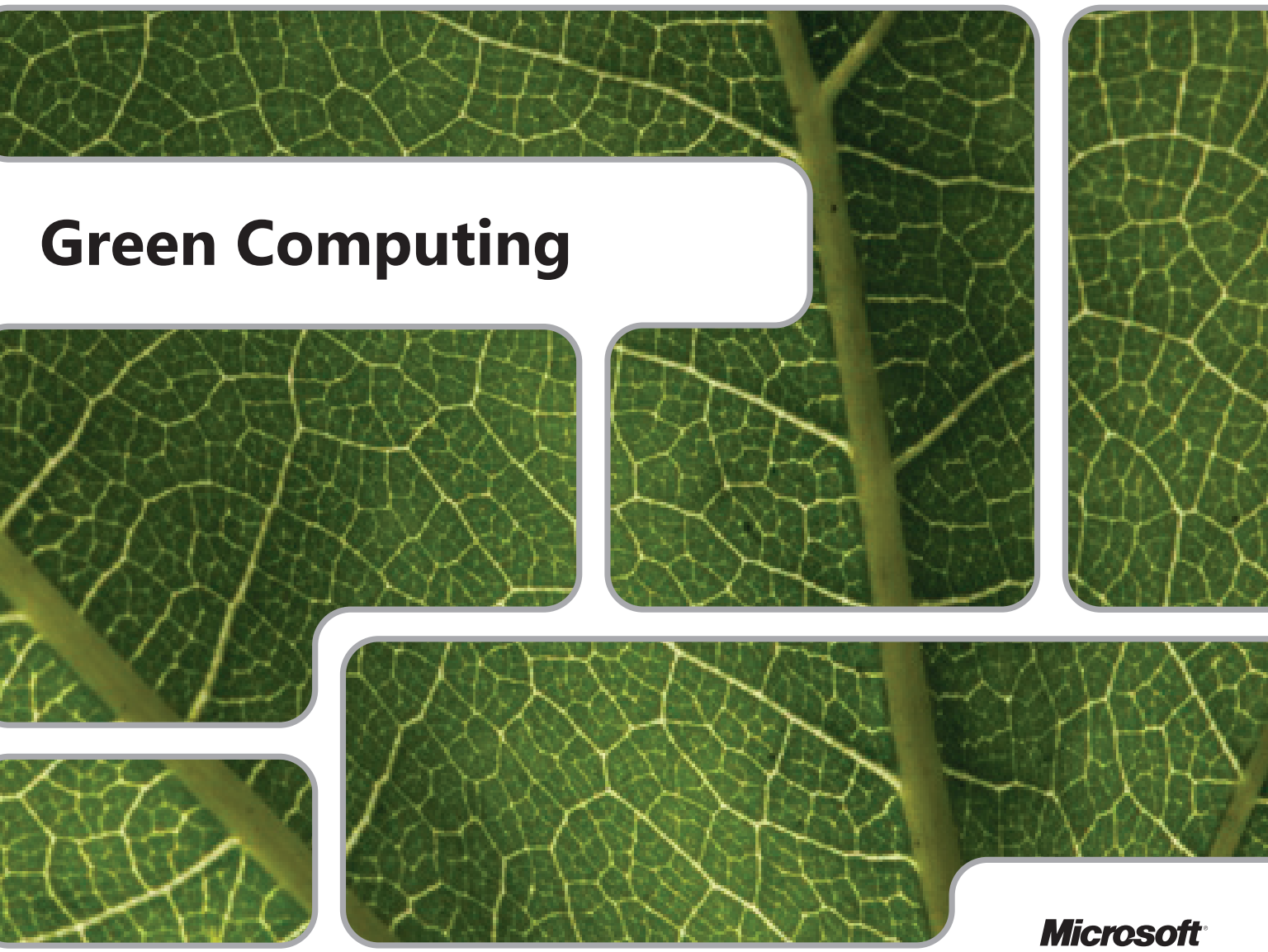
The Architecture Journal

Learn the discipline, pursue the art, and contribute ideas at www.architecturejournal.net

input for better outcomes

18

Green Computing



Microsoft

Environmentally Sustainable Infrastructure Design

Green Maturity Model for Virtualization

Application Patterns for Green IT

Architecture Journal Profile: Udi Dahan

Profiling Energy Usage for Efficient Consumption

Project Genome: Wireless Sensor Network for Data Center Cooling

Green IT in Practice: SQL Server Consolidation in Microsoft IT

Contents

Foreword 1
by Diego Dagum

Environmentally Sustainable Infrastructure Design 2
by Lewis Curtis
A comprehensive understanding of environmental sustainability needs for IT infrastructure system design.

Green Maturity Model for Virtualization 9
by Kevin Francis and Peter Richardson
The authors present a study on the maturity of virtualization practices and offer a sustainability roadmap for organizations planning a green agenda.

Application Patterns for Green IT 16
by Dan Rogers and Ulrich Homann
A complete analysis on power-efficient applications design, considering tools that help the architect achieve scalability without deriving in energy waste.

Architecture Journal Profile: Udi Dahan 22
For this issue's interview, we catch up with Udi Dahan, noted expert on SOA and .NET development, and four-time Microsoft MVP.

Profiling Energy Usage for Efficient Consumption 24
by Rajesh Chheda, Dan Shookowsky, Steve Stefanovich, and Joe Toscano
This article suggests that tracking energy consumption at every level will become the factor of success for green architecture practitioners.

Project Genome: Wireless Sensor Network for Data Center Cooling 28
by Jie Liu, Feng Zhao, Jeff O'Reilly, Amaya Souarez, Michael Manos, Chieh-Jan Mike Liang, and Andreas Terzis
Learn about a research project on an interesting adaptive cooling technique.

Green IT in Practice: SQL Server Consolidation in Microsoft IT 35
by Mark Pohto
This Microsoft IT case study offers useful guidelines for effective resource consolidation for environmental sustainability.

Dear Architect,

The theme of this issue, "Green Computing," is especially important and timely: As computing becomes increasingly pervasive, the energy consumption attributable to computing is climbing, despite the clarion call to action to reduce consumption and reverse greenhouse effects. At the same time, the rising cost of energy — due to regulatory measures enforcing a "true cost" of energy coupled with scarcity as finite natural resources are rapidly being diminished — is refocusing IT leaders on efficiency and total cost of ownership, particularly in the context of the world-wide financial crisis.

We are pleased to have Lewis Curtis, Principal Architect on the Microsoft Platform Architecture Team, as our subject matter expert on this topic and co-editor for this issue. In the first article (page 2), Lewis provides a holistic approach to greener architectures, and he also helped us subdivide the topic in five perspectives:

- **Physical.** The Genome research project, described in "Wireless Sensor Network for Data Center Monitoring" (page 28), uses heat distribution data from a wireless sensor network to optimize data center design and server provisioning in order to avoid overcooling the entire data center.
- **Operating Platform.** Hardware resources are often allocated based on a worst-case scenario that may happen with a low frequency. As a result, complete farms may be as much as 90 percent underutilized. Read Mark Pohto's article on SQL Server consolidation (page 35) to learn more about virtualization and other consolidation tactics.
- **Sustainable Intelligence.** An Energy Usage Profile (EUP) is an essential tool for measuring energy consumption in various domains such as hardware, operating systems, users, and applications, as the Software Architect of the Future explains in the article by Steve Stevanovich and coauthors (page 24).
- **Application Development.** Solution architects also have an opportunity here, as coauthors Dan Rogers and Ulrich Homann point out in "Application Patterns for Green IT" (page 16). Green computing discussions today tend to focus on the platform, hardware, and data centers. However, application inefficiencies, such as suboptimal algorithms and inefficient usage of shared resources causing contentions, are originators of higher CPU usage and, therefore, energy consumption.
- **The Cloud.** In their article on green design principles (page 9), Kevin Francis and Peter Richardson also cover utility computing-based delivery models. Insofar as this these models consolidate organizations, consumption has the potential to be remarkably reduced as the Internet-scale data centers in which services are hosted can make efficient use of shared resources (servers, storage, cooling mechanisms, and so forth).

Now and in the future, green computing will be a key challenge for businesses and presents a leadership opportunity for all architects. It is an exciting time to be an architect. Celebrating the relaunch of the Architect MVP (Most Valuable Professional) program, we interviewed Udi Dahan (Architect MVP for four years now) in our Architecture Journal Profile (page 22).

We hope you enjoy these thoughtful articles on green computing. We invite you to visit the Microsoft Environmental Sustainability portal at http://www.microsoft.com/environment/, and as always, we welcome your feedback at editors@architecturejournal.net.

Diego Dagum

Environmentally Sustainable Infrastructure Design

by Lewis Curtis

Summary

As the commitment to reduce environmental impact and power consumption are becoming increasingly important objectives for organizations, architecture leaders are now proactively considering environmental resource constraints along with more traditional IT business goals.

This article exams significant architectural decision points in the infrastructure and discusses discuss holistic issues for environmental sustainability.

Rather than viewing the environmentally sustainable data center as a product feature checklist for a one-time win, serious IT architects are adopting a more comprehensive sustainability plan in their data center system design. While new technology from the industry continues to drive efficiency into the IT infrastructure, environmental systemic quality metrics need to be built into at every part of the IT architectural process, and an ongoing architectural commitment is required at all levels of the infrastructure, beyond a typical product procurement strategy.

For Architects: What's Different About "Green"

One cannot achieve a Sustainable Strategy with a product, it takes an Architectural commitment.

The corporate architect must realize that the impact corporations make on the environment now is engrained in doing business:

- Executives are allocating time, energy and money to invest in environmental initiatives.
- Governments are allocating research and regulations, and laws are being written to address the efficiency of data centers and other critical components of IT infrastructure.
- Consumer advocates, policy makers and influential industry leaders are promoting IT organizations to significantly confront the impact that computing and electronics make on the environment.

This is a vastly different landscape than for other galvanizing themes such as SOA, agile design, or Web 2.0 and SaaS. Those themes did not elicit the same degree of regulatory, legal, and advocacy. Ten years from now, those initiatives may not exist in their present incarnation, but commitments to reduce environmental impact and power consumption will continue to be important objectives for organizations.

IT professionals must shed the traditional view of the environmentally sustainable data center as a product feature checklist to gain one-time wins. While new technology from the industry will help drive efficiency into the IT infrastructure, it will not replace the necessary ongoing architectural and process commitment.

For example, a virtualization or a blade environment product decision has the potential to reduce power consumption. Yet, if there are no processes or architectural guidance to go with it, this can encourage virtual server sprawl and eventually increase power consumption at a higher rate due to additional physical servers allocated to meet the virtual sprawl needs. And of course, increasing rack power density without an aligned cooling architecture is a recipe for data center disaster.

Environmental impact and power consumption are becoming crucial architectural systemic quality metrics.

In the past, IT architects gave too little attention to security, eventually suffering the consequences. Like security, environmental sustainability design qualities are quickly becoming pervasive architectural issues for new projects.

New Architectural Decision Points

The need to reduce power consumption is obvious. Gone are the days of measuring data centers by square foot of space. Now, data centers are increasingly sized by the watt. More efficient technologies with new capabilities are being promoted as magic cures. Yet, saving energy is a much more complex architectural problem, requiring a coordinated array of tactics, from architectural power management capacity planning techniques to optimizing operational processes and facilities design.

Continuously reducing environmental impact is more challenging. There is a consensus that serious negative environmental repercussions are the consequence of manmade pollution. From examining the atmosphere, soils and oceans: governments, partners, consumers and industry organizations want companies to have a more positive impact on the environment. The most common environmental impact measurement is labeled carbon footprint, usually measured in tCO_2eq (metric tons of CO_2 equivalent) based on the source of energy and amount consumed, manufacturing and logistics impact (often labeled embodied cost), as well as end-of-life impact (e-waste, environmental externalities, and so on).

Commitment to a Sustainable Technology Strategy

While new technology from industry helps drive efficiency into the IT infrastructure, an ongoing commitment to a sustainable technology strategy is required in IT architecture and process. Environmental systemic quality metrics need to be built into every part of the IT architectural process.

Traditional IT architecture goals persist in the waste-conscious era of sustainable data center design:

- Encourage IT reuse
- Reduce IT complexity

- Align stakeholders
- Optimize functional and non-functional (systemic quality goals)
- Spend the organization's money wisely

To design successful IT solutions that reduce power consumption and environmental impact, IT architects must also consider the environmental impact on other systemic architectural quality metrics as a part of every design goal. This includes (but by no means limited to) name services, backup and recovery, management systems and network infrastructure.

Focus on Business Objectives

Research on different environmentally sustainable endeavors from internal activities, customer projects, and industry experts indicates architectural leaders leverage three main themes that differentiate successful environmentally sustainable projects:

- *Know specifically who and what you want to impact (as well as not impact):*
  - Regulatory entities
  - Business units and activities
  - Specific public demographic groups.
- *Know specifically which metrics you will target and ignore:*
  - Customer-focused metrics
  - Operational-focused metrics
  - General public (non-customer) perception focused metrics.
- *Use a holistic plan of action for developing an environmentally sustainable solution that leverages:*
  - Technologies
  - Processes
  - Strategies.

