocessor systems, resource management for parallel and distributed systems, and operating system services for large scale distributed systems.

Our work on the DIVIRS/CERPASS effort was conducted during the period of May 1, 1988 through July 1995. The funding of this effort covered work during this period only. The contract was subsequently extended with a no-fund extension through May 1996 to allow completion of work on a separate AASERT award, supporting a graduate student working on electronic commerce. The AASERT award was attached to this contract and required that the contract remain in place, but funding for the graduate student was provided independently.

This report will summarize our efforts and accomplishments in all areas.

Phase 1 - May 1, 1988 Through November 30, 1991

OS Development for Multiprocessor systems

The Center for Experimental Research in Parallel Algorithms, Software and Systems (CERPASS) provided an experimental facility for researchers on the Internet. The Center's parallel computing resources included a Connection Machine CM-2 and a Symult Series 2010. Both the CM-2 and the Symult S2010 were made accessible from the Internet via their VAX and Sun Front Ends, respectively.

The first machine installed at ISI was a Connection Machine Model CM-2. This CM-2 was a data-parallel, single-instruction, multiple data-stream (SIMD) machine. It consisted of 16K (16,384) processors, each with 64K-bit local memory. These processors were interconnected via a hypercube network that allowed configuration at the software level as an N-dimensional cube where N is an integer from 1 to 31 (including linear array and mesh configurations when N is 1 and 2 respectively). The flow control was handled by the front-end. Both a VAX 6210 and a Symbolics 3600 were used, each of which could control either an 8K partition or a 16K partition of the machine. The upper 8K partition of our CM-2 was equipped with a floating-point accelerator, and had two high-bandwidth I/O boards connecting to two I/O devices: the Framebuffer and the Data Vault. The Framebuffer is a direct mapping color display device, which included a SONY color display, and the Data Vault is a 5GB fault-tolerant disk storage system, which was used for fast file transfer. The software available on ISI's CM-2 included *Lisp, C* and PARIS. Most of the programs that we wrote for this machine are in *Lisp or PARIS, each of which were supported on both front-end machines.

The Symult Series 2010 system (S2010) used for this project is a distributed-memory, message-passing, multiple-instruction, multiple data-stream (MIMD) computer based on a rectangular mesh topology. ISI's S2010 has 32 processing nodes, each node consisting of a Motorola 68020 microprocessor as its CPU, and a Motorola 68881 floating-point coprocessor. Each processing node has

8MB of local memory and a custom-designed VLSI routing chip that manages the message routing wholly on the mesh network. The S2010 utilizes a Sun-3 as a front-end. The programming languages available are C and SPCL (Common LISP) with parallel extensions defined in the Cosmic Environment and Reactive Kernel (CE/RK) software.

The results of our efforts in phase 1 of the CERPASS/DIVIRS contract include the provision of computing services on the CM-2, porting of Mach to the Simult Series 2010 system and provision of computing service on the machine, and consulting for others on the problems associated with such environments. Each of these results will be described separately below.

Connection Machine activities

The Connection Machine Model CM-2 is a massively parallel SIMD machine. Its hardware consists of 16,384 data processors interconnected via a hypercube network. Each processor has a bit-serial ALU and a 64K-bit local memory. This CM-2 has two front-end subsystems -- one VAX 6210 and one Symbolics 3600, a 5-Gigabyte mass storage system and a graphic color display device. Half of the machine (i.e., 8,192 processors) is also equipped with floating-point accelerator. Programming languages available on this machine include *Lisp, C* and PARIS interfaces to Common Lisp and C.

Results - Connection Machine activities

Both the Connection Machine Data Vault and floating-point accelerator were installed during the first year of the CERPASS effort. Software release 5.1 for the VAX 6210 and the Symbolics 3600 was installed by Thinking Machines Corporation (TMC) personnel and Dr. Yu-Wen Tung. ISI assisted TMC in installing version 5.1's patch files to both our front-end systems and prepared several bug reports for this new software release.

Initially, ISI had problems with the chip failure rate for the lower 8K partition of the 16K machine, but this was resolved in September when the floating-point accelerator was installed and the TMC people move the original 16 upper boards to where the 16 lower boards had been.

Dr. Tung improved previous implementations for bitonic sorting algorithms by using the "get-from-news" functions for hypercube bitonic sort, and the "get-from-power-two" functions for mesh bitonic sort. These functions were made available only in versions 5.0 and 5.1, respectively, and did show significant speed up over previous implementations.

Dr. Tung also wrote a quicksort algorithm and a bubble sort algorithm in the new version of PARIS, as well as a set of testing programs. He ran them on the Connection Machine. These programs automatically measured the timing and performance of his previous sorting algorithms on the Connection Machine, under various values of VPR (virtual processing ratio) and various argument lengths. (The total execution time was four to five days.)

Dr. Tung conducted research with Dan Moldovan of USC's Department of Electrical Engineering on using the Connection Machine to implement the marker propagation rules for retrieving information stored in semantic networks.

ISI continued to seek new applications for parallel processing research using the Connection Machine. One such research area was the development and testing of a cognitive model of how a novice acquires perceptual skill for visual data. Both the low-level vision system and the high-level neural system in this model will require the computing speed of the Connection Machine and its graphics and direct input/output capabilities.

Richard Bisbey explored the feasibility of using the Connection Machine frame-buffer as a visualization tool, due to its high-bandwidth connection to CM-2 processor memory. He also investigated the use of ISI's 16K processor Connection Machine (CM-2) for processing and displaying 1024x1024x24 (HDTV) video images.

His studies concluded that the CM-2 was too slow in both processing power and I/O bandwidth to do real-time processing of the video images. Non-real-time processing is possible, if the entire video sequence is cached in CM-2 memory prior to display. The current 16K processor configuration (and 128MB of memory) would permit slightly more than one second of 30 frame-per-second video. Upgrading the CM-2 to 64K processors (and 512MB of memory) would permit a commensurately longer sequence. However, the people at TMC believe that real-time display of HDTV images is not possible with the above configuration, and that real-time HDTV display would only be possible with a 64K processor, 8 frame buffer machine. In this case, external hardware would have to be built to merge the video signals from the independent frame buffers.

Dr. Tung provided technical advice and support to the Connection Machine programming group. The Connection Machine programming group acquired several new members during the first year of the CERPASS effort, including: USC's Olivier Bourdon (image processing), UCLA's Rajive Bagrodia and Edmund Kwan (parallel language), and Hughes's Michael Oyster and associates (for their DARPA contract). Old users such as USC's J. S. Chen (image processing) and S. H. Chung (semantic network array processor project) continued to use the machine heavily during the first year of the effort.

In the Spring of 1990, DARPA exchanged a newer model CM-2 for ISI's older CM-2 machine. Engineers from the Thinking Machine Corporation upgraded ISI's CM-2 with a 16K floating-point accelerator. They also installed software release 5.2 on the Symbolics front-end.

During this period, ISI continued to seek new applications for parallel processing research using the Connection Machine. Dr. Tung contacted Professors Prasannakumar and Kai Hwang at USC's Electrical Engineering Department, and Professor Selim Akl at Queen's University, Canada, regarding use of the Connection Machine in their research activities. Prasannakumar was interested in collaborating on parallel vision and other non-numerical algorithms, and Professor Akl wanted to try a few of his algorithms on the Connection Machine.These algorithms include parallel geometry on a grid, optimal parallel algorithms for b-matchings in trees, parallel binary search, optimal parallel algorithms for computing a vertex of the Hitchcock transportation polytope, a parallel algorithm for the assignment problem on complete weighted bipartite graphs, and so on.

Dr. Tung continued his daily technical and administrative support for CM-2 users. He created a mailing list for ISI's CM users for easier information exchange. This list included, price@iris.usc.edu,rom@iris.usc.edu,mzerroug@pollux.usc.edu,wdl@rana.usc.edu, plin@iris.usc.edu, peng@iris.usc.edu, kwan@cs.ucla.edu, rajive@cs.ucla.edu, ato@bellcore.com, thanh@vlsi-cad.isi.edu, bfreeman@silver.ucs.indiana.edu, efreeman@silver.ucs.indiana.edu, and marek@iuvax.cs.indiana.edu

Porting Mach to the Symult 2010

There are a great number of parameters that characterize parallel architectures and their resident operating systems that can be expected to have an influence on system performance. Examples of these include whether a machine is MIMD or SIMD, whether it has shared or local memory, the ratio of its computation speed to its communication speed, I/O bandwidth, processor speed, types of low-level parallelism available within processors, tolerance of memory latency, the effect of caches, synchronization overhead, and operating system characteristics such as context switching speed or the ability to migrate processes for load balancing.

The Mach port methodology involved specifying a systems architecture that supported a NORMA multicomputer hardware environment, including identifying strategies whereby filesystem support could be made available to jobs executing in that environment and internode communications between cooperating tasks was supported. To accomplish this, we (1) treated the underlying communications mesh as a local-area computer network, allowing us to take advantage of existing software resources, (2) placed a minimal instance of the Mach kernel on each node with additional operating system services supplied by the executing servers on selected nodes of the simulated LAN, and (3) provided a global filesystem environment to each node by supporting a diskless workstation strategy for the nodes with remote NFS support handled by the front-end machine.

