RECOVERY ACT: DYNAMIC ENERGY CONSUMPTION MANAGEMENT OF ROUTING TELECOM AND DATA CENTERS THROUGH REAL-TIME OPTIMAL CONTROL (RTOC)

Final Scientific/Technical Report

DOE Award Number: DE-EE0002888
Project Period: 01:10 – 03:11

Prepared by:

BAE SYSTEMS

Space Products & Systems
9300 Wellington Road
Manassas, VA 20110

Submitted to:
Industrial Technologies Program
Energy Efficiency and Renewable Energy
U.S. Department of Energy

Principle Investigator (PI): Ron Moon, (703) 367-4687
ronnie.moon@baesystems.com
Team Members: Cisco Systems®, Inc.
TRG Consulting, LLC

Data Centers Energy Reduction and Management Through RTOC

Acknowledgement: This material is based upon work supported by the U.S. Department of Energy under Award No. **DE-EE0002888**

Document Availability:
This report is available free via the U.S. Department of Energy (DOE) Information Bridge Website: http://www.osti.gov/bridge

Reports are also available to DOE employees, DOE contractors, Energy Technology Data Exchange (ETDE) representatives, and Informational Nuclear Information System (INIS) representatives from the following source:

Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 47831
Tel: (865) 576-8401
FAX: (865) 576-5728
E-mail: reports@osti.gov
Website: http://www.osti.gov/contract.html

Data Centers Energy Reduction and Management Through RTOC

(U) DOCUMENT CHANGE HISTORY

Date	Rev	Description
30 June 2011	(-)	Original release.

(U) SOURCE INFORMATION

Application: Microsoft Word for Windows (converted to PDF) 97-2003
Name Version

Name	Date
File: BAE_DOE RTOC_Final Scientific Report.pdf	30 June 2011

Data Centers Energy Reduction and Management Through RTOC

Table of Contents

Table of Figures

Data Centers Energy Reduction and Management Through RTOC

Table of Tables

Data Centers Energy Reduction and Management Through RTOC

Executive Summary

This final scientific report documents the Industrial Technology Program (ITP) Stage 2 Concept Development effort on Data Center Energy Reduction and Management Through Real-Time Optimal Control (RTOC). Society is becoming increasingly dependent on information technology systems, driving exponential growth in demand for data center processing and an insatiable appetite for energy. David Raths noted, "A 50,000-square-foot data center uses approximately 4 megawatts of power, or the equivalent of 57 barrels of oil a day[1]." The problem has become so severe that in some cases, users are giving up raw performance for a better balance between performance and energy efficiency[2]. Historically, power systems for data centers were crudely sized to meet maximum demand. Since many servers operate at 60%-90% of maximum power while only utilizing an average of 5% to 15% of their capability, there are huge inefficiencies in the consumption and delivery of power in these data centers. The goal of the "Recovery Act: Decreasing Data Center Energy Use through Network and Infrastructure Control" is to develop a state of the art approach for autonomously and intelligently reducing and managing data center power through real-time optimal control.

Advances in microelectronics and software are enabling the opportunity to realize significant data center power savings through the implementation of autonomous power management control algorithms. The first step to realizing these savings was addressed in this study through the successful creation of a flexible and scalable mathematical model (equation) for data center behavior and the formulation of an acceptable low technical risk market introduction strategy leveraging commercial hardware and software familiar to the data center market. Follow-on Stage 3 Concept Development efforts include predictive modeling and simulation of algorithm performance, prototype demonstrations with representative data center equipment to verify requisite performance and continued commercial partnering agreement formation to ensure uninterrupted development, and deployment of the real-time optimal control algorithm.

As a software implementable technique for reducing power consumption, the RTOC has two very desirable traits supporting rapid prototyping and ultimately widespread dissemination. First, very little capital is required for implementation. No major infrastructure modifications are required and there is no need to purchase expensive capital equipment. Second, the RTOC can be rolled out incrementally. Therefore, the effectiveness can be proven without a large scale initial roll out. Through the use of the Impact Projections Model provided by the DOE, monetary savings in excess of $100M in 2020 and billions by 2040 are predicted. In terms of energy savings, the model predicts a primary energy displacement of 260 trillion BTUs (33 trillion kWh), or a 50% reduction in server power consumption. The model also predicts a corresponding reduction of pollutants such as SO2 and NOx in excess of 100,000 metric tonnes assuming the RTOC is fully deployed. While additional development and prototyping is required to validate these predictions, the relative low cost and ease of implementation compared to large capital projects makes it an ideal candidate for further investigation.

[1] Raths, David. Govtech.com. 19 December 2006. Web. http://www.govtech.com/magazines/pcio/100560229.html. 20 June 2011.
[2] Hodgin, Rick. geek.com. 26 May 2009. Web. http://www.geek.com/articles/chips/new-server-farm-trend-power-savings-over-performance-20090526/. 20 June 2011.

Data Centers Energy Reduction and Management Through RTOC

Introduction

The Dynamic Energy Consumption Management of Routing Telecom and Data Centers through Real-Time Optimal Control (RTOC) project is a *concept definition study* in the utilization of advanced control methods to minimize telecom and data center power consumption. The advances in control theory and microelectronics over the past several years are enabling a new generation of control algorithms. These modern control algorithms have the capability to operate autonomously and provide optimal solutions given a set of user defined rules or constraints. This study will look at applying the advanced RTOC algorithm to reduce the power consumption of Telecom and Data Centers. The project is funded from the Recovery Act Energy Efficient Information and Communication Technology (ICT) funding opportunity announcement from Department of Energy's Industrial Technologies Program (ITP).

The DOE Industrial Technologies Program sponsoring and overseeing this research and development effort utilizes the Stage-Gate Innovation Management Guidelines. The ITP's goal is to accelerate the use of innovative, energy-efficient, industrial technologies. The Stage-Gate Innovation Management Guidelines map out an effective pathway for innovative technology and new technical information to reach the end-user. These guidelines mature and introduce the technology to market through a series of stages, referred to as stage-gates. This research and development project is Stage 2 Concept Definition. Concept definition is early research exploring and defining technical concepts, focusing on thoroughly understanding and describing the capabilities of the technology.

This project's concept definition efforts set the foundation to enable the commercial application of an energy management RTOC algorithm into the telecom and data center commercial market. RTOC algorithms have reached a mature stage for implementation in today's systems due to the advances in computational mathematics and microelectronic processors. Traditional control method approaches continue to remain in an archaic state because of a failure to challenge the community's basic mathematic assumptions that unduly constrain innovation and fracture the controlled system, reducing efficiency. In order to commercialize an energy management RTOC algorithm, critical research focus areas required development.

The objective of the RTOC algorithm project was threefold. The first objective is to document requirements for implementing a RTOC algorithm in Routing and Telecom Data Centers. The second objective is to formulate a Routing and Telecom Data Center control equation. The third objective is to identify potential avenues for market introduction of the RTOC algorithm and an outline of work applicable for Stage 3 Concept Development efforts.

Data Centers Energy Reduction and Management Through RTOC

Background

The U.S. Environmental Protection Agency (EPA) Report to Congress on Server and Data Center Energy Efficiency clearly identifies the opportunity to save power through improved control algorithms. The EPA report states, "Widespread underutilization of servers is one of the most often-cited reasons for suboptimal energy efficiency in data centers." This occurs because servers typically operate at an average processor utilization level of only five to fifteen percent. The typical U.S. volume server will consume anywhere from 60 to 90 percent of its maximum system power at these low, five to fifteen percent, utilization levels[3]. The concept of maximizing the utilization of processors within a server and removing (i.e. powering down) unloaded servers is called Virtualization. Virtualization has assumed a varied definition that goes beyond this concept, but this study defines it as the process of maximizing the use of the minimum number of servers and powering down idle servers. In this context, virtualization provides one of the most significant opportunities for energy savings in data centers and server installations. It is important to note at this point that the number and footprint of traditional Telecom hardware is declining as the implementation of Voice over IP (VoIP) increases. VoIP hardware equipment is practically identical to that of server and data center hardware, VoIP telecom implementation becoming an application that runs on data server hardware. While standing in an equipment center and viewing the hardware, most individuals cannot tell the difference between VoIP and data centers. The proposed RTOC algorithm is applicable in both telecom and data centers. However, due to their similarity in hardware and significantly greater numbers, this research and resulting reports refer to the application as a data center. When differences between telecom and data center affect the proposed algorithm's application, the differences will be noted.

In order to realize significant energy savings improvements, virtualization software must be expanded to control and coordinate power-management capabilities across the virtualized servers. Virtualization software is the automated tool that assigns and shifts server workloads in the data center, forcing them to operate in an energy efficient region of their overall range. This is a continuous process due to constantly changing computational demand. The implementation of a power management RTOC algorithm would provide the virtualization software control required to realize these energy savings, autonomously and in real time.

Existing Routing and Telecom Data Center power control systems are designed and implemented in a disjointed manner, based upon archaic methodologies, and require significant development efforts to update. Data center power control systems range from manual to automated hierarchical systems, or some hybrid thereof. The manual data centers systems rely on operator action to facilitate power savings, e.g. turn servers off during off-peak hours. Automated data center power saving control systems often operate on crude rule sets, e.g. time-based rule sets. In modern data centers, automated power control systems operate at each equipment hierarchy, each layer of power control attempting to minimize equipment power consumption but often adversely impacting the attempts of associated hierarchical power control layers, resulting in

[3] Environmental Protect Agency. (2007). Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431: U.S. Environmental Protection Agency ENERGY STAR Program

reduced power efficiency. With the advent of remote equipment monitoring and power management, the technology exists to begin the implementation of systems of systems power management approach at the macroscopic level. A modern, autonomous RTOC algorithm coupled with existing microelectronics provides the computational capability to implement a system of systems resource loading and power management control system, providing the capability to realize significant power savings not only in the data center but also across the entire enterprise. Additionally, modern, automated RTOC algorithms provide the opportunity for power savings beyond that of archaic and traditional time–based automated control systems.

Newer enterprise and routing and telecom data center equipment, such as Cisco's EnergyWise open source initiative, provides remote control authority to a wider range of hardware, and to lower hierarchical levels. As newer equipment is deployed, the automated control system can be expanded to move across the micro and macroscopic power systems, providing the capability for these systems to work in a unified manner reducing inefficiencies of competing control loops. The RTOC control algorithm could monitor desired, key control indicators to shift equipment into different power states (off, standby, on, as well as other intermediate states) in order to maintain the required quality of service and minimize power consumption. With modern optimal control algorithms, this power-management control is not limited to the data center, but can be extended throughout the enterprise as the ability to control power states of equipment and devices increases.

In order to begin this control algorithm implementation, the key monitoring parameters, the elements to be controlled and the system mathematical model must be developed. Expecting to immediately autonomously control every power consuming device in a data center campus (e.g. servers, server fans, HVAC, office lighting, phones, etc.) is a daunting and reckless approach. When introducing a significant advance in control methodology, it is prudent to begin with a manageable list of the major power consumers. For this research effort, the system was bounded to the data center servers, chassis, racks, associated fans and Heating, Ventilation, and Air Conditioning (HVAC). Expanding the power management control to the data center campus facility office space lighting, phones, computers, printers, copy machines, etc. is possible given the control authority, control wiring, to this devices, but is better left to a follow-on development iteration or release of the control algorithm. This approach is very common within the software community in spiral development or commercial planned product improvements. With the data center controlled system bounded by the data center servers, chassis, racks, associated fans and HVAC, optimal control theory can be applied.

