

Position Summary

Bossa: a DSL framework for Application-Specific Scheduling Policies

Luciano Porto Barreto, Gilles Muller
COMPOSE group, <http://www.irisa.fr/compose>
IRISA/INRIA, Campus de Beaulieu, 35042 Rennes Cedex, France
{lportoba,muller}@irisa.fr

Emerging computing models and applications are continuously challenging the operating system scheduler. Multimedia applications require predictable performance and stringent timing guarantees. Embedded systems need to minimize power consumption. Network routers demand isolated execution of active network programs. Meeting all these requirements requires specialized scheduling policies, which traditional scheduling infrastructures are unable to provide.

While it is clear the need for customized scheduling policies, there is a lack of tools that capture the design singularities of schedulers to ease the development process. Moreover, writing schedulers requires deep OS knowledge and involves the development of low-level OS code, which frequently crosscuts multiple kernel mechanisms (*e.g.*, process synchronization, file system and device driver operations, system calls).

We present a framework for easing the development of adaptable process scheduling infrastructures. This framework permits the development and installation of basic scheduling policies, which can be specialized using application-specific policies.

We base our approach on a Domain-Specific Language (DSL) named Bossa. A DSL is a high-level language that provides appropriate abstractions, which captures domain expertise and eases program development. Implementing an OS using a DSL improves OS robustness because code becomes more readable, maintainable and more amenable to verification of properties [2]. Our target is to specialize process schedulers for an application with soft-real time requirements that is able to specify adequate regulation strategies for its CPU requirements.

Our framework architecture relies on two basic components: (i) Virtual Schedulers (VSs) and (ii) Application-Specific Policies (ASPs). We consider a process to be the minimal schedulable entity. Every process in the system is associated with a VS. During initialization, a process registers with a VS either by joining an existing VS or by loading a new VS and joining it afterwards.

A VS is a loadable kernel module that controls the execution of a set of processes. A VS manages processes by selecting a process for execution and determining its CPU quantum. To permit the coexistence of multiple and independent schedulers, VSs can be stacked, forming a tree hierarchy as in [1]. A VS also provides an *interface*. This interface consists of a list of functions and events that can be used in the specification of an ASP. An ASP uses interface functions to exchange data with a VS. These functions provide a local view of the process state stored by a VS.

An ASP is a Bossa program that specializes the behavior of a VS with respect to specific application needs. An ASP is organized into two parts: (i) a list of global variable declarations and (ii) a collection of event handlers. Event handlers are executed either periodically or in response to specific OS events.

To improve robustness, we perform static and dynamic verification on Bossa code. By construction, Bossa programs are not allowed to have unbounded loops and recursive functions. This constraint ensures that event handlers terminate.

We are currently implementing Bossa in Linux. To provide flexibility while retaining acceptable performance, we are implementing a JIT compiler that runs in the kernel. The JIT compiler translates ASP code into machine code, which is then executed in the kernel. We plan to assess our framework by conducting experiments on scheduling infrastructures using real workloads. We will evaluate Bossa by analyzing the behavior of soft real-time applications, such as video players.

References

- [1] B. Ford and S. Susarla. CPU Inheritance Scheduling. In *OSDI'96*, pages 91–105, Oct. 1996.
- [2] G. Muller, C. Consel, R. Marlet, L. Barreto, F. Méryllon, and L. Réveillère. Towards robust oses for appliances: A new approach based on domain-specific languages. In *ACM SIGOPS European Workshop (EW2000)*, pages 19–24, Sept. 2000.