Multi-Epoch Scheduling Within the Real-Time Execution Performance Agent Framework

Sam Siewert siewerts@thinker.colorado.edu
Department of Computer Science *University of Colorado, Boulder, CO 80309-0520*

Gary Nutt nutt@cs.colorado.edu
Department of Computer Science *University of Colorado, Boulder, CO 80309-0430*

Abstract

In earlier work, we have described the Real-Time Execution Performance Agent (RT EPA) which accommodates mixed hard and soft real-time processing with measurable reliability by providing a confidence-based scheduling and execution fault handling framework. Based on experience with the RT EPA with a space telescope application, a theory was formed for determining scheduling feasibility for a set of services partitioned into multiple scheduling epochs, which are simultaneously active, yet mutually exclusive. This paper explains the multi-epoch theory whose utility has been demonstrated, but for which a formal implementation within the RT EPA framework remains to be completed.

1.0 Introduction

The RT EPA is a framework for ongoing research for systems that support mixed hard and soft real-time services, especially systems which include processing pipelines between source and sink device interfaces . The RT EPA theoretical results include three major new real-time theories [Si00]:

1) An engineering view of real-time scheduling that inherently includes mixed hard and soft services with quantifiable reliability and confidence in the system.
2) Confidence-based thread admission and monitoring reducing RM (Rate Monotonic) pessimism in execution time bounds and providing reliable response prior to deadlines.
3) Evidence that thread sets can be admitted to multiple on-line epochs with priority changes between epochs, but fixed within an epoch.

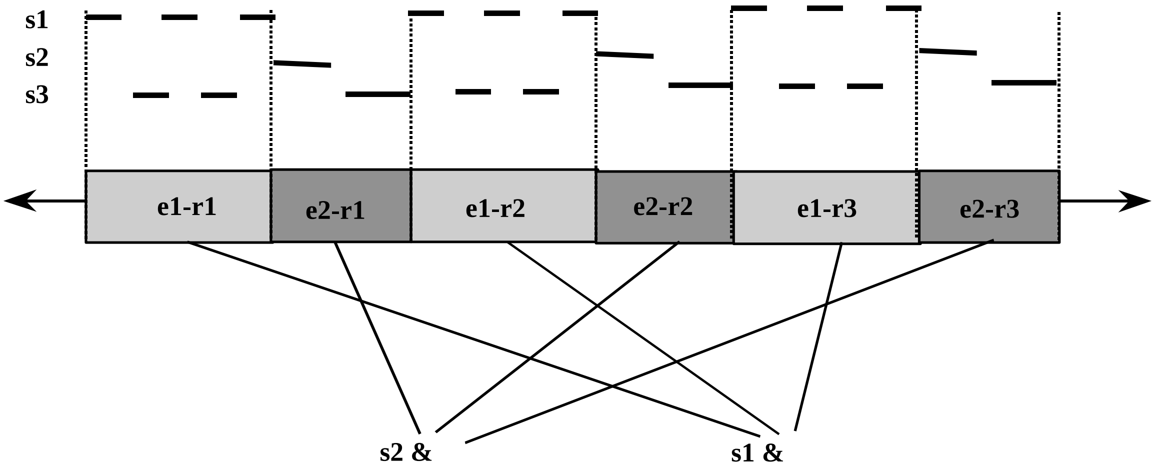
Research contributions 1 and 2 are discussed elsewhere [SiNu96], [SiNuHa99]. The third research contribution, multi-epoch scheduling, is described in this paper. Multi-epoch scheduling is orthogonal to the confidence-based scheduling features of the RT EPA, but was derived from experimentation with the RT EPA kernel monitoring technology as applied to the NASA SIRTF/MIPS (Space Infrared Telescope Facility / Multi-band Imaging Photometer for SIRTF) space telescope instrumentation processing [Si00b]. Without multi-epoch scheduling, the SIRTF/MIPS application would not provide reliable telescope exposure data processing and the science of the mission would have been adversely affected. This paper describes RT EPA multi epoch scheduling theory, presents preliminary results from the SIRTF/MIPS application, and discusses how the theory can formally be integrated into the existing RT EPA scheduling and execution monitoring framework.