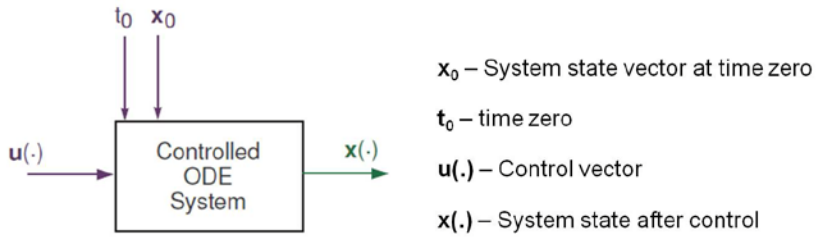
Data Centers Energy Reduction and Management Through RTOC

RTOC Algorithm Overview and Project Analysis

The following sub-sections document the research and development performed by the BAE Systems team under this Stage 2 Concept Definition effort. First, to facilitate a better understanding of the overall research efforts and results, a brief overview of the Real-time optimal control (RTOC) algorithm followed by a tutorial on optimal control is provided. The explanations are provided at a very high level, not all-inclusive, but will provide sufficient background information to assist in the presentation of the research efforts. Each topic below is, in itself, a discipline worthy of a lengthy presentation.

A RTOC algorithm is software code compiled to run with a system model on a computer whose outputs are used to control a system's behavior. Figure 1 is an example of a basic controlled system. The system is at some state or condition (x_0) at time zero (t_0). A control authority issues a set of control commands (u(.)) represented in this example as a mathematical vector. The control devices (e.g. the steering wheel, throttle, brakes etc.) on an automobile are manipulated to their new positions, and the system responds and now is in a new state (x(.)). In this automobile example, the car could be moving slower and traveling in a different direction after the applied control, moving the steering wheel or changing the throttle postion.

Figure 1: Basic Controlled System



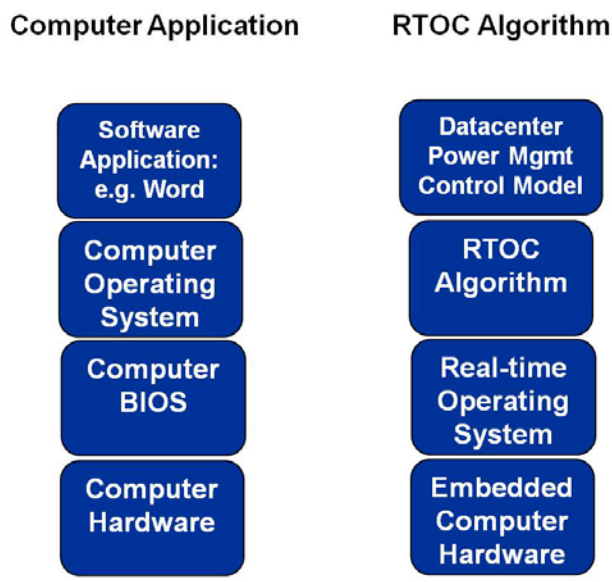
In the case of this research project, the controlled system within the box in Figure 1 is a data center and the control commands (u(.)) are generated and inserted into the system by an automated control application. The control application is the combination of the RTOC control algorithm and the data center system model. The system state within the data center to be controlled are the server power states (on, off or some point in between), the sever rack and chassis cooling fans and center HVAC systems. The power control application software code generating control commands may be compiled to run on a desktop, laptop server computers or within an embedded computer. Examples of embedded computers in other applications are the avionics of an airplane or engine control module of an automobile. The term real-time, within the RTOC acronym, refers to the algorithm providing control commands within the system's control loop cycle, thus, enabling the system to be controlled. In order for the control commands to be useful, the control commands must be generated, issued to the system, and the system respond all within system's real-time control cycle. It is important to note, that the measured elapsed clock time that defines real-time for a system is different from system to system. The

Data Centers Energy Reduction and Management Through RTOC

"real-time" required to control a tractor plowing a field in most cases is much longer than the "real-time" required to control an airplane during a landing. If a control algorithm is unable to meet its particular system's real-time criteria, the system is uncontrollable and will eventually oscillate or drift into an unwanted operating region, such as an airplane banking or diving uncontrollably until it finally crashes.

The following is a simplified analogy to display how the RTOC algorithm is hosted in hardware, but is useful for demonstrating the interaction between computational hardware, software and control algorithms. The RTOC algorithm is hosted on computational hardware much like a computer program runs on a modern personal computer (PC), see Figure 2. A computer program such as a word processing program (e.g. Microsoft Word or Apple's Pages) runs atop a computer operating system (e.g. Windows or OS X) that, in turn, runs atop the computers BIOS, which finally runs on the computer hardware. The data center power management control model is akin to the word processor program analogy and runs on top of the RTOC algorithm that, in turn, runs atop a real-time operating system, which finally runs on the computer hardware. Real-time operating systems may or may not contain a BIOS layer.

Figure 2: RTOC Algorithm Comparison Analogy



Within the RTOC algorithm implementation, real-time is used in two different aspects. Real-time in terms of the control algorithm is explained above, the update time lapse and response of the controlled system to be useful. The second aspect of real-time in this algorithm application is real-time operating system. The “real-time” is similar in context but different in application and the two must be kept in their proper context. Real-time used in the context of an operating system means that the operating system accepts and executes a task in a repeatable, finite amount of time. A real-time operating system is used when the application values how quickly or how

predictably it can respond more so than for the amount of work it can perform in a given period of time. Within the real-time operating system realm, different operating systems are characterized as "soft" or "hard" real-time operating systems depending on whether they can "usually" or "definitely" meet a processing deadline, respectively. How this characterization of an operating system is made is beyond the scope of this research or need to know understanding. The simple knowledge of this difference suffices. One last note on this topic is that PC operating systems (Windows or OS X) are not real-time operating systems. PC and server operating systems focus on maximizing the amount of work it can perform in a given period. Timed execution repeatability of a common task is not as important. With an understanding of software application implementation on real-time operating systems and hardware, it is important to understand optimal control and algorithms.

One does not need to be a control theory expert to understand the concept of optimal control. Control theory is an engineering discipline that attempts to control Nature or inanimate objects while obeying the physical laws. In the abeyance of physical laws, the control engineer is presented constraints. For example, should the control engineer exceed structural loading of an airliner wing with excessive passenger and baggage weight coupled with an excessive allowance in banking or turning rate, the material capabilities of the wing will be exceeded; the wing will deform or fail, causing catastrophic loss. Optimal control introduces the next level of complexity in control theory by not only obeying constraints, physical laws, but also maximizing the performance of the system. In our project's case, we desire to maximize the amount of work assigned to a data center while minimizing the amount of power required to complete this work. Conceptually, the simplest way to look at accomplishing this goal is to fully load the minimum number of servers and supporting cooling equipment to complete the work and turn off the rest of the servers, fans and supporting HVAC equipment. In order to accomplish this, one must formulate the control problem that accounts for rapid and dynamically changing workloads assigned to a data center. First, one must understand the general control problem formulation.

Optimal control problems are dynamic optimization problems[4]. The control algorithm implemented in this project is based upon satisfying Pontryagin's Maximum Principle[5]. Figure 3 below displays the structure or framework of a fundamental control problem. Many control problems can be formulated into this structure model. The bold variables in Figure 3 represent vectors. Upper case variables represent cost functions and lower case variables represent constraints. "**x**" is a system state vector defining the state of the system, which must reside within the constrain set (state space) "**X**". "**u**" is a control vector whose control authority is bounded by the control space "**U**".

Figure 3: Fundamental control problem structure

[4] Ross, I.M. (2005). "Control and Optimization: An Introduction to Principles and Applications", Electronic Edition, California: Naval Postgraduate School.

[5] Kopp, R.E. (1962). George Leitman (Ed.) "Pontryagin Maximum Principle," in Optimization Techniques. New York: Academic Press, Inc.

Data Centers Energy Reduction and Management Through RTOC

$$\mathbf{x} \in \mathbb{X} = \mathbb{R}^{N_x}, \quad \mathbf{u} \in \mathbb{U} \subseteq \mathbb{R}^{N_u}$$

$$(\mathsf{F}) \begin{cases} \text{Minimize} & J[\mathbf{x}(\cdot), \mathbf{u}(\cdot), t_f] &= E(\mathbf{x}(t_f), t_f) + \displaystyle\int_{t_0}^{t_f} F(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \\ \text{Subject to} & \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \\ & \mathbf{x}(t_0) &= \mathbf{x}^0 \\ & t_0 &= t^0 \\ & \mathbf{e}(\mathbf{x}_f, t_f) &= \mathbf{0} \end{cases}$$

where the functions,

$$\begin{aligned} E: &\quad \mathbb{R}^{N_x} \times \mathbb{R} \longrightarrow \mathbb{R} \\ \mathbf{e}: &\quad \mathbb{R}^{N_x} \times \mathbb{R} \longrightarrow \mathbb{R}^{N_e} \\ F: &\quad \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \times \mathbb{R} \longrightarrow \mathbb{R} \\ \mathbf{f}: &\quad \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \times \mathbb{R} \longrightarrow \mathbb{R}^{N_x} \end{aligned}$$

"**J**" is a scalar cost function that is calculated for each potential control solution. The scalar costs for each solution are compared to determine the "optimal" solution for a given system state and desired end state. "**J**" is comprised of an Event (or Mayer) cost (**E**) and running (LaGrange) cost (**F**). This cost formula acknowledges that there is a fixed cost to take a system from one state to another (no free lunch) and, in most control system cases, an increased cost the longer the system takes to transition from the initial sate to the final state. "x_dot(t)" is a system model that maps the state vectors to the new state as a function of time and control vector. $x(t_0)$ is the state vector status at the problem initiation. t_0 is the time at initiation. **e** is an endpoint, or event constraint, function vector. The vector functions **E**, **e**, **F**, and **f** are assumed continuously differentiable with respect to the state variable. Constructing an accurate and properly formatted system model function is critical to the development and implementation of an optimal control algorithm. Developing the system model was the focus of this Stage 2 Concept Development effort.

The proposed algorithm achieves an optimal control solution by maintaining a set of Maximum Principle conditions during the calculation of control vectors transferring the system from the initial condition to the final, while minimizing the Bolza cost function. The Bolza cost function is comprised of functional parameters identified by the user to be maximized or minimized, i.e. server loading and energy usage. The Maximum Principle converts the infinite dimensional problem into instantaneous finite dimensional mathematical programming problem in terms of the control parameter **u**[6]. The Legendre Pseudospectral method[7] is employed to derive solutions that meet, and maintain, the Maximum Principle requirements throughout the solution space. Additional mathematical conditions, such as the Hamilton-Jacobi-Bellman equation, complete the verification of sufficient conditions for optimality, problem formulation, and establish a boundary-value-problem.

[6] Ross, I.M. (2005). "Control and Optimization: An Introduction to Principles and Applications", Electronic Edition, California: Naval Postgraduate School.

[7] Ross I. M. and Fahroo, F. (2003). "Legendre Pseudospectral Approximations of Optimal Control Problems," Lecture Notes in Control and Information Sciences (Vol. 295). New York: Springer-Verlag.

Data Centers Energy Reduction and Management Through RTOC

Many engineering applications can follow this generic problem formulation process, code into computational software and solved within a reasonable amount of time to establish a useful control algorithm. The questions to be investigated in this concept development project were:

1. could the data center energy management problem formulation be accomplished (more specifically, can the data center system model be formulated), and
2. were there data center hardware and software capable, with deterministic periodicity, to calculate the control solution?

1.1 Technical Requirements (SOPO Task 1.0)

The research team investigated, explored and present in this final scientific report the following:

1. Current and projected future computer resource requirements to host an RTOC algorithm in a Data Center environment.
2. Current and projected future hardware requirements to implement an RTOC algorithm in Routing and Telecom Data Centers.
3. Existing and projected future control authority elements available in a Data Center environment.

Note: unless otherwise clarified or defined, "projected future" items are considered capabilities that are expected to be available or deployed to the community within the next five years.