Results - Porting Mach to the Symult 2010

This section of the report describes the first port of CMU's Mach operating system to a NORMA multicomputer. The target machine was a Symult 2010 multicomputer, a distributed-memory, multiple-instruction, multiple data-stream (MIMD) computer based upon a two-dimensional mesh topology. The machine was installed at ISI as part of the CERPASS project and eventually contained 32, 25MHz 68020 computational node boards, each with 8 MBytes of memory and a resident Motorola 68881 floating-point coprocessor. Each board was attached to the system via a communications coprocessor, connected to a custom-designed VLSI routing chip known as an automatic message routing device (AMRD). Neighboring AMRD chips were interconnected on a backplane via four bidirectional channels in the form of a two dimensional communications mesh, with an additional channel connecting the AMRD to the computational node board. The data path between AMRDs is eight bits wide with advertised data rates on the order of 20 MBytes per second per direction. Access to the communications mesh was via a Sun-3 front-end attached to one corner of the communication mesh.

Unlike Mach, the Cosmic Environment/Reactive Kernel (CE/RK), the native operating system for the 2010 provided by Symult, is primarily designed to optimize message passing between applications running on several nodes. As such, the OS did not provide preemptive scheduling, general purpose virtual memory or interprocess communication, multiprocessing, or paging. To maintain compatibility between CE/RK and Mach, several hybrid software architectures combining Mach and CE/RK were considered. However, due to fundamentally different assumptions made in both operating systems, no approach was found that would successfully combine both operating systems without negatively impacting the implementation of Mach.

For these reasons, it was decided avoid a design combining both systems. Instead, effort was focused on avoiding the introduction of features in the port that would later prevent a developer from partitioning the 2010 into nodes that would run one OS or the other. Furthermore, with careful planning it was felt that the Mach nodes could run existing CE/RK applications with little or no change to existing Symult software.

With respect to Mach itself, another decision was made to run an instance of the Mach operating system on each node, as opposed to developing a multinode kernel. This decision predated information on CMU's microkernel development work, but was made with the assumption that unnecessary features might be stripped from node kernels in the future, with the minimum software required to support a job remaining on each node or provided via user-level servers.

Other features of the CE/RK influenced the Mach port. In particular, the lack of UNIX support severely restricted available software upon which to draw and the multiple binary executable formats and multiple versions of utilities complicated program development. While the 2010 and Sun-3 were substantially different at the hardware level, it was realized that the virtual machine presented by Mach to the application could be nearly identical on both machines. Supporting compatibility for Sun-3 binaries on the nodes provided access to a substantial software base and eliminated incompatibilities between Mach applications running on the 2010 nodes and the front-end.

Other major design issues remained, specifically: the form of interface adopted for the mesh, how the file access and paging were to be supported, how the system was to be initialized, how nodes were to be made available to user processes, and what would be the role of the Front End Sun-3.

It was reasoned that if the mesh interface was implemented as a network interface and supported standard TCP/IP access, many existing software mechanisms could be utilized intact to help expedite the porting effort, including all existing Mach network-based servers. Furthermore, NFS could be utilized to provide a global file system that spanned the node environment and to provide paging on the front-end. Finally, the mesh could be implemented as a subnet on the ethernet with access through the Front End Sun-3 workstation, with each node assigned its own network address and supporting remote user login and command execution from any interior node or exterior host.

While the TCP/IP family of communications protocols was realized to be a poor match to the characteristics of the mesh, the immediate advantages brought to the environment in supporting the protocols far outweighed the anticipated performance penalties incurred in their use. Later effort could be applied to support a better communications protocol.

The model used to implement the network interface stemmed from an existing Sun-3 UNIX device driver provided by Symult for handling mesh I/O for the front-end. Originally, the driver only supported character I/O, but, during Symult's dissolution, ISI obtained an undebugged, unreleased, experimental device driver containing a network interface. This module, designed to drive the Sun Interface Board (SIB) connected to the mesh from the Sun-3 Front End, formed the basis for all mesh communication for both the Sun-3 Front End and the Mach port running on each node. Basing mesh communication on this module not only provided complete BSD network functionality to the port but also provided a high speed messaging mechanism that could be used both by Mach and CE/RK applications.

The experimental SIB driver code was designed to interface to an earlier version of the network distribution software than was supported under the current Mach release. The interface had to be adapted to the current software distribution and debugged. The process resulted in the detection of various hard-to-find coding errors, including apparent random modification of communications buffers, and their eventual correction. The code was integrated with the Mach operating system, and modified to execute in both the front-end and node environments.

As part of this process, the SIB driver had to be modified to utilize Mach blocking and wake-up mechanisms. In addition, the network interface code had to be extended to support simulated broadcasting, since the underlying mesh, unlike the Ethernet, provided no hardware support for this feature, and the Mach operating system relied upon it to extend Mach IPC over the local area network. Further work was also required to support the Internet routing of messages between nodes in the mesh.

During this activity, a significant design defect was uncovered in the message passing subsystem that limited kernel error recovery. Insufficient context shared between the Message Packet Processor and the node CPU prevented a kernel from resetting it's message buffers. Furthermore, the Front End had no means to restart a single node. This precluded any support for debugging or restarting a node remotely using the mesh or diagnosing a kernel bug on a node from a *postmortem* period dump.

Critical to the Mach port was the need for diskless operation. Only a fraction of the nodes on the 2010 have disks yet Mach relied exclusively on local disks for booting. Fortunately, CMU had added vnode support to the evolving kernel to support NFS and one group at CMU had modified the inode pager to support remote paging. However, the code quickly became obsolete as new kernels were released. Several modifications at ISI allowed paging off of a NFS-mounted file system with minimal changes needed to keep current with new kernels.

The standard Mach porting activity of reimplementing the PMAP module for the underlying MMU hardware was also undertaken. Deficiencies with the Symult 2010 memory management unit complicated this task. The Symult memory management unit is optimized to provide rapid message remapping between the kernel and a few user processes and not the functionality for virtual memory in a general-purpose OS.

The 2010 MMU was found to provide an inconvenient page table implementation for Mach that could result in very expensive context switching. In addition, it did not incorporate page-referenced or page-modified bits and did not provide kernel read-only protection. These deficiencies complicated the development of the machine-dependent VM support.

A major advantage in porting Mach to a target machine is the prior existence of a UNIX implementation on the same hardware platform. Much of the machine-dependent code in UNIX, such as system initialization and the device drivers, can be used in the port; no prior UNIX implementation existed for the 2010 multicomputer. This also resulted in a rewrite of the Mach real-time clock code as well introduction of special mechanisms to support the system console and emulate software interrupts. The latter mechanism was also used to facilitate kernel debugging in the node environment.

One issue that had to be addressed was how to debug the mesh communications driver, given that any error in the front-end version of this driver would almost invariably mean an error in the node version of the driver, and vice versa. A strategy was adopted utilizing several test programs running under CE/RK to facilitate debugging the driver on the Front End. Loopback code was introduced in the standard RK software, designed to validate and record an arriving message and associated checksum and turn the message around to the sender after computing the new checksum. This approach enabled the diagnosing and correction of an especially insidious error that periodically corrupted message packets on a random basis. A side-effect of this activity was the detection of an error in the *ping* utility used at ISI and various other sites that resulted in incompatible checksums in even-byte-count versus odd-byte-count messages. Once the front-end version of the driver was known to function correctly, the node version of the driver was much easier to debug since modifications that were introduced were designed with compatibility in mind.

A major question early on in the effort was how to debug the Mach kernel being ported to the 2010. Since the porting effort involved development of mesh support in addition to more traditional Mach porting activities, it was clear that relying on the mesh driver software for kernel debugging was not a practical solution. The existence of two serial ports on each node board offered an alternative access path that could assist in the debugging process. With Symult's assistance, ISI was able to use the serial ports to track and monitor console output as well as interface to the kernel debugger. Achieving this debugging capability required additional effort in several areas. Real-time clock software was modified to provide an alternate mechanism to invoke the Mach Kernel debugger on the nodes as well as simulate console interrupts lacking in the hardware. A Symult-supplied utility, SIPG, was modified to provide a mechanism for downloading symbol information and utilizing this kernel symbol table with the kernel debugger software supported within Mach. And a new mechanism for procedure invocation that facilitated use of the debugger in the node environment was developed.

During the course of the effort, various hardware problems were encountered with the Symult 2010. Initially, node failures were detected during the booting of the node environment. These node failures did not necessarily manifest themselves as hard or repeatable failures, and were often immune to detection by Symult diagnostic routines. Most of these failures started showing up in the last few months prior to Symult's termination of business operations, since that was the point when the porting effort began to utilize the Symult more regularly as part of the porting effort. Suspect node boards were replaced, even if hard failures could not be produced. Eventually Symult terminated operations, and the long term support of the 2010 hardware became an issue. Project resources resulted in a decision to acquire support on a time and material basis, hoping that complete system failure did not occur and that lesser failures could be dealt with by reconfiguring offending nodes out of the system. Eventually, failures appeared that disabled the entire system, producing the suspicion that the communications backplane itself was at fault. Custom chip pullers were not available to swap AMRD ICs to pinpoint the failure, but project staff experimentation with hardware reconfiguration of the Symult to a 4x4 mesh, with the suspect AMRD configured out of the reduced system, eliminated the problem while still affording a useful operating environment.