There are infrastructural architectural design approaches to start analyzing environmentally sustainable goals.

Sustainable Intelligence: Understanding Energy Consumption and Environmental Impact

As an industry segment, data centers are one of the fastest growing energy consumers. Why?

- IT systems are demanding increasing amounts of energy to power larger and larger solutions. Architects are designing systems with significantly more complex processing elements and dependencies.
- Energy consumption from physical servers has increased dramatically in the last five years.
- New IT solutions are being introduced into the enterprise at a velocity that significantly outpaces solution retirement.

Energy Consumption

Organizations are realizing that the source and amount of their energy consumption significantly contributes to green house gas (GHG) emissions. In response to this awareness, organizations are currently using the following equation:

Reduced energy consumption
= reduced green house gas emissions
= reduced operational costs for the data center and business

For architecture models, it means adopting fewer and more energy efficient systems while refactoring application environments to make optimal use of physical resources (doing more work with less code and systems) as well as leveraging providers that are more energy- and GHG-efficient.

A typical data center consumes energy in four basic areas:

- Critical computational systems (servers, networks, storage)
- Cooling systems
- Power conversion such as power distribution units (PDU)
- Hoteling (everything else: lighting, and so on).

Environmental Monitoring

Leaders cannot manage what they cannot measure. Therefore, an organization needs good environmental measurement solutions. They need to use environmental monitoring to measure consumption and output, and to develop actionable metrics and forecasting.

The following technology exists for measuring energy consumption and thermal output for data center elements:

- circuit meters (data center zone area or group of racks)
- power strip meters (group of systems or rack)
- plug meters (one physical system)
- base board controller energy consumption metering (one physical system)
- external thermal sensor metering (predefined floor or rack area)
- internal server thermal metering (one physical system).

Extensible Architecture

An architecture that considers environmental impact should be extensible. Due to the proprietary nature of most environmental metering interfaces from separate vendors, IT architects should aggregate these communication models into extensible communication architecture. As new metering interfaces and technologies evolve, the solution can be extended as well as normalized to reduce complexity.

To design an efficient environmental metering environment, it is important to assemble a functionally decomposed environment that leverages existing services.

Proprietary energy API services. Most vendors have their own proprietary API model to interface with the metering devices. Because energy metering architectures differ with many data centers, larger organizations may have more than one proprietary interface environment. It is important to set up a reliable design standard that reaches across data centers and technologies.

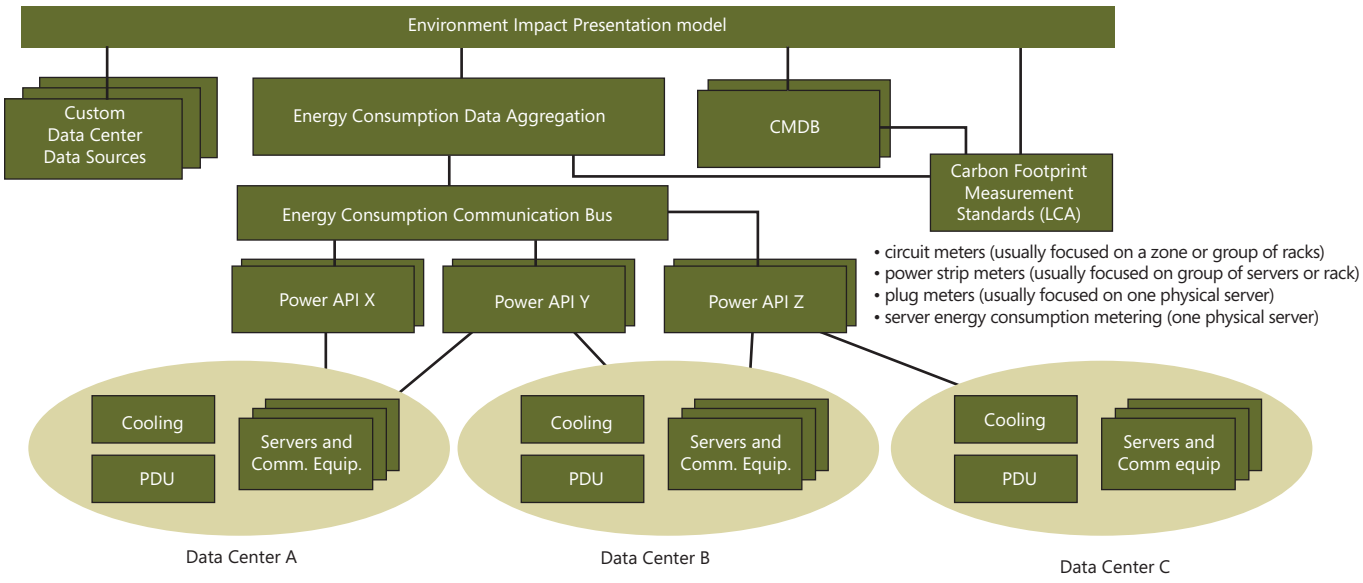
Environmental consumption communication bus. Because of diverse proprietary environmental metering interface systems (as well as versioning changes issues), organizations should assemble a common communication bus to assemble the environmental monitoring solution into a common interface model for different metering and reporting systems.

Environmental consumption data aggregation zone. This is a common data collection repository designed for frequent updates. This area is the collection point for environmental data across data centers.

Configuration management database environment (CMDB). As the names suggests, CMDB environments store mostly static (or infrequently updated) system configuration information. It is important to be able to associate this information with metering systems to understand the impact of configuration decisions on environmental metrics.

GHG/environmental measurement standards. Most organizations have or are in the process of defining the algorithms for measuring GHG impact. This equation usually depends on the source and amount of energy utilized. However, environmental life cycle assessment models could expand in scope as cap and trade programs mature.

Figure 1: Designing a common environmental metering environment

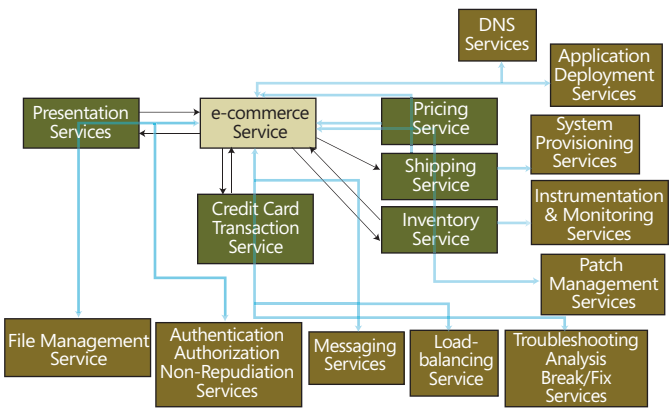


Often, this data depends on the organization's environmental life cycle assessment to lock down the scope of impact on metering parameters. However, it is important to keep these equations loosely coupled with the existing environmental metering environment. This allows the organization to adapt as metering standards change

Custom data center data sources. In designing a common environmental metering environment, there often are unique data sources that are important for the solution. Examples include the price and source of energy for that specific data center, operational logistics data, and common data center performance data. It is usually best to keep these systems separate with some common interface standards rather than grouping them together.

*Environmental impact presentation model.*This is the presentation aggregation point for different user personas (Figure 1). While the architectural principles are the same, the architect can leverage many different design options to accomplish the task.

Figure 2: Solutions having an environmental impact on data center services



Using a Holistic Design Approach

It's easier to make environmental impact decisions at specific engineering and development granular points in the architecture. However, it becomes more difficult understand how those decisions interact and impact the IT and business ecosystem.

A holistic design means the architect sees the big picture impact as well as how all the pieces fit together productively. Figure 2 shows how a sample e-commerce architecture has an environmental impact on supporting data center services.

Also, granular areas in the design will sometimes conflict. For example: PUE (power usage effectiveness) metrics will encourage efficient physical data center design by measuring the total datacenter consumption compared to the amount of energy utilized for critical systems (servers, storage, communications, etc..).

This is a popular metric today and produces valuable data for IT organizations.

However, the design of this metric encourages critical systems using more energy in the datacenter, not less. In effect: Replacing systems with more efficient servers could hurt PUE scores.

PUE can be positive for measuring overall physical datacenter operations but not a great match for server efficiency models. It is important to utilize the metric that drives the right behavior for the right areas.

It is critical that infrastructure leaders have this holistic view when leading environmentally sustainable efforts.

System SKUs

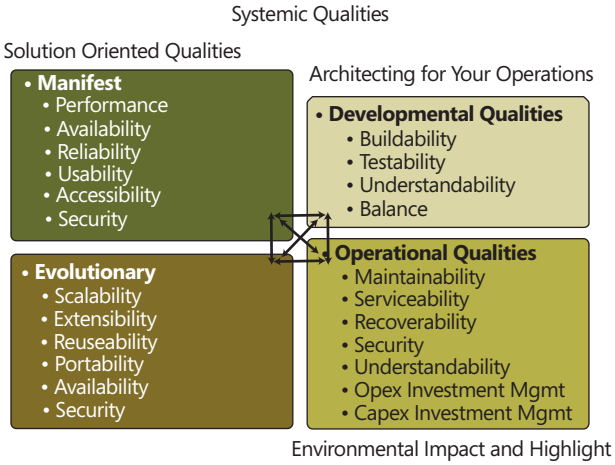
Establishing a focused set of system stock keeping units (SKUs) for each tier and layer area will help you to enforce energy efficiency and consumption standards, as well as environmental impact standards for your organization.

For example, when considering hardware purchases, ACPI 3.0 Systems can use advanced power management capabilities from Windows Vista and Windows Server 2008 to reduce energy consumption. For the server, consider reducing or eliminating redundant power supplies and acquiring the most efficient power supplies available.

Optimizing the Infrastructure Platform

Traditional modeling promotes component diagrams and physical machines between tiers. However, architects need additional information to

Figure 3: Systemic Qualities Incorporating Environmental Impact



make informed decisions about environmental optimization.

Architects often elect for more redundancy to improve performance, availability, and scalability. While this can improve some specific systemic qualities, a culture of excessive redundancy can lead to problems. One of those problems is complexity. A small increase in architecture complexity can yield unintentional energy consumption results in large scale solutions. This is one reason why many large-scale environments use significantly simplified designs (it also usually decreases operational brittleness). Energy consumption pressures and environmental impact needs are other incentives for architects to minimize complexity.

Decomposing the Infrastructure Environment

To reduce the impact of key systems in the architecture, the infrastructure should be decomposed into finer grain areas for environmental focus (Figure 4).

By examining each tier (Client, Presentation, Business, and Resource), the team can analyze architecturally significant decision points to identify environmentally sustainable best practices for their organization. To determine environmental impact optimization for the overall platform architecture, it is important that each tier be examined by:

- Solution environment
- Operating system environment
- Physical system environment.

Each architectural tier can be divided into five system service areas of focus:

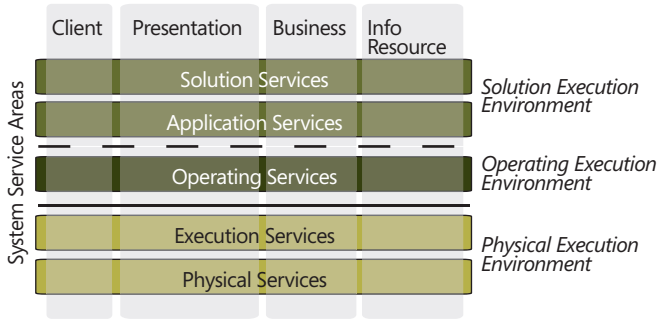
- Physical services
- Execution services
- Operating services
- Application services
- Solution services.

Client Tier Optimization

There are many ways to optimize a client environment to save energy and reduce environmental impact (Figure 5).

Client physical execution environment. Acquiring an Energy Star 4.0 system which recognizes ACPI 3.0 power management capabilities from Windows Vista allows the operating system to manage power for processors (and

Figure 4: Decomposing an n-tier architecture between tiers and systems service areas for focused environmental optimization



multiple cores), attached devices, and allows advanced capabilities for hibernation and sleep. Also, administrators can use group policies to throttle back the maximum CPU load to reduce energy consumption when needed.

Operating execution environment. To leverage advanced energy efficient computer hardware, the operating environment must be capable of using the new ACPI 3.0 hardware functionality, and it must deliver advanced performance and power management capabilities for the user and administrator. The operating execution environment is the configuration and standardization of the operating system and the supporting utilities. It is crucial to leverage the most aggressive power savings capabilities possible while accomplishing computing goals of the organization. When setting up a standardized configuration, minimize the number of running system services to reduce energy consumption.

Note: Software vendors have developed various solutions that can selectively turn client systems on and off to minimize energy use.