The Accomplishment of this task turned out to be extremely more difficult than expected by the Principle Investigator (PI). The Cisco team members quickly educated the PI and other team members regarding data center equipment configurations. Data center hardware, configurations and functional construct vary so widely that it is extremely rare to find two identical data centers, even within the same company. This situation is beginning to change with the introduction of new data center designs such as containerized computing and company standardized data center designs (Yahoo's Computing Coop). Regardless, this project's RTOC algorithm approach remains valid; however, the findings of this section tend to be more generalized than first anticipated at the onset of this research effort. When the experimental data center is finalized for a Stage 3 Concept Development project, the specific software and hardware configuration can be established.

1.1.1 Current and Projected Future Computer Resource Requirements

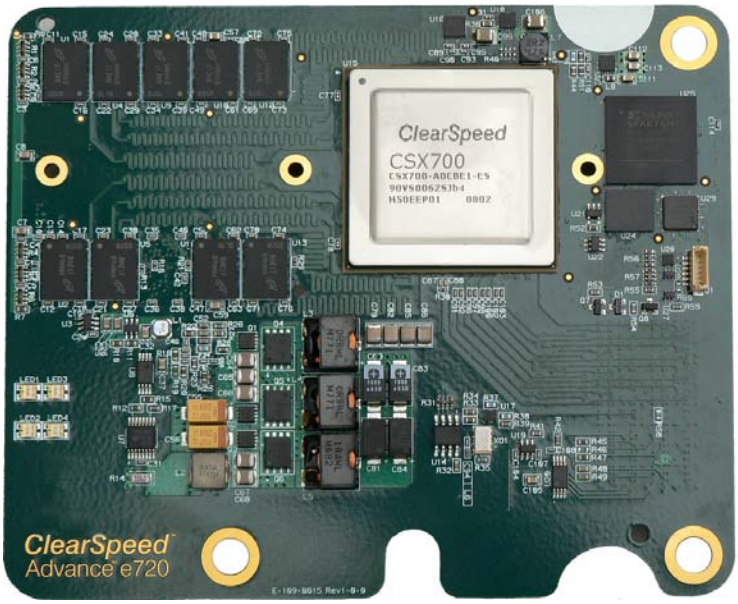
Computationally, the current state of computer hardware is capable of hosting and executing the RTOC algorithm. Moore's law has been a RTOC algorithm enabler concerning microelectronic processing capability. Data centers themselves host computationally powerful microprocessors. At the core of data centers are servers or blade servers. Blade servers are generally smaller, stripped down versions with lower performance of their server counterparts. However, and more importantly, at the heart of either is a powerful Intel™ or AMD™ microprocessor. The key component within the microprocessor that enables the performance of a RTOC algorithm is the microprocessor's floating-point unit (FPU). The RTOC algorithm is dependent upon numerous double-precision floating-point calculations per control solution. Therefore, the faster the

Data Centers Energy Reduction and Management Through RTOC

microprocessor's FPU and input-output (I/O) bus, the faster the RTOC algorithm's execution cycle. A RTOC algorithm variant for use in an International Space Station (ISS) application utilized a PC with a single-core Intel 1.8 GHz Pentium 4 microprocessor, circa 2001. Today, the servers shipping from both Intel and AMD contain 2 - 16 microprocessor cores ranging in clock speeds from 1.8 GHz to 3.1 GHz.

If, for some reason, additional floating-point processing performance is required to augment the algorithm's own microprocessor performance and reduce the algorithm's computation time, a commercially available hardware accelerator, such as the ClearSpeed™ CSX700 can be added. The CSX700 digital signal processor (DSP) was specifically designed to accelerate financial modeling algorithms and scientific applications. The latest CSX700 configuration is the e720 card, see Figure 4, specifically fabricated to be installed inside a Hewlett Packard (HP) blade server without power or cooling modification.

Figure 4: ClearSpeed e720 HP Blade DSP board



(Photo courtesy ClearSpeed™)

There are two exceptions to data center microprocessor viability for RTOC algorithm implementation. First is the introduction of the Intel Atom processor-based servers. The Intel Atom microprocessor line is power efficient at the expense of a robust FPU. Data centers are finding these Atom based servers attractive due to their notable power savings. For general webpage services, a rapidly growing market, server microprocessors do not require robust FPUs, rather rapid integer operation. Second, some data center functional organization or construct does not allow a server on the data center floor to back feed the control algorithm control signals back into the data center's other servers. In both exceptions noted in this paragraph, one solution

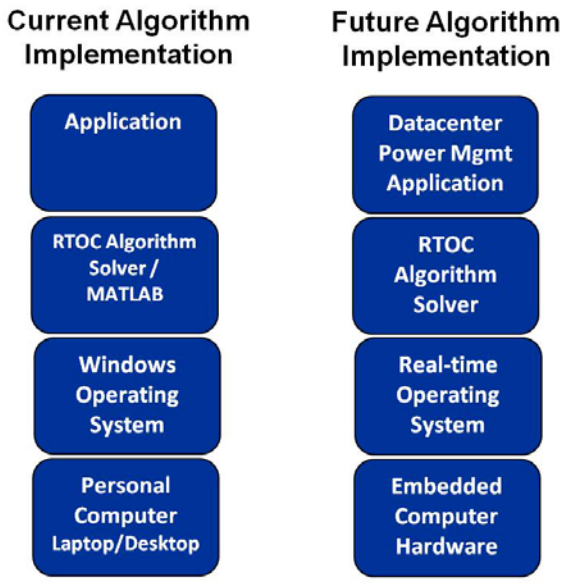
is to add a capable microprocessor (server, blade, PC or embedded computer) to the data center's Load Balancer function to perform the control calculations and dissemination. The Load Balancer will be discussed in more detail later.

Because of the rapid advances in computational data center hardware and software, and the uncertainty of the commencement date of the next development phase, it is recommended that the configuration and control host for the power management application prototyping be identified in the test data center selected for the next project stage. The research team views the technical challenge to host the RTOC algorithm limited by the functional construct and organization of the data center and not the commercially available computational hardware or operating system. Numerous commercially available microelectronic processors (general purpose, digital signal processors, hybrids, or embedded) possess capable double-precision floating-point units sufficient to host the control algorithm. The range and capability of microelectronic offerings will continue to increase with time, taking advantage of Moore's Law.

1.1.2 Current and Projected Future Hardware Resource Requirements

The current configuration of the RTOC algorithm implementation used as the basis of this Concept Definition effort is displayed in the left-hand side of Figure 5. The RTOC algorithm runs on a personal computer (PC), laptop or desktop. Algorithm performance improves when the host computer possesses a CPU with a fully capable, double-precision FPU. The computer hardware may utilize Windows 7 or Windows XP.

Figure 5: RTOC Algorithm Implementation



The RTOC Algorithm operates in the current, and previous, version of MATLAB.

Data Centers Energy Reduction and Management Through RTOC

The algorithm consists of MATLAB script files, not Simulink modules, leveraging basic mathematic and matrix library elements. The application that calls the RTOC algorithm is the system model under control. Currently implemented applications range from controlling and balancing an inverted pendulum for demonstration purposes to performing an International Space Station (ISS) Zero Propellant Maneuver.

The right column of Figure 5 displays the conceptual implementation of the data center power management application in future hardware. This application is the fusion of the RTOC algorithm and the system model developed in section 1.3 below. Formulating and verifying a system model for the system to be controlled with the RTOC algorithm is the most critical element of implementing a RTOC algorithm. Other notable differences between current and future hardware resource requirements is the removal of a generic operating system for a Real-time operating system (RTOS), MATLAB and the integration of the data center power management application. For a data center power management application, the Windows operating system introduces an unacceptable time variance in the control calculation. It is anticipated that the control loop calculation time is on the order of milliseconds, or below. The Windows operating system is a very powerful, very flexible generic operating system. This flexibility hosts the ability to have multiple processes running in the background, many inherent to the operating system itself. These processes place demands on the CPU at what appear to be random times and can have over-riding priority interrupts to the desired applications, e.g. anti-virus processes, which is inherent to the task-scheduling scheme within the operating system itself. This loading and unloading the CPU adversely influence time-critical applications' computation time. Real-time applications desire a very deterministic computation cycle to generate a solution, e.g. real-time control applications. The time variance delay in the task calculation time is called jitter. While a credible software programmer can tailor or tune a general-purpose operating system to minimize these jitter sources, real-time operating systems were developed for this purpose and possess a scheduling scheme that minimizes jitter and focuses on computing the critical application in a very time-deterministic manner. Real-time operating systems are classified as a "soft" or "hard" real-time operating system based on how well they minimize computational jitter. While there are many control applications that can afford the jitter of a general purpose operating system, the data center power management application is not one.

MATLAB™ removal is another dependency that will unburden future hardware resources. While MATLAB is a very powerful development platform, it incurs a notable financial and hardware resource cost. Deploying a MATLAB-dependent control algorithm incurs a licensing cost per data center application. Deploying an application dependent upon operating within MATLAB is not practical. The full MATLAB environment is dependent upon Windows or Linux generic operating systems, the unacceptable control jitter introduced is discussed in the paragraph above. MATLAB provides for a method of extracting code developed within the environment, both manual and automated. The automated tools for extracting algorithm developed within MATLAB, while automated, come at a price, bloated code. Regardless, either method maintains a dependency on the MATLAB library file, in one configuration or another. While not a technical constraint, this dependency maintains a financial license burden for each

application deployment. In the technical realm, the MATLAB library files are often large, relative to the embedded processing market. This roughly correlates to cost in hardware, memory, and power, increased memory, and increased latency, accessing through the memory to obtain the desired library function. Each of these factors, cost, power and latency, are undesired in a high performance, real-time application.

The coding of the actual algorithm will require porting to a mainline software programming language, such as C. The C programming language is a credible language selection based on widespread commercial tools and support in the PC, server and embedded hardware markets. Selecting a programming language with such a wide support base provides flexibility in algorithm deployment among dissimilar data centers whereby the Load Balancer function may be implemented in embedded hardware in one data center and a server in another. Recompiling the algorithm across these different hardware platforms will be easier with a well-supported and established programming language such as C. Additionally, the algorithm itself is primarily dependent upon basic math functions that are supported within the C math libraries. The higher-level matrix operations inherent to MATLAB and not found in C libraries can be reconstructed with common indexing programming methodologies.

Current and future hardware resources are capable of hosting a data center power management control application.

Note: stating that future hardware resources are capable of hosting the control algorithm is made to convey that the control algorithm possesses no identified sunset technology dependency.

1.1.3 Existing and Projected Future Control Authority Elements

Control authority elements for power management are the fastest growing component of this concept definition effort. Advanced Configuration and Power Interface (ACPI) specification is an early, operating system controlled, open standard for device configuration and power management, first introduced in 1996. One can effectively argue that ACPI was not the first power management implementation on personal computers; however, ACPI does appear to be the first attempt at a widespread open source computer power management topology. The standard started by Microsoft™, Intel™ and Toshiba™ and was soon joined by HP™ and Phoenix™ (a major BIOS developer at the time). This power management scheme was originally targeted for implementation on a single computer, providing multiple commandable hardware power state levels, CPU and other computer devices. Since that time, the concept of software control of hardware power states has proliferated into data center servers, smart building control systems, facility HVAC systems, and much more.

The implementation of widespread power management requires more than software. The proliferation of low power, low cost, highly capable embedded microprocessors has helped accelerate the extension of power management from the major power consumption devices to a much broader hardware set. These newer generations of microprocessors are enabling power management and control to permeate all aspects of modern society. Where there once was only

power management of the computer's CPU, hard drive and display, the bounds of a power managed system can be expanded to include server fans, building HVAC systems, building lighting, VoIP phone systems and other consumer appliances.

