Operating System Support for Multimedia Applications

Clifford W. Mercer

School of Computer Science, Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
cwm@cs.cmu.edu

Abstract

This paper briefly describes a resource reservation operating system abstraction that supports real-time multimedia applications alongside non-real-time applications. This approach facilitates predictable behavior in applications while still maintaining the flexibility required by multimedia applications in a dynamic system. The accompanying video tape shows how to reserve processor capacity for multimedia applications and how to control processor usage even when real-time and non-real-time applications compete for access to system servers.

1 Reservation for predictability

We address the problems of providing predictable real-time behavior based on high-level quality of service specifications by structuring the system in two parts:

1. a quality of service layer which uses information about user preferences and the relative importance of various types of applications to make resource allocation decisions, and

2. a resource reservation mechanism which takes resource allocation specifications in the form of capacity requirements and schedules contention for the resource accordingly.

Our research focuses primarily on the resource reservation mechanism, specifically processor capacity reservation. The processor capacity reservation system has three components:

1. an admission control policy to determine whether new reservation requests can be accepted,

2. a scheduling policy to arbitrate contention for the resources, and

3. a reservation enforcement mechanism that measures resource usage to ensure that reserved programs don't use more than their reserved capacity.

Reservations can be requested from the system and terminated dynamically, and reservations can even be adjusted on the fly to fit the changing needs of their applications. Even with the flexibility of being able to change reservations, the system assures predictability – because between reservation requests, the application mix and its real-time requirements are static and known to the system.

An initial reservation system has been implemented in Real-Time Mach [3] (an extension of Mach 3.0 [1]) running on a Gateway 2000 66MHz 486-based machine with 16MB of main memory. The demonstrations shown in the video used this machine configuration.

The resource capacity reservation mechanism associated with the processor resource is described in more detail elsewhere [2].

2 The video

In the video, we show two clips of video applications running. The first shows three instantiations of a QuickTime video player displaying a short video clip from main memory. Figure 1 shows these three players as they appear on the screen in the video. These players are scheduled under a time-sharing policy, and their behavior is, as expected, unpredictable.

The second clip shows that the system can reserve processor time for multimedia applications, and then enforce that reservation in the face of competition for the processor. Also, this clip shows that reservations are flexible and can be adjusted dynamically. The three players, arranged on the screen as in the first clip (Figure 1), are running under the reservation system. One of the video players has a reservation of 90% of the processor capacity, so it has ample time to display its frames. The other two players are not reserved; they run under a "background" time-sharing scheduling policy. So their performance is again unpredictable.

During the second demo clip, we show the flexibility of the reservation system by changing the reservations of

This work was supported in part by a National Science Foundation Graduate Fellowship and by the U.S. Naval Ocean Systems Center under contract number N00014-91-J-4061. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of NSF, NOSC, or the U.S. Government.

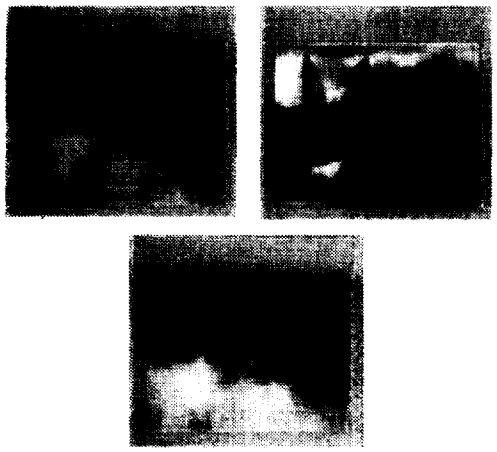

Figure 1: Video Players (from video clip)

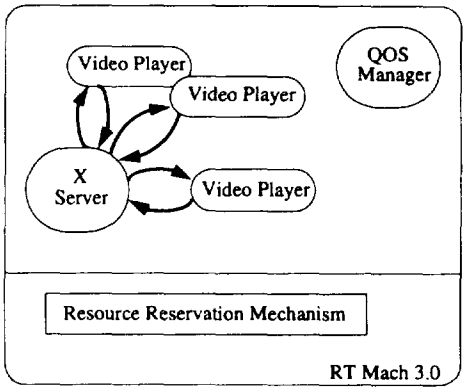

Figure 2: Demo System Software Configuration

the video players on the fly. We remove the 90% reservation from the reserved video players, and give the capacity to one of the previously unreserved players. Now the newly reserved player has the resources to produce a constant frame rate output while the previously reserved player suffers unpredictability in unreserved time-sharing mode. A quality of service manager which keeps track of all the reservations in the system facilitates the coordination of reservation changes.

The scenario depicted in this demonstration is similar to the situation in a video conference with several participants, where the user or a conference manager application would like to be able to change the quality of service parameters of various multimedia data streams based on information like which participant currently has the floor. Our reservation mechanism enables that kind of resource management and control.

3 System configuration

The demonstration shows how the reservation system handles the allocation of processor capacity, but we should note that the end-to-end predictability shown in the demo requires coordination among several system components. Figure 2 shows the structure of the software components in the demo. The players keep their short video clips in memory to avoid interaction with a file server or with a disk.

The important thing to note about this demo is that there is contention in the X server between the frames of the reserved player and the frames of the other players. To achieve end-to-end predictability, the X server must work within the framework of the reservation system; it must service incoming work according to the priority of the client, and it must charge its computation time to the client for whom the service is performed.

The inter-process communication mechanism that clients use to talk to the server must also understand reservations and deliver messages in the order prescribed by the reservation system, otherwise the reservation system can be circumvented, resulting in unpredictable performance.

4 Conclusion

The video shows that are processor reservation system facilitates predictable behavior in real-time multimedia applications, even in the face of competition for system resources from non-real-time programs.

Acknowledgements

The author would like to express his appreciation to Jim Kocher, Chen Lee, Ragunathan Rajkumar, and Jim Zelenka for their help in producing the accompanying video, and to the researchers who have worked on RT-Mach and its associated programming environment: Takuro Kitayama, Tatsuo Nakajima, Ragunathan Rajkumar, Stefan Savage, Hideyuki Tokuda, and Jim Zelenka.

References

[1] D. L. Black et al. Microkernel Operating System Architecture and Mach. In *Proceedings of the USENIX Workshop on Micro-kernels and Other Kernel Architectures*, pages 11–30, April 1992.

[2] C. W. Mercer, S. Savage, and H. Tokuda. Processor Capacity Reserves: Operating System Support for Multimedia Applications. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS)*, May 1994.

[3] H. Tokuda, T. Nakajima, and P. Rao. Real-Time Mach: Toward a Predictable Real-Time System. In *Proceedings of USENIX Mach Workshop*, pages 73–82, October 1990.