Application services environment. To reduce the amount of resources a client must use to run a fully installed application, architect teams can leverage client application virtualization from solutions such as Microsoft's

Figure 5: Energy consumption approaches on the client tier

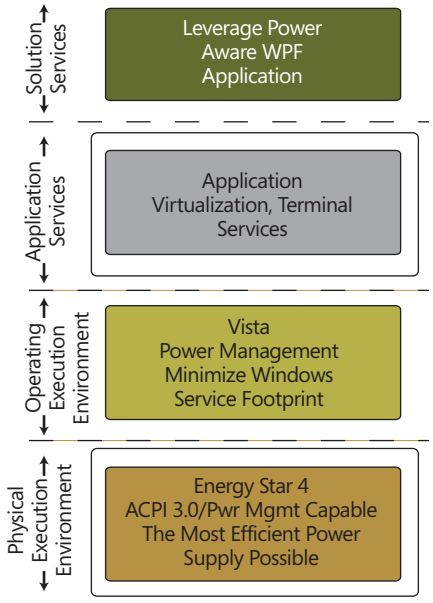


Figure 6: Presentation Tier Example

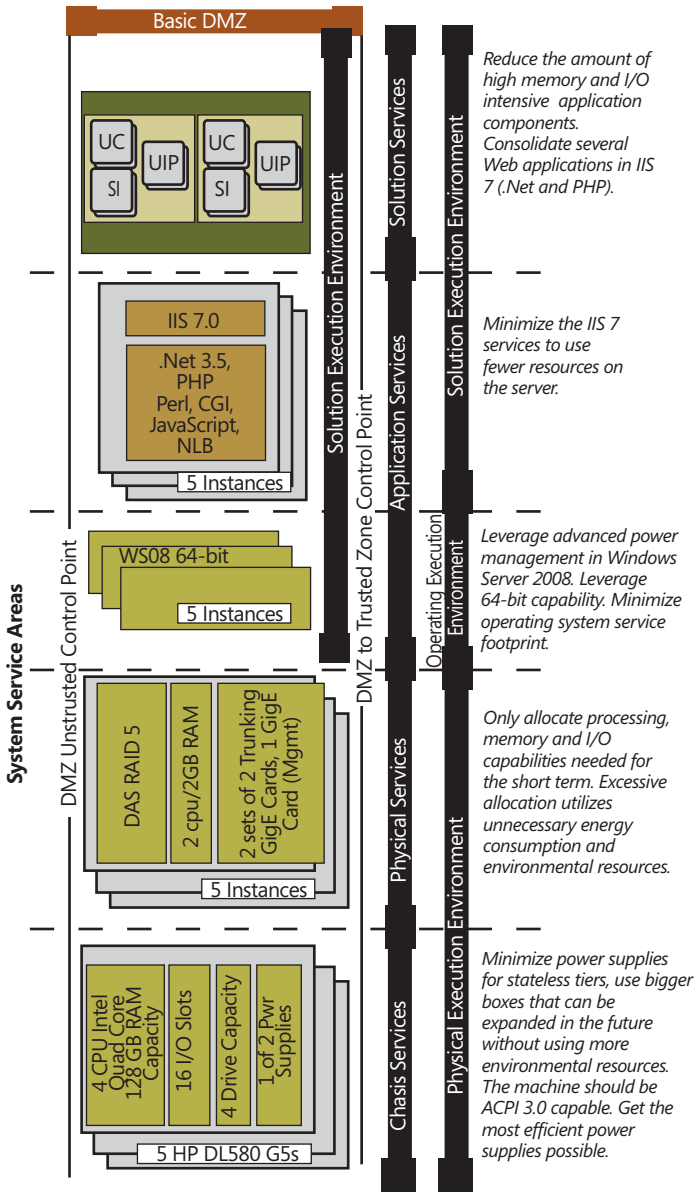
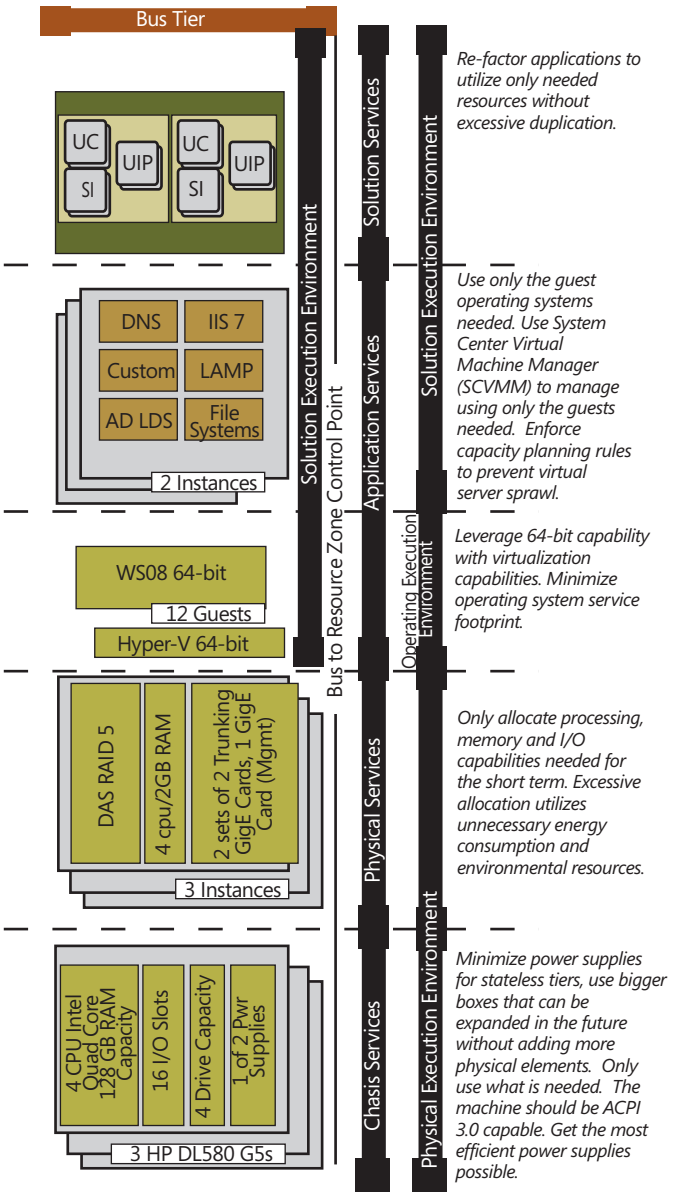


Figure 7: Virtualization Example, Business Tier



Application Virtualization solution for client systems and remote client interface solutions from new Windows Server 2008 Terminal services. However, this takes careful planning and works in a focused set of scenarios. It is important to carefully meter the full GHG and energy consumption tradeoffs for the actual deployment.

Software environment. Power-aware WPF applications can use less power-intensive presentation experience based on the power state of the client. Also, some are aggregating application development best practices to minimize energy resource consumption on the client environment.

From Load Balancing to Load Aggregation and Optimization

Typical N-Tier design is often plagued by competing and siloed budget allocations. This design creates unnecessary duplication and produces extensive energy consumption waste in the organization.

Most organizations use multiple n-tier designs with many underutilized servers. This approach consumes more energy and increases the organization's carbon footprint.

Aggregating tier areas reduces capital and operating costs, energy consumption, and environmental impact, and it simplifies management with consistent builds across the organization.

Consolidating separate n-tier solutions often takes unique approaches for specific tier areas. In the next sections, we will investigate common approaches for each tier area.

Presentation Tier Optimization

The presentation tier represents those data center services which provide and manage a consolidated server-based user experience for users (Figure 6). The traditional example is the Web server; other examples include common portals to Wireless Access Protocol (WAP) gateways. It is in this tier that server sprawl can happen quickly as scalability demands room to

grow. Today, many architects are consolidating their work into a multihosting environment (each server manages multiple Web site environments). Not only does this reduce energy consumption through consolidation, this also promotes energy efficient configuration standardization across servers (for example, limiting Web servers to only one power supply).

Business Tier Optimization

The business infrastructure tier is called by many names (the business, application or transaction tier). It is where critical application business rules, workflow rules, transaction management, and integration coordination take place. Consolidating multiple applications at this tier is more complex. Often, this can involve virtualization to ensure significant separations of concern for business tier systems (reducing the impact of cross-application interference activity; see Figure 7). With business and resource tier architecture, organizations often make the mistake of physically over-provisioning the server configuration, using excessive amounts of energy with no real processing benefit. For example, most departments buy more processors, memory, and disk capacity for the servers even when the server only needs a small fraction of these resources. This is commonly done for immediate capital budgetary reasons rather than scalability needs. In addition to wasting energy, often the department pays a premium price for the added components. Waiting for these upgrades usually decreases cost over time (saving the department capital expenditures as well as reducing energy consumption). However, capital budgetary models in organizations usually prevent such good financial behavior in the enterprise market.

Information Resource Tier Optimization

The information resource tier represents important data management systems in the infrastructure. Common examples include databases, directories, file-systems, and flat files; a database example is shown in Figure 8.

Information resource tier environments are usually high in I/O and memory-intensive activity. Windows Server 2008 Active Directories, File Servers, and SQL Server 2008 have the capacity to consolidate databases and directories on a single physical environment.

Energy and Environmental Transference

Environmental transference leverages software and services with a cloud application resource for solution resource optimization.

Transference makes it possible for an organization to transfer power, processing, and environmental impact to another entity. This approach can potentially reduce cost, complexity, and reduce impact on the environment (if the service provider has better environmental impact metrics than the enterprise). Today, infrastructure architectural solutions can take advantage of transference strategies with service providers with every tier.

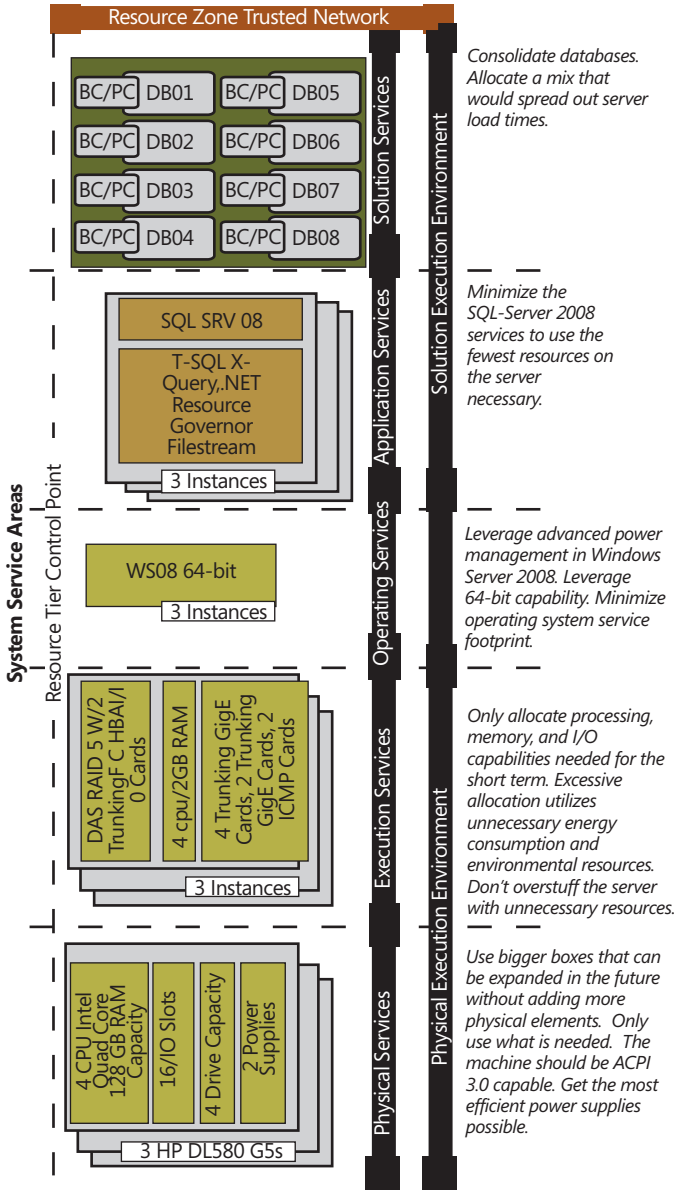
Increasingly, environmental communities are promoting measuring the embodied energy/environmental cost as well as the operational environmental cost of a solution. The embodied environmental cost represents those costs involved in the manufacture of the specific service or product. Calculation of embodied environmental costs will become more accurate and prevalent as life cycle assessments mature in the industry.

Looking at the Whole System

While it is important to examine each layer of each tier carefully, it is essential to look at the architectural model as a whole to understand how your efforts are affecting specific targeted environmental impact metrics (Figure 9).