In order to effectively manage the research and development effort to implement a control algorithm, the system to be controlled must be bounded. Setting the system boundaries provides the first order of control element reduction. The approach utilized by the research team was to select a manageable system boundary containing the most significant data center power consumers and account for remaining power consumer elements in a future development cycle or planned product improvement. This approach allowed the team to focus on providing a solution to a manageable problem. The system boundary was set at the data center racks and servicing HVAC system. The data center rack control includes the server power state and chassis fans. The HVAC system control can be comprised of large HVAC systems external to the data center building, individual computer room air conditioning (CRAC) units, or some hybrid configuration depending on the data center HVAC configuration. The left column of Figure 6 provides a summary of control elements addressed in this iteration of the control algorithm implementation. The right column provides an expanded list displaying the next level of power consumer/control authority elements for future algorithm iterations.

Figure 6: Control Authority Elements

Project Control Authority Elements	Future Control Authority Elements
Server/blades (network, storage, processing)	Server/blades (network, storage, processing)
Chassis fans	Chassis fans
Rack	Rack
HVAC (includes CRAC unit(s))	HVAC (includes CRAC unit(s))
	Networking devices
	Data center lighting
	Office space lighting
	VoIP phone systems
	Office computers
	Office equipment (fax machines, projectors)

1.2 Identification of Critical Elements (SOPO Task 2.0)

The research team identified 3 critical elements of the RTOC algorithm requiring a feasibility demonstration in later Stage 3 effort. Key critical condition states and control authority elements are indentified and test cases are proposed to verify operating assumptions.

1.2.1 Blade-load Curve Verification

The first critical element requiring testing and verification is the server load curve shown in Figure 10. The same curve appears to the right of the y-axis in Figure 11. This curve displays blade power consumption (y-axis) versus blade loading (x-axis). This curve will be vendor dependent since no two manufactures servers consume the same amount of power. Vendors

such as Sun™, HP™ , Cisco™, and IBM™ will have data sheets with generalized curves. Experimentally verifying the information would be prudent. This verification will help ensure that the algorithm model tracks actual hardware performance. The blade power consumption versus blade loading curve would be constructed utilizing a test server (SUN, HP, Cisco, IBM, etc.). The selection of the particular server brand will be dependent on the test-case data center. First, the standard power supply rails to the server will be monitored over a period of time to determine the nominal voltage fluctuation during operations. Next, the server voltage will be regulated tightly at a fixed voltage within the operating parameter range of the server. Then, the server will be issued applications, starting at the idle but fully powered state. Applications will be distributed to the server at an increasing rate. A current measuring device will measure and record the server current consumption. This test will be repeated to verify consistent and repeatable results. Then, the server input voltage will be incremented to another acceptable voltage input level and the application-loading test repeated several times. This process will be repeated, incrementing the input voltage across the full span of acceptable input voltages, repeating the test multiple times for each selected voltage. There are three types of servers: processing, storage and network. These tests would be performed on several servers of each type to verify consistency. The test results will develop a representative server power versus load curve for use in the RTOC algorithm model.

1.2.2 Server State Transition Curve Verification

The second critical testing element is proving the server power state step transition, see the stair step curve representing various non-application executing server states on the left side of Figure 11. This step curve represents the server power consumption state prior to reaching the normal operating state, executing applications. Servers can be placed in a multitude of idle or standby states. Each non-application executing server state draws a differing amount of power and requires a different amount of time to transition to the next higher or lower readiness state. Understanding and accurately modeling these power consumption states (relative vertical height differentials) and transition times between states (slope of the curve steps) will ensure that the RTOC algorithm will be capable of maximizing power savings through powering down the maximum number of servers with the ability to restore servers fast enough to maintain quality of service metrics. Like the first test scenario, this test process will be repeated, incrementing the input voltage across the full span of acceptable input voltages, repeating the test multiple times for each selected voltage. All three types of servers (processing, storage and network) will be tested, multiple times. The test results will develop a representative server power versus non-application executing server states curve for use in the RTOC algorithm model.

1.2.3 Calculation Time & System Response

The final critical testing element for Stage 3 Concept Development is control loop cycle-time modeling. The results from the first two testing events identified above build and accurate representation of Figure 11 server power-load curve, for each of the server types (processing, storage, and networking). These curves are then incorporated into the overall data center system model; the system model is discussed in detail in section 1.3. The system model would then be

combined with the RTOC algorithm, test cases generated and control loop calculation time and system response modeled in MATLAB. The control loop calculation time, even though calculated in the MATLAB environment, will provide confidence in the RTOC algorithm viability. This development approach is consistent with those for previous control algorithm implementations, e.g. the Zero Propellant Maneuver for the ISS.

1.3 Translation of Routing and Telecon Data Center Behavior (SOPO Task 3.0)

This task was the primary effort of this research and development project. The research team utilized findings from Task 1.0 and Task 2.0 to translate Routing and Telecom Data Center system behavior into a preliminary control model, equation. This equation serves as the basis of the power management application and contains the critical state and control parameters. Initial system control model equations are normally formulated as an ordinary differential equation (ODE), family of ODEs, or, commonly, Differential Algebraic Equations (DAE). Once formulated, the system can be controlled.

In future research and development efforts, Stage 3, this equation would be integrated with the RTOC software algorithm and modeled, obtaining projected Data Center system efficiencies and allowing for iterative algorithm improvements to meet desired efficiency goals.

1.3.1 Model Development Overview

The following efforts identify the development, analysis and the final Stage 2 set of differential-algebraic equations (DAE) that model a generic Telecom or Data Center, also generically referred to within this report as a processing center. The DAE model is motivated by the possibility of applying optimal control theory for minimizing energy consumption of a processing center while maintaining some basic constraints such as quality of service.

The primary purpose of a processing center is to run applications, commonly referred to as “apps”. With the advent and rapid advance of Voice over Internet Protocol (Voice over IP, VoIP), traditional telephone hardware centers are rapidly changing to resemble data centers. The exception to this change is the final hardware that drives the data signals over the long haul wire distribution network. During this early Stage 2 Concept Definition effort of the control algorithm development, the research team sought to maximize the application of the control algorithm for power savings on the most prolific equipment; therefore, the team focused efforts on power savings for processor center hardware common to both telecom and data centers.

The number of racks contained within a processing center varies widely depending on site and location, even within the same company. A processing center is primarily composed of racks that run applications, additional discussion regarding applications occurs later. Each rack within the processing center can contain a varying number of chassis, which, in turn, host multiple servers or blade server computers. The modeling of the chassis to blade is important. Direct addressing servers without accounting for chassis to server correlation risks handicapping the processing center model’s ability to minimize power consumption due to intermediary fans tied

to individual chassis. Also of important note and account, as the computational processing density continues to increase, some racks are supplied with a chilled water supply. Figure 7 below displays a representative example of a cluster of processing racks, a rack with multiple blades and a representative rack with an optional chilled water heat exchanger for increased cooling capacity.

Figure 7: Representative Processing Center Racks, Blades and Chilled Water Cooling

Because the configuration of processing center racks, chassis, servers and support equipment vary so widely, any control model development for widespread use must be not only comprehensively descriptive for the elements intended to be controlled, but scalable as well. The processing center model development approach used in this research is both. For the purpose of this report discussion, the following configuration is assumed: 16 racks, each rack contains 7 chassis, and each chassis contains 8 blades. Ultimately, all apps run on blades. All other powered equipment (e.g. HVAC, fans, lights etc.) that is part of the processing center supports the blades.

1.3.2 Basic Indexing Model

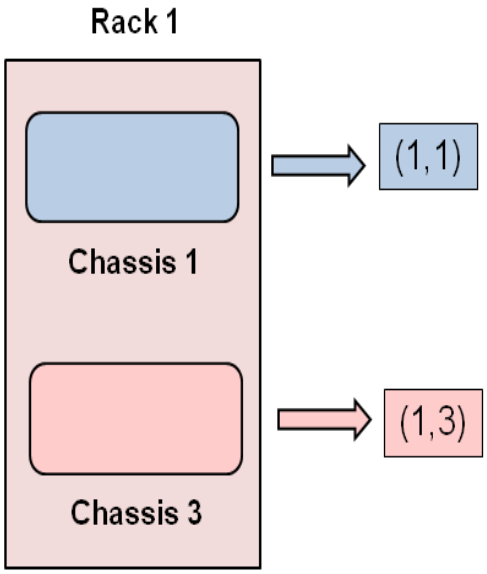
In order to construct generic models, we let Nr be the number of racks in the theoretical processing center. That is, even though Nr = 16, we do not limit the modeling to 16 racks. Similarly, we let Nc be the number of chassis in a rack, and Nb be the number of blades in a chassis. For our reference processing center example, those number are Nr = 16, Nc = 7, and Nb = 8. Furthermore, a rack has Nc ∗ Nb blades, and the reference processing center example has Nr ∗ Nc ∗ Nb blades.

Data Centers Energy Reduction and Management Through RTOC

Let r, c, and b, be the indices for a rack, chassis and blade respectively. Then, r runs from 1 to Nr; c runs from 1 to Nc; and b runs from 1 to Nb.

1. Rack and Chassis Location Scheme: Let each rack in the reference processing center example be numbered from 1 to Nr. Then a number, r, $1 \leq r \leq Nr$, identifies a specific rack. Let each chassis within a rack be numbered from 1 to Nc. Nc may depend upon r, but we avoid the notation Nc(r). Then, a chassis in the reference processing center example is located by a pair: (r, c).

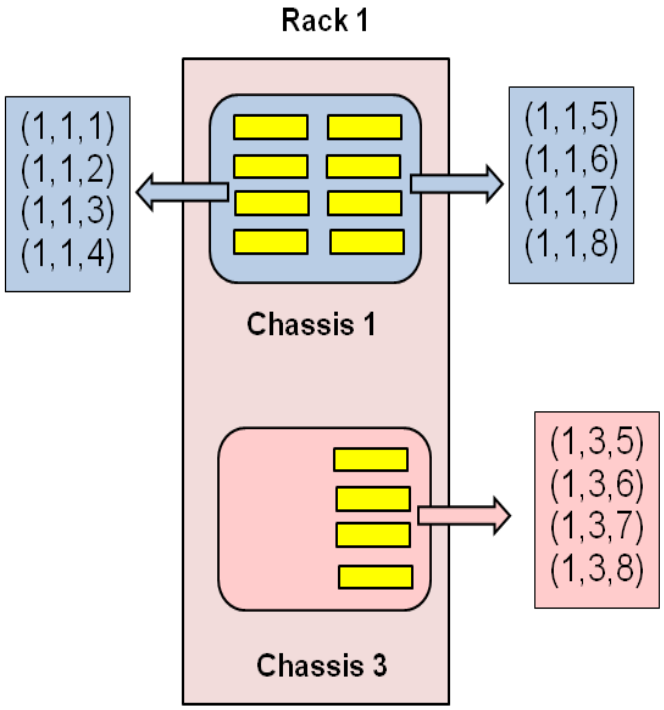
Figure 8: Numbering scheme for a rack and chassis



2. Blade Location Scheme: Within a chassis, a blade is numbered from 1 to Nb. The blades are numbered sequentially from top to bottom, and left to right. A blade can be identified by the triplet: (r, c, b).

Figure 9: Numbering scheme for blades

Data Centers Energy Reduction and Management Through RTOC



3. **Blade "Coloring" Scheme (Functionality):** A blade can be classified as a Processing blade, a Network blade, or a Storage blade. Let I_p be the set of all triplets (r, c, b) that identify a blade to be a processing blade. Then, any blade, identified by a triplet and associated with I_p is a processing blade. Similarly, we let I_n and I_s be the set of all triplets that identify a blade to be either a network blade or a storage blade respectively. Thus, any blade can now be located by a triplet and identified (by association) to be processing, networking or storage.

