REAL-TIME GLOBAL VIRTUAL MACHINES

by

Prof. John A. Stankovic
Department of Computer Science
University of Virginia
Charlottesville, Virginia

The world is replete with the infrastructure for distributed computing. Such computing takes place every day in various forms and is growing exponentially. The future will bring more and more virtual distributed systems that execute on and across this worldwide hardware-software platform. These virtual distributed systems will grow in sophistication, have different purposes, and exist to provide different levels of services, fault tolerance, security, and performance. Each of these virtual distributed systems will continue to evolve on a daily basis and need to interact with other virtual distributed systems.

For example, many military, real-time industrial control, and decision support applications have at their core requirements for managing and analyzing massive amounts of data residing in many data repositories throughout the world. Much of this data has timing attributes. A particular satellite image may be valid for no more than 5 minutes. Stock quotes may be timestamped. Audio and video are continuous media data types with various timing constraints such as end-to-end delay and jitter limits. Since audio and video data are growing in importance for many applications, this gives rise to the need for Quality of Service (QoS) support. Of course, for most applications there is also a need for continued operation in the presence of faults (especially given the large infrastructure) and security.

What software requirements are needed to support a vision of the future for distributed computing that centers around real-time global virtual machines? First, software that supports a distributed virtual execution platforms so that applications can make better use of the vast processing power that exists in the world is necessary. For a discussion of the issues for these platforms the reader is referred to [1]. What is also required is the need for virtual (multimedia) real-time databases (not just WEB browsing or single system queries). It is exciting that products which are moving us towards this vision are appearing. For example, dbANYWHERE (from Symantec) is providing database connectivity via the Java Database Connectivity standard.

This permits downloaded Java applets to make standard SQL calls to various databases such as Oracle and Sybase. The next step that I am proposing is to be able collect sets of databases, producing in essence, a federated distributed database with *tailored* and possibly *specialized* features for a given enterprise. In particular, the specialized properties would be along four dimensions: real-time, Quality of Service (QoS), adaptive fault tolerance, and security.

To implement a real-time global virtual database, new protocols and algorithms are needed that embody (i) a new notion of data deadlines [2] (rather than only transaction deadlines commonly found in real-time database work), (ii) parallel and real-time recovery, (iii) use of reflective information [4] to support flexibility and adaptivity, (iv) a specification language to support security, (v) composible and tailorable fault tolerant entities that synergistically operate with the transaction properties of databases and with the real-time logging and recovery, (vi) a new architecture and model of interaction between multimedia and transaction processing, (vii) a uniform task model that permits hard and multimedia tasks to be treated together in one virtual system, and (viii) new scheduling, resource management and renegotiation algorithms to support various types of end-to-end QoS.

Consider the real-time issue in more depth. Transactions may have deadlines and operate on data whose validity expires with time. Transactions that process data with time validity intervals must use timely and relatively consistent data in order to achieve correct results. New and effective scheduling algorithms have been developed which modify transaction priorities based on data deadlines and forcing transactions to delay processing if data is about to become invalid [2]. These ideas have also been combined with "similarity" notions which says that as long as the data (sensor) value is not "too different" from the value used in the processing, then the transaction can be committed. This enables much more efficient and time cognizant transaction processing especially in overload or crisis situations. Such work must be expanded to consider virtual distributed systems.

In these virtual database systems, real-time aspects will undoubtably become increasingly more pervasive [3] especially as audio and video become commonplace database outputs. The real-time solutions will need to support an increasingly broader spectrum of applications. These solutions will support admission control, resource reservations, synchronization, and renegotiation of QoS. Many applications have widely varying time and resource demands and must deliver dependable and adaptable services with guaranteed temporal qualities. This will be a key challenge.

Queries, update transactions, multimedia playout, and analysis tools may access many objects in this global database. To support fault tolerance on a global basis across these services and objects, adaptive fault tolerance is required. Since fault tolerance is expensive it must be tailored to a user's requirements for it.

Security is a necessity, but it is difficult to achieve especially when it has to be consistent with timing constraints and remain in place even in the presence of faults.

In summary, a global virtual machine can provide a degree of real-time performance, QoS, fault tolerance, and security needed by a certain application class. It can also interact with vast facilities in the global infrastructure. Global database systems can be considered either as adding functionality to distributed and virtual execution platforms or as co-residing with them so that the results of the virtual database queries might be passed to the distributed execution platform for analysis.

References

[1] A. Grimshaw, W. Wulf, and the Legion Team, The Legion Vision of a Worldwide Virtual Computer, *CACM*, Vol. 40, No. 1, January 1997, pp. 39-45.

[2] M. Xiong, R. Sivasankaran, J. Stankovic, K. Ramamritham and D. Towsley, Scheduling Transactions with Temporal Constraints: Exploiting Data Semantics, *Real-Time Systems Symposium*, December 1996.

[3] J. Stankovic, Strategic Directions: Real-Time and Embedded Systems, *ACM Computing Surveys*, Vol. 28, No. 4, December 1996.

[4] J. Stankovic and K. Ramamritham, A Reflective Architecture for Real-Time Operating Systems, *Advances in Real-Time Systems*, Prentice Hall, pp. 23-38, 1995.