In looking at the system, the following questions should be asked when determining if the architecture design will meet environmental sustainability goals:

Figure 8: Database Example

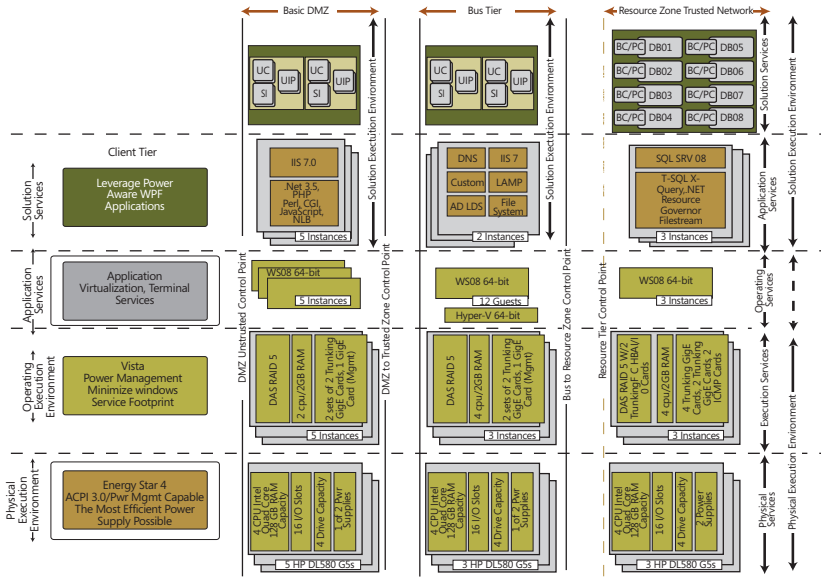


- What is the energy consumption per node / per tier at predicted loads?
- Can the architecture be translated into specific environmental metrics per the organization's policy on environmental life cycle assessments?
- How does this architecture affect the energy consumption and environmental impact of both supporting data center systems and business operational activity?

Often, architects focus too much of their time on answer-based patterns. The basic formula is: In a given condition, do this. While each answer by itself can be effective, combined, these architectural answer patterns can lead to unusable or unstable solutions.

This is the reason that architects are increasingly leveraging question-based patterns to study the holistic impact of architectural decisions. What are the consistently good environmental impact questions to address? As environmental impact analysis becomes increasingly

Figure 9: Whole System View



important, it will be crucial to leverage question-based analysis technique. (For more information on question-based architectural impact analysis frameworks, see Perspective Based Architecture Reference, listed in Resources.)

Best Practices for Sustainable Architecture Design

When it comes to designing environmentally sustainable architecture, it can be overwhelming organizing the complexity. However, we've narrowed down a list that can be leveraged when studying an architectural design (no matter the complexity or scale).

The following best practices summarize the guidelines discussed in this article:

- ***Know your environmental business objectives.*** Determine who and what you want to impact (or not impact), with regard to regulatory entities, businesses, the public, as well as the environment.
- ***Understand energy consumption and environmental impact.*** Understand where energy is consumed in the data center and how a data center affects its surrounding environment.
- ***Develop an environmental monitoring strategy.*** Use environmental monitoring to measure consumption and output and develop metrics. Know what metrics you want to target (and ignore).
- ***Establish a focused set of system SKUs.*** Establish a focused set of system SKUs for each tier area to enforce energy efficiency and environmental impact standards.
- ***Build environmental sustainability into change and configuration management processes.*** Incorporate environmental sustainability into the data center's change management and configuration management processes. Many IT organizations have a process operational model encapsulating change and configuration management. This is to ensure an adequate examination process for new technology adoption (change management) as well as encourage standardization to reduce complexity (configuration management). Strong operational process models have been attributed to reducing security vulnerabilities and understanding the impact of new technology adoption decisions. Today, energy consumption and environmental impact/differentiation need to be included in these operational processes for IT organizations to evolve.
- ***Enforce environmental impact standards into architectural capacity planning models.*** Getting one-time wins is relatively easy, promoting a continuous strategy of reducing energy consumption and GHG impact is quite difficult. Start at the heart of the problem: Reducing over-provisioning during the capacity planning phase will significantly help to prevent out-of-control growth leading to excessive GHG production and energy consumption.
- ***Optimize details in each infrastructure tier.*** Focus on the details of each tier of the infrastructure to reduce energy and environmental impact of key systems.
- ***Reduce architectural complexity.*** Reduce the number of tiers and component dependencies to reduce excessive system use, and aggregate tier systems through consolidation techniques.
- ***Leverage environmental transference when possible and acceptable.*** Use transference to leverage software plus services with external cloud providers to significantly reduce GHG and energy consumption impact in the enterprise. Ensure proper environmental service level agreements (SLAs) are in place for your organization.
- ***Use a holistic design approach.*** Use a holistic design approach to the architecture: Carefully examine the environmental components in each tier, as well as the environmental impact on external systems supporting the overall solution. Use a plan of action that leverages technologies, processes, and strategies.

Environmental impact and energy consumption are quickly becoming crucial systemic qualities for IT architectural design considerations. As this happens, it will be important for architects to understand this new systemic quality well and document successful patterns to analyze and design environmentally sustainable solutions in the future.

By focusing environmental objectives and systemically analyzing the infrastructure with proper design rigor, architects can effectively lead environmentally sustainability IT projects with a higher probability of success.

Resources

Perspective Based Architecture References
http://msdn2.microsoft.com/en-us/library/bb245776.aspx
http://www.perspectivebasedarchitecture.com

About the Author

Lewis Curtis is a principal architect for the DPE Platform Architecture Team at Microsoft focusing on next generation enterprise infrastructure architecture and architectural issues for environmental sustainability. A speaker and writer, he has published in several journals (including The Architecture Journal, IEEE ITPro) investigating best practices and challenging issues for IT Architects. As one of the first Microsoft Certified Architects (MCA), Lewis has served on the board of advisors since its inception with a focus on training and growing senior architectural leaders. He has also promoted question-based patterns for architects as the founder of Perspective Based Architecture and Systemic Quality Impact Analysis. Besides work, Lewis enjoys time with his wife and two Saint Bernards. A little known fact is that Lewis was a professional musician playing saxophone and managing bands (jazz and rock 'n' roll) before his IT career. His blog is at http://blogs.technet.com/lcurtis.

Green Maturity Model for Virtualization

by Kevin Francis and Peter Richardson

Summary

The biggest challenge facing the environment today is global warming, caused by carbon emissions. About 98 percent of CO_2 emissions (or 87 percent of all CO_2–equivalent emissions from all greenhouse gases) can be directly attributed to energy consumption, according to a report by the Energy Information Administration (see Resources). Many organizations today are speaking openly about a desire to operate in a "green" manner, publishing principles for environmental practices and sustainability on their corporate Web. In addition, many companies are now paying (or will pay in the near future) some kind of carbon tax for the resources they consume and the environmental impact of the products and services they produce, so a reduction in energy consumed can have a real financial payback.

In this article, we focus on reduction in energy consumption over the full equipment life cycle as the prime motivator for "green" application design, with energy reduction as the best measure of "green-ness." Our sole motivation is reducing energy consumption, without regard to economic impact. However, we do observe that improving energy efficiency will also reduce economic costs, as energy costs are a significant contributor to the life-cycle cost of a data center, but this happy coincidence is not explored further in the paper.

Using Architectural Principles

There are a variety of ways of performing architectural design. Regardless of the actual approach used, the use of architectural principles is both valid and useful. Architectural principles are key decisions that are made and agreed upon, generally at an organizational level, within the Enterprise Architecture domain. This enables important decisions to be made away from the heat of the moment and these decisions are clearly documented and understood by all Architects in an organization. Within an individual project, the use of architectural principles serves to keep project architecture on track and closes down pointless technical arguments. It is also important to align architectural principles with the organization's core principles to ensure that the work done by development teams furthers the organization's larger goals.

Therefore, we need architectural principles from an environmental perspective that can be aligned with an organization's environmental principles.

Why Most Applications Aren't Green

Good architects consider a range of factors when designing applications, such as reliability, scalability, security, and usability. Environmental factors have not generally been key drivers. The issue tends to be deeper than not giving sufficient weight to environmental factors. Green architectural design requires careful consideration, at a level of detail greater than what generally takes place today, and it requires software architects and infrastructure architects to work together.

Before we examine the approaches to designing environmentally efficient systems, it is useful to spend some time considering common examples of applications that make poor use of resources.

The first example is where applications are run alone on servers not because of capacity constraints, but to avoid potential conflicts with other applications — because of complex application installation issues, or simply because corporate purchasing or budgeting policies make sharing servers difficult. This obviously wastes resources through the underutilization of hardware, as well as through the power consumed to run those servers. Depending on utilization levels, the energy directly attributed to running an application may be small while most of the energy consumed by the computer is used to run the operating system, base services, and components, such as disk drives, regardless of what the application software is doing.

Another example is applications that run on multiprocessor computers but only effectively make use of a single processor. This is the case for many applications that were developed on single-processor computers and that now run on computers fitted with multiple processors. This can also result where applications are developed on computers with multiple processors but are not designed in such a way as to make use the full capabilities of the hardware. Such applications waste processing power and electricity, and can even limit the ability to run more than one application on a server in an efficient manner, which again results in applications that need to be run on dedicated servers.

Yet another common occurrence is computers that are underutilized or are not utilized at all, such as servers that run applications that only run at certain times of the day, servers that run at night to provide file and print capabilities that are only needed during the day, test environments that are used infrequently but left running permanently, or computers that run because nobody is quite sure what they do.

Most large organizations today probably have examples of all of the above categories, consuming valuable resources and producing emissions from the energy that they consume.

Table 1 Levels of virtualization maturity

Virtualization Maturity	Name	Applications	Infrastructure	Location	Ownership
Level 0	Local	Dedicated	Fixed	Distributed	Internal
Level 1	Logical	Shared	Fixed	Centralized	Internal
Level 2	Data Center	Shared	Virtual	Centralized	Internal
Level 3	Cloud	Software as a Service	Virtual	Virtual	Virtual

Reference Model for Virtualization

Virtualization is a term used to mean many things, but in its broader sense, it refers to the idea of sharing. To understand the different forms of virtualization and the architectural implications for creating and deploying new applications, we propose a reference model to describe the differing forms of the concept. In this model we observe a number of different layers of abstraction at which virtualization can be applied, which we describe as increasing levels of maturity, shown in Table 1. We assert that higher levels of virtualization maturity correspond to lower energy consumption, and therefore architectures based on higher levels of maturity are "greener" than those at lower levels, which we discuss further on.

Level 0 ("Local") means no virtualization at all. Applications are all resident on individual PCs, with no sharing of data or server resources.

Level 1 ("Logical Virtualization") introduces the idea of sharing applications. This might be, for example, through the use of departmental servers running applications that are accessed by many client PCs. This first appeared in the mainstream as mainframe and then "client/server" technology, and later with more sophisticated N-tier structures. Although not conventionally considered virtualization, in fact, it is arguably the most important step. Large organizations typically have a large portfolio of applications, with considerable functional overlaps between applications. For example, there may be numerous systems carrying out customer relationship management (CRM) functions.

Level 2 ("Data Center Virtualization") is concerned with virtualization of hardware and software infrastructure. The basic premise here is that individual server deployments do not need to consume the hardware resources of dedicated hardware, and these resources can therefore be shared across multiple logical servers. This is the level most often associated with the term virtualization. The difference from Level 1 is that the hardware and software infrastructure upon which applications/servers are run is itself shared (virtualized). For server infrastructure, this is accomplished with platforms such as Microsoft Virtual Server and VMware among others, where a single physical server can run many virtual servers. For storage solutions, this level is accomplished with Storage Area Network (SAN) related technologies, where physical storage devices can be aggregated and partitioned into logical storage that appears to servers as dedicated storage but can be managed much more efficiently. The analogous concept in networking at this level is the Virtual Private Network (VPN) where shared networks are configured to present a logical private and secure network much more efficiently than if a dedicated network were to be set up.

Level 3 ("Cloud virtualization") in the virtualization maturity model extends Level 2 by virtualizing not just resources but also the location and ownership of the infrastructure through the use of cloud computing. This means the virtual infrastructure is no longer tied to a physical location, and can potentially be moved or reconfigured to any location, both within or outside the consumer's network or administrative domain. The implication of cloud computing is that data center capabilities can be aggregated at a scale not possible for a single organization, and located at sites more advantageous (from an energy point of view, for example) than may be available to a single organization. This creates the potential for significantly improved efficiency by leveraging the economies of scale associated with large numbers of organizations sharing the same infrastructure. Servers and storage virtualized to this level are generally referred to as Cloud Platform and Cloud Storage, with examples being Google App Engine, Amazon Elastic Compute Cloud, and Microsoft's Windows Azure. Accessing this infrastructure is normally done over the Internet with secure sessions, which can be thought of as a kind of virtualized discrete VPN.

Each level of maturity has a number of significant technology "aspects" of the computing platform that may be virtualized. A summary of the virtualization layers as they map to the server, storage, and network aspects is shown in Table 2.