4. **Blade "Coloring" Scheme (Customer):** A blade can also be classified according to which customers are assigned. Thus, a blade can be designated to process apps from one particular customer, for example Bank of America, but not another customer, for example eBay. Let J_k be the set of all triplets (r, c, b) that identify a blade to serve customer k. Then, any blade, identified by a triplet and associated with J_k is a blade that processes only apps form customer k. Apps from customer k1 will not be allocated to any blades that belongs to customer k2, where $k1 \neq k2$. For example, apps from customer 1 will only be allocated to blades belong to set J_1.

1.3.3 Power State of a Server

The power state of the blade can be defined in terms of discrete and continuous values. The discrete values are identified as P0-P4 (corresponding to different sleep stages) and the continuous state is described as P5. It will be apparent shortly that the mathematical state of a blade is more than just the power state of the blade.

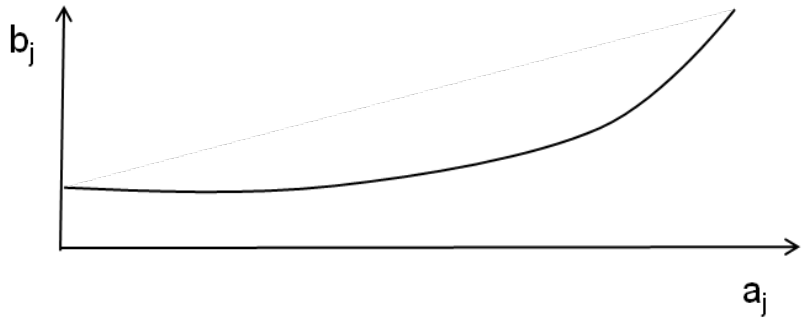
Data Centers Energy Reduction and Management Through RTOC

Let j be a generic triplet, (r, c, b). That is, j is a shorthand notation for the triplet (r, c, b) that identifies a specific blade. Let $b_j(t)$ be the power state of the generic blade j at any given time, t. Then, by definition, $b_j(t)$ can take five discrete values, 0 – 4, corresponding to P0 – P4, and any value in the continuous range [5,max].

The decision to put a blade in any of the discrete states is currently performed in some ad hoc manner. The factor largely affecting this decision is maintaining a high quality of service. To maintain a high quality of service (i.e. timeliness of processing an app), the blade needs to be in state P5. When the blade is in state P0-P4, it takes time to power up to achieve state P5 when it is ready to accept apps for processing. When a blade is in state P5, its "value" is determined autonomously based on the number of apps it is processing. Thus, the number of apps processed by the blade "controls" the power state of the blade in the P5 region. Thus, the power state of the blade is indirectly controlled by the apps in the P5 region and "directly" controlled in the P0-P4 region.

The P5 state of the blade depends upon the number of apps running on the blade. When the blade is in the P5 state, it is possible to obtain a server-load curve, generically represented, in Figure 10. The variable a_j is the app running on the server. Ideally, the curve $L_5(a_j)$ must be obtained from the server manufacturer, experimentally verified, or through experimental data.

Figure 10: Server-load curve, $b_j = L_5(a_j)$.

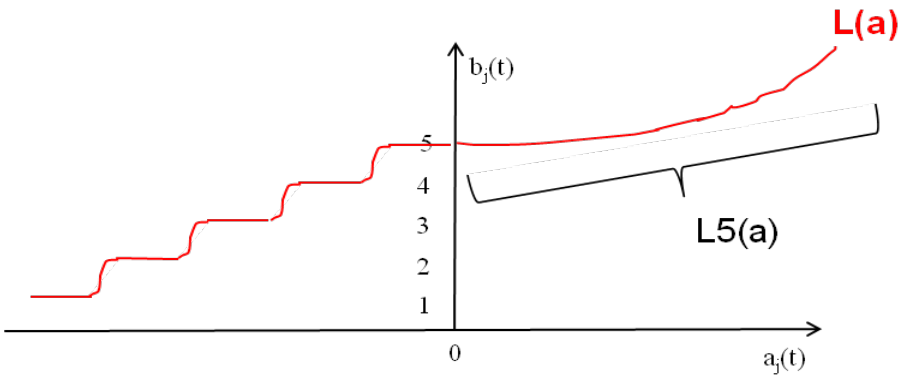


1.3.4 Generalized Server-Load Concept

In order to perform unified mathematical operations over the entire spectrum, we propose to extend the L5 curve to negative apps to form a generalized server-load curve as in Figure 11.

Figure 11: Generalized blade-load curve, $b_j = L(a_j)$

Data Centers Energy Reduction and Management Through RTOC



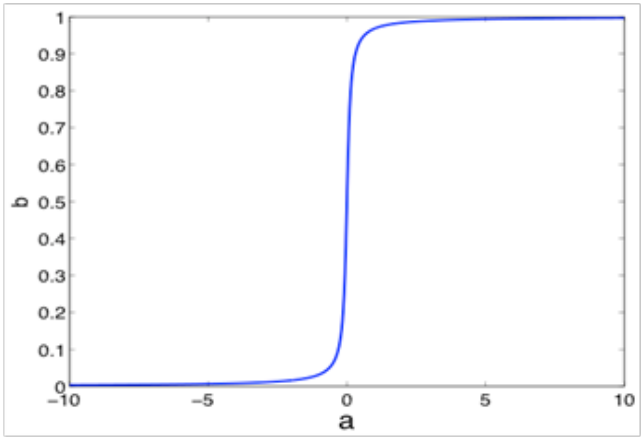
If negative apps are running on a server, it takes on the discrete states, P0-P4. The concept of negative apps allows the extension of the P5 notion of using apps to autonomously control the server state.

The server-load curve b=L(a) is made smooth so that the partial derivative of L(a) with respect to a(t) can be performed. To this end, we will use the following function as a building block to smooth the discrete part of the states.

$$b = L(a) = \frac{1}{\pi}\tan^{-1}(c_1 a) + 0.5$$

In this function, constant C1 is used to control how smooth the function is; when C1 = 10, the shape of the smoother function is shown in Figure 12.

Figure 12: Illustrating smoother function used as a building block in the negative app region



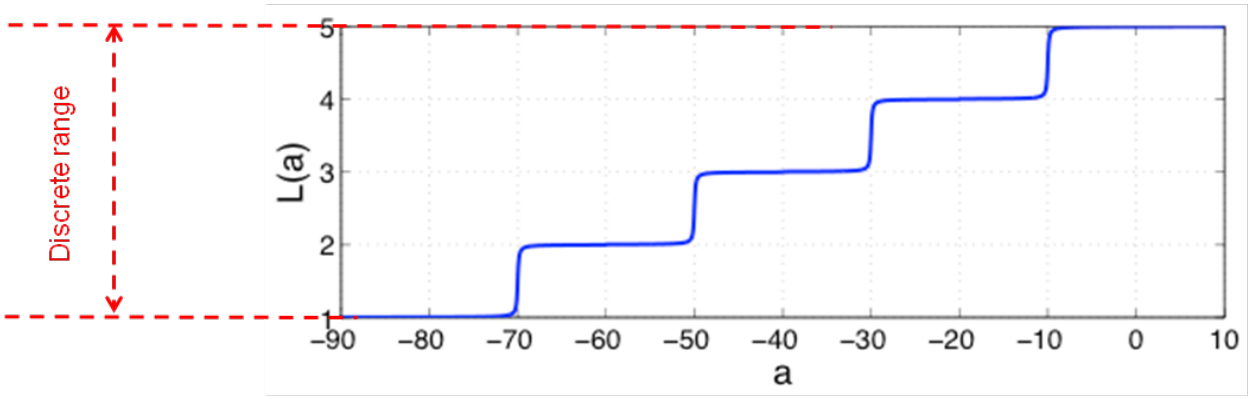
Based on the smoother function, we model the discrete blade-load curve as (by choosing c1 = 10):

Data Centers Energy Reduction and Management Through RTOC

$$L(a) = \begin{cases} \frac{1}{\pi}\tan^{-1}(10(a+70)) + 1.5, & \text{if } -90 \leq a \leq -60 \\ \frac{1}{\pi}\tan^{-1}(10(a+50)) + 2.5, & \text{if } -60 \leq a \leq -40 \\ \frac{1}{\pi}\tan^{-1}(10(a+30)) + 3.5, & \text{if } -40 \leq a \leq -20 \\ \frac{1}{\pi}\tan^{-1}(10a+10) + 4.5, & \text{if } -20 \leq a \leq 10 \end{cases}$$

The shape of this function is shown in the Figure 13.

Figure 13: Illustrating negative apps to control the discrete power state of a blade



Thus, by extending the L5 curve to negative apps, we can construct a generalized blade-load curve, where L5 is obtained from experimental data. Thus, the dynamics of a generic blade can now be modeled as:

$$\dot{b}_j = \partial_a L(a_j)\dot{a}_j$$

1.3.5 Controlling the Blade Dynamics

It is clear from the blade dynamics equation that the blade is now controlled by the apps (real and fake) running on it. By controlling the apps running on the blade, we control the power blade state. We now discuss techniques to control the real and fake apps running on a blade.

Consider the equation,

$$\dot{a}_j(t) = -c_2 \cdot a_j(t) \cdot s$$

where c_2 is a positive constant and s is a decision variable that takes values 0 or 1. When s = 0, then from the above equation and blade dynamics, we have the b_j = constant (and can hence be in any of the P0-P4 states). When a_j is negative (fake) and s = 1, then the derivate of a_j is positive; thus b_j will gracefully jump as previously illustrated before. Thus, the decision variable, s, can

be used to model the control and the "latency" can be modeled by a proper choice of the constant c_2.

1.3.5.1 The Dynamics of Real Apps (without Incoming Apps)

The apps in the blade are processed or "killed". When there are no incoming apps, all apps are eventually processed. Hence, we control real apps by controlling the incoming apps (i.e. the allocation to a specific blade).

Let ($a_p(t)$, $a_n(t)$, $a_s(t)$) be generic processing, networking and storage apps running on a generic blade. The generic kill dynamics may be written as:

$$\begin{cases} \dot{a}_p(t) &= k_p(a_p(t), b) \\ \dot{a}_n(t) &= k_n(a_n(t), b) \\ \dot{a}_s(t) &= k_s(a_s(t), b) \end{cases}$$

where dot denotes the derivative and k(.) are functions to be modeled. The choice of k(.) needs to ensure

$$\lim_{t\to\infty} (a_p(t),\ a_n(t),\ a_s(t)) \ = \ 0$$

so that all apps will eventually be processed. As an example, we can choose the generic kill dynamics to be

$$\begin{cases} k_p &= c_{3p} \cdot b \cdot (a_p(t))^{c_{4p}} \\ k_n &= c_{3n} \cdot b \cdot (a_n(t))^{c_{4n}} \\ k_s &= c_{3s} \cdot b \cdot (a_s(t))^{c_{4s}} \end{cases}$$

where (c_{3p}, c_{3n}, c_{3s}) are positive constants and (c_{4p}, c_{4n}, c_{4s}) are positive odd numbers. Based on Lyapunov stability theorem, such choice of k(.) can guarantee

$$\lim_{t\to\infty} (a_p(t),\ a_n(t),\ a_s(t)) \ = \ 0$$

The constants (c_{3p}, c_{3n}, c_{3s}) and (c_{4p}, c_{4n}, c_{4s}) determine the rate of apps be processed.
The exact value of these constants can be determined from experiments using the following method:

1. Write down the analytic solution of the kill dynamics with C_3 and C_4 as unknowns.
2. Given some typical data, we can find the unknown constants C_3 and C_4 through curve fitting.

In the simple case, (c4p, c4n, c4s) can all be chosen as 1 resulting in a linear dynamical system. Here, we do not make this assumption to allow for possible nonlinear effects. Based on given typical data, it is also possible to modify k(.) to incorporate further complexities in the apps' dynamical behavior.

