

To these challenges we add an important constraint on the nature of the solution: one that is introduced by business reality. Previous research efforts have attempted to address simultaneous management of power and performance through centralized solutions that manage power and performance jointly. However, although perhaps feasible in prototypes, centralized approaches do not recognize the reality of today’s IT environments, which typically contain multiple management products specialized to different disciplines including performance and power as well as other characteristics such as availability [15].

In the abstract, it seems clear that multi-agent systems should provide an excellent and fundamentally appropriate paradigm for addressing the data center management challenges we have cited, especially given the business constraints described above. Indeed, this position is consistent with our earlier claim [3] that autonomic computing is a *killer app* for multi-agent systems, given that data center management is one important, specific application domain for autonomic computing.

The purpose of this paper is to support our theoretical argument with a practical demonstration. Specifically, we describe an implemented system and experiments that demonstrate coherent automated management to specified power and application performance objectives in a medium-sized server cluster housed in a single IBM BladeCenter chassis. The agents are commercial or freeware products, or sub-components thereof, to which has been added a thin layer of communication and some fairly simple algorithms.

The remainder of this paper is structured as follows. Section 2 sets the stage for our current work by providing a bit of historical perspective. In section 3, we describe our multi-agent architecture. Then, in section 4, we describe the results of two experiments that demonstrate successful management of the system to performance and power objectives that are specified using utility functions. Finally, in section 5, we summarize our main points and provide thoughts on future directions.

2. BACKGROUND

Industry is beginning to provide advanced power management features for servers, of which IBM Active Energy Manager (AEM) Version 3.1 extension¹ of the IBM System Director (Version 5.20) [6] is an example. IBM Systems Director features open, standards-based design and broad platform and operating support to enable customers to manage heterogeneous environments from a central point. Director is typically run on a dedicated server and remotely monitors other servers in the data center. In our work, we measure power and control the server performance state using internal, prototype software that is somewhat in advance of the power management features available commercially today.

AEM provides power measurements for servers by using either power measurement circuitry built into IBM servers, or by communicating with rack-level power distribution units to monitor AC power of 3rd party servers. Based on such information, customers can allocate lower power usage on select IBM servers by two different techniques: *power capping* and *power savings*. Power capping lets users set a maximum power level per system while power savings mode lets users manage power usage by running the processors on the server

¹AEM was formerly known as Power Executive.

at their lowest frequency and/or voltage setting. Today, AEM is manually operated by the systems administrator and does not have the capability to use performance information to make power management decisions on demand. In our work, we show that, by casting an advanced prototype of AEM as an agent and situating it in a multi-agent systems with other agents possessing the knowledge and capability to manage performance and other aspects of system behavior, we can achieve autonomic management of power and performance in accordance with specified objectives.

Our approach differs from previous literature on performance and power management in several respects. First, we power servers off when they are not idle, which is inherently able to achieve greater energy savings than clock frequency manipulation techniques that have been studied previously [9]. Moreover, we differ from previous approaches that use intelligent placement of virtual machines or applications to consolidate servers [2] in that we employ load balancing to route work away from servers, allowing them to be powered off. To our knowledge, we are unique in offering a multi-agent, more de-centralized approach, which we believe is increasingly necessary as the system scales up in size, as well as an implemented system in which real (as opposed to simulated) servers are powered on and off, and in which actual power measurements are taken.

3. ARCHITECTURE

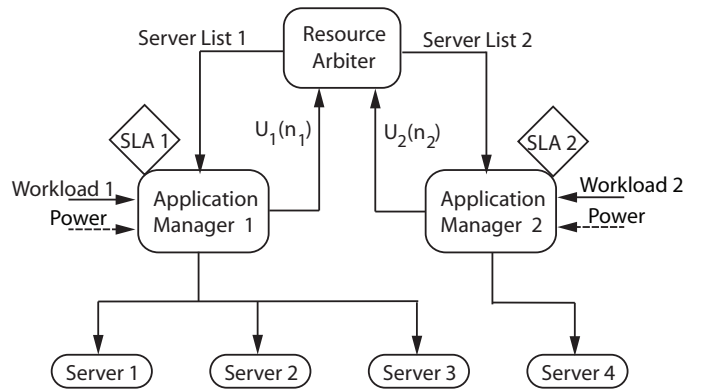


Figure 1: Unity architecture for optimal resource allocation in a data center.

This paper builds upon our prior “Unity” architecture for implementing self-management in a data center via interactions amongst a population of autonomous agents [16]. Unity supports multiple, logically separated *Application Environments*, each providing a distinct application service. Each Application environment is represented by an *Application manager* agent, which is responsible for local performance optimization within the application and communicating with a *Resource Arbiter* agent regarding resource needs (Figure 1). Agent-based resource allocation in the data center is achieved as follows. Each Application Manager i periodically computes and reports to the Arbiter a utility function $U_i(n_i)$ estimating expected business value to the application of receiving an allocation of n_i homogeneous servers. Business value within an application may be defined, e.g., by a performance-based Service Level Agreement (SLA), which stipulates payments or penalties as function of one or more performance metrics. Unity assumes that all