The intent throughout the porting effort was to attempt to stay current with new Mach kernel releases, both to take advantage of CMU's and other participating site's bug fixes, as well as to acquire new Mach system functionality as it was developed, such as support for NFS. When ISI first got involved in the porting effort, the Mach 2.0 system was the current release and Mach 2.5 was to be released shortly for experimentation and examination to selected research sites.

While CMU attempted to minimize the impact of OS changes within the machine-dependent code, the nature of the ISI porting effort could not be localized in this area. Furthermore, some functionality needed for the 2010 port was in development at CMU. To complicate the porting process, CMU customized standard software development tools and libraries that resulted in incompatibilities when used in different computing environments. As a result, a significant effort was expended in updating new releases of the Mach kernel, the accompanying libraries, servers, and subsystems.

The Mach/2010 port was completed in early 1991, with the initial goals of a multinode Mach machine connected to the Internet realized. With that task finished, new goals were developed that addressed issues that surfaced during the porting effort with regard to properly supporting an MIMD computer of this sort. Specifically, it was realized that the internode I/O performance might not be adequate for finer-grained multinode parallel program execution due to the intrinsic overhead of the network IPC environment. Also, the need for more global job and system management concepts than those provided for in the traditional Mach environment were identified. As a consequence, two new activities were initiated: investigation of existing communications performance between individual nodes and between the front-end and the node environment and concomitant experimentation with reduced overhead communication strategies; and development of job and system management concepts that supported a more global view of a job than simply an ad hoc, disconnected collection of processes launched or spawned dynamically on the various nodes of the system.

In the area of high-performance I/O, various experiments were initiated to gather better information on the performance of the existing system. Various mechanisms for accumulating performance information were explored including the Mach Kernel Monitoring feature and the UNIX Profiling mechanism. Excess and experimental code was trimmed from the software to improve system performance. A set of token ring programs were also written to measure interprocessor communications speed on the 2010 nodes. One program was written for the 2010 running the CE/RK environment and the experiment carried out on Caltech's 92-node 2010. Another program was written for the ISI's 32-node 2010 running the newly ported Mach 2.6 system that exploited the Mach port mechanism. As expected, the performance of the latter was found to be unacceptably slow for finer-grained parallelism. An alternative communications mechanism that bypassed Mach IPC was also tested which yielded a better than 500% improvement but still fell short of the CE/RK performance figures[1]. Additional work continues in this area under other funding.

In the area of job and system management, experimental software was developed that described a job as a multi-node collection of tasks that were to be dynamically interconnected via Mach IPC based upon an accompanying script that defined both the tasks and their interconnectivity. The software would process the job script, establish unique port names, launch both the requested tasks as well as necessary support tasks designed to assist in interconnecting the components of the job, and pass on the unique port names to the appropriate tasks for subsequent IPC. This work is being used as a preliminary mechanism for exploring and developing the much more elaborate concepts of job and system management required for large multicomputers such as those of the joint DARPA/Intel Touchstone effort.

Communications

A significant component of the Mach/Symult porting effort involves communications, including communications between Mach hosts outside of the Symult multicomputer environment and the Front End Sun, communications between the Front End Sun and the processor nodes on the mesh, and communications between individual nodes within the mesh. The approach adopted was to

[1] The IPC speed on the S2010 using CE/RK was measured as 0.24 milliseconds, and that using Mach ports, which utilizes TCP/IP communication, required 5 milliseconds. Our new approach interfaced directly at the device level. The IPC speed under this approach was 0.72 ms.

implement the concept of a subnet within the Symult node set and exploit existing Internet mechanisms and techniques to effect communications between nodes, with the Sun Front End, and (through that front-end) to other hosts in the Internet. Mach runs on the selected node set as well as the Front End Sun workstation. This communications infrastructure provides the requisite communications support for higher level Mach functions. Where necessary for performance, a special communications mechanism bypasses the Internet mechanisms in order to effect direct communication between user processes, but this requires more active communications management on the part of the user process.

The effort required development of new software as well as modification of existing software to interface standard Internet support code to the Symult communications mesh, both at the front-end host and at Mach nodes within the S2010. Also important was optimization of selected high-level Mach components that implement network IPC and remote file access to improve the efficiency with which they utilize the underlying communications infrastructure in implementing their services. These services included the Network Message Server, the Network Name Server, and the Network File Server. Internet routing mechanisms were examined with regard to communication between external hosts and the Front End Sun, and the forwarding of messages by the Front End Sun to destination nodes.

With respect to intra-S2010 node connectivity, the focus of the work was on the manner in which CMU Mach networking software best interfaced to Symult Reactive Kernel subroutines that drive the Symult communications mesh. Investigation of Symult Inner Kernel and Reactive Handler software regarding message communications functionality took place. Detailed analysis of Inner Kernel software use of message buffers and Symult 2010 Message Packet Processor hardware buffer placement requirements, and its impact on Mach buffer allocation routines was completed.

The principal software component involved with communications over the mesh, and from the Front End Sun to the mesh, is the SIB driver. Standard software that treats the Sun Interface Board as a UNIX I/O device was supplied by Symult. An experimental version of this code that additionally interfaces to SunOS Internet software was also acquired from Symult. Project members analyzed and modified this code to integrate the SIB network interface with Mach/Internet software. During the process, a mismatch between SIB networking software and Mach networking software was discovered and corrective action undertaken.

Modifications to the Reactive Kernel to support loopback communications with the Mach SIB network interface were developed for message transmission testing. This involved development and incorporation of a loop-back extension to the Reactive Kernel, implemented as a separate handler under the inner kernel.

A new Network Message Server was built utilizing the (previously experimental) message server provided with Mach release 2.5. Several problems were encountered during server creation that were reported to CMU. Tests were run on the Network Message Server to establish proper operation. These included registering and retrieving ports with the Network Name Server and sending messages between the Mach/Sun-3 Front End and a NeXT workstation at ISI that also runs Mach. The tests were successful, with one exception, resulting from an implementation inconsistency, that required changing the broadcast address definition for one of the machines.

Consideration was given to reconfiguring the Network Message Server to utilize VMTP instead of TCP as its communications protocol, in order to improve communications efficiency and reduce communications overhead. To this end, a Mach kernel was configured with the VMTP protocol option in anticipation of evaluating the protocol for exchanging data between processors in the Symult 2010. Several small bugs were detected and corrected during this process and reported to CMU.

Front-end software to support communications between the Sun and the Symult node environment was completed and successfully tested via a loopback mechanism introduced into the Symult Reactive Kernel software. This software supported Internet connectivity with the Symult communications mesh driver by completing experimental software made available by Symult. Nodes in the mesh are assigned Internet addresses consistent with implementation of a subnet within the mesh. Standard Internet services are used to establish and support both stream- and datagram-based communications. With the exception of modifications to conform to the memory management support requirements of the Symult/Mach kernel for Mach/UNIX *mbufs* and Symult control and message blocks, the code runs essentially unaltered on the Mach system that executes on the nodes and complete the high-level Internet communications path between processes in the Mach/Symult environment.

The Mach operating system requires a broadcast capability to support Mach IPC. To satisfy this requirement, a broadcast mechanism was introduced into the communications driver to simulate the broadcast facility normally available on the Ethernet. This mechanism was tested in the front-end environment by exploitation of the loopback mechanism introduced into the Reactive Kernel for communications checkout. Minor modification to this loopback mechanism was required to support testing.

During subsequent testing, a serious communications bug was detected that resulted in intermittent and apparently unpredictable corruption of communications buffers. The result was proper transmission of a number of uncorrupted messages, followed by transmission of a corrupted message with concomitant message rejection due to an incorrect checksum. Diagnosis of the bug was complicated by the fact that a standard internet utility, *ping*, was improperly generating checksums for odd-sized messages but was functioning correctly for even-sized messages. This problem with *ping* was diagnosed and communicated to Internet management personnel. Extensions were introduced into the loopback code previously incorporated into the Reactive Kernel to help determine whether the extant communications bug was associated with the sending or the receiving of messages, or both. Additional data on the nature and frequency of message buffer corruption was accumulated to help in error diagnosis through the generation of fixed-format test messages and the logging of corrupted messages by the loopback code.

The communications bug was eventually isolated to the Symult-supplied network SIB driver message transmission code. The error was associated with an incorrectly updated counter involved with the transfer of data from operating system *mbufs* to driver output buffers. The consequence was apparently random, frequent corruption of data in output communications buffers depending upon message size and location in the output stream. The erroneous software was modified appropriately, tested, and subsequently functioned correctly.