1.3.5.2 Incoming Apps - Facts and Assumptions

A processing center accepts a given number of apps, A(t), at time t. Technically A(t) is an integer; however, for the purposes of modeling we will treat it as a real, nonnegative number. The justification for this assumption is based on A(t) being a "large" number. A typical number for A(t) is 40,000. The allocation of A(t) to the blades is a decision variable, commonly accomplished in modern processing centers by a "Load Balancer", a function normally implemented with an embedded hardware and software device.

The variable A(t) belongs to K number of customers. Let $[A(t)]_i$, i=1,2,…,k be the apps from customer i. Then

$$A(t) = [A(t)]_1 + [A(t)]_2 + \cdots + [A(t)]_K$$

where $[A(t)]_k$ is a vector of "processing", "network" and "storage". Let $[A_p]_k$, $[A_n]_k$ and $[A_s]_k$ be the number of incoming processing apps, network apps and storage apps from customer k. Then,

$$[A(t)]_k = [A_p]_k + [A_n]_k + [A_s]_k$$

Let A_p, A_n and A_s be the total number of incoming processing apps, network apps and storage apps respectively, i.e.,

$$\begin{aligned} A_p &= [A_p]_1 + [A_p]_2 + \cdots + [A_p]_K \\ A_n &= [A_n]_1 + [A_n]_2 + \cdots + [A_n]_K \\ A_s &= [A_s]_1 + [A_s]_2 + \cdots + [A_s]_K \end{aligned}$$

Thus, it follows that,

$$\begin{aligned} A(t) &= [A(t)]_1 + [A(t)]_2 + \cdots + [A(t)]_K \\ &= A_p(t) + A_n(t) + A_s(t) \end{aligned}$$

Each color of A(t) contains processing, networking and storage components with different percentage. For example, a typical processing app may contain 90% of processing part, 5% of networking and 5% of storage.

Table 1: Application (app) to Customer Modeling Construct

Data Centers Energy Reduction and Management Through RTOC

	$[A_p]_k$	$[A_n]_k$	$[A_s]_k$
Processing Part	$[\alpha_{11}]_k$	$[\alpha_{12}]_k$	$[\alpha_{13}]_k$
Networking Part	$[\alpha_{21}]_k$	$[\alpha_{22}]_k$	$[\alpha_{23}]_k$
Storage Part	$[\alpha_{31}]_k$	$[\alpha_{32}]_k$	$[\alpha_{33}]_k$

A typical set of numbers for all three types of apps is:

$$\begin{pmatrix} [\alpha_{11}]_k & [\alpha_{12}]_k & [\alpha_{13}]_k \\ [\alpha_{21}]_k & [\alpha_{22}]_k & [\alpha_{23}]_k \\ [\alpha_{31}]_k & [\alpha_{32}]_k & [\alpha_{33}]_k \end{pmatrix} = \begin{pmatrix} 0.9 & 1/3 & 0.2 \\ 0.05 & 1/3 & 0.1 \\ 0.05 & 1/3 & 0.7 \end{pmatrix}$$

These numbers satisfy

$$\sum_{i=1}^{3}[\alpha_{i,1}]_k = \sum_{i=1}^{3}[\alpha_{i,2}]_k = \sum_{i=1}^{3}[\alpha_{i,3}]_k = 1$$

Let $[a_p^{in}(t)]_k$, $[a_n^{in}(t)]_k$ and $[a_s^{in}(t)]_k$ be the total amount of processing, networking and storage components of all incoming apps from customer k. Then,

$$\begin{aligned} [a_p^{in}(t)]_k &= [\alpha_{11}]_k \cdot [A_p]_k + [\alpha_{12}]_k \cdot [A_n]_k + [\alpha_{13}]_k \cdot [A_s]_k \\ [a_n^{in}(t)]_k &= [\alpha_{21}]_k \cdot [A_p]_k + [\alpha_{22}]_k \cdot [A_n]_k + [\alpha_{23}]_k \cdot [A_s]_k \\ [a_s^{in}(t)]_k &= [\alpha_{31}]_k \cdot [A_p]_k + [\alpha_{32}]_k \cdot [A_n]_k + [\alpha_{33}]_k \cdot [A_s]_k \end{aligned}$$

From the relation

$$\sum_{i=1}^{3}[\alpha_{i,1}]_k = \sum_{i=1}^{3}[\alpha_{i,2}]_k = \sum_{i=1}^{3}[\alpha_{i,3}]_k = 1$$

it can be proved that

$$[A(t)]_k = [A_p]_k + [A_n]_k + [A_s]_k = [a_p^{in}(t)]_k + [a_n^{in}(t)]_k + [a_s^{in}(t)]_k$$

Data Centers Energy Reduction and Management Through RTOC

The quantities $[a_p^{in}(t)]_k$, $[a_n^{in}(t)]_k$ and $[a_s^{in}(t)]_k$ need to be relocated to processing, networking and storage blades respectively. Note that they are constrained to blades designated to customer k.

Let

$$\left(a_{p,j}^{in}(t),\ a_{n,j}^{in}(t),\ a_{s,j}^{in}(t)\right)$$

be the total incoming processing, network and storage components of the apps on blade j. Note that a processing blade may still take some networking and storage apps. They are the decision variables that need to be determined.

Based on these facts/assumptions, we have the following equations

$$\begin{aligned}
\left[a_p^{in}(t)\right]_k &= \sum_{j\in J_k} a_{p,j}^{in}(t) \\
\left[a_n^{in}(t)\right]_k &= \sum_{j\in J_k} a_{n,j}^{in}(t) \\
\left[a_s^{in}(t)\right]_k &= \sum_{j\in J_k} a_{s,j}^{in}(t)
\end{aligned}$$

where J_k is the set of all triplets (r, c, b) that identify a blade designated to customer k

1.3.5.3 The Apps Allocation Equation

Let a_j^{in} be the total incoming apps components on blade j. Then,

$$a_j^{in}(t) = a_{p,j}^{in}(t) + a_{n,j}^{in}(t) + a_{s,j}^{in}(t)$$

Combining this equation with the equations previously introduced, we have

Total number of apps from customer k $= [A(t)]_k = \sum_{j\in J_k} a_j^{in}(t)$

$A(t)$ Incoming apps from all customers $= \sum_{k=1}^{K}[A(t)]_k = \sum_{k=1}^{K}\sum_{j\in J_k} a_j^{in}(t) = \sum_{j=(1,1,1)}^{N_r,N_c,N_b} a_j^{in}(t)$

Data Centers Energy Reduction and Management Through RTOC

All equations identified previously serve as constraints on the allocation of incoming apps. They are part of the constraints in the final formulation of optimal control problem. With incoming apps, the dynamics of apps and blades will be influenced by the decision variables

$$\left(a^{in}_{p,j}(t),\ a^{in}_{n,j}(t),\ a^{in}_{s,j}(t)\right)$$

1.3.5.4 Blade Dynamics with Incoming Apps

The apps dynamics is itself given by,

$$\dot{a}_j(t) = \begin{cases} k_p(a_{p,j}(t), b_j) + k_n(a_{n,j}(t), b_j) + k_s(a_{s,j}(t), b_j) & \\ \quad +a^{in}_{p,j}(t) + a^{in}_{n,j}(t) + a^{in}_{s,j}(t), & \text{if } a_j(t) \geq 0 \\ l(a_j(t), s), & \text{if } a_j(t) \leq 0 \end{cases}$$

From the blade dynamics given by,

$$\dot{b} = \partial_a L(a(t))\dot{a}$$

we can write the full equation by substituting the apps dynamics in the blade dynamics; thus, we have,

$$\begin{aligned} \dot{b}_j(t) &= \begin{cases} \partial_{a_j} L(a_j) \cdot [k_p(a_{p,j}(t), b_j) + k_n(a_{n,j}(t), b_j) + k_s(a_{s,j}(t), b_j) & \\ \quad +a^{in}_{p,j}(t) + a^{in}_{n,j}(t) + a^{in}_{s,j}(t)], & \text{if } a_j(t) \geq 0 \\ \partial_{a_j} L(a_j) \cdot [l(a_j(t), s)], & \text{if } a_j(t) \leq 0 \end{cases} \\ a_j &= a_{p,j} + a_{n,j} + a_{s,j} \\ j &= (1,1,1),\ \cdots,\ (N_r, N_c, N_b) \end{aligned}$$

From an optimal control perspective, the blade dynamics yields the "insight" that the state, x, of the blade is not merely b but the "vector" [b, a]. Vector b is of dimension Nr × Nc × Nb corresponding to the number of the blades in a container. Vector, a, is of the same dimension as b. The incoming apps on blades,

$$\left(a^{in}_{p,j}(t),\ a^{in}_{n,j}(t),\ a^{in}_{s,j}(t)\right)$$

and s are decision variables to be determined through an application of optimal control theory and computation. Thus, we have a differential-algebraic model for the optimal control system.

Data Centers Energy Reduction and Management Through RTOC

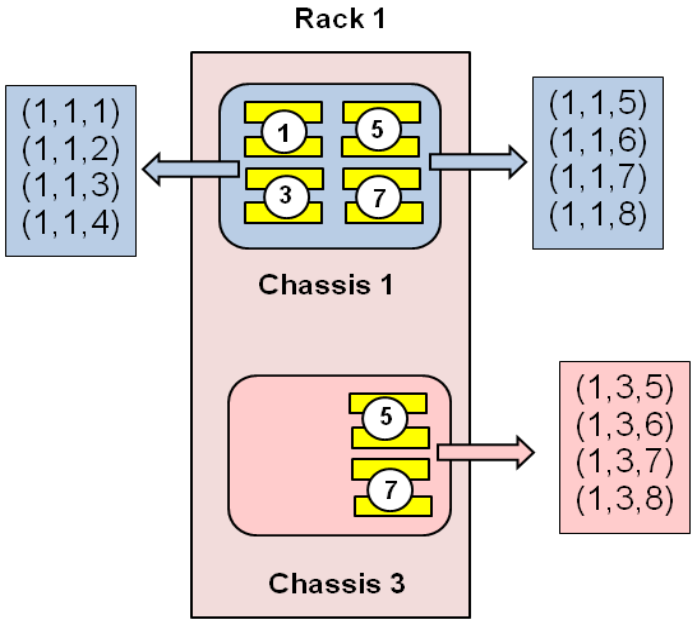
1.3.6 Equations for the HVAC

The HVAC system in a processing center has significantly slower dynamic response than the blades. While it is possible to include the dynamics of the HVAC into the system differential equation model, this incorporation does not generate a significant gain; hence, it is preferential to model it as part of the algebraic component of the DAE previously developed.

The main feature of constructing the algebraic model for the HVAC is the number and location of the components that constitute the HVAC. As these depend upon the specific design of the data center, we focus here on the rack and chassis fan to illustrate the concept.

The "dynamics" of the fan is modeled via decision variables. To perform this decision, a number scheme similar to the servers is adopted, see Figure 14.

Figure 14: Illustrating numbering scheme for the chassis fans



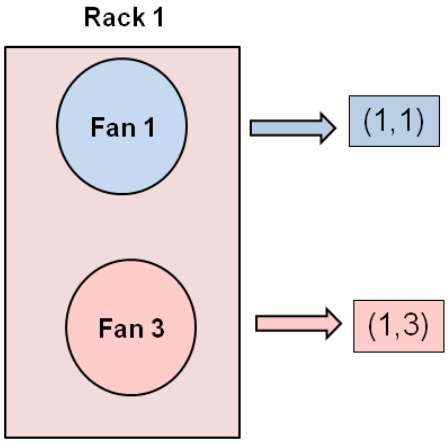
1. **Fan Numbering Scheme:** Because chassis fans are paired with the blades, they are numbered 1, 3, 5, 7 etc. For examples, (1, 1, 1) for a fan corresponds to fan # 1 in chassis # 1 on rack # 1; (1, 1, 3) implies fan # 3 in chassis # 1 on rack # 1; (1, 3, 5) implies fan # 5 in chassis # 3 on rack # 1. A chassis fan is thus identified by the triplet, (r, c, f). Thus, we can define a fan index set of triples. Denote this by CFI for Chassis Fan Index.