2.0 Multi-Epoch Scheduling Theory and Related Research

Traditional RM theory establishes the window of scheduling feasibility to be the longest overall period based upon the theorem formalized by Lehoczky, Sha, and Ding [BriRoy99]. In contrast, the goal of Pfair is to ensure that all tasks make progress at a steady rate, proportional to the weight of the task (utility) [Baruah97]. The Pfair algorithm reduces the window of scheduling feasibility to a window shorter than even the shortest release period. *Multi-epoch scheduling* lies between the Pfair extreme and RMA (though it is defined at a much higher level of granularity than either. MES, considers the possibility that multiple, independent chains of dependent real-time tasks – scheduling epochs – can be active at the same time, but only one of the epochs can be released at any given point in time (see **Figure 1**). This scenario is common to any system, which has mutually exclusive operational modes as the

SIRTF/MIPS application does. Epochs, then, are application-dependent sets of services independently scheduled and defined by the application programmer (based on design principles and on-line monitoring).

Figure 1: Multiple Epochs of Scheduling Active Simultaneously



s2 & s3 => admitted to all e1 epoch releases with unique negotiation
s1 & s3 => admitted to all 32 epoch releases with unique negotiation

The RT EPA concept for epochs assumes that they are all active, but mutually exclusive such that there is no need to consider the possibility of two epochs being released in a critical instant. Epoch phasing is specified to the RT EPA and the RT EPA admits services to an individual epoch just as in a conventional single epoch system based upon:

1) Service code to run.
2) Release frequency within the epoch, resource needs, and deadline requirements.
3) Desired service quality (for RT EPA mixed hard and soft service scheduling).
4) Event or time release source of the epoch.

For each service admitted into an epoch, the RT EPA must assign a resource usage priority, monitor actual usage and control overruns just as it does for a single system. If a service is admitted to more than one epoch, then it will have multiple release and execution contexts and therefore at the system level (over all active epochs), it has dynamic priority, but fixed priority within a single epoch. The dynamic priorities of MES are more constrained than dynamic schemes such as EDF (Earliest Deadline First) since EDF must adjust priorities at each new service release whereas MES must only adjust them between epochs as a known set [LiuLay73].

Just like a service in a system, an epoch must have a policy for becoming the active epoch (equivalent of a service being dispatched). Since epoch overhead is not insignificant and since it is envisioned that most applications will have limited need for multiple epochs, the proposed RT EPA active epoch policy is simple:

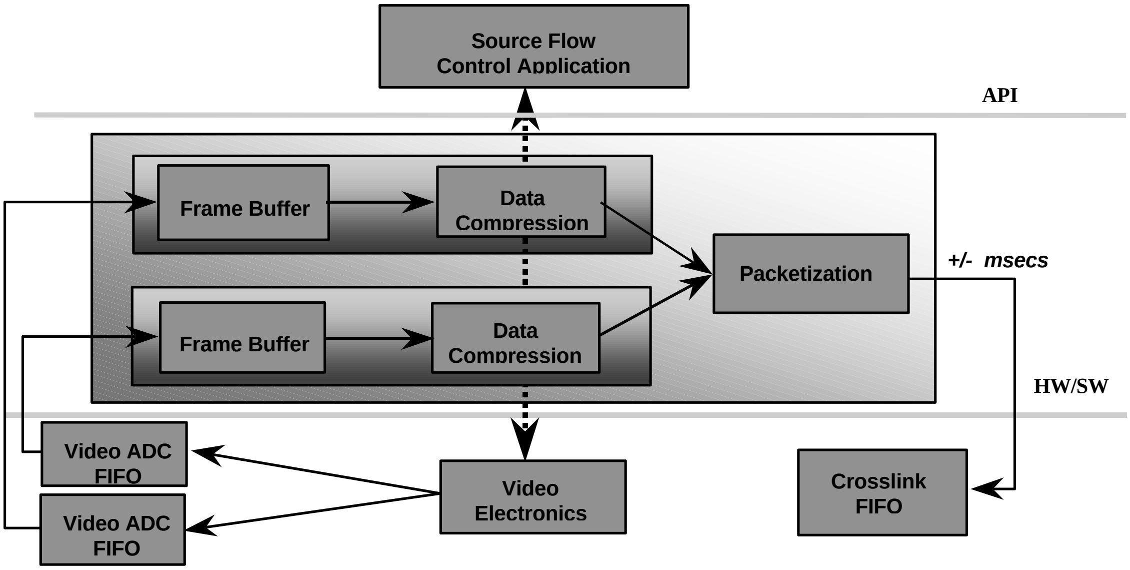
1) The epoch must be based upon an event that is part of the normal system event stream.
2) The epoch is active up to a deadline that must be greater than or equivalent to the longest period service admitted to the epoch.
3) Epochs are activated and deactivated and priorities adjusted according to specified event release and duration.
4) If an epoch releasing event occurs before the current epoch has reached its duration deadline, this is a phasing error and the RT EPA system will execute an error handler or ignore the event.