Starting at Home – Level 0

Level 0 ("Local") in the virtualization maturity model means no virtualization at all. Even with no virtualization, there is plenty of scope for energy savings. Traditional design and development approaches may lead to applications that are less efficient than they could be. There are also a number of other design issues that can be readily recognized in applications, and therefore, a set of rules, or principles, can be recommended to be implemented by architects and developers for all applications.

Enable power saving mode: Most PCs are idle for the majority of time, and theoretically can be turned off when idle. This can generate enormous energy savings. This has some implications for application design, as applications designers need to consider the platform (client and/or server) going to sleep and waking up. For example, if a client or server goes to sleep while an application user session is still active, what are the session timeout implications and policies when the platform wakes up?

Principle: Always design for transparent sleep/wake mode.

Examples of power-saving features that all applications should enable include:

- Testing to ensure that applications do not restrict a computer from entering sleep mode.
- Testing to ensure that an application can execute successfully when a computer has left sleep mode.
- Making sure that applications do not unnecessarily hold network

Table 2: Technology aspects for virtualization

Virtualization Maturity	Technology Aspects			
	Name	Server	Storage	Network
Level 0	Local	Standalone PC	Local disks	None
Level 1	Departmental	Client/Server, N-tier	File server, DB server	LAN, Shared services
Level 2	Data Center	Server virtualization	SAN	WAN/VPN
Level 3	Cloud	Cloud platform	Cloud storage	Internet

connections open and do not require full-screen animation, for example. Both of these may stop a computer from entering sleep mode.
- Make sure that your application uses disk access sparingly as constant disk access will stop hard drives from powering down automatically.

Minimize the amount of data stored: Data uses power because data stored in application databases or file systems needs disks to be operating to store the data. Therefore, reducing the amount of data stored can reduce the amount of power used by an application by reducing the amount of data storage infrastructure. Efficient data archiving approaches can assist here.

Principle: Minimize the amount of data stored by the application and include data archiving in application design.

Design and code efficient applications: In the past, developers were forced to code carefully because computers were so slow that inefficient code made applications unusable. Moore's Law has given us more powerful computer hardware at a faster rate than we can consume it, resulting in applications that can appear to perform well even though internal architecture and coding may be wasteful and inefficient. Inefficient code causes greater CPU cycles to be consumed, which consumes more energy, as we describe in more detail later. Tools such as code profilers are available that can automatically review the performance profile of code.

Principle: Design, develop, and test to maximize the efficiency of code.

Sharing is Better – Level 1

Level 1 ("Logical Virtualization") in the virtualization maturity model introduces the idea of sharing applications. This might be for example through the use of departmental servers running applications that are accessed by many client PCs. This first appeared in the mainstream as "client/server" technology, and later with more sophisticated N-tier structures. Although not conventionally considered virtualization, in fact, it is arguably the most important step. Large organizations typically have a large portfolio of applications, with considerable functional overlaps between applications. For example, there may be numerous systems carrying out "CRM" functions.

"Cloud computing provides the next big thing in computing — some interesting architectural constructs, some great potential from a monetary aspect, and a very real option to provide a more environmentally friendly computing platform. "

Moving to Level 1 ("Logical Virtualization") is all about rationalizing the number of applications and application platforms where there is overlapping or redundant functionality, and increasing the use of common application services so that shared application components can be run once rather than duplicated multiple times. For large organizations, this will produce much bigger payoffs than any subsequent hardware virtualization. The best way to do this is to have a complete Enterprise Architecture, encompassing an Application Architecture identifying the functional footprints and overlaps of the application portfolio, so that a plan can be established for rationalizing unnecessary or duplicated functions. This may be accomplished by simply identifying and decommissioning unnecessarily duplicated functionality, or factoring out common components into shared services. As well as solving data integrity and process consistency issues, this will generally mean there are fewer and smaller applications overall, and therefore, less resources required to run them, lowering the energy/emissions footprint and at the same time reducing operational costs. The increased use of shared application services ensures that common functions can be centrally deployed and managed rather than unnecessarily consuming resources within every application that uses them.

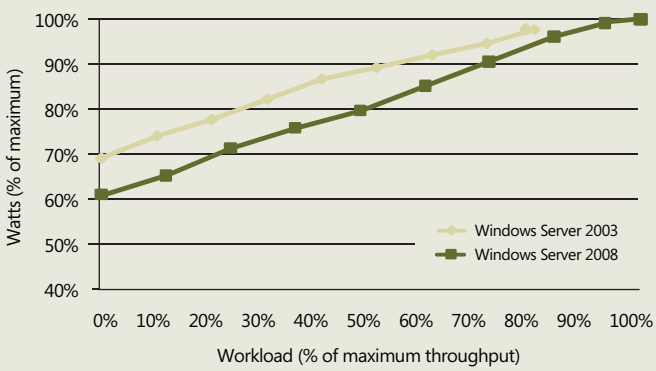
Server Efficiency

Servers are more energy efficient per unit of work done when run at higher utilization. This is because even though each processor consumes more electricity when it is working harder, there is a base level of power that is used regardless of what a computer is doing. Increasing the utilization does not increase power consumption at the same rate. Moreover, most servers operate under quite low utilization. Therefore there are huge gains to be had from using up spare server capacity.

As described in *Windows Server 2008 Power Savings* (see Resources), multiple virtual machines can run on a single physical machine without consuming significantly more power than would have been consumed if that machine was operating as a standalone server. This means that for comparable levels of throughput, you can add virtual machines at very low power cost, up to the load limits of your equipment. So long as there is excess peak CPU capacity, the savings continue to scale with the number of servers you are able to virtualize. According to the Windows Server 2008 document, "Running 4 virtual machines means saving the equivalent power output of three physical servers; running 10 virtual machines means saving the equivalent power output of 9 physical servers."

In addition, physical computers take up more space (even if they are blades) and need more cooling than their virtualized equivalents. Additional hardware is a considerable waste of power and resources if it can be avoided.

Table 1: Power usage comparison between OOB installations of Windows Server 2003 and Windows Server 2008



Principle: Develop a plan to rationalize your applications and platform portfolio first.

A single computer running at 50 percent CPU usage will use a considerably less power than two similar computers running at 25 percent CPU usage (see sidebar, "Server Efficiency, page 11). This means that single-application servers are not efficient and that servers should ideally be used as shared resources so they operate at higher utilization levels. When aiming for this end result, it is important to ensure that sociability testing is executed to make sure that applications can work together. Also, performance testing should be executed to ensure that each application will not stop the others on the server from running efficiently while under load. In effect, it is important to ensure that the available CPU can be successfully divided between the applications on the server with sufficient capacity for growth.

As a by-product of executing this form of design, it is recommended that a set of architectural standards be introduced to require that applications install cleanly into an isolated space and not impact other applications, and that testing takes place to ensure that this is the case. However, we have all seen examples where this is not the case regardless of how simple this task would appear.

Principle: Consolidate applications together onto a minimum number of servers.

Level 2: Infrastructure Sharing Maximized

As defined earlier, Level 2 ("Data Center Virtualization") is the level most often associated with the term "virtualization." Through platforms like Microsoft Virtual Server, VMware and others, server and storage virtualization does provide more efficient solutions for organizations that have the size and capability to develop a virtualized infrastructure. The bar for virtualization is low and is becoming lower all the time as virtualization software becomes easier to manage and more capable. The price/performance of servers has now reached the point where even smaller organizations hosting only 3 or 4 departmental applications may reduce costs through deployment into a virtualized environment.

It would appear straightforward, by extending the previous arguments about avoiding single-task computers, that data center virtualization would provide a simple answer and approach to share resources. This is indeed a good starting point, but is not as simple as it appears.

The lowest-hanging fruit in the transition to virtualization are in test, development, and other infrequently used computers. Moving these machines into a single virtual environment reduces the physical footprint, heat produced and power consumed by the individual servers. Each physical server that runs a virtualized infrastructure has finite resources and consumes energy just like any other computer. It is therefore apparent, from extending the same set of rules outlined above, that the aim is to load as much as possible onto a single physical server, and to make use of its resources in the most efficient way possible.

Creating virtual servers does not come at zero energy or management cost. Computers should not be running unless they are needed, even in a virtual environment. This will extend the limit of the available resources. It is particularly efficient to do this in a virtual environment as virtual machines can be easily paused, restarted, and even moved. This adds, as would be expected, the requirement for applications running on the virtual machines to be able to be paused, along with the base operating system. It could be possible, for example, to pause the operation of a file and print server at night while application updates run on another virtual machine, making use of the now available resources.

Data Center Efficiency

Bigger data centers can be made much more energy efficient than smaller data centers.

Standards are emerging for measuring this, such as the concept of Power Usage Effectiveness (PUE). PUE is defined as the ratio of total facility power divided by IT equipment power. Thus, it is a measure of how much of the power being consumed by the facility is actually being used to power the IT equipment itself rather than all the other things. By IT equipment, we mean the equipment actually delivering useful value, including servers, storage devices, switches, routers, telco network equipment, workstations/PCs, and so on. Other elements of the facility that consume power but can be considered overhead include UPS, transformers, distribution frames, lighting, cooling, fire suppression equipment, security devices, building management, and so on. Typical IT facilities have a PUE of about 2. Best practice is currently considered to be about 1.2. This represents a huge difference, with a potential savings of about 40 percent, however, it can only be achieved with sophisticated equipment and on a large scale. For all but the largest organizations, the best way to achieve these levels is to aggregate IT infrastructure across many organizations to reach the required economies of scale.

Principle: Eliminate dedicated hardware infrastructure, and virtualize servers and storage to obtain energy economies of scale and improve utilization

A Brighter Shade of Green: Cloud Computing

Cloud computing provides the next big thing in computing — some interesting architectural constructs, some great potential from a monetary aspect, and a very real option to provide a more environmentally friendly computing platform. Fundamentally, cloud computing involves three different models:

- Software as a Service (SaaS), refers to a browser or client application accessing an application that is running on servers hosted somewhere on the Internet.
- Attached Services refers to an application that is running locally accessing services to do part of its processing from a server hosted somewhere on the Internet.
- Cloud Platforms allow an application that is created by an organization's developers to be hosted on a shared runtime platform somewhere on the Internet.

All of the above models have one thing in common — the same fundamental approach of running server components somewhere else, on someone else's infrastructure, over the Internet. In the SaaS and attached services models the server components are shared and accessed from multiple applications. Cloud Platforms provide a shared infrastructure where multiple applications are hosted together.

Level 3: Cloud Virtualization

Cloud virtualization in the virtualization maturity model can significantly improve efficiency by leveraging the economies of scale associated with large numbers of organizations sharing the same infrastructure. Obtaining energy efficiencies in data centers is highly specialized and capital intensive. Standard metrics are emerging such as Power Usage Effectiveness (see sidebar, "Data Center Efficiency") which can be used to benchmark how much energy is being usefully deployed versus how

much is wasted on overhead. There is a large gap between typical data centers and best practice for PUE.

The other major advantage of Level 3 is the ability to locate the infrastructure to best advantage from an energy point of view.

Principle: Shift virtualized infrastructure to locations with access to low emissions energy and infrastructure with low PUE.

Note that applications need to be specifically designed for Level 3 to take full advantage of the benefits associated with that level. This is an impediment to migrating existing functions and applications that may limit the degree to which organizations can move to this level.

Principle: Design applications with an isolation layer to enable cloud-based deployment later on, even if your organization is not yet ready for this step.

Making Sure Your Cloud has a Green Lining

In applying the principles provided in this article, it is apparent that some cloud computing models are more attractive than others, keeping in mind that even running applications on servers that are located "somewhere else on the Internet" and are owned by someone else still produces an environmental cost. Cloud data centers may be more efficient to cool, but CPU and disks still need power, and therefore, the less used, the better.

There are four aspects of efficiency that should be considered in cloud computing:

- The placement and design of the cloud data center
- The architecture of the cloud platform
- The architecture and development approach of the applications that are hosted.

When a customer elects to purchase electricity from a supplier, in most cases it is possible for the customer to elect to buy green electricity, and in doing so, it is possible to verify just how green that electricity is. Hopefully, your organization has made a decision to buy green power.

In the same vein, you should make an informed decision about which cloud provider to use. That is not our purpose here as this article is being written when cloud services are evolving. We can, however, outline the environmental facets of cloud data center design that an organization should evaluate in selecting a platform.

Firstly, the location of the data center is important. All other things being equal, a data center in a relatively hot climate such as Central Australia will require far more resources to cool than a data center that is located in a colder environment, like Iceland. Of course, there may be other considerations such as access to other "green" mechanisms for site cooling — the point is that the characteristics of a site can help reduce the energy footprint substantially. Similarly, a location near renewable power sources, such as hydro or wind power allows for a significant reduction in carbon emissions.

