Bachelor Thesis Proposal

# Loadable Plan-Based Scheduler Kernel Module Feasibility Study

**Michael Zent**

Supervisor
**Barry Linnert**

Institute for Computer Science
Freie Universität Berlin

–

## 1. Introduction

In the Grid Computing paradigm of *Advanced Reservation* it is necessary to appoint a *Program Execution Plan* according to which jobs are distributed and run on the compute Grid nodes. The advantage hoped for, over traditional approaches based on job-queues, is a guaranteebility in Quality of Service, esp. regarding the fulfilment of deadlines [1].

That demands for a novel *Plan-Based Scheduler* in addition to the common Queue-Based Schedulers in nowadays Operating System (OS) Kernels [2].

Given the striven for flexibility of a Grid's structure and the heterogenity of the participating nodes it seems, that the required planning functionalities should be addable to the nodes' OS Kernel in runtime. Therefore the on-demand integration of a Plan-Based Scheduler in an already running OS via a *Loadable Kernel Module* [3], as simplifiedly depicted in Fig.1, shall be discussed in this thesis.

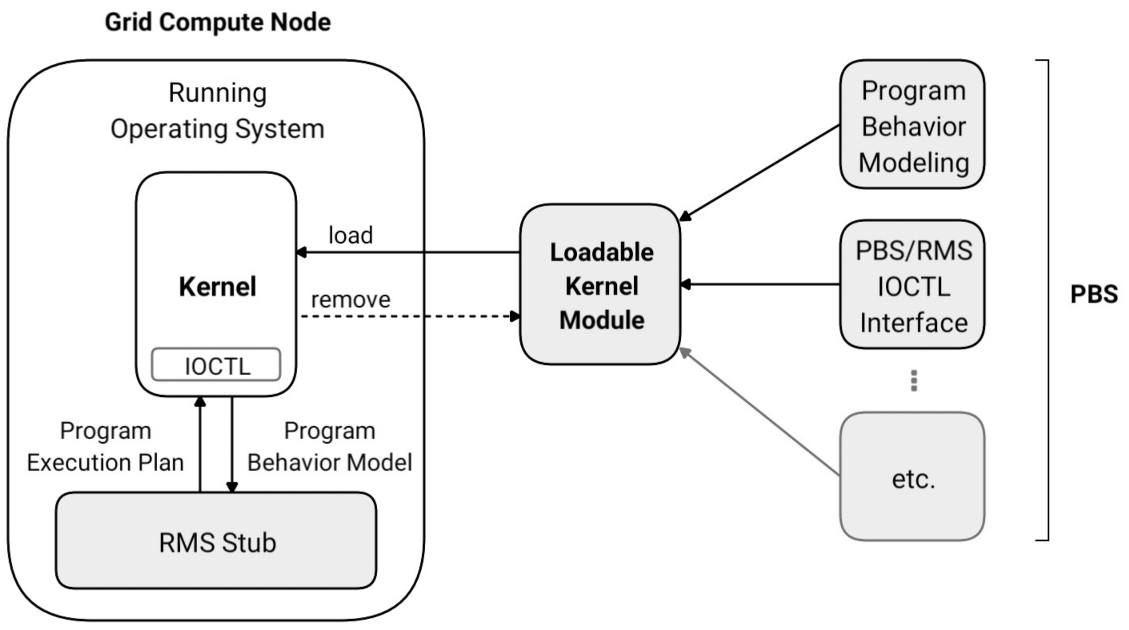

**Fig.1** Implementing a Plan-Based Scheduler (PBS) as a Loadable Kernel Module, communicating with a Resource Management System (RMS) Stub via an interface employing the Kernel's I/O Control (IOCTL) facility. The parts to be developed in this thesis are marked grayish.

## 2. Objective

The thesis' goal is to examine the feasibility of a placement of the Plan-Based Scheduler into a Loadable Kernel Module. That requires the following.

1. Development of a Loadable Kernel Module to extend the OS Kernel by a Plan-Based Scheduler,
2. Implementation of a RMS Stub as a user space program to monitor the functionality of the loaded PBS kernel module, because a fully functional RMS is not, resp. not yet, available,
3. Simple test of the such extended Kernel regarding the acceptance, execution and monitoring of a plan-based program run, and
4. Testing the integrity of the OS after the Loadable Plan-Based Scheduler Kernel Module's removal, i.e. basically that the usual mode of working of the OS is ensured.

## 3. Environment

In principle the intended procedure should be applicable likewise on a series of OSs, inter alia Linux [3], FreeBSD [4], and even Windows [5]. Thereby of course also the heterogenity of Grids would be supported.

In this thesis it will be experimented with Debian GNU/Linux firstly -- as a Guest OS in a QEMU-based Virtual Machine to avoid peril for the integrity of the computer's OS.

## 4. References

[1] L.-O.Burchard, M.Hovestadt, O.K.A.Keller, and B.Linnert, "The virtual resource manager: An architecture for SLA-aware resource management", in *4th Intl. IEEE/ACM CCGrid 2004,* Chicago, USA, 2004.

[2] J.Schneider and B.Linnert, "List-based Data Structures for Efficient Management of Advance Reservations", Int. J. Parallel Prog. 42:77–93, 2014. [Online] Available: http://dx.doi.org/10.1007/s10766-012-0219-4

[3] P.J.Salzman et al., *The Linux Kernel Module Programming Guide*, 2023. [Online] Available: https://sysprog21.github.io/lkmpg

[4] "Dynamic Kernel Linker Facility", in *FreeBSD Architecture Handbook*, 2023, ch.9, sec.2, pp.157-158. [Online] Available: https://docs.freebsd.org/en/books/arch-handbook

[5] R.D.Reeves, "Kernel Mode Drivers", in *Windows 7 Device Driver*, Boston, MA, US: Addison-Wesley, 2010, pp.148-321.