To permit continuation of the porting effort during diagnosis of the communications bug, the kernel was reconfigured to force NFS to utilize slower, special-purpose code in the Mach kernel that performs UDP checksumming as part of its normal operation (instead of the *fast-send* code normally utilized by Mach to reduce Internet communications overhead). The assumption was that communications could successfully continue, even if a percentage of the transmitted messages were rejected and message retransmission was required. Unfortunately, this *slow-send* code did not function correctly, resulting in frequent system crashes. As an alternative approach, checksumming software was temporarily introduced into the *fast-send* code to permit continued progress in the porting effort.

User interface

Some effort was directed toward insuring that the remote user interface to the system was fully functional. Problems were found to exist in the operation of various servers that provide remote access to a Mach host. For the most part, these problems are a consequence of CMU-specific authentication procedures and requirements that differ from standard BSD 4.3 authentication procedures.

Specifically, the *rshd* did not allow remote shell execution, but instead, terminated with no apparent action; *rexecd* exhibited similar behavior to *rshd*; and *syslogd* filled up logging files with reinitialization messages issued every 60 seconds. Similarly, *rlogind* required an inordinate amount of time to complete the login process.

Investigation of the problem with *rshd* indicated fundamental differences between CMU access authorization procedures and those typically supported in the BSD 4.3 UNIX operating environment. In fact, interaction with CMU personnel revealed that the CMU Computer Science Department does not, in fact, support *rshd* as part of normal operations and that sources for the server were not available from CMU. Upon request, CMU made available the sources for the internet super server, *inetd*, and advised us to utilize publicly available BSD 4.3 versions of the *rshd* server. Recompilation under Mach of *rshd, rexecd,* and *rlogind* sources retrieved from the standard BSD 4.3 UNIX distribution appeared to correct their problems.

Problems with *syslogd* appeared to be endemic to its use with the Nanny super server. Experiments with the *syslogd* log server revealed that the software exhibited the behavior described above when executed under the control of the Nanny, but not when started independently as it would be under non-Mach system operation. The latter has been adapted as the standard mode of operation in our environment.

Investigation of the problem of abnormal termination of the system utility *ps* revealed a problem with the implementation of the kernel-mode copy-on-write mechanism during the porting effort. This problem has been corrected.

Several errors specific to the node environment surfaced as the Mach node software became more fully operational. In particular, a problem with the Nanny server arose in which the server would crash without any diagnostic messages when executed on a Symult node. Sources for the Nanny server were obtained from CMU and the problem was diagnosed as resulting from an improperly formatted data file describing the services managed by the Nanny server. The erroneous file was reformatted, and the Nanny server subsequently to functioned correctly in the node environment.

Additionally, various bugs associated with the spawning of processes and console output were identified and corrected.

Hardware issues

To facilitate debugging of the node version of the Mach kernel, a second TTY line between the Sun-3 Front End and the Symult was installed. This required interaction with Symult personnel to identify the appropriate communications lines to use on the on-board UART chip interface.

A replacement board for a marginal node board at location 3,0 in the Symult was obtained immediately prior to Symult's terminating business operations. This board (without memory) was swapped with a defective node board in the S2010, by scavenging the memory from the bad board and using it to populate the replacement board. This corrected the problem of the marginal board that resulted in occasional difficulty in booting the S2010.

Provision of service and consulting

With both platforms installed and running at ISI, it was the role of the CERPASS staff to provide service to others needing the use of our computing resources, to explore the best ways to use these resources, and to transfer the knowledge acquired from our use of these resources to others in the community.

Results - Provision of service and consulting

Dr. Tung completed his extensive studies of instruction set execution on the Connection Machine. Dr. Tung and Dr. David Mizell completed their experiments with mapping parallel sorting algorithms onto the Connection Machine. Their joint paper, "Two Versions of Bitonic Sorting Algorithms on the Connection Machine," [30] was presented by Dr. Mizell at the IEEE Parallel Processing Symposium in Fullerton, California, where he also chaired a session on multiprocessor performance evaluation.

Extending their previous results in conservative methods for parallel discrete-event simulation, Rivi Sherman, working with Mani Chandy of the University of Texas (currently visiting at Caltech), developed a theoretical model of discrete-event simulation [2] that characterizes all discrete-event simulation methods as special cases of a very general "relaxation" algorithm -- one in which time can be run either forwards or backwards. Dr. Mizell, worked with Richard Lipton of Princeton, and obtained "worst case" results that provide a limited theoretical comparison of conservative versus optimistic simulation methods. Their results are summarized as follows: while examples can be found for which optimistic methods outperform conservative methods by an arbitrary amount, the converse is not true. It ws proven that conservative methods can only outperform the optimistic methods on the same simulation model by a constant factor.

Susan Coatney and Walid Najjar produced a first prototype of "Simple," an object-oriented concurrent programming language based on C++, which they are designing to relieve programmers of many of the tedious details of writing parallel programs for message-passing systems like the Ametek 2010.

Dr. Mizell and Rivi Sherman, along with Richard Lipton of Princeton, continued their joint work on a theoretical extreme-case comparison of conservative and optimistic distributed simulation methods. A paper on their results was submitted to the 1990 SCS Conference on Distributed Simulation [11].

In the first part of calendar year 1989, Sherman attended the Eastern Multiconference of the Society for Computer Simulation in Tampa, Florida, and presented two papers based on her collaboration with Mani Chandy of Caltech [2, 1].

Dr. Mizell gave a lecture, "Sorting Algorithms on the Connection Machine," at the Institute for Defense Analyses (IDA) Supercomputing Research Center in Lanham, Maryland.

Steve Farnworth and Dr. Mizell completed the preliminary design of an instrumentation and debugging system for MIMD multiprocessors using an in-circuit emulator system.

Walid Najjar and Paraskevas Evripidou worked with Jean-Luc Gaudiot of USC's Department of Electrical Engineering on techniques for mapping functional language programs onto multicomputer architectures [9].

Rivi Sherman and A. Pnueli began and completed a study of efficient implementations of linear temporal logic model checkers for verifying finite-state parallel and distributed programs [25]. Susan Coatney developed a compiler for the state machine and proposition notation used by Rivi Sherman's experimental prototype model checker.

Dr. Tung and JPL's P. Peggy Li prepared an unpublished research note, "Parallel Sorting on Symult S2010" [28]. The paper is on the performance analysis of three sorting algorithms implemented on the Symult S2010. Each of these algorithms is a combination of a fast sequential algorithm and a parallel version of either Bitonic sort, Shell's sort, or Quicksort, and is used to sort M keys on an N-node machine where M >> N.

Paraskevas Evripidou's presentation, "Data-Flow Computing: A Status Report," which was presented in June 1989, is currently being prepared as a chapter in a book titled, *Advanced Topics in Data-Flow Computing* to be published by Prentice Hall [3].

The paper, "A Single-Assignment Language in a Distributed-Memory Multiprocessor," by Evripidou, Najjar and Gaudiot that appeared in the *Proceedings of the Parallel Architectures and Languages Europe*, was reprinted by USC/Information Sciences Institute in December 1989 [9].

Paraskevas Evripidou's paper, coauthored with J-L. Gaudiot of USC's Electrical Engineering Department, was accepted for publication in the proceedings of the 1990 International Conference on Parallel Processing, St. Charles, Illinois, the week of 13-17 August 1990. "A Decoupled Graph/Computation Data-Driven Architecture with Variable-Resolution Actors" [4] presents a hybrid multiprocessor architecture that combines the advantages of the dynamic data-flow principles of execution with those of the control-flow model of execution. Two major design ideas are utilized by the proposed model: asynchronous execution of graph and computation operations, and variable-resolution actors.

We have responded to two surveys regarding our parallel computing resources and research. ISI has become known to researchers by means of the listing on the NNSC's Internet Resource Guide.

The first query was from Von Neumann Supercomputer Center's Bob Bijoy, who asked detailed questions on the CM-2 configuration, research, and applications.

The second query came from MITRE Corporation's Bede McCall, who was surveying the various NSFNET Supercomputer Centers with an eye toward their possible use by various MITRE projects. He asked detailed questions on ISI's resources (CM-2 and S2010), network access, and user account policies.

Dr. Tung attended the Connection Machine Workshop held at UCLA on 20-21 April. This workshop, hosted by Professor Charles Taylor of UCLA, provided Connection Machine researchers a chance to interact with and discuss their research work with others. Of particular interest was joint work by UCLA's Charles Tong and NASA-Ames's Paul Swarztrauber on ordered-FFT, which illustrates some research points in our work on machine performance.

Other work by Paraskevas Evripidou and J-L. Gaudiot of USC involves the implementation of scientific programs on a decoupled data-driven architecture with vectors and macro actors. The compiler generates graphs with various-sized actors in order to match the characteristics of the computation. For instance, vector actors are proposed for many aspects of scientific computing while lower resolution (compiler-generated collection of scalar actors) or higher resolution (scalar actors) is used for unvectorizable programs. A block-scheduling technique for extracting more parallelism from sequential constructs is incorporated in the decoupled architecture. In addition, a graph-level priority-scheduling mechanism is implemented that improves resource utilization and yields higher performance.