Different vendors are approaching the design of their data centers in different ways. Different approaches that can be used to reduce power consumption in data centers include:

- Buying energy-efficient servers
- Building energy efficient data centers that use natural airflow, water cooling (ideally using recycled water and cooling the water in an efficient manner)
- Efficient operation by running lights out, by moving load to the cooler parts of the data center and by recycling anything that comes out of the data center, including equipment.

Some data center operators already publish statistics on their power usage, such as Google. These operators use an industry standard for measuring the efficiency of a data center through the ratio of power used to run IT equipment to the amount of power used to run the data center itself (PUE). As this space grows, it is expected that other organizations will do the same, allowing a comparison to be made.

Another area that can be considered is the technical architecture of the cloud platform, as different organizations provide different facilities and these facilities can determine the efficiency of the application and therefore impact the efficiency of the overall platform.

Some cloud vendors, for example, provide services that are controlled by the vendor; such is the case with SaaS vendors. In this case it is up to the architect of the calling application to ensure the efficiency of the overall architecture.

"The lowest-hanging fruit in the transition to virtualization are in test, development, and other infrequently used computers. Moving these machines into a single virtual environment reduces the physical footprint, heat produced and power consumed by the individual servers."

Embodied Energy

Embodied energy refers to the quantity of energy required to manufacture, and supply to the point of use, a product, material or service. When considering the energy/emissions impact of IT equipment, the embodied energy of each element must be considered as well as the operational energy consumption for that equipment. Embodied energy is significant. About 80 percent of the overall emissions over the complete product life cycle for a laptop are attributable to embodied energy, only 20 percent relate to in-use energy consumption. For a desktop PC and monitor about 70 percent is attributable to embodied energy. Embodied energy for a server is lower, about 25 percent of the life cycle emissions, but is still significant and must be considered.

We assert that larger servers are more efficient in terms of the embodied energy per unit of capacity, compared to the equivalent number of smaller servers. Therefore, apart from any direct energy consumption savings associated with reducing the number of servers, there will be a reduction in embodied energy in reducing the number of servers through virtualization.

Embodied energy means there can be a big sacrifice in deploying an application in a data center if it subsequently gets migrated to the cloud (meaning the embodied energy of the original deployment hardware is "thrown away"). A driving factor in moving through the virtualization levels from an energy point of view is to consider the embodied energy implications of moving up the levels. If you have to purchase a lot of new hardware to move from one level to the next, the embodied energy in these additions may negate the benefits in operational savings. This is why it is critical to consider the embodied energy as part of the entire lifecycle energy footprint during application design, and in selection of deployment architecture.

Other cloud vendors host virtual machines that run on a scalable cloud infrastructure. There is no doubt that this is more efficient than executing physical servers, and likely more efficient than executing virtual machines in a local data center. This approach, as has already been discussed, can still lack efficiency because of the amount of power consumed by operating system services within the virtual machine images (an inefficiency that could be multiplied across the many virtual machines hosted).

Other cloud vendors provide an application hosting platform where a single platform provides a shared infrastructure for running applications and shared facilities such as messaging and storage. As has been outlined earlier, this approach is fundamentally more efficient than a single application per machine — physical or virtual.

Therefore, when examining cloud computing as an option, the better options lie with those platforms that have the most efficient software architectures (by, for example, sharing as much infrastructure as possible across applications) and have the most efficient data center architecture (through efficient servers, lower PUE and management).

Principle: Take all steps possible to ensure that the most efficient cloud vendor is used.

Getting a Return on the Embodied Energy Costs of Buying a New Virtualization Server

We define the following:

N – the number of servers to be virtualized on a single new physical server

B – embodied energy ratio (embodied energy of new server divided by total energy consumption of that server over its life cycle)

E – efficiency factor (energy consumption of a single new server with capacity equivalent to the original N servers divided by energy consumption of N original servers, assuming the same technology and utilization, for the projected life)

T – technology factor (energy consumption of new servers per unit CPU capacity divided by energy consumption of old servers per unit CPU capacity)

U = utilization factor (utilization of old servers divided by utilization of new server)

To pay back the cost of embodied energy and realize a net gain, you need:

$E \times U \times T < (1 - B)$

If a typical B value is 25 percent, then total improvement factors needs to be better than 0.75. This is easy to achieve since even if the technologies of old and new servers are similar (T= 1) and there is no efficiency gains (E=1) you would still expect U to be lower than 0.5 if N is greater than 2 since nearly all servers are grossly underutilized. Thus as soon as you can virtualize more than two servers you can probably justify the embodied energy of buying a new server over the life cycle of that server, from an energy point of view.

"If you have to purchase a lot of new hardware to move from one level to the next, the embodied energy in these additions may negate the benefits in operational savings. This is why it is critical to consider the embodied energy as part of the entire life cycle energy footprint during application design, and in selection of deployment architecture."

Moving to Increasing Levels of Virtualization

Referring to the model in Table 1, most IT organizations are now at Level 1 with more advanced organizations moving in whole or in part to Level 2. Only a small proportion of organizations are at Level 3. Although we argue that these levels of increasing maturity in virtualization correspond to reduced energy footprint, we note that Level 3 is not necessarily an endpoint for all organizations — in some cases there may be good business reasons for not moving to Level 3 at all.

We omit the transition from Level 0 to Level 1 as the vast majority of medium to large organizations have already taken this step.

Moving from Level 1 to 2 involves replacing individual dedicated servers with larger platforms running virtual servers. If more than one physical server is required, additional benefits can be achieved by grouping applications on physical servers such that their peak load profiles are spread in time rather than coincident. This enables statistical averaging of load since normally the sizing of server capacity is driven by peak load rather than average load. The trade-off here is the embodied energy cost of the new server to host the virtualized environments (see the sidebar, "Getting a Return on the Embodied Energy Costs of Buying a New Virtualization Server"). Storage infrastructure can be approached in a similar manner.

In general, with regard to server virtualization at least, the gains in virtualizing multiple servers onto a physical server are substantially greater than the embodied energy costs associated with buying a new server to host the virtualized platform.

Level 2 virtualization can also be used to leverage the embodied energy in existing infrastructure to avoid the need for procuring more infrastructure. This may seem counter intuitive but it may be better to use an existing server as a virtualization host rather than buy a replacement for it. For example, if you have four servers that could potentially be virtualized on a single new large server, there may be advantages in simply retaining the best two servers, each virtualizing two server instances (thus avoiding the embodied energy in a new server and reducing operational consumption by about a half) rather than throwing all 4 out and buying a new server. This can be possible because the existing servers are probably running at such low utilization that you can double their load without impact on system performance. Obviously, this will ultimately depend on the antiquity of the existing servers and current and projected utilization levels.

Principle: measure server utilization levels – low utilization is a strong indicator for potential virtualization benefits.

Note also that if you can skip Level 2 altogether, rather than deploy to level 2 and then migrate to Level 3 later, you can save on the embodied energy costs of the entire Level 2 infrastructure.

Moving from Level 2 to 3 is fundamentally about two things —

sharing the data center infrastructure across multiple organizations, and enabling the location of the data center infrastructure to shift to where it is most appropriate. Sharing the infrastructure across multiple organizations can deliver big benefits because achieving best practice efficiencies in data center energy usage requires complex, capital-intensive environments.

Another advantage of Level 3 is that the infrastructure can be dynamically tuned to run at much higher levels of utilization (and thus energy efficiency) than would be possible if dedicated infrastructure was used, since the dedicated infrastructure would need to be provisioned for future growth rather than currently experienced load. In a cloud structure, hardware can be dynamically provisioned so that even as load for any individual application grows, the underlying hardware platform can be always run at optimal levels of utilization.

The trade-offs here are:

- Increased load and dependence on external network connectivity, although this is largely energy neutral
- (Perceived) loss of local control because the infrastructure is being managed by an external organization (although they may commit to service levels previously unachievable through the internal organization)
- (Perceived) security or privacy concerns with having the data hosted by an external party (such as managing hospital records, for example).

In summary, we argue that energy consumption will reduce at each increase in virtualization level, therefore organizations should be looking to move up the levels to reduce their energy footprint.

Principle: Design and implement systems for the highest level you can, subject to organizational, policy and technological constraints.

Conclusion

There is a compelling need for applications to take environmental factors into account in their design, driven by the need to align with organizational environmental policies, reduce power and infrastructure costs and to reduce current or future carbon costs. The potential reduction in energy and emissions footprint through good architectural design is significant.

The move to more environmentally sustainable applications impacts software and infrastructure architecture. The link between the two is strong, driving a need for joint management of this area of concern from infrastructure and software architects within organizations. These issues should be considered at the outset and during a project, not left to the end.

An interesting observation is that our principles also align well with the traditional architectural drivers. Does this mean that energy reduction can be used as a proxy for all the other drivers? An architecture designed solely to reduce energy over the full lifecycle would seem to also result in a "good" architecture from a broader perspective. Can we save a lot of time and effort by just concentrating solely on energy efficiency above all else?

Resources

Emissions of Greenhouse Gases in the United States 2006, Energy Information Administration
ftp://ftp.eia.doe.gov/pub/oiaf/1605/cdrom/pdf/ggrpt/057306.pdf

Google Data Center Efficiency Measurements, Commitment to Sustainable Computing
http://www.google.com/corporate/datacenters/measuring.html

"The Green Grid Data Center Power Efficiency Metrics: PUE and DCiE," The Green Grid
http://www.thegreengrid.org/gg_content/TGG_Data_Center_Power_Efficiency_Metrics_PUE_and_DCiE.pdf

"Guidelines for Energy-Efficient Datacenters," The Green Grid
http://www.thegreengrid.org/gg_content/Green_Grid_Guidelines_WP.pdf

Microsoft Environmental Principles, Corporate Citizenship
http://www.microsoft.com/About/CorporateCitizenship/US/ResponsibleLeadership/EnvironmentalPrinciples.mspx
Microsoft Windows Server 2008 Power Savings
http://www.microsoft.com/downloads/details.aspx?FamilyID=61d493fd-855d-4719-8662-3a40ba3a0a5c&displaylang=en

Moore's Law, Wikipedia
http://en.wikipedia.org/wiki/Moore%27s_law

Object Consulting
http://www.objectconsulting.com.au/beliefs.aspx

"The Path to Green IT," Mike Glennon
http://www.fujitsu-siemens.com/ps/itfuture2008/presentations_benefr/GARTNER_Green-IT.ppt

Performance Analysis, Wikipedia
http://en.wikipedia.org/wiki/Performance_analysis

Telstra Corporation
http://www.telstra.com.au/abouttelstra/csr/environment.cfm

About the Authors

Kevin Francis is the National Practices and Productivity Manager for the Object Group, located in Melbourne, Australia. He has 23 years experience in IT as an architect, enterprise architect and delivery manager. Kevin's expertise lie in the areas of SOA, Enterprise Architecture, and the Microsoft product stack, and he has lead teams up to 120 people. He's been a solutions architect MVP from 2005 to 2008. He is currently assisting Object to become Australia's leading software company. Kevin was also published in Journal 7 of The Architecture Journal and has a blog: http://msmvps.com/blogs/architecture/.

Peter Richardson is Chief Intellectual Property Officer of Object Consulting, and Executive Director of InfoMedix Pty. Ltd. He has responsibility for products and IP across the Object Group. He is also Object's sustainability thought leader and leads their sustainability initiatives and solution offerings. Previously Object's Chief Technology Officer, he has over 23 years experience as an architect, primarily in the area of large-scale enterprise systems. In 2000, Peter founded InfoMedix, a subsidiary of Object Consulting delivering electronic medical records products in the healthcare industry. From 1996 to 1999, Peter was the Australian Partner of the Object Management Group. Before joining Object in 1991 to found the Melbourne office, Peter was a Principal Engineer with Telstra Research Labs.

Application Patterns for Green IT

by Dan Rogers and Ulrich Homann

Summary

Energy is an increasingly scarce and expensive resource. This reality will continue to have a profound effect on how IT solutions are designed, deployed, and used, particularly at the data center level. While virtualization and other power-saving technologies may go part of the way to solving the problem, virtualizing inherently inefficient applications has obvious limits.

This paper presents ideas for how to design power-efficient applications. Reducing the resources required to get work done will soon not only save companies money by reducing the need to build new data centers, it will be an absolute necessity for continued growth.

The Problem

Companies around the world are either already facing, or are close to reaching, hard limits in the amount of power they can afford or are allowed to consume. In Figure 1, we see that server management and administration costs appear to consume the largest part of data center costs, and we would think of them as the limiting factors in data center growth. However, if you cap the power spend, it draws a box around the growth chart, with power becoming the limiting factor — and that is exactly what the survey data on the right of the figure shows.

With world governments looking at energy and the U.S. Congress mulling capping data center expenditures as a part of gross national power, available power and cooling are indeed key constraints to future growth. To deal with these constraints, organizations are attempting to limit or reduce the amount of energy they use. However, today's most utilized approaches — primarily focused on infrastructure optimization — may be too narrow to deal with the power challenges of tomorrow. Methods of optimizing infrastructure usage are required that run the entire ecosystem, spanning the disciplines of application architecture and design, data center management, and IT operations.

