A New Generation Real-Time Decentralized Operating System

E. Douglas Jensen

Concurrent Computer Corporation
Westford, MA
508-692-6200
edj@cs.cmu.edu, uunet.uu.net!masscomp!jensen

Abstract

Alpha is a new kind of operating system, which is unique in two highly significant ways. First, it is decentralized, providing reliable resource management transparently across physically dispersed nodes, so that distributed applications programming can be done largely as though it were centralized. And second, it provides comprehensive, high technology support for real-time system integration and operation, such as C2 and supervisory-level industrial automation, which consist predominately of aperiodic activities having critical time constraints such as deadlines. Alpha is extremely adaptable so that it can be easily optimized for a wide range of problem-specific functionality, performance, and cost. Alpha is the first systems effort of the Archons Project, and a prototype was created at Carnegie-Mellon University directly on Sun workstation hardware modified to be multiprocessors. It has been successfully demonstrated with a real-time C2 application written by General Dynamics Corp. Continuing research by Concurrent Computer Corp. is leading to a series of enhanced follow-ons to the Alpha prototype: these are portable but initially hosted on Concurrent's MASSCOMP line of multiprocessor products. Both the initial and the subsequent versions of Alpha are sponsored in part by SDIO through the USAF Rome Air Development Center, and are in the public domain for government use.

A Decentralized OS is New

Alpha is oriented towards systems having on the order of 10 to 100 nodes which are physically dispersed on the order of 1 to 100 meters (longer distances are possible). Alpha is for the most demanding kind of situation: mission-oriented systems where all nodes are contributing to the same application, not simply for the network case of individual users at each node doing unrelated computations. Our focus is on having nodes be logically integrated together rather than autonomous. Alpha provides this logical integration by executing on the bare hardware and managing resources in the same sense as a uniprocessor OS does, not by being just a "UNIX-style" user process and providing standard application interfaces and protocols for simple inter-node resource sharing like conventional computer network style distributed OSs do. Resources must often be managed by Alpha across node boundaries in the best interests of the whole application, not just on the usual per-node basis. This necessitates that Alpha also accept responsibility for handling certain fundamental asynchronous concurrency and reliability issues which arise in distributed systems, instead of passing them all up to the users for recurring, lower performance solutions. Alpha provides mechanisms which are necessary and sufficient to maintain consistency of data and correctness of operation at both the OS and application levels despite concurrent execution, and node or communication path failures, using techniques similar to those normally found far above the OS in distributed database systems—e.g., nested atomic transactions, replication. With Alpha, the nodes collectively form a single computer, not a computer network; thus, distributed application software can be written as though it were for a conventional uniprocessor—without even knowing about, much less having to manage, distributed resources.

Alpha is decentralized in another valuable and difficult sense. It does not depend on the existence of any physically or even logically centralized resource management entity or service, such as a "location broker."
"Real-Time" is Different in the System Integration and Operation Context

The term "real-time" is usually intended to mean "deterministic behavior" and "faster is better", particularly in the area of interrupt handling and context swaps. Real-time control in this sense applies only to computer systems which simply do low-level sensor/actuator sampled-data loop applications, and are traditionally designed to have rigidly periodic behavior. But real-time system integration and operation is far more difficult because it encompasses not just such static periodicity but also predominantly dynamic and aperiodic activities which nonetheless have critical time constraints, such as deadlines. These constraints are part of the correctness criteria of the computation, and failure to meet them is a threat to the system's mission and to survival of property and human life. Alpha personnel invented a novel approach whereby the application's time constraints are expressed in terms of the value to the system of completing each activity as a function of its completion time (deadlines are a simple special case—a step function). In addition, activities have relative importance which are also time-dependent. These time value functions and importances are dynamic and must be continuously re-evaluated. Every evaluation is performed for all executing and pending activities collectively so as to maximize the total value to the system across the whole time period represented by the expected durations of all these activities. This sophisticated and explicit treatment of real time has been conclusively shown in both theory and practice to be exceedingly cost-effective. The conventional and seemingly simpler notions of "priority" in real-time systems are zeroth order approximations which extensive experience has consistently demonstrated introduces massive and uncontrollable complexity into all but the most trivial real-time systems. Alpha employs this new real-time management technique to resolve all contention for resources such as processor cycles, communication access, secondary storage, and synchronizers (e.g., semaphores, locks). Time constraints and importance are among the attributes propagated with computations which cross node boundaries so that resource management can be global. The ubiquitous client/server model is unsuitable in this respect since it does not maintain such essential correspondences between the service and client on whose behalf that service is being provided.

Alpha exhibits a fundamental philosophy which is contrary to that of OSs for other application environments. Instead of optimizing performance of the normal cases at the expense of infrequent ones, it does the opposite. It is in the exception cases such as emergencies (e.g., being in danger due to attack or failure) when a real-time OS must be depended upon to perform best, even if the system's routine behavior must be compromised to ensure that. This is one of the principal reasons why real-time UNIXs are inherently limited.

