ELSEVIER

Journal of Systems Architecture 46 (2000) 305–325

JOURNAL OF SYSTEMS ARCHITECTURE

www.elsevier.com/locate/sysarc

The meaning and role of value in scheduling flexible real-time systems [1]

A. Burns [a,*], D. Prasad [a], A. Bondavalli [b], F. Di Giandomenico [b], K. Ramamritham [c], J. Stankovic [d], L. Strigini [e]

[a] *Real-Time Systems Research Group, Department of Computer Science, University of York, Heslington, York Y010 5DD, UK*

[b] *CNR, Pisa, Italy*

[c] *Department of Computer Science, University of Massachusetts, USA*

[d] *Department of Computer Science, University of Virginia, Charlottesville, VA USA*

[e] *Centre for Software Reliability, City University, UK*

Abstract

The real-time community is devoting considerable attention to flexible scheduling and adaptive systems. One popular means of increasing the flexibility, and hence effectiveness, of real-time systems is to use value-based scheduling. It is surprising however, how little attention has been devoted, in the scheduling field, to the actual assignment of value. This paper deals with value assignment and presents a framework for undertaking value-based scheduling and advises on the different methods that are available. A distinction is made between ordinal and cardinal value functions. Appropriate techniques from utility theory are reviewed. An approach based on constant value modes is introduced and evaluated via a case example.

Keywords: Flexible scheduling; Utility of service; Value assignment

1. Introduction

The specification and design of any computer-based system (real-time or not) inevitably involves trade-offs and compromises. Not all possible services (to use a general term) can be supported. And those services that are implemented will not all have maximum functionality.

With most current real-time systems, the implementation phase has little flexibility. A fixed set of 'hard' services must be mapped on to the available resources, and pre run-time checks must be made to ensure that all timing constraints (typically, deadlines) are satisfied. This, of course, requires a predictable (bounded) model of the environment's impact on the computer system.

1.1. Need for dynamic scheduling

In this paper we attempt to increase the flexibility of real-time systems by allowing certain decisions about the system's behaviour to be made at run-time. This will require some form of dynamic scheduling. The notion of 'value' (of a service or sub-service) will be used to control the run-time decision process. Although certain aspects of

* Corresponding author. E-mail: burns@cs.york.ac.uk

[1] This research was initiated by the AFERT Project (041), supported by a joint EU US funding arrangement, and is now being carried out under the auspices of the VASTU project (GR/L73456) funded by the UK EPSRC.

1383-7621/00/$ - see front matter
PII: S 1 3 8 3 - 7 6 2 1 (9 9) 0 0 0 0 8 - 9

value-based scheduling have been addressed before [21,4,1,38,7,15], no systematic study has yet been published. In this paper we attempt to lay the foundation for the systematic approach that is needed.

The motivation for incorporating run-time decisions is largely based on two observations about the behaviour of typical static schedules.

- They use resources inefficiently. As there are sufficient resources to cope with the maximum possible load on the system (i.e. worst-case execution times, worst possible phasings, and worst-case arrival of sporadic work) the average resource utilization is low. Hence, there is considerable scope for value-added computation.
- They react inflexibly to failures and overloads. Although the static scheme may be able to cope with certain failures (as defined in its failure model), once the system moves outside its failure model or load hypothesis then no level of service can be depended upon. Hence, it is desirable to allow graceful degradation (of service) when resources are scarce.

The need to support value-added computation and graceful degradation is necessarily complicated by the reality that not all services have equal utility. When an implementation is constrained to support all services (all of the time), this is not an issue. For run-time decision making, it becomes an almost insurmountable one (because of the excessive overheads it inevitably introduces, at least in the general case).

1.2. Need for value-based scheduling

To keep the terminology clear we shall use the term ‘utility’ to mean the actual benefit that accrues from the delivery of services. In general this measure will not be known with absolute certainty. The value of a service will be some approximation for utility, used to influence the real-time scheduling behaviour of the computer system. During the development of any system many design trade-offs are made. The system designer(s) must articulate preferences between services (or groups of services). Such preferences may be static or vary depending on operating conditions. They are based upon an informed belief in the utility of the system as represented by its services. Due to the complexity of run-time decision making most trade-offs are evaluated at design time. To postpone such decisions until run-time can only be worthwhile if

- The decision becomes much more effective if it takes into account run-time data/conditions.
- The decision would lead to pessimistic resource usage if taken pre run-time.

To facilitate run-time decision making requires the preference relationships to be stored and searched. However, this is only feasible if the overheads involved in making such decisions do not overshadow the potential benefits. This issue is a critical one, many published results in dynamic scheduling have ignored run-time overheads and are therefore misleading. As Chen and Muhlethaler [16] discuss, such scheduling is invariably NP-hard.

There are clearly a number of ways of holding preference relationships. In this paper we consider the use of a value attribute assigned to services and subservices. We present a framework (in Section 3) for value-based scheduling, and use this, in Section 4, to evaluate a number of approaches that have been reported in the literature. Section 5 of the paper then returns to the key issue of assigning values so that they represent an adequate approximation of the designers' preferences. An extensive example is introduced in Section 2 and considered in detail in Section 6. Conclusions are presented in Section 7.

2. Value-based scheduling

In simple terms, value-based scheduling is a decision problem involving the choice of a collection of services to execute so the ‘best possible’ outcome ensues. At various decision points (at run-time) there are a set of services that are available for execution. Unfortunately there may not be enough resources to satisfy all services. And hence, a decision must be made. This decision may involve picking out the ‘extra’ services to support when resources are spare, or which services to sacrifice when resources are scarce. Note that the particular scheduling regime is not significant; it