Technology transition activities for phase 1

Papers

[1] Chandy, K. M. and R. Sherman. *Space-Time and Simulation.* Proceedings of the SCS Multiconference on Distributed Simulation, Tampa, Florida, March 1989.
[Reprinted as USC/Information Sciences Institute Reprint Series ISI/RS-89-238, June 1989.]

[2] Chandy, K. M., and R. Sherman. *The Conditional-Event Approach to Distributed Simulation.* USC/Information Sciences Institute Research Report ISI/RR-88-226, June 1989.

[3] Evripidous, P. *Data-Flow Computing: A Status Report.* Published? by Prentice Hall, June 1989.

[4] Evripidou, P. and J-L. Gaudiot. *A Decoupled Graph/Computation Data-Driven Architecture with Variable-Resolution Actors.* Was presented at the International Conference on Parallel Processing, Saint Charles, Illinois, August 1990.

[5] Evripidou, P. and J-L. Gaudiot. *Decoupled Data-Driven Architectures with Vectors and Macro Actors.* Presented at the CONPAR 90-VAPP IV Joint Conference on Vector and Parallel Processing held in Zurich, Switzerland, 10-13 September, 1990.

[9] Evripidou, P., W. Najjar, and J-L. Gaudiot. *A Single-Assignment Language in a Distributed-Memory Multiprocessor.* Proceedings of the Parallel Architectures and Languages.

[11] Mizell, David and Richard J. Lipton. *An Extreme-Case Comparison of Optimistic and Conservative Parallel Discrete-Event Simulation Methods.* Submitted to the 1990 Society for Computer Simulation (SCS) Conference on Distributed Simulation.

[25] Sherman, R., and A. Pnueli. *Model Checking for Linear Temporal Logic: An Efficient Implementation.* USC/Information Sciences Institute Research Report ISI/RR-89-241 (in preparation).

[28] Tung, Y. and P. Li. *Parallel Sorting on S2010.* Presented at the Fifth Distributed Memory Computing Conference, Charleston, South Carolina, April 1990.

[30] Tung, Yu-Wen, and David Mizell. *Two Versions of Bitonic Sorting Algorithms on the Connection Machine.* Presented at the IEEE Parallel Processing Symposium in Fullerton, California.

Seminars

Dr. Tung presented a seminar surveying several recently developed distributed operating systems on January 17, 1991. Research issues of communication and resource management on the Amoeba, Sprite, V and Mach operating systems were addressed and discussed.

Dennis Hollingsworth gave a seminar at ISI on Mach messaging concepts that was attended by individuals from Trusted Information Systems and TeraData Corporation.

Discussions

Dr. Gaines discussed with Intel Corporation the porting of Mach to their Touchstone machines. Experience gained through our efforts was passed on to Intel staff.

Dr. Tung worked with Thanh Tu, a USC graduate student, on the performance analysis of the CM-2 machine.

ISI met with researchers from the VISCOM group on the USC campus and provided technical advice concerning Mach and their design for an orthogonal multiprocessor based on the Intel i860 processor.

Visitors

In early 1990, Dr. Tung discussed with Professor Redekopp, who is head of USC's Aerospace Engineering and Mechanical Engineering departments, how researchers in his departments can use ISI's Connection Machine and Symult S2010.

Gil Weigand of DARPA visited ISI on 11 January 1990 to discuss the parallel processing projects at ISI.

On 7 February 1990, Stephen M. Griffin of the Division of Advanced Scientific Computing at the National Science Foundation visited with several project team members.

Covers period 1 November 1991 through 30 April 1992
Phase 2 - October 1, 1991 Through May 30, 1996

Resource Management for Parallel and Distributed Systems and OS Services for Distributed Systems

To reflect the shift in focus indicated in the continuation proposal, the name of the Center for Experimental Research in Parallel Algorithms, Software, and Systems (CERPASS) Project changed to the DIstributed VIRtual System (DIVIRS) Project.

As outlined in the continuation proposals 91-ISI-57 (revised) and 92-ISI-50R (revised) on cooperative agreement NCC 2-539, we (1) developed software, including a system manager and a job manager, that manages available resources and that enables programmers to program parallel applications in terms of a virtual configuration of processors, hiding the mapping to physical nodes; (2) developed communications routines that support the abstractions implemented in item one; (3) continued the development of a file system based on the Virtual System Model; and (4) incorporated appropriate security measures to allow the mechanisms developed in items 1 through 3 to be used on an open network.

The goal throughout our work was to provide a uniform model that can be applied to both parallel and distributed systems. We believe that multiprocessor systems should exist in the context of distributed systems, allowing them to be more easily shared by those that need them. Our work provides the mechanisms through which nodes on multiprocessors are allocated to jobs running within the distributed system and the mechanisms through which files needed by those jobs can be located and accessed.

The results of our efforts in these areas take the form of several research prototypes: The Prospero Resource Manager, the Prospero File System and Directory Service, and integration of security and electronic commerce mechanisms with distributed system services. Each of these results is described separately below.

The Prospero Resource Manager

Conventional techniques for managing resources in parallel systems perform poorly in large distributed systems. To manage resources in distributed parallel systems, we have developed resource management tools that manage resources at two levels: allocating system resources to jobs as needed (a job is a collection of tasks working together), and separately managing the resources assigned to each job. The Prospero Resource Manager (PRM) presents a uniform and scalable model for scheduling tasks in parallel and distributed systems. PRM provides the mechanisms through which nodes on multiprocessors can be allocated to jobs running within an extremely large distributed system.

The common approach of using a single resource manager to manage all resources in a large system is not practical. As the system grows, a single resource manager becomes a bottleneck. Even within large local multiprocessor systems the number of resources to be managed can adversely affect performance. As a distributed system scales geographically and administratively, additional problems arise.

PRM addresses these problem by using multiple resource managers, each controlling a subset of the resources in the system, independent of other managers of the same type. The functions of resource management are distributed across three types of managers: system managers, job managers, and node managers. The complexity of these management roles is reduced because each is designed to utilize information at an appropriate level of abstraction. Our approach to resource management is based on a separation of duties across the three entities.

The system manager controls a collection of physical resources. In our present work we are concentrating solely on processors, though eventually the system manager might also manage memory, communication links, and I/O channels. Usually a system manager will control the resources on a single multiprocessor, but when necessary for performance or other reasons, multiple system managers may exist, each controlling a disjoint subset of the processors on the machine. A system manager might also control resources across a collection of systems.

The job manager provides the abstraction of a virtual system to a job. A job is a collection of tasks working together to perform a computation. When a job is initiated, the job manager contacts one or more system managers to obtain the necessary resources. The job manager then initiates the loading of programs onto the appropriate processors, causes the jobs to be executed, and monitors the continued execution of the program. If user I/O is required the job manager starts a task on the local node that manages I/O to the terminal. If additional resources are required, tasks within the job request them from the job manager, which is responsible for allocating resources across the tasks in the job.

The node manager runs on each processor in the system, eventually as part of the kernel but in the current implementation as a user level process. The node manager accepts messages from the system manager identifying the job managers that will load and execute programs. When requested by an authorized job manager, the node manager loads and executes a program. The node manager notifies the job and system managers about events such as the termination and failure of tasks. In the current prototype, since nodes are shared with other users, the node manager keeps the system manager informed about the availability of the node for assignment (i.e. whether any interactive users are logged in to the workstation). The node manager also caches information needed to direct messages for other tasks to the node on which the task runs.

Results - Prospero Resource Manager

During the contract period the job manager, system manager, and node managers were designed, implemented, deployed, and several releases were distributed by FTP through the Internet. The release through the end of the CERPASS/DIVIRS effort runs on a collection of Sun3, SPARC, and HP9000/700 workstations running various versions of the Unix operating system, and a single Intel486 personal computer running Mach. Communication between the job, system, and node managers, and between tasks in a job is supported by a reliable delivery protocol based on the user datagram protocol (UDP) running over local and wide-area networks. Heterogeneous execution environments are supported - a system manager may manage nodes of more than on processor type. In the common case there is one system manager for each site. For example, our test setup consisted of one system manager responsible for a set of SPARCstations on USC's main campus, another managing a collection of Sun3, SPARC, and HP700 workstations at ISI, 15 miles away from the main campus, while a third managed a set of HP700 workstations at MIT, across the country. We have run applications that use processors at all three sites.

Initially we had planned to move our software to a Touchstone class multicomputer when such a machine became available at the DARPA High-Performance Computing demonstration center , and we considered applications of our model to shared-memory multiprocessors, discussing the possibility of access to such a machines from the Information Sciences Division of NASA Ames. Unfortunately, these resources were not ultimately made available to us, so development and releases remained focussed on collections of workstations throughout the term of the contract.

Programmers link executables for their tasks with the communication library we provide. To run a parallel application, users invoke the job manager passing it the name of the application. I/O to the terminal and to files that are not otherwise accessible to the application is handled through an I/O task that runs on the user's workstation.