Of course, Alpha also has all the features usually sought in real-time operating systems, including a fully pre-emptable kernel, synchronization, asynchronous notification, i/o directly to/from user space, contiguous files on disk, memory-locked objects, pre-allocatable resource pools, low interrupt latency and services times, etc.

Extraordinary "Adaptability" is Essential to Real-Time System Integration and Operation

Real-time system integration and operation applications are very complex, and are not (perhaps cannot be) well understood; in addition, the environment and technology are always in a state of flux. Thus, the functional and performance requirements for their computers evolve continuously throughout the entire life cycle of the system, which can be decades. Alpha accommodates this situation through a variety of techniques, many of which are quite innovative. Its design is kernelized and strictly adheres to the principle of policy/mechanism separation. Specific OS policies are carefully excluded from its kernel level mechanisms so that a wide range of different service facilities, and indeed entire DOSs, can be effectively constructed using Alpha's kernel, in accordance with application needs. For example, Alpha's kernel provides atomicity, serializability, and permanence as orthogonal mechanisms. Conventional atomic transaction facilities bundle all three properties together, with correspondingly high overhead, as the only choice of policy regardless of need and affordability. But the client layers of Alpha's kernel can base their policies on other combinations of these mechanisms. For example, there are many instances in real-time systems when problem-specific consistency constraints yield correct results more efficiently than serializability would, or when permanence is not worth its cost. This same philosophy is followed in scheduling, communications, and all other types of resource management.

Computers embedded in real-time systems usually must produce the highest possible performance from the
allowable hardware size, weight, and power, including memory space for the OS. A general-purpose computer system can easily be an order of magnitude lower performance than a special-purpose one for a particular application. Thus, to achieve the balance of performance and flexibility needed for cost-effectiveness in a multiplicity of changing system integration and operation applications, Alpha is general-purpose but unusually malleable so as to exploit all the problem-specific static and dynamic information available from the application. In addition, application functionality can readily be migrated downward into the OS, and even into its kernel, for increased performance when necessary.

Alpha's internals are organized so that its subsystems such as scheduling, communications, secondary storage, etc. can all execute truly concurrently at each node. We intend that these separate hardware points of control within Alpha are a mixture of dynamically assigned general-purpose processors (i.e., each node in the decentralized computer can be a multiprocessor) and algorithmically specialized hardware accelerators (co-processors and other forms of augmentation). Alpha extends to its client applications the same opportunities for taking advantage of multiple special-purpose as well as general-purpose processors at each node.

Alpha presents a programming model which is object oriented, in the sense of abstract data types. This imposes a structure and discipline conducive to modular software at both the DOS and application levels, as well as improving fault isolation. The active entity, or unit of logical computation, is a *thread* stringing through objects via operation invocation, without regard for address spaces or node boundaries; fundamental distribution and reliability issues are the responsibility of Alpha instead of the user. This network uniformity and transparency greatly aids the creation and modification of distributed applications.

**Status**

Alpha Release 1 (done at CMU) has been demonstrated to DoD agencies since late 1987 with a real-time C2 application written by General Dynamics Corporation. Concurrent Computer Corporation is creating Releases 2 and 3 of Alpha which are significantly enhanced and commercial quality; these will be available for experimental use on their multiprocessor products by the Fall of 1989 and 1990, respectively. Alpha is an open operating system in the sense of being both intended and designed for portability to multiple vendors' hardware, and has begun to emerge as the *de facto* standard for next-generation mission-oriented real-time operating systems.

**Acknowledgment**

Alpha is currently funded in part by the SDIO through the USAF Rome Air Development Center, with additional support from Concurrent Computer Corporation, General Dynamics Corporation, and others.

**References**

Northcutt, J. D., Clark, R. K., Shipman, S. E., Maynard, D. P., Lindsay, D. C., Jensen, E. D., Smith, J. M., Kegley, R. B., Keleher and Zimmerman, B. A.

*Alpha Preview: A Briefing and Technology Demonstration for DoD.*

Jensen, E. D., Northcutt, J. D., Clark, R. K., Shipman, S. E., Maynard, D. P. and Lindsay, D.C.

*The Alpha Operating System: An Overview.*

Northcutt, J. D.

*The Alpha Operating System: Requirements and Rationale*
Northcutt, J. D. and Clark, R. K.
*The Alpha Operating System: Programming Model.*

Northcutt, J. D., Clark, R. K., Shipman, S. E. and Lindsay, D. C.

Northcutt, J. D.

Trull, J. E., Northcutt, J. D., Clark, R. K., Shipman, S. E. and Lindsay, D. C.
*An Evaluation of Alpha Real-Time Scheduling Policies.*

Clark, R. K., Kegley, R. B., Keleher, P. J., Maynard, D. P., Northcutt, J. D., Shipman, S. E. and Zimmerman, B. A.
*An Example Real-Time Command and Control Application on Alpha.*

Northcutt, J. D. and Shipman, S. E.

Northcutt, J. D. and Shipman, S. E.
*The Alpha Operating System: Programming Utilities.*

Northcutt, J. D.
*The Alpha Distributed Computer System Testbed.*

Northcutt, J.D.