Establishing a Real-Time Distributed Benchmark

John Drummond
NCCOSC RDT&E Division (NRaD)
San Diego, CA 92152-5000
(619) 553-4131
drummond@lyra.nosc.mil

Abstract

There are currently very few well-known approaches for performing measurements on real-time distributed environments, none of which can be considered as widely accepted at this point in time. The establishment of an acceptable approach to evaluation for real-time distributed environments will be a far-reaching enterprise. There is much debate about what constitutes a valid performance level evaluation of a real-time distributed system. The eventual specific measurement methodology, however controversial, must be structured upon a basic foundation from which accuracy and correctness can be built. This paper will present elemental foundation details which must be considered in the process of developing and evaluating a real-time distributed benchmarking proposal for possible future standardization.

Introduction

A benchmark is intended to afford us with a mechanism whereby a realistic analysis and comparison can be accomplished. As with any non real-time measurement benchmark, the application being investigated will prescribe the net worth and intrinsic value of the benchmark itself. This will also hold true for real-time, as well as distributed real-time, analysis. For example, from investigation of the design of mission critical application the schedulability and timing correctness are considered of paramount importance, this may or may not be true of other applications. To this end then the realistic analysis goal being pursued will be within a real-time distributed environment and applications specific to this environment.

In the past three distinct levels of system analysis and benchmark classification have been considered[1]. Architectural, Operating System Primitive, and High-level operating system services.

The focus of our analysis must be directed toward the specific characteristics which comprise the distributed real-time operating system itself. The third level consisting of high-level operating system services must obviously be included in this analysis, however, this emphasis may also include ambient factors and conditions of first and second levels consisting of overall architecture and operating system primitives. Other significant issues include utilization of synthetic or application based measurement approach and which one of these can provide the most realistic foundation. The eventual aim is to create a true real-time performance benchmark which is platform independent, or at the very least can isolate, and exclude from impact upon analysis, the components/details which are platform specific.

Real-time Assessment

Critical characteristics of a given real-time system, which can be considered as foundational elements, are those which accentuate timing, guarantee latencies, and predictability. These characteristics must be examined during a creditable real-time analysis and formulate measurement at the foundation level. Specific investigation of these real-time system characteristics must also include a detailed look at a given systems capability to mitigate priority inversion situations. For instance, is Priority Inheritance[7] employed or some other restorative approach. Underlying all these elemental characteristics are the low-level primitives which may have significant impact on the foundational elements including: context switch time; preemption; interrupt latency; semaphore exchange; deadlock resolution; and interprocess communication latency. At the task level, the real-time analysis must also include both periodic and aperiodic processing.

Two reasonably well known examples of real-time analysis benchmarking efforts include Rhealstone[3] and Hartstone[5]. The Rhealstone benchmark has
been categorized as a "fine-grained benchmark"[4] which seeks to examine low level kernel primitives (as listed above) while Hartstone provides a performance view. Any proposed measurement approach should be based upon a figure representative of the culmination of low-level characteristics as well as application level elements. This benchmark measurement proposition must directly address the performance of a real-time task set from the tasks inception (at the application level) through requesting services phase, to completion. The measurement will provide insight into the total expenditures required and give a performance level realistic analysis and comparison can be accomplished.

The Distributed Factor

The accurate and acceptable measurements performed on a typical real-time system soon become ineffective when a distributed qualities are introduced into the environment being measure. Elements from a distributed system perspective can easily overburden a processor to the point of dereliction of servicing the critical characteristics which form the foundational makeup of a real-time system, (timing, guarantee latencies, and predictability). These elements include a significant increase in network protocol activity tasking. Also the degree of shared memory access uniform memory access, non_uniform memory access, to no remote memory access and the variable nature of that access across a real-time distributed architecture will have a significant effect.

All of these elements contribute to a distributed real-time systems performance and must be a fundamental part of any realistic distributed real-time benchmark analysis proposal. By selectively isolating components (explained below) which form the makeup of a real-time distributed operating system the fundamental basis can be developed. As detailed in Distributed Hartstone[1], the isolation of the various functional components found within real-time distributed systems is a sound technical approach. These components consist of: communication latency; priority queuing; protocol engine preemptability; communication bandwidth; and media contention. Once isolated, these functions can decompose a typical real-time distributed architecture into digestible elements. These elements in-turn can be examined in detail to reveal any task level roadblocks, determine timing faults, deadline misses, and failed execution primitives which will provide a foundational base for a sound distributed real-time analysis.

Conclusion

The work this paper is based upon has been concentrated in researching a limited number of real-time operating systems and distributed environments providing for mission critical applications. Among the systems examined in varying detail include RTMach[6], a precursor to today's real-time operating systems, developed by Carnegie Mellon University, continues to provide the real-time community problem solving approaches of the difficult task of combining distributed and real-time characteristics, LynxOS1, and MARUTI developed by University of Maryland. Almost exclusively the noteworthy systems available today are real-time thread based and allow choice of scheduling algorithms. There is also a movement to evolve POSIX into the real-time operating system arena which should lend to a formal standardization format which can be more easily measured. From the viewpoint of one who is working to evolve highly complex system architectures for mission critical applications, the standardization of an agreed upon and acceptable benchmark measuring methodology for real-time distributed systems will be a truly welcome event. Evaluating these architectures and their critical applications require some method of performing a true evaluation. This paper suggests some points which hopefully contribute to this end.

References

[1] Mercer C. W., Ishikawa,Y., Tokuda, H., "Distributed Hartstone Real-Time Benchmark Suite",TR 90, Mar90.
[2] Kar, R & Porter, K., "Rhealstone: A Real-Time Benchmarking Proposal", Dr Dobb's Journal, Feb. 1989.
[3] Kar, R., "Implementing the Rhealstone Real-Time Benchmark", Dr Dobb's Journal, April, 1990.
[4] K. Ghosh, B. Mukherjee, and K Schwan, "A survey of real-time operating systems -draft" TR, Feb, 1994, pp41.
[5] Weiderman, N., "Hartstone: Synthetic Benchmark Requirements for Hard Real-Time Applications" TR, CMUSEI-89-TR-23, June, 1989.
[6] Tokuda H., and Nakajima T., and Rao P. "Real-Time Mach: Towards a Predictable Real-Time System" Proceedings of USENIX Mach Workshop, Oct, 1990.
[7] Sha, L., Rajkumar, R. and Lehoczky, J. P., "Priority inheritance protocols: An approach to real-time synchronization," IEEE Transactions on Computers, Vol 39, 1990, 1175-1185.

Copyright Lynx Real-Time Systems