In the initial prototype, information about the requirements of a job was read from a job description file. This file specified the number of tasks to be initially created, the location of the executable for each task, and a list of system managers from which resources might be obtained. Work was completed in 1993 that allowed this information to be obtained from the Prospero directory service, developed in another part of the project. Now, depending on how PRM has been configured, users create a job description file or they make suitable entries in the Prospero Directory Service.

In the final funded year of the project, we extended the Prospero Resource Manager to support remote execution of programs that are not written specifically for PRM or relinked with the PRM library. Sequential applications, including those for which source code is not available, can be run under PRM. Interactions between the user and unmodified sequential jobs are supported by redirecting stdin, stdout and stderr streams through a separate task running on the same node. This task accepts output from the unmodified task and sends it to the terminal-I/O task and it accept input from the terminal I/O task and writes to the stdin stream of the unmodified task. A parallel application may be composed of both unmodified and relinked program modules. For such applications, a user-designated task receives terminal input; unmodified tasks communicate through their standard I/O streams as described above, and relinked tasks use PRM's message passing mechanisms.

We also added support for suspension and subsequent migration of tasks running under PRM. To suspend a task, the execution state is captured in a checkpoint file and the task is killed. To migrate the task, a checkpoint is created by the node manager, which informs the concerned job manager. The job manager then acquires another processor from the system manager and requests the new node manager to restart the task from the checkpoint. Our implementation builds upon the checkpointing mechanisms from the Condor package of the University of Wisconsin. We have integrated this package with PRM and enhanced PRM's communication libraries to correctly handle messages addressed to migrating/migrated tasks. We have also implemented an alternative task suspension/ resumption method (but not migration) for programs that cannot be relinked with Condor's checkpointing library.

Also, in the final funded year of the effort, we extended PRM's resource allocation and release policies. Depending on its configuration, the system manager may allocate a node to a job for the entire duration of the job's execution, or for executing a designated set of tasks. The former policy is more efficient for jobs in which tasks are dynamically spawned. The latter policy enables the system manager to preempt nodes from a job, and force its tasks to checkpoint (for tasks capable of checkpointing). It is then the job manager's responsibility to find an alternate set of nodes to migrate its tasks to. The implementation allows for a default policy to be configured to suit the users' requirements.

Also during the final funded year we began development of an alternate job managers to support debugging of parallel applications and we improved the performance of the communications primitives that provide the appearance of a virtual systems to tasks using our software. We also explored the benefits of creating separate I/O tasks to control I/O to remote files.

To debug PRM applications we developed debugging tools consisting of a command interface at the front-end and task-monitors at the back-end. The interactive front-end enables the application programmer to monitor and control program execution. At the back-end each target task is controlled by a separate task monitor that is co-located with the target. We have adapted the Gnu Debugger (gdb) to function as the task monitor and interact with the front-end. Use of gdb gives the task monitor all the features of a traditional debugger.

We added support for playback debugging using traces. When applications are linked with an instrumented version of the communication library the communication activity of the program can be captured in trace files. Invoking 'replay' at the command interface causes task-monitors to use these trace files to replay programs and exactly recreate the sequence of events in a program's history.

We also implemented a library that allows existing programs written for the Parallel Virtual Machine (PVM) parallel computing environment from Oak Ridge National Laboratory to run unmodified over PRM. The programs must be relinked with our version of the communication library, but the interface to the library is the same as that for PVM. The PRM libraries fully support the interprocess communication interface provided by PVM Version 3.3.5. A group-server process handles collective operations such as broadcast, barrier synchronization and global reduction. The group server is automatically spawned by the job manager when a group operation is first invoked by one of the tasks.

Products and papers - Prospero Resource Manager

Two papers about the Prospero Research manager were published and are included with this report:

[1] B. Clifford Neuman and Santosh Rao. Resource Management for Distributed Parallel Systems. In *Proceedings of the 2nd International Symposium on High Performance Distributed Computing.* Spokane, July 1993.

[2] B. Clifford Neuman and Santosh Rao. The Prospero Resource Manager: A scalable framework for processor allocation in distributed systems. *Concurrency: Practice and Experience.* Summer 1994.

The latest release of the Prospero Resource Manager (as updated since completion of the contract), can be obtained from the web page:

http://gost.isi.edu/info/prm

PRM has been used by a parallel computing class on the USC campus. We have also used it to run the Ocean program from Stanford University's SPLASH benchmark suite, which studies the role of eddies and boundary currents in influencing large-scale ocean movements by solving a set of partial-differential equations. We started with a message-passing version of Ocean available for the Connection Machine (CM-5). To port this program to the PRM platform, we wrote a set of macros and routines to implement the CM-5 communication library functions using equivalent calls from our own library. The host program from the CM-5 version was incorporated into a terminal I/O task and handles interactive input. We are also using PRM to develop a simulator for large networks of neurons, to crack encryption keys, and to run the public domain ray tracing program, POVRAY.

Our efforts to extend and support the Prospero Resource Manager will continue under new projects.

The Prospero File System and Directory Service

The Prospero file system and directory service is a file system and directory service based on the Virtual System Model. In the context of multiprocessor systems, files are scattered across multiple I/O nodes. The Virtual System Model allows users to organize files with names that are independent of the storage site. This allows files to be moved around as needed for optimal performance without constraining the ability of the user to organize the files. Our work in this area is also important for organizing information in distributed systems in general since it is difficult for users to find the files that are of interest among the huge number of files scattered across the Internet.

Results - The Prospero File System and Directory Service

The development of Prospero moved from the University of Washington to ISI and several new versions of the software were released from ISI during the contract period. Changes in the first release from ISI included bug fixes and extensions to support the needs of specific users. Among these changes was a new option to directory queries that allows attributes to be returned for all files in a directory together with the directory listing. This change was requested by the maintainers of the Australian Academic and Research Network who developed an FTP server that uses Prospero to shadow archive sites in the U.S. and Europe. This change greatly improves the performance of their server and reduces the number of packets sent across their trans-pacific connection to the rest of the Internet. Other extensions to Prospero included the addition of support for multi-threading of the Prospero server, and caching of directories and attributes.

In August 1992, a protocol specification for version 5 of the Prospero protocol was released. Version 5 of the Prospero protocol is more rigorously specified, and the implementation removes many of the assumptions made in earlier releases. These changes allow additional information sources to be made available through Prospero and will allow additional applications (e.g. document and file system browsers, hypertext systems) to be implemented on top of Prospero. In March 1993, software supporting Version 5 of the Prospero protocol was released. A prototype server supporting NFS access to files named by a Prospero virtual system was implemented.

During the contract period several new access methods were added to the Prospero file system. The first allows access by FTP. Though Prospero has supported Anonymous FTP for some time, the extensions supporting FTP in general allow the use of Prospero to organize information that is not publicly available. We also added support to retrieve data maintained as part of the Wide Area Information Service (WAIS). Many full text databases are available over the Internet using this service. Our extensions to Prospero allow documents found through WAIS queries to be treated as files in the virtual file system. Support for access to files stored in Gopher, a distributed internet menu browser, was also added. Also an e-mail method supporting non-real-time access to files available through e-mail requests to designated electronic mail addresses is now supported. Finally, we added support for access to files on the world wide web, retrievable using HTTP.

In addition to support for external data access methods, we have added support for a new access method based on the CONTENTS attribute that is suited to the retrieval of data from files, where the contents of a file can be sent as an attribute of the file, reducing the number of exchanges needed to access a file. In support of the CONTENTS access method we have extended the Prospero implementation to support binary data values for attributes. This technique is suitable primarily for small files, and access to larger files is best performed using an access method separate from the directory service itself.

In addition to FTP, WAIS, Gopher, e-mail, and HTTP, and the CONTENTS method, other access methods supported include the Andrew File System (AFS), and Sun's Network File System (NFS).

In July 1993, an initial version of a menu browser was released together with a gateway that made information from the Gopher service available to Prospero users. Gateways from the Gopher Menu Browser and the X-Mosaic hypertext browser to information exported using the Prospero protocol have been implemented by Pandora systems and Bunyip Information Systems.

During the contract period we designed and implemented the Prospero Data Access Protocol (PDAP) to support secure retrieval of data from systems running Prospero. Initially, the Prospero Directory Service provided only directory information about files in the Prospero File System. Access to the files was supported automatically using existing access methods. The PDAP provides a common protocol for access to local files and gateway access to remote files using alternative access methods, thus reducing the number of access methods that must be supported by Prospero applications.

We also integrated the Prospero Directory Service with the Prospero Resource Manager (PRM) allowing information about the configuration of parallel programs to be maintained as attributes of the programs themselves, rather than in a separate configuration files, and providing for parallel applications to be invoked the same way sequential programs are, simply by typing their name; the configuration information stored in the directory entry for the program is used to automatically invoke the appropriate job manager. The Prospero File Access Protocol was also integrated with PRM allowing access to local files that are not otherwise exported, by tasks running on remote nodes as part of a job initiated from the local node.

Steven Augart implemented a new database/directory format for Prospero that combines attribute information previously associated with a file or object, with the information associated with a directory. Sung-Wook Ryu developed a database module that allows information in the format just described to be stored in a common dbm database. The first change reduces the storage and i-nodes required to maintain information on a Prospero server and improves performance. The dbm extension further reduces storage requirements, but locking and reliability issues make it suitable for only certain applications.