2. **Fan Decision Variables:** As fan 1 corresponds to the blade pair 1 and 2, it is clear that if the power states of blades 1 and 2 are zero, fan 1 can be turned off. Thus, for a given data center, a matrix of decision variables can be constructed based on the numbering

scheme. These decision variables will depend upon how the blades and fans are paired up. For our example, this corresponds to the simple logic: if $b_{(r, c, b)} + b_{(r, c, b+1)} = 0$, then $f_{(r, c, b)} = 0$, if (r, c, b) is an index set of CFI. Thus $f_{(r, c, b)}$ (zero or nonzero) implies if the corresponding fan is turned off or on. Note that additional layers of granularity can be easily added to this model by pairing up the state of the fan with the states of the blades (provided the fan has different states beyond an on and off state). For instance, a fan can be turned on to run with less power depending upon the power state of the blade and the design of "states" of the fans.

3. **Rack Fan Numbering and Decisions:** The rack fans are numbered similar to the chassis. As there are 7 chassis to a rack, a conservative decision is to turn off Fan # 1 if all the blades (and hence the corresponding chassis fans) in chassis 1, 2, and 3 are off. Similarly, for Fan # 2 to be off, chassis, 3, 4 and 5 are required to be off. Fan # 3 is off if chassis 5, 6 and 7 are off.

Figure 15: Illustrating number scheme for rack fans



1.3.7 Model Development Summary

The state-space dynamics of a processing center is given by:

$$\dot{a}_j(t) = \begin{cases} k_p(a_{p,j}(t), b_j) + k_n(a_{n,j}(t), b_j) + k_s(a_{s,j}(t), b_j) \\ \quad +a^{in}_{p,j}(t) + a^{in}_{n,j}(t) + a^{in}_{s,j}(t), & \text{if } a_j(t) \geq 0 \\ l(a_j(t), s), & \text{if } a_j(t) \leq 0 \end{cases}$$

Data Centers Energy Reduction and Management Through RTOC

$$
\begin{aligned}
\dot{b}_j(t) &= \begin{cases} \partial_{a_j} L(a_j) \cdot [k_p(a_{p,j}(t), b_j) + k_n(a_{n,j}(t), b_j) + k_s(a_{s,j}(t), b_j) \\ \qquad +a^{in}_{p,j}(t) + a^{in}_{n,j}(t) + a^{in}_{s,j}(t)], \qquad \text{if } a_j(t) \geq 0 \\ \partial_{a_j} L(a_j) \cdot [l(a_j(t), s)], \qquad \text{if } a_j(t) \leq 0 \end{cases} \\
a_j &= a_{p,j} + a_{n,j} + a_{s,j} \\
j &= (1,1,1), \ \cdots, \ (N_r, N_c, N_b)
\end{aligned}
$$

By itself, the state-space dynamics are incomplete; they require the following algebraic equations that constrain the state space:

$$
\begin{aligned}
\left[a^{in}_p(t)\right]_k &= [\alpha_{11}]_k \cdot [A_p]_k + [\alpha_{12}]_k \cdot [A_n]_k + [\alpha_{13}]_k \cdot [A_s]_k \\
\left[a^{in}_n(t)\right]_k &= [\alpha_{21}]_k \cdot [A_p]_k + [\alpha_{22}]_k \cdot [A_n]_k + [\alpha_{23}]_k \cdot [A_s]_k \\
\left[a^{in}_s(t)\right]_k &= [\alpha_{31}]_k \cdot [A_p]_k + [\alpha_{32}]_k \cdot [A_n]_k + [\alpha_{33}]_k \cdot [A_s]_k
\end{aligned}
$$

$$
[A(t)]_k = [A_p]_k + [A_n]_k + [A_s]_k = [a^{in}_p(t)]_k + [a^{in}_n(t)]_k + [a^{in}_s(t)]_k
$$

These equations constitute the differential-algebraic equations (DAEs) for a state-space model of the processing center.

The algebraic part of the DAE is further augmented with additional algebraic equations that connect the power state of the blades to the chassis fan via a decision matrix. The decision matrix is specific to a specific construction of the processing center. A similar decision matrix was augmented for the rack fans. Additional decision matrices can be added to specific HVACs. The collection of all these equations constitutes a system DAE for a generic processing center.

1.4 Mitigate Market Introduction & Technical Risks (SOPO Task 4.0)

The research team mitigated market introduction risks by identifying existing, or projected future, processing center product lines capable of hosting an RTOC algorithm. The team formulated and outlined a test scenario for a future, Stage 3, program phase implementation in Section 1.2. The testing and implementation path leverage commercial market products, e.g. IBM, HP or SUN servers. The following sections identify a potential path to incorporate the RTOC algorithm into the data center with minimal data center impact while using commercial products.

1.4.1 Host Data Center Product Lines

The introduction of a data center power management RTOC algorithm is not as disrupting to the data center market due to the well-established concept and implementation of “Load Balancing”

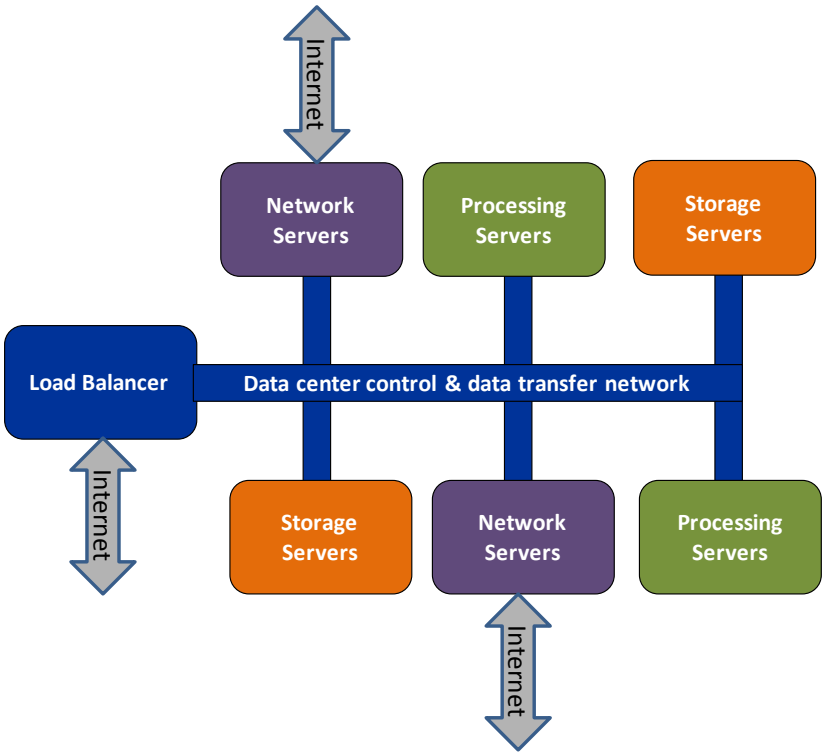
or Application Delivery Controllers (ADC). Load Balancing is implemented in many different schemes, but the underlying concept remains the same, efficient utilization of multiple computers and traditionally focuses on optimizing CPU cycles and bandwidth. Data center ADC control is the next generation approach, moving beyond the Load balancing CPU cycles and bandwidth and focuses on the availability of the host applications. With rare exception, every data center possesses a functional device, hardware and software that controls and assigns incoming work assignments, tasks, and jobs to the available cluster of computers, regardless of Load Balancing or ADC implementation. The data center power management RTOC algorithm presented in this research effort introduces the next generation beyond ADC by integrating the work assignment, customer application quality of service (QoS) and, now, macro level power management functions.

The term Load Balancer is used generically, referring to traditional Load Balancing, ADC and the RTOC power management algorithm interchangeably, throughout the remainder of this report. This utilization, while not technically accurate in a purist's viewpoint, is not out of line with or uncommon in the community's technical writings, especially one at this overarching concept definition level. However, to fairly represent and appreciate the differences between these terms prior to generic use, a brief discussion on the Open Systems Interconnection model (OSI model), Load Balancing and ADC is warranted. The International Organization for Standardization developed the OSI model. The model divides the communication system into smaller sections, called layers. The OSI model contains 7 layers, from physical (Layer 1) to application (Layer 7) layers. Each "Layer" provides service to the layer above or receives service from the layer below. Load Balancing was an early method of data center optimization and focused on CPU cycles and bandwidth, as mentioned above, thus operating at the Transport Layer (Layer 4). ADC is a newer, expanded method of Load Balancing or "Equalizing" that moves well beyond the Transport Layer and accounts for the host applications, customer QoS, etc, operating at the Application Layer (Layer 7). The overall implementation, whether true Load Balancing or ADC, is commonly referred to as Load Balancing, even by companies that provide ADC data center products. Therefore, remaining consistent with this approach and in an effort to provide a readable document with laborious qualification statements for every nuance, the remainder of this report adopts the common practice of utilizing the generic term Load Balancing. Should an important differentiation be required, it will be clearly stated.

Figure 16 depicts a simplified functional representation of a load balancer within a data center. All incoming applications enter the data center through network/internet connections. Each application entering the data center is comprised of a processing, networking and storage component; see section 1.3.5. The load balancer controls the loading of each type of sever (processing, networking & storage) such that the minimal number of servers are online and the appropriate number of servers are off or in varying standby states, ready to increase its readiness state with a surge of working entering the data center. The data center power management RTOC algorithm assigns applications to the computational hardware in such a manner as to minimize the number of operating servers and, in turn, fans and HVAC services. The RTOC power management algorithm would be hosted within the load balancer function.

Data Centers Energy Reduction and Management Through RTOC

Figure 16: Simplified Data Center Network Highlighting Load Balance Function



The load balancer function may be hosted within the data center's native servers or dedicated computer platform (independent hardware and software). Figure 17 below is one such dedicated platform offering from the commercial load balance market.

Figure 17: Barracuda Load Balancer

Photo courtesy of Barracuda

Data Centers Energy Reduction and Management Through RTOC

As long as the devices to be controlled (servers, fans, HVAC) possess a network layer connection, the introduction of a data center power management RTOC algorithm becomes primarily a software integration effort into a Load Balancer.

1.4.2 Market Introduction Mitigation

The research performed in this early Concept Definition (Stage 2) resulted in a clear, low-risk market introduction path. The data center power management RTOC algorithm is a natural progression of the data center Load Balancer evolution. There are two primary reasons for this claim: market need and existing community utilization.

First, dramatic data center power consumption is trending toward unsustainable levels, generating a market changing need. Our society, business and consumers, is becoming increasingly dependent on information technology systems, driving exponential growth in demand for data center processing and an insatiable appetite for energy. Utilizing the well-established data center Load Balancer concept to assign incoming applications in real time to the least capable and power consuming hardware devices while concurrently minimizing the number of servers in operation to meet QoS requirements will significantly reduce the disturbing power consumption trend due to data centers.

Secondly, the RTOC power management algorithm utilizes existing community practices, hardware and software. The data center power management approach advocated in this research is based upon managing the assignment of applications, or tasks, across available server clusters within the data center. This approach is consistent with the function of Load Balancers, distributing workload. Load Balancing is a commonplace and mature function within data centers, mitigating many market introduction risks for data center power management RTOC algorithm. The power management algorithm can be introduced as a software pre-planned product improvement deployment to an existing data center Load Balancer or as a new hardware and software combination (a.k.a. “appliance”) during a data center recapitalization period. Cisco research development team members concurred that the logical implementation and least market introduction risk of the RTOC power management algorithm is within the Load Balancer function.