Theory of Constraints

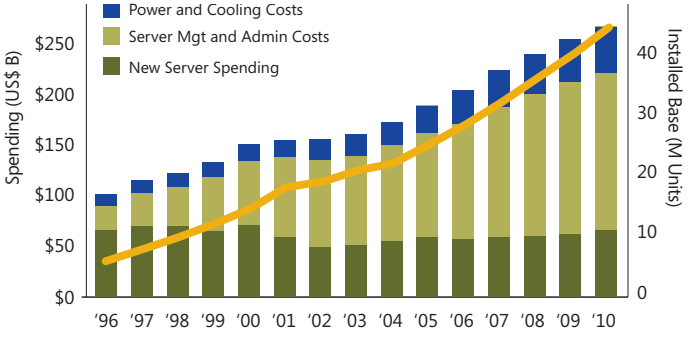
A useful starting point is Dr. Goldratt's Theory of Constraints (TOC), familiar to many readers. Essentially, TOC stipulates that identifying and managing system-level constraints (or interdependencies) is a more effective route to optimization than trying to optimize each subsystem individually. Originating in factory production management, TOC has been extended to general application and serves us well as an introduction to green IT. The most relevant principle of TOC here is throughput optimization, which assumes that you achieve maximum throughput by controlling system bottlenecks — that is, by arranging bottlenecks in the system so that they maximize the possible throughput of the available resources. In the case of IT, the constraints are infrastructure components — servers, images, virtual machines, memory, bandwidth, and so on — that control the speed with which valuable business output can be produced.

Challenges and Realities

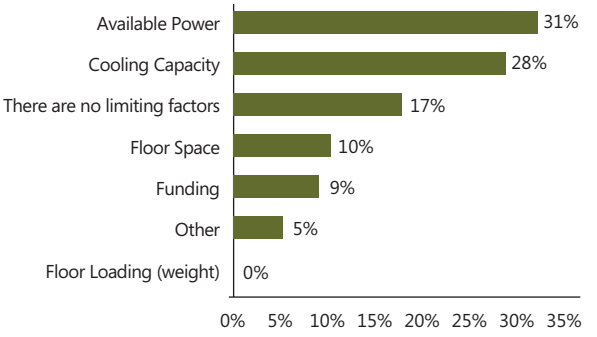
There exist well-respected and proven application and solution design patterns and behaviors in use today that conflict with the constraints of today and will only become more so as time goes on.

Synchronicity is dead. Today's application developers and architects came of age in a world where resources were typically available whenever they were needed. Synchronicity of needs and resources has been a core staple of our trade since we left the mainframe. As power capacity becomes increasingly scarce, governments have begun discussing hard limits on how much energy data centers can consume. This indicates that

Figure 1: Data Center Costs

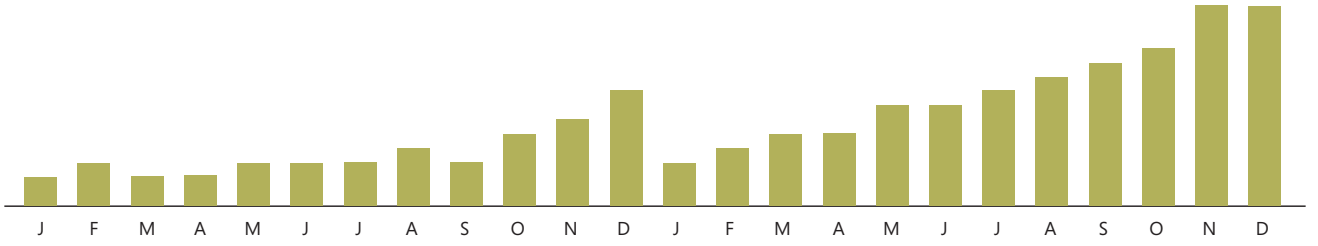


Rate of Server Management and Power/Cooling Cost Increase
(Source: IDC)



(Source: Intel Technology Journal)

Figure 2: Load profile for a deployed solution



in the future developers will be forced to expend significant additional time optimizing application power consumption, simply because this costs less than exceeding the limits on consumption.

Envisioned application usage success as a design tenet. Most solutions today are designed around a hypothetical end state rather than as an organic system that grows to meet demand. For example, a company with 150,000 users may deploy a SharePoint solution with 50 GB of disk space for each user. The solution design, capacity, and deployment plan assume "success" — that is, that all 150,000 users will actually collaborate and use their 50GB of disk space immediately.

Figure 2 shows a typical load profile for a fictitious usage of a deployed application/solution with load ramping up over two years. Extra capacity may be reassuring, but it is typically wasteful: Resources — power, storage, and so forth — are being allocated for the ultimate usage profile on Day One of the solution deployment rather than on a just-in-time basis that conserves precious resources and saves costs.

What is required? And when is it required?. In the mid-'90s, Microsoft IT used to run an HR application with a separate payroll solution that required substantial number of servers — servers that ran for only two days per month and sat idle the rest of the time. Frugal managers pointed this out, but the IT staff didn't feel comfortable reusing the systems for other tasks. Given that everyone likes getting paid, the IT folks won every time. Today, this has changed. The first response is no longer, "Can we add another computer or data center?" Instead, the approach is, "Can we design it to use fewer resources?" To do that, fluctuations in capacity demands need to be well understood — that is, one needs to ask, "What is required when?"

Design by seat of your pants (SOYP). Planning by SOYP is a sure way to introduce inefficiency into a system. Yet even major installations are often designed today using what is fundamentally guesswork based on past experience and vague capacity data. The combination of input factors generally leads to a solution design that is targeted at peak load plus n% [20% < n < 50%]. The same algorithm used to plan project timelines gets applied to capacity because nobody wants to be the one that messes up the project by underestimating.

'Belt-and-Suspenders' solution design approach. Early client/server books taught us to be over-cautious. Because our system solutions were not quite as robust as needed, we therefore went to great lengths to avoid risk, embarrassment, or exposure: If the belt should break, the suspenders will keep the pants on. In the application design arena, we devised application and infrastructure partners with multiple recovery solutions for every possible failure. The phrase "single point of failure" became the impetus for countless spending sprees: three Web servers where two would do, two load balancers because the one might just smoke someday, two databases on clustered, redundant storage area networks (SANs), and so on.

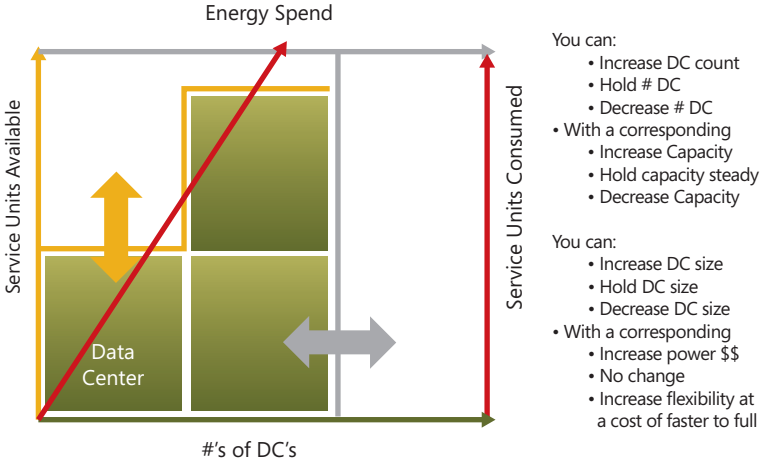
The need for recovery and robustness solutions doesn't go away, but needs to be balanced with environmental objectives of the organization. As power use grows in importance, application architects will be required to pay more attention to balanced designs — we can't count on virtualization or hardware advancements to keep the focus off of our discipline. Everyone will have to pitch in for more energy-efficient IT.

Which Constraints to Optimize?

With the law of synchronicity challenged by power constraints, we have to ask what optimizations will allow us to provide desired services while running applications and systems at ever-lower cost. As Figure 3 shows, the levers are energy spend, service units available, and service units consumed.

The way to read this figure is to recognize that it is a four dimensional graph. (In this rendering, the cost unit is the data center. If your scale is based on smaller units, adjust appropriately.) The small blocks represent scale units, meaning that they represent the fixed capacity gained as a result of a fixed amount of capital spent. You cannot add less than one unit, although you could vary the size of the units added. The lesson is that capacity does not move up in smooth fashion to match demand but rather in graduated jumps, leaving a gap between service units consumed and energy spend. The overall spend is your key optimization factor. You need to "run them full" — the corollary to the IT department's mantra of "turn them off." To maximize energy efficiency, more work must be accomplished without simply adding units.

Figure 3: Servers use vital resources whether on or off.



A Note on Power Cost Models

Power is paid for differently based on how much you use. Smaller customers pay for actual power used. They will typically try to reduce the ratio of power used to services rendered, and also to shut down non-required services. Large power consumers commit to a certain level of consumption at a given rate for a given time period. They will try to reduce the amount of power for which they contract or try to increase the number of services they can provide based on the power they have purchased. They will not typically think

first of shutting off resources since they do not get any immediate benefit from it.

Tactics

Now that we understand what drives optimization and what the application architecture and design-specific challenges are that stand in the way of this optimization, it is time to look at things we can actually do to effect energy efficient design.

Measure, measure, measure

Without the ability to gather and analyze data on the behavior of systems and their components, optimization is guesswork. The following guidelines are from a Microsoft-internal paper by James Hamilton summarizing learning from our large-scale Internet properties:

- Instrument everything. Measure every customer interaction and transaction that flows through the system and report anomalies. Runners (user simulations) are useful but insufficient.
- Code instrumentation (alerts, warnings, and status) should be done in conjunction with operations during development. Ensure that alerting can be modulated dynamically during live operations.
- Support configurable logging that can optionally be turned on or off as needed to debug issues. Deploying new builds to enable monitoring during a failure is very dangerous.
- Implement ways to easily monitor the health of your component in production.
- Record all significant actions. Every time your system does something important, particularly on a network request or modification of data, log what happened. This includes the user command and the system's response. This record helps debugging immensely. Even more importantly, you can build mining tools to discover critical information such as what kinds of queries users are making (which words, how many words, and so on).

A simple approach is best. Decide which results you want measured, and then measure them. Don't settle for approximations or statistical sampling alone. The key to successful measurement is using throughput measures instead of point-in-time samples.

A useful tool to support your efforts is the logging application block of the Enterprise Library released by the Microsoft patterns & practices team. This building block, among others, can support instrumentation investments in application code.

Composition models

Composition is a wide field in the land of application architecture. For our purposes, we simply wish to encourage architects and designers to challenge their assumptions about components, services, and system design rather than debating the relative merits of specific models.

One of the challenges identified earlier is the "what is required and when" approach. This problem is as cogent for code as it is for hardware. Application code, components, and modules can be deployed and consume resources despite being necessary only a fraction of the time — or not at all. Even systems highly suitable for partitioning are often run as monoliths. This approach may be the least risky from an availability standpoint, but it makes the dangerous assumption that resources are unlimited. The well-known approaches and methods of composition and factorization can help arrive at a resource-friendly design that still meets user needs.

Granularity

Factorization and composition are powerful tools in the quest to minimize resource usage. The goal is to achieve the minimal default application surface that can be run as a stand-alone unit. Yet it is important not to over-optimize for too limited a set of goals. To allow for selective deployment of functionality, your application code should be factored into functional groups. This approach allows components to execute at specific points in time without sacrificing overall function. Work can then be scheduled in well-identified, resource-friendly units.

Windows Server 2008 follows this model. Figure 4 shows the taxonomy used by the Windows Server development team to help them work toward the goal of enabling customers to optimize based on their different respective constraints.

Windows Server Manager uses roles to deploy and manage features. Deployment choices can be made on the feature and role level, no longer bound tightly together in unnecessarily large chunks such as workload, solution, or product. (Components in the Windows Server world are effectively invisible.) All Windows applications released with Windows Server 2008 follow this taxonomy. Other Microsoft products such as Exchange 2007 follow a similar role-based factorization approach.

While this taxonomy might appear specific to Microsoft and Windows, it works for other operating systems and applications as well. In fact, it is a general composition framework for software development — for applications, operating systems, databases, or line-of-business solutions such as ERP or CRM systems.