Sio-Man Cheang developed a configuration package for Prospero that provides functionality similar to that provided by the X Window system. All user runnable commands call the configuration package to determine configuration parameters for network communication, gateways, debugging, priorities, and security and payment options. The configuration package determines the configured values by reading, in order, command line options, user-specific and system-specific configuration files, and compile time definitions.

In the final funded year of the effort, Sio-Man Cheang implemented a prototype transitive indexing system. Transitive indexing is a scalable technique for generating high-level indices that direct users to information of interest. An indexer was completed that when executed with a depth and a list of attributes to be indexed as arguments, builds an index by traversing the Prospero directory structure and retrieving the values of the named attribute from each file, directory, or object. An index is then generated mapping values of the attribute to the first level links through which objects with matching attribute values may be found. A reference to this index is bound as an attribute to the node from with it was generated. When generating an index, if a node is reached that already has an associated transitive index for the same attribute and at least the requested depth, the data from the existing index is propagated upward and the traversal is pruned at that point.

Information from a transitive index is obtained using the TQ filter on a Prospero directory query. When querying a directory, Prospero allows the user to specify filters that modify the list of objects that are returned. By specifying an attribute and a value as arguments to the TQ filter, the list of links that appear in a directory will be restricted to only those subdirectories through which objects with the requested attributes can be reached.

Products and papers - Prospero File System and Directory Service

Four papers about the Prospero File System and Directory Service were published and are included with this report:

[16] B. Clifford Neuman. *Prospero: A Tool for Organizing Internet Resources.* Electronic Networking: Research, Applications and Policy, Volume 2, Issue 1, Spring 1992. USC/ Information Sciences Institute Research Report ISI/RS-92-421 (in preparation).

[18] B. Clifford Neuman. *The Prospero File System A Global File System Based on the Virtual System Model.* Workshop on File Systems, May 1992.

[19] B. Clifford Neuman and Steven Seger Augart. *Prospero: A Base for Building Information Infrastructure.* In Proceedings of INET'93. San Francisco, August 1993. USC/Information Sciences Institute Research Report ISI/RS-93-415 (in preparation).

[20] B. Clifford Neuman, Steven Seger Augart, and Shantaprasad Upasani. *Using Prospero to Support Integrated Location-Independent Computing.* In Proceedings of the Usenix Symposium on Mobile and Location-Independent Computing. Cambridge Massachusetts, August 1993. USC/Information Sciences Institute Research Report ISI/RS-93-416 (in preparation).

The latest release of Prospero (as updated since completion of the contract), can be obtained from the web page:

http://gost.isi.edu/info/prospero

On the technology transfer front, negotiations were completed with CyberSafe Corporation who has licensed Prospero from USC and intends to integrate it with several of their products. Bunyip Information Systems is working with several large publishers to make available information provided by the publisher to Internet users on a subscription basis. This information will be accessed by users of the service using Prospero. Also, America Online (AOL) introduced their Internet service, allowing AOL users to access information from Gopher, WAIS, and Prospero servers on the Internet. AOL's information gateway, developed by Pandora systems, uses a Prospero server that translates data from the Gopher and WAIS formats, returning data to a Prospero client using the Prospero protocol. The Prospero server caches data from other services, reducing network and server load, and improving performance and reliability. In the Spring of 1994, the Prospero gateway for AOL handled roughly 100,000 queries per day, and has a cache hit ratio of 88 percent. Our work to optimize performance of the Prospero server was driven, in part, by the requirements for production use by AOL, but the improvements were incorporated into the standard Prospero release and, therefore, benefit all users of Prospero.

Our efforts to extend and support the Prospero Resource Manager will continue under new projects.

Security for Distributed Systems and Electronic Commerce

The computing and information infrastructure we are developing as part of the DIVIRS project requires an underlying security infrastructure to provide fine-grained access control mechanisms, to protect such resources, and to provide accounting in order to manage their use. Such security infrastructure is also necessary to support secure commerce on the internet, and to protect electronic forms of payment.

Though not specifically a task under the CERPASS/DIVIRS effort, separate funding was obtained through an Augmentation Award for Science and Engineering Research Training (AASERT) to support a graduate student, Ari Medvinsky, to work on the problems of electronic commerce. His security related efforts - as needed for his work on electronic commerce - were also applied to improve the security of the information and computing infrastructure under development as part of the DIVIRS effort.

Results - Security for Distributed Systems and Electronic Commerce

During the contract period, we incorporated new security mechanisms into both Prospero and the job, system, and node managers. Support for strong authentication through Kerberos was added to Prospero. Significant changes to a beta release of Version 5 of Kerberos from MIT were made and these changes were fed back to MIT and were included in the subsequent release from MIT. These changes to Kerberos included support for the forwarding of authentication credentials, a mechanism that is necessary to securely allow remotely executing tasks to perform operations with the privileges of the user.

Support was added for password based authentication for Prospero directory queries. Although Kerberos authentication was already supported, a weaker form of authentication was needed for users at sites that don't run Kerberos, and passwords are better than no authentication at all. Support was also added to the Prospero Directory Service protocol for billing. These changes support a range of billing methods from the inclusion of credit card numbers, to direct billing options, to mechanisms that will be supported in the future such as proxy checks, and anonymous electronic currency. While we discourage the use of techniques that are vulnerable to compromise, such as sending a credit card number unencrypted on the network, we recognize that application developers will use such methods anyway. By providing a common billing framework within which they can use such methods, the hooks for more secure billing mechanism will already be present in application protocols when those secure billing mechanisms are ready.

We separately received funding to develop wide scale security infrastructure supporting authorization and related security services on the Internet, and within the scope of the DIVIRS effort we implemented the Prospero Resource Manager and the Prospero Directory Service so that it can take advantage of such infrastructure when it becomes available.

To support for-hire services, it must be possible to send payments over the network when information or other on-line services are requested. As part of an AASERT award attached to the DIVIRS contract, and in conjunction with our efforts on a separate contract for Security Infrastructure, Gennady (Ari) Medvinsky developed electronic payment mechanisms for the Internet. As part of this effort, a prototype implementation of NetCheque and NetCash have been released on the Internet. Users registered with NetCheque accounting server are able to write checks to other users. When deposited, the check authorizes the transfer of account balances from the account against which the check was drawn to the account to which the check was deposited.

Products and papers - Security for Distributed Systems and Electronic Commerce

Two papers about the security end electronic commerce were published based on work from this contract, and are included with this report:

{14] Gennady Medvinsky and B. Clifford Neuman. *NetCash: A design for practical electronic currency on the Internet.* In Proceedings of the first ACM Conference on Computer and Communications Security. Fairfax Virginia, November 1993 (copy included with last semi-annual report). USC/Information Sciences Institute Research Report ISI/RS-93-413 (in preparation).

[17] B. Clifford Neuman. *Proxy-Based Authorization and Accounting for Distributed Systems.* In Proceedings of the 13th International Conference on Distributed Computing Systems. Pages 283-291. Pittsburgh, Pennsylvania, May 1993 (copy included with last semi-annual report). USC/Information Sciences Institute Research Report ISI/RS-93-419 (in preparation).

The latest release of Kerberos as obtained from MIT, with our changes included, and the latest release of the NetCheque system, can be obtained from the web pages:

http://gost.isi.edu/info/kerberos

http://gost.isi.edu/info/netcheque

Staff

The following individuals contributed to and were employed by the DIVIRS/CERPASS effort at various points during the contract period.

Celeste Anderson

Steven Augart

Richard Bisbey

Ben Britt

Kwynn Buess - Work study

Scott Carter

Sio-Man Cheang, GRA

Susan Coatney

Danny Cohen

Paraskevas Evripidou

Manjiri Gadagkar - DR

Stockton R. Gaines

Frances Henderson

Dennis Holingworth

Sukumal Imudom, GRA

Sanjay Joshi - DR

Joe Kemp

Konstadinos Kutsikos

Charlie Lai

Wei-Ming Lin

Kathleen McLaughlin

Gennady (Ari) Medvinsky

David Mizell

Paul Mockapetris

Walid Najjar

Clifford Neuman

Ron Ohlander

Bruce Onder

Jon Postel

Santosh Rao, GRA

Sung-Wook Ryu, GRA

Rivka Sherman

Carole Sumler

Yu-Wen Tung

Shantaprasad Upasani, GRA

Peter Will

Suzanne Woolf

Jeanine Yamazaki

APPENDIX A - GLOSSARY

AASERT	Augmentation Award for Science and Engineering Research Training
ACM	Association for Computing Machinery
ALU	arithmetic logic unit
AMRD	automatic message routing device
AOL	America Online
ARPA	Advanced Research Projects Agency (see also DARPA)
BSD	Berkeley Standard Distribution
CERPASS	Center for Experimental Research in Parallel Algorithms, Software and Systems
CM	Connection Machine
CM-2	Connection Machine Model 2
CMU	Carnegie Mellon University
DARPA	Defense Advanced Research Projects Agency (see also ARPA)
DIVIRS	Distributed Virtual Systems
EV	Embeddable Variant
FTP	File Transfer Protocol
GDB	Gnu Debugger
HDTV	high-definition television
IEEE	Institute of Electrical and Electronics Engineers, Inc.
IMP	internal macroinstruction procedures
INET'93	The 1993 annual conference of the Internet Society
I/O	Input/Output
IOCTL	I/O Control
IPC	inter-process communication
ISI	Information Sciences Institute
JPL	Jet Propulsion Laboratory, Pasadena, CA
MIMD	multiple-instruction, multiple data