Due to the overhead of multiple levels of dispatch and preemption and the overhead of adjusting service priorities for the currently active epoch, it is envisioned that most systems would make use of only a few epochs. However, as demonstrated by SIRTF/MIPS, a few epochs can be used to solve an otherwise difficult problem that is not so easy to handle efficiently with single epoch RM methods such as period transform and sporadic servers [BriRoy99]. Furthermore, the concept of operational modes, used for example in Space Transportation System flight software is a well-proven approach to dealing with real-time demands that change over the course of a longer period (e.g. Space Shuttle mission from ascent to orbit, to de-orbit, to descent with different services provided by software in each mode) [Carlow84]. However, in the case of the Shuttle flight software, the transition between operational modes is a one-time transition and MES extends this to include on-line transitions with memory resident services which may be periodic.

3.0 SIRTF/MIPS Preliminary Results with MES

The importance of scheduling epochs was discovered during RT EPA monitoring experiments with the SIRTF/MIPS software. The SIRTF/MIPS software processes two video streams and forms combined compressed packets for down-link as shown in Figure 2. Initially, exposures could not be scheduled reliably with the software due to four distinct modes of processing with differing deadlines and execution variances that were causing processing to miss deadlines and ultimately time out.

Figure 2: SIRTF/MIPS Dual Stream Pipeline



The SIRTF/MIPS four distinct modes of processing were used to define epochs, which included:

- e0: The instrument ready state where telemetry is processed and clocking hardware is ready to start clocking out exposure and digitizing exposure data.
- e1: An exposure start event ends e0 ready and starts e1 exposure start preprocessing and software/hardware synchronization.
- e2: An exposure collection cycle start event ends e1 and starts e2 data collection and processing for the steady state prior to a detector electronics reset to avoid saturation.
- e3: A programmed detector reset event ends e2 and starts e3 data processing completion. The e3 epoch is ended by one of two events – either return to the ready state and e0 or return to data collection and e2.

The segmentation of the SIRTF/MIPS scheduling into these 4 service epochs made it possible to analyze the scheduling feasibility in each independently. Table 1 enumerates the service releases in the steady-state exposure processing mode epochs e2 and e3. To solve the SIRTF/MIPS timing problems, priority was adjusted between epochs and one service allocation change was made between epochs to provide system level scheduling feasibility based on this epoch decomposition. Without this epoch-based dynamic priority adjustment and this reallocation of services to independently scheduled epochs, the SIRTF/MIPS instrumentation software would never have been able to meet exposure processing deadlines reliably. The RT EPA kernel level monitoring capabilities were used to identify the loading in each epoch and to determine priorities in each epoch.

Table 1: Multiple Epoch Design of the SIRTF/MIPS Steady-State Video Processing

task ID	**Description**	**Release Period (msecs)**
Epoch 2	Reset and Initial Sample	589.824
1	Si FIFO driver	inactive
2	Ge FIFO driver	233.02
3	Science Link FIFO driver	32.768
4	Ge Processing and Compression	131.072
5	Si Compression	589.824
6	Science Grouping	589.824
Epoch 3	Sample Frame Period	2031.616
1	Si FIFO driver	65.536
2	Ge FIFO driver	233.02
3	Science Link FIFO driver	inactive
4	Ge Processing and Compression	131.072
5	Si Compression	589.824
6	Science Grouping	1048.576

The details of all 4 epochs are given in [Si00], but here we focus on e2 and e3, the steady state processing epochs, which coexist on-line during data processing. Once the system is running, it does on-the-fly data compression from three video sources and has 2 clear epochs: 1) compressed image computation and cross-link (*e2*) and 2) image-ramping data collection (*e3*). The deadlines for processing during the steady-state epochs are such that the processing falls behind no more than one Si frame (1048 msecs) during e2 and such that all data collection cycle data (between detector resets) is fully processed before the next cycle begins to avoid buffer overflows.