The RTOC power management algorithm is capable of running on existing commercial hardware, software, supports accommodating data center customer QoS requirements, all within an existing data center Load Balancer function, thus mitigating market introduction. The state of practice computational hardware exists to deploy the algorithm. As discussed earlier in this report, the existing RTOC algorithm, non- data center power management applications, has been operated on a single Intel Pentium IV™ 1.8 GHz desk-top PC CPU. The key computational performance component of the processor is the double-precision floating-point unit (FPU) and CPU high-speed input-output (I/O). The Data center power management algorithm can be hosted within either data center Load Balancer implementation: software Load Balancer utilizing the data centers own server or a dedicated Load Balancer appliance. Current data center servers are several generations beyond the Intel Pentium IV™, e.g. Intel Xeon™ or AMD Opteron™

server processors. These newer generation CPUs contain a FPU with significantly improved performance. Dedicated Load Balancer appliances often, but not always, utilize commercial Intel, AMD, Texas Instruments™, etc. microprocessors. These dedicated appliances are advertised to perform complex and high-speed data encryption and compression. Leveraging these commercial processor products and high-end functionality suggests that the highly proprietary Load Balancer designs could contain a processor possessing double-precision floating-point capability. If not, including a double-precision floating-point capable processor in a future, but near-term Load Balancer design is possible. The commercial market contains a multitude of affordable and capable double-precision floating-point processors.

The RTOC power management algorithm is capable of being translated and complied into common data center software targets. The current RTOC algorithm is hosted within MATLAB™. The functional code is written using low-level math functions, for which there are multiple library support packages and programming languages. The data center model developed during this research phase currently exists as a math model; however, it would be programmed and integrated with the RTOC algorithm, forming a complete RTOC power management application. The research team's current intention is to program the RTOC power management application (integrated RTOC algorithm and data center model) in the C programming language, utilizing minimal or tailored mathematical libraries, in order to facilitate deterministic, real-time implementation in either data center servers or dedicated Load Balancer appliance. In the 2011 TIOBE Programming Community index, an indicator of programming language popularity, the C programming language is the second most popular programming language. C has dominated the top position for many years, finally losing to Java in 2011. Java's library support does not lend itself as readily to complex mathematical operations as does C. Regardless, the dominance of the C programming language for many years has brought with it widespread and sustainable software tool support.

The RTOC power management application supports accommodating data center customer QoS requirements. Any power management scheme or application will fail if it does not account for or support customer QoS requirements. Meeting customer QoS requirements drives a data center's business profitability; therefore, it was imperative that the RTOC power management application accounts for maintaining customer QoS requirements, and it does. RTOC algorithm contains an inherent cost function whereby the control characteristics are governed by user-defined parameters. In this RTOC power management application; these parameters will include the customer QoS requirements. The cost function definition and programming naturally occurs in the program development phase, Stage 3.

1.5 Project Management and Reporting (SOPO Task 5.0)

BAE Systems provided all required reports and other specified deliverables in accordance with the Federal Assistance Reporting Checklist (DOE F 4600.2), following the instructions included therein. These deliverables address Federal and American Recovery Reinvestment Act (ARRA) requirements, as well as ITP programmatic requirements including but not limited to program status reports and financial status reports.

Data Centers Energy Reduction and Management Through RTOC

The research and development team attended and presented results at the yearly project review meeting held March 17-18, 2011 in San Francisco, CA. The Principal Investigator presented results of tasks completed along with a project overview to the DOE project team and other attendees.

1.6 Impact Projections Model

The Impact Projections Model provided by the DOE was utilized to predict the theoretical monetary and energy savings possible through the implementation of the RTOC approach. Assuming that the growth rate of demand increases by 5% per year and a 100% addressable market (with 60% likely market penetration), the model predicts large monetary and energy savings with a corresponding reduction in pollutants. By 2040, with full roll out, the modeled impact is

- Total primary energy displaced: 259.2 trillion Btu
- Direct electricity displaced: 32.85 billion kWh
- CO displaced: 5399.13 metric tonnes
- SO2 displaced: 67987.28 metric tonnes
- NOx displaced: 44637.93 metric tonnes
- Overall savings: approximately $2B
- Overall reduction in server energy consumption: >50%

While achieving these projected levels of savings must still be demonstrated and actual savings will likely be less than predictions, the potential is significant and capturing even half the predicted savings would be significant. Further, the effort required to prototype the system is less expensive and faster to implement than other energy savings approaches that rely on large capital investments or long implementation schedules, making this project a desirable candidate for further investigation.

Data Centers Energy Reduction and Management Through RTOC

Conclusions

The research and development team for the RTOC power management application project has met the Stage 2 Concept Definition goals identified in the program proposal and is qualified to move the technology development forward to Stage 3 Concept Development. The team researched and documented the control algorithm hardware requirements, meeting technical requirements. The team translated end user data center QoS needs into preliminary model and algorithm technical specifications. Critical elements of the model and algorithm requiring feasibility demonstration were identified. The team investigated and understands commercial market and technical risks involved in deploying an RTOC power management application. The team, along with community experts consulted during this project, understands the potential energy savings, environmental emissions reductions and economic advantages in efficient data center operations this RTOC power management application concept provides.

All Statement of Project Objectives (SOPO) outlined in the Concept Definition (ITP Stage-Gate 2) proposal were achieved:

Task 1.0: Technical Requirements - Successful
The research team investigated, explored and documented:

1) Current and projected future computer resource requirements to host an RTOC algorithm in a Data Center environment. *The current generation of microelectronic processors is capable of meeting the algorithm's floating-point computational requirements (**See section 1.1.1**).*
2) Current and projected future hardware requirements to implement an RTOC algorithm in Routing and Telecom Data Centers. *The RTOC power management application supports either data center Load Balancer implementation: software or dedicated appliance. In the case of software Load Balancer implementation, with the exception of the Intel Atom™ based servers, current generation of data center servers are capable of hosting the RTOC power management application. In the case of a completely Intel Atom™ based data center, a dedicated Load Balancer appliance is anticipated to be required due to the Atom's weak floating-point performance. In the case of a dedicated Load Balancer appliance, the vendor appliance selection will need to account for an embedded processor with a floating-point unit performance equal to or better than Intel Pentium 4 1.80 GHz. This processor was built circa 2000. All Intel or AMD server or mid-level PC processors purchased today easily meet this floating-point unit performance (**See section 1.1.2**).*
3) Existing and projected future control authority elements available in a Data Center environment. *Control authority elements for power management are the fastest growing component of this concept definition effort. The concept of software control of hardware power states has proliferated into data center servers, smart building control systems, facility HVAC systems, and much more. The control authority elements targeted for this research effort are available. The incorporation of HVAC control will require a modern, IP addressable HVAC control card. (**See section 1.1.3**).*

Data Centers Energy Reduction and Management Through RTOC

Task 2.0: Identification of Critical Elements - Successful
The research team identified critical elements of the RTOC algorithm requiring a feasibility demonstration in later stage-gate phases. The team documented initial key critical condition states, control authority elements, and suggested test cases (**Milestone 2)**. *The research team identified 3 critical elements of the RTOC algorithm requiring a feasibility demonstration in a later Stage 3 effort: Blade-load Curve Verification, Server State Transition Curve Verification, and Calculation Time & System Response. Key critical condition states and control authority elements are indentified and test cases proposed to verify operating assumptions (**See section 1.2**).*

Task 3.0: Translation of Routing and Telecon Data Center Behavior - Successful
The research team utilized findings from Task 1.0 and Task 2.0 to translate Routing and Telecom Data Center system behavior into a preliminary control model (equation) (**Milestone 2**). *The Differential Algebraic Equation (DAE) resulting from this research will be incorporated with the RTOC algorithm to create the RTOC power management application in later development stages. The DAE model contains the critical state and control parameters (**See section 1.3**).*

Task 4.0: Mitigate Market Introduction & Technical Risks - Successful
The research team documented a RTOC algorithm Data Center test scenario and potential plan for incorporation into new and existing Data Centers (**Milestone 3**). *The team formulated and outlined a test scenario for a future, Stage 3, program phase implementation in Section 1.2. The testing and implementation path leverage commercial market products, e.g. IBM, HP or SUN servers. The identified RTOC power management application implementation path into data centers minimizes data center impact (**See section 1.4**).*

Task 5.0: Project Management and Reporting
The research team provided reports and contractually required deliverables in accordance with the Federal Assistance Reporting Checklist (DOE F 4600.2). These deliverables address Federal and American Recovery Reinvestment Act (ARRA) requirements, as well as ITP programmatic requirements *(**See section 1.5**)*.

Projected Energy Savings
While the Impact Projections Model may provide an overly optimistic estimate of energy reduction, the ability to prototype quickly at a much lower cost than large scale capital projects makes it a desirable candidate for further investigation.

Data Centers Energy Reduction and Management Through RTOC

Recommendations

The research and development team achieved the Stage 2 Concept Definition goals and recommends Stage 3 Concept Development. Recommended Concept Development efforts include:

1. Technical concept proof of feasibility through programming the RTOC algorithm and data center model. These efforts include refining technology specifications or information requirements.
2. Performing predictive modeling and simulation of algorithm performance to verify desired operation. Obtain feedback from end users and incorporate results into revised technical specifications.
3. System engineering study to assess scale-up challenges or issues and risk mitigation management. This effort supports scale-up of prototype design to field test unit, identifies data gaps for scale-up, and determine feasibility of scale-up through models and analysis.
4. Prototype and demonstrate the algorithm feasibility on representative data center hardware. This effort includes developing a prototype according to technical specifications and cost goals. Prototype testing exercises critical components under simulated operating conditions and assists in qualifying the technology. The prototype demonstrations to end users also serve to assess user feedback and identify technology production partners for next stage efforts.
5. Explore critical financial, legal and regulatory issues. These efforts include understanding all potential financial, legal and regulatory issues. Review market information (e.g., end user needs, market potential) and refine market impact. Establish commercial partnering agreements.
6. Prepare field test and information verification plans for Stage 4.

The BAE Systems' led research and development team contains the requisite technical skills, systems engineering discipline and program management acumen necessary to pursue successful RTOC power management application development through commercial deployment.

Data Centers Energy Reduction and Management Through RTOC

References / Bibliography

Environmental Protect Agency. (2007). Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431: U.S. Environmental Protection Agency ENERGY STAR Program.

Hodgin, Rick. geek.com. 26 May 2009. Web. http://www.geek.com/articles/chips/new-server-farm-trend-power-savings-over-performance-20090526/. 20 June 2011.

Kopp, R.E. (1962). George Leitman (Ed.) "Pontryagin Maximum Principle," in Optimization Techniques. New York: Academic Press, Inc.

Raths, David. Govtech.com. 19 December 2006. Web. http://www.govtech.com/magazines/pcio/100560229.html. 20 June 2011.

Ross, I.M. (2005). "Control and Optimization: An Introduction to Principles and Applications", Electronic Edition, California: Naval Postgraduate School.

Data Centers Energy Reduction and Management Through RTOC

Appendix: Acronyms

Acronym	Definition
AC	Alternating Current
ADDR	Adder
Apps	Applications
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central processing unit
CRAC unit	Computer room air conditioning unit
DAE	Differential-Algebraic Equations
DC	Direct Current
DOE	U.S. Department of Energy
DSP	Digital signal processor
FPU	Floating-point unit
HVAC	Heating Ventilation and Air Conditioning
IC	Integrated Circuit
I/O	Input / Output
ITP	Industrial Technologies Program
MULT	Multiplier
ODE	Ordinary Differential Equation
QoS	Quality of Service
OSI model	Open Systems Interconnection model
PC	Personal computer
RTOC	Real-Time Optimal Control
RTOS	Real-time operating system
SOPO	Statement of Project Objectives