MIT	Massachusetts Institute of Technology
NFS	network file system
NNSC	NSF Network Service Center
OCSG	Open Computing Security Group
PRM	Prospero Resource Manager
PROM	programmable read-only memory
PVM	Parallel Virtual Machine
RISC	reduced instruction set computer
SIB	Sun Interface Board
SIMD	single-instruction, multiple data
SIMMs	single in-line memory module
SIPG	Sun Interface Monitor to Ginzu
SPCL	Symult 2010 Parallel Common Lisp
SUP	System Update Protocol
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TMC	Thinking Machines Corporation
TQ	Transitive query
UART	universal asynchronous receive transmitter
UCLA	University of California, Los Angeles
UDP	User Datagram Protocol
USC	University of Southern California
VMTP	Versatile Message Transport Protocol
VPR	virtual processing ratio
WAIS	Wide Area Information Service

APPENDIX B - PUBLICATIONS

The following papers report research that was conducted wholly or in part through the DIVIRS/ CERPASS effort. Copies of each paper are attached.

[1] Chandy, K. M. and R. Sherman. *Space-Time and Simulation.* Proceedings of the SCS Multiconference on Distributed Simulation, Tampa, Florida, March 1989. [Reprinted as USC/Information Sciences Institute Reprint Series ISI/RS-89-238, June 1989.]

[2] Chandy, K. M., and R. Sherman. *The Conditional-Event Approach to Distributed Simulation.* USC/Information Sciences Institute Research Report ISI/RR-88-226, June 1989.

[3] Evripidous, P. *Data-Flow Computing: A Status Report.* Published by Prentice Hall, June 1989.

[4] Evripidou, P. and J-L. Gaudiot. *A Decoupled Graph/Computation Data-Driven Architecture with Variable-Resolution Actors.* Presented at the International Conference on Parallel Processing, Saint Charles, Illinois, August 1990.

[5] Evripidou, P. and J-L. Gaudiot. *Decoupled Data-Driven Architectures with Vectors and Macro Actors.* Was presented at the CONPAR 90-VAPP IV Joint Conference on Vector and Parallel Processing held in Zurich, Switzerland, 10-13 September, 1990.

[6] Evripidou, P. and J-L. Gaudiot. *Decoupled Multilevel Data-Flow Execution Model.* Presented at the workshop Data-Flow Computing: A Status Report, Eliat, Israel, June 1989.

[7] Evripidou, P. and J-L. Gaudiot. *Some Scheduling Techniques for Numerical Algorithms in a Simulated Data-Flow Multiprocessor.* Published in the proceedings for Parallel Computing 89, Leiden, August 1989.

[8] Evripidou, P. and J-L. Gaudiot. *The USC Decoupled Multilevel Data-Flow Execution Model.* Chapter to appear in Advanced Topics in Data-Flow Computing, Prentice Hall, in press.

[9] Evripidou, P., W. Najjar, and J-L. Gaudiot. *A Single-Assignment Language in a Distributed-Memory Multiprocessor.* Proceedings of the Parallel Architectures and Languages.

[10] John T. Kohl, B. Clifford Neuman, and Theodore Y. Ts'o. *The Evolution of the Kerberos Authentication Service.* IEEE, Distributed Open Systems, Editors F.M.T. Brazier and D. Johansen, pp. 78-94, 1994. USC/Information Sciences Institute Research Report ISI/RS-94-412 (in preparation).

[11] Mizell, David and Richard J. Lipton. *An Extreme-Case Comparison of Optimistic and Conservative Parallel Discrete-Event Simulation Methods.* Submitted to the 1990 Society for Computer Simulation (SCS) Conference on Distributed Simulation.

[12] Mizell, David, and Richard J. Lipton. *Time Warp vs. Chandy-Misra: A Worst-Case Comparison.* In Distributed Simulation, Proceedings of the SCS Multiconference on

Distributed Simulation, San Diego, CA, Simulation Series, Vol. 22, No. 2, January 1990. Reprinted as USC/Information Sciences Institute Reprint ISI/RS-90-253.

[13] Gennady Medvinsky and B. Clifford Neuman. *Electronic Currency for the Internet.* Electronic Markets, No. 9-10, pp. 23-24, October 1993. USC/Information Sciences Institute Research Report ISI/RS-93-414 (in preparation).

[14] Gennady Medvinsky and B. Clifford Neuman. *NetCash: A design for practical electronic currency on the Internet.* In Proceedings of the first ACM Conference on Computer and Communications Security. Fairfax Virginia, November 1993 (copy included with last semi-annual report). USC/Information Sciences Institute Research Report ISI/RS-93-413 (in preparation).

[15] Najjar, W. and J-L. Gaudiot. *Comparative Performance Evaluation of Three Voting Schemes*, 1989. Prepared for submission to a special issue of IEEE Transactions on Computers.

[16] B. Clifford Neuman. *Prospero: A Tool for Organizing Internet Resources.* Electronic Networking: Research, Applications and Policy, Volume 2, Issue 1, Spring 1992. USC/Information Sciences Institute Research Report ISI/RS-92-421 (in preparation).

[17] B. Clifford Neuman. *Proxy-Based Authorization and Accounting for Distributed Systems.* In Proceedings of the 13th International Conference on Distributed Computing Systems. Pages 283-291. Pittsburgh, Pennsylvania, May 1993 (copy included with last semi-annual report). USC/Information Sciences Institute Research Report ISI/RS-93-419 (in preparation).

[18] B. Clifford Neuman. *The Prospero File System A Global File System Based on the Virtual System Model.* Workshop on File Systems, May 1992.

[19] B. Clifford Neuman and Steven Seger Augart. *Prospero: A Base for Building Information Infrastructure.* In Proceedings of INET'93. San Francisco, August 1993. USC/Information Sciences Institute Research Report ISI/RS-93-415 (in preparation).

[20] B. Clifford Neuman, Steven Seger Augart, and Shantaprasad Upasani. *Using Prospero to Support Integrated Location-Independent Computing.* In Proceedings of the Usenix Symposium on Mobile and Location-Independent Computing. Cambridge Massachusetts, August 1993. USC/Information Sciences Institute Research Report ISI/RS-93-416 (in preparation).

[21] B. Clifford Neuman and Santosh Rao. *Resource Management for Distributed Parallel Systems.* In Proceedings of the 2nd International Symposium on High Performance Distributed Computing. Spokane, Washington, July 1993 (copy included with last semi-annual report). USC/Information Sciences Institute Research Report ISI/RS-93-417 (in preparation).

[22] B. Clifford Neuman and Santosh Rao. *The Prospero Resource Manager: A Scalable Framework for Processor Allocation in Distributed Systems.* Concurrency: Practice and Experience. Summer 1994. USC/Information Sciences Institute Research Report ISI/RS-94-410 (in preparation).

[23] B. Clifford Neuman and Stuart G. Stubblebine. *A Note on the Use of Timestamps as Nonces.* In Operating Systems Review, 27(2): 10-14, April, 1993. USC/Information Sciences Institute Research Report ISI/RS-93-418 (in preparation).

[24] B. Clifford Neuman and Theodore Ts'o. *Kerberos: An Authentication Service for Computer Networks.* IEEE Communications Magazine, Volume 32, Number 9, pp. 33-38, September 1994. USC/Information Sciences Institute Research Report ISI/RS-93-413 (in preparation).

[25] Sherman, R., and A. Pnueli. *Model Checking for Linear Temporal Logic: An Efficient Implementation.* USC/Information Sciences Institute Research Report ISI/RR-89-241 (in preparation).

[26] Tung, Yu-Wen. *Mapping Parallel Algorithms to Parallel Computers: An Analysis Using Sorting Experiments.* Currently under review.

[27] Tung, Y., S-H. Chung, and D. Moldovan. *Modeling Semantic Networks on the Connection Machine*, 1989. Submitted to IEEE Transactions on Parallel and Distributed Processing.

[28] Tung, Y. and P. Li. *Parallel Sorting on Symult S2010.* Presented at the Fifth Distributed Memory Computing Conference, Charleston, South Carolina, April 1990.

[29] Tung, Y. and D. Mizell. *Performance Analysis and Modeling of Parallel Sorting Algorithms on the Connection Machine.* Submitted to a special session of the Hawaii Information and Computing Systems Symposium.

[30] Tung, Yu-Wen, and David Mizell. *Two Versions of Bitonic Sorting Algorithms on the Connection Machine.* Presented at the IEEE Parallel Processing Symposium in Fullerton, California.