The importance of scheduling epochs is clearly demonstrated by the RT EPA monitoring experiments with the SIRTF/MIPS instrument video processing software. Without the epoch analysis and redistribution of releases, this instrument would not have been able to meet its requirements for real-time processing at all. The system has been operating for many months without a single missed deadline (since March 2000). The identification of the MIPS steady state exposure processing epochs using the RT EPA kernel monitor are presented along with bus analyzer timing analysis for comparison in [Si00] [Si00b].

4.0 MES RT EPA Integration

The current RT EPA API (Application Programmer's Interface) is documented in detail in [Si00]. The interface supports initial negotiation for service, renegotiation, specification of on-line monitoring and control, and configuration of inputs and outputs for pipelining services.

In the current implementation, the RT EPA supports a single epoch system;the current API and framework must be extended to accommodate MES with three major changes. First, the API will require a new function to define system epochs including epoch release event and epoch duration and modification to the rtepaTaskAdmit function to specify the admission epoch. The modified rtepaTaskAdmit will then determine scheduling feasibility for each epoch – in the degenerate case of a single epoch, the RT EPA should function as it does now. Second, priorities must be assigned to services in each epoch and services which are admitted to multiple epochs must have multiple priorities in each epoch adjusted automatically by the RT EPA. Finally, the RT EPA must monitor the multi-epoch system to handle an errant release of an epoch prior to completion of the current epoch and to handle release of a service that is not a member of the current epoch set of services.

Work to extend the RT EPA API and on-line functionality as described above is in progress, but based upon the success with the SIRTF/MIPS application, it is believed that this single application solution using MES can be extrapolated to provide a generic MES capability within the RT EPA framework.

5.0 Conclusion

The viability and significance of MES has already been shown for the SIRTF/MIPS application. The system is scheduled to be launched in late 2001 and without MES, the SIRTF/MIPS instrumentation would not be able to safely perform the full range of exposures it was designed to perform on the SIRTF telescope. With MES, the SIRTF/MIPS system has been under test since March 2000 and has not had a timing glitch. Even if the highly loaded system were to experience a very occasional timing glitch, this is much more acceptable than not being able to perform the full range of exposures at all.

The success of the SIRTF/MIPS application of MES demonstrates that the inclusion of MES in the RT EPA for a general MES implementation is a worthwhile pursuit, and furthermore, given that single epoch MES is equivalent to the current RT EPA, the extension of the RT EPA to include MES will simply enhance the already well demonstrated advantages of the RT EPA for pipelined mixed hard and soft deadline applications. Finally, an alternative testbed (RACE [Si00]) will be used to evaluate the RT EPA, and to provide a rigorous experimental evaluation of the MES as it was used to evaluate the initial RT EPA. Given the flexibility of RACE to simulate conditions similar to that of SIRTF/MIPS (e.g. variable frame rates and multi-phase frame acquisition where frames can be ignored or processed on a configurable periodic basis), the evaluation of the more generic MES will provide additional evidence that the MES solution to SIRTF/MIPS scheduling problems can be extrapolated to a generic solution for a whole class of applications.

6.0 References

[BriRoy99] Briand, Loïc and Roy, Daniel, Meeting Deadlines in Hard Real-Time Systems – The Rate Monotonic Approach, IEEE Computer Society Press, 1999.

[Carlow84] Carlow, Gene, “Architecture of the Space Shuttle Primary Avionics Software System”, Communications of the Association for Computing Machinery, Vol. 27, No. 9, September, 1984.

[LiuLay73] Liu, C., and Layland, J., “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment”, Journal of the Association for Computing Machinery, pp. 46-61, Vol. 20, No. 1, January 1973.

[Si00] Siewert, S., “A Real-Time Execution Performance Agent Interface for Confidence-based Scheduling”, Ph.D. Dissertation , Dept. of Computer Science, Univ. of Colorado Boulder, 2000.

[Si00b] Siewert, S., “MIPS Flight Software Reference Timing Model”, Ball Aerospace and Technologies Corp. Systems Engineering Report, MIPS.SER.SW.00.002, June 1, 2000.

[SiNu96] Siewert, S., and Nutt, G., “A Space Systems Testbed for Situated Agent Observability and Interaction”, In the Second ASCE Specialty Conf. on Robotics for Challenging Environments, Albuquerque, New Mexico, June 1996.

[SiNuHa99] Siewert, Sam, Gary Nutt, and Elaine Hansen, "The Real-Time Execution Performance Agent: An Approach for Balancing Hard and Soft Real-Time Execution for Space Applications," *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, June, 1999.