Figure 4: Composition and Dependencies for a Single Server or Workload

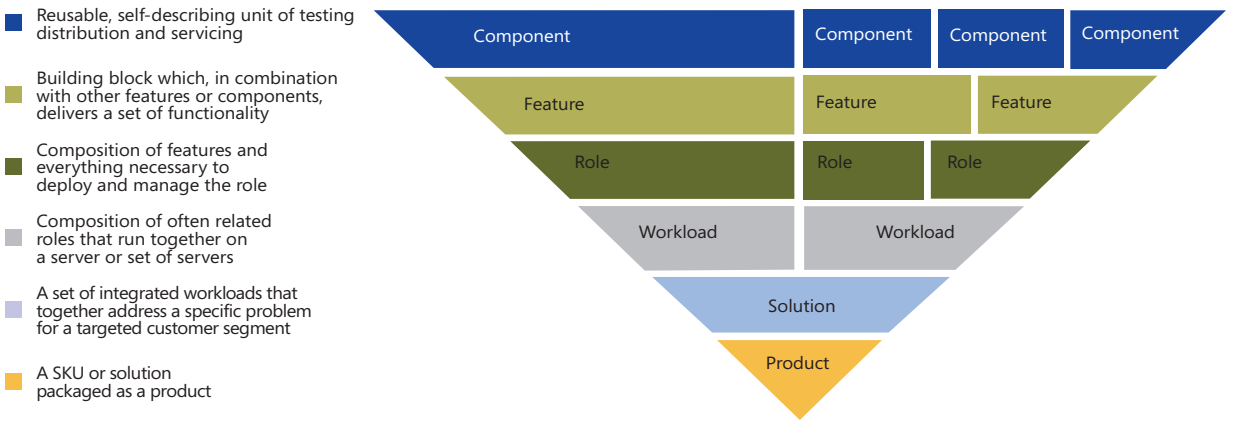
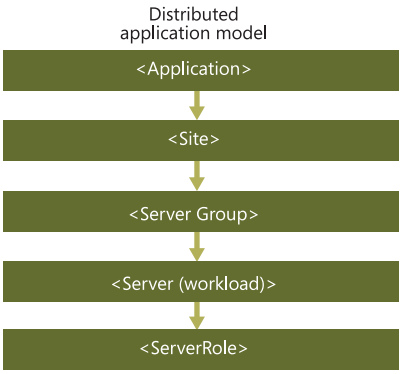


Figure 5: Distributed Application Model Dimensions



Dependencies

The model could be too rich for some applications, but the basic concept of roles is familiar to most developers. Even a one-role application is typically dependent on other roles (a database, for example) or interacts with other roles or applications (such as a Web site interacting with an ERP system). In traditional client/server applications, however, the relationships are hidden in the compiled bindings. To enable predictable and automated resource management, these dependencies must be surfaced as modeled elements included with the application bits.

The taxonomy in Figure 4 describes composition and dependencies for a single server or workload. Most solutions in today's data centers are distributed and involve multiple tiers. In Figure 5, we add dimensions to the model allowing us to describe dependencies across servers and tiers. Server/Workload and Server Role are the same as in the Windows Server taxonomy in Figure 4, enabling us to seamlessly join the models.

- Application: a logical representation of the exposed application or functionality exposed. Physically, applications expose endpoints of some kind — HTTP/HTTPS, TCP/IP and so on. Applications can also express dependencies on other applications — for example, SharePoint requires SQL Server and Active Directory.
- Site: represents a deployment of functionality into a data center or data center room. An application can have many sites associated with it.
- Server group: represents a grouping of servers with common attributes, such as high availability (network load balancing or failover cluster). An application can have any number of server groups to express tiers contained within a site.

This kind of information allows the operations team to understand critical resource dependencies and usage. It also enables automated provisioning of resources aligned with functional and non-functional requirements as well as operational patterns such as user or transaction load at any given point in time.

The application architect's job is to structure the application ito allow functional elements to be deployed on demand, just-in-time, whether this is within a single server or across multiple servers. We believe that this provides operators with the flexibility to optimize for total spend no matter which energy purchasing model is in effect. It also enables operators to clearly understand and differentiate required functionality and dependencies within applications and across applications. In this way, the deployment will meet the user requirements with minimal usage of resources.

Scale Unit-Based Architecture and Design

Earlier, we discussed the problem of systems designed around "success" — a hypothetical maximum number of users. Building an application to serve every user that will accrue over time is a sure path to waste; an oversized system also increases the risk of point failure and may be fiscally unfeasible at the outset.

To take a simple example, you can't buy a load balancer that scales dynamically. They all have a fixed number of ports. The capacity of your physical equipment becomes a scale limit that has a huge impact on cost. You can buy a 500-port balancer, but if you only need 100 ports the first year, the waste will eat the money you saved through buying the more capacious hardware. Just as you wouldn't build a wall with one huge brick, you don't want to design your IT deployments for envisioned maximum capacity.

Smart scaling requires using an approach based upon scale-units. Scale-units trigger "managed growth" based on well-defined "growth factors" or "growth drivers." Combining this approach with instrumentation and composition/factorization described earlier deployed in a just-in-time fashion, aligns resource use with the actual user demand without compromising the architecture. This approach is practiced in large-scale Internet properties.

This is where scale-unit planning comes in. A scale unit is a defined block of IT resources that represents a unit of growth. It includes everything needed to grow another step: computing capacity, network capacity, power distribution units, floor space, cooling and so on. A scale unit must support the initial resource needs while enabling you to support the necessary delta of key growth factors, be they transactions, number of users, ports, or what have you. The size of the scale unit is chosen to optimize the trade-off between resources needed immediately and those that can grow over time.

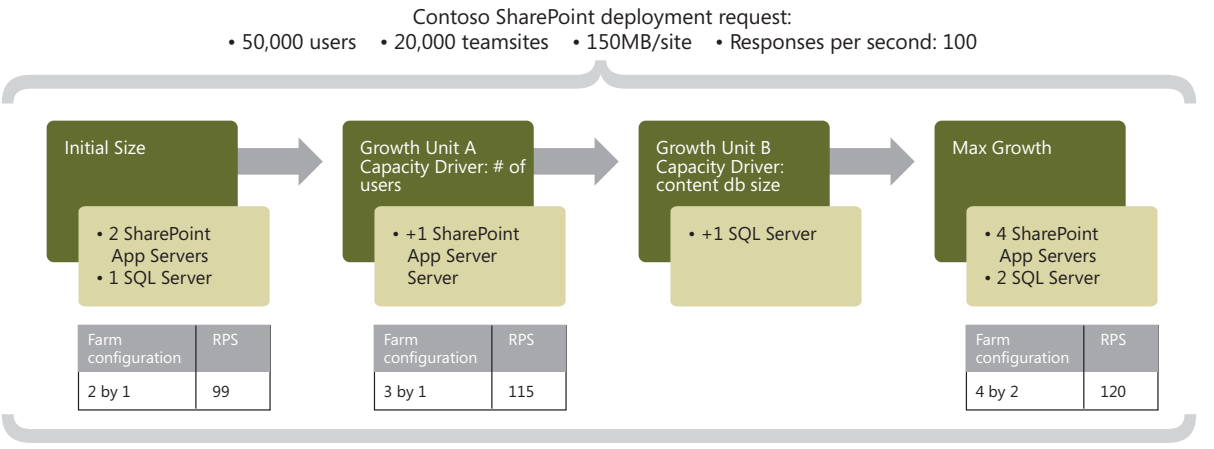
Growth drivers are generally hidden in capacity planning documents for any given product. For example — and speaking generally — the Microsoft SharePoint Server planning guide tells you to plan for resources (servers, storage, bandwidth, and so forth) based on the number and sizes of SharePoint sites, desired responses per second, and expected number of active users. These are the growth drivers for SharePoint Server. Based on the "success as a design tent" model, capacity projections are typically matched to architecture which is then transformed into resource requests and deployment. Our challenge is this: We don't want to deploy for "success." We want to deploy just-in-time to use resources most efficiently.

The design process is actually quite simple and straightforward:

1. Design the deployment of the solution based on available capacity-planning information.
2. Identify growth drivers for the solution. Ideally, the software vendor or development team will provide them. If they don't, you can dissect the capacity planning guidance and see what changes in the solution architecture are triggered based on scale (typically user traffic, request processing, "static" elements such as Web sites, content within Web sites, and so forth).
3. Identify and design appropriate scale-units for each identified growth driver. If this is your first scale-unit deployment, you might want to start with only one or two growth drivers.
4. "Partition" the design based on the scale-units that you defined.
5. Verify that the initial deployment and subsequent scale-units add up to fully functional deployments.
6. Deploy the solution with only the initial scale-unit.

Effective scale-unit deployment requires effective monitoring, deployment (provisioning) that is automated (or at least well-managed), and a clear cause-and-effect relationship between growth drivers and scale-units.

Figure 6: Scale-Units Grow



Monitoring counters in the operational configuration and monitoring environment trigger growth (or shrink) provisioning once the specific capacity driver hits 80% of speciified value:
– Growth based upon RPS (growth type A): initial size – 99 RPS; counter is set to 80 RPS
– Growth based upon content db size (growth type B): initial size – 0.8 TB; counter is set to 0.7 TB

Example

Here is an example using SharePoint. Let's assume a fictional SharePoint Server deployment for our much-beloved Contoso manufacturing company. The desired end state is supposed to be reached in 2010, as business units across the globe adopt the solution. Scale-units and growth drivers for the solution are shown in Table 1.

Figure 6 shows the scale-units that grow the deployment based on the growth drivers identified in Table 1 as well as the eventual maximum deployment.

The solution is initially deployed as an end-to-end solution with two SharePoint application server instances and one SQL Server instance. While the graphic appears to show sequential triggering, the growth drivers are independent of each other, although the deployment of one scale-unit might affect some other aspects of the deployed solution.

Putting it all together

To examine the effectiveness of this approach, let's combine some of the ideas we've discussed and revisit the model we used earlier to discuss scale-out units and capacity planning.

Look at the scale labeled "Service Units Consumed," represented by the red line on the right hand side of the graph in Figure 3. At time T-0, the data center has been built and no capacity is being used. We begin to consume the resources available in the data center and as time passes, use outstrips the capacity. The area below the purple line is used capacity. The area above the purple line is unused capacity. Effectively managing unused capacity means real power savings.

Virtualization

For application architects, virtualization is not the starting point for greener IT — it is a tool that helps efficient applications use fewer resources. At the start of capacity growth, virtualization enables us to match hardware resources more precisely to user needs. This reduces power spend by reducing provisioned but unused hardware capacity. In the future, virtualization will allow you to shut down excess resources that are already in place. Such a solution requires switchable, computer-controlled power circuitry; dynamic network configuration; and software designed to scale up and down in parts that can be composed and separated as necessary. The solution must be linked to and driven by monitoring systems that not only track application health but "understand" capacity drivers and capacity consumption. With those core elements, computing resources (and therefore power) can be throttled based on how well the solution is meeting its service level agreement (SLA). If it is performing within the necessary parameters, spare capacity can be shut down.

This is the core scenario driving a massive industry shift toward virtualization. It's amazingly complex, requiring more control and more accurate models (capacity models, consumption models, monitoring on multiple dimensions, safety margins, and power plans, to name a few) than we've ever seen in software systems.

Even if your work is not measured in units the size of data centers, the challenge remains: Use less to get the same amount of work done. This is where the architectural partitioning of applications becomes critical. Today, the problem is one of excess resources: "Find the unused ones and turn them off." It is a problem that can be solved by IT pros in the trenches. In the future, "more" will not even be an option. Only designers and architects can solve that problem — and it's one that must be solved before software is ever deployed or even built. Instead of building applications that rely on synchronicity — the "get more to get more" approach that uses virtualization to solve the problem of waste in off-peak hours — we can drive the wave to the optimization side of the resource equation by enabling the applications that are used in the business to consume resources as fully and effectively as possible. Instead of thinking ,"buy a lot of computers," constrain the problem by asking yourself how to get the job done on a third or a quarter you think is necessary. This is the world that will be a reality in the next 20 years — a world where five servers worth of software has to be virtually scheduled onto two computers worth of

Table 1: Scale Units and Growth Drivers for Example SharePoint Deployment

Growth Driver	Scale-Unit
Number of active users putting pressure on responses per second	SharePoint application server
1 TB limit on size of all content databases for a single SQL Server instance	SQL Server instance

capacity. The way to do this is to make sure you can use all of the waste — the space above the purple line, so that you are using all of the resources available all of the time — efficiently and effectively.

Futures

Store and forward, queue and resume

Architects must design applications the parts of which can function independently without losing the ability to share data and generate work product in the proper order. Returning to the example of the payroll system that is only active two days in a month, our challenge is to find a way to use those resources during the other 28 days. Instead of dedicated software configurations on fixed hardware, we need to design applications which can be stored on the virtual shelf in pieces, ready to be pulled out and activated in various combinations just-in-time.

Achieving this requires a powerful, flexible set of control software. The control software and the application have to work together to track and profile loads across functional components (such as server roles) as well as topology (such as sites and server groups). We have to track and profile static usage, as well as dynamic usage, introducing time as a key resource management dimension. Figure 7 shows a simplified load profile over a 24-hour period. Using time enables us to break the synchronicity principle and increase the service output despite a fixed resource budget.

Portfolio management

Managing individual applications is a critical step in the right direction. However, the real power is in static and dynamic resource management across the entire the portfolio of applications that provide the desired service output. The control software has to be enabled to track resource usage, identify scale-units that can be managed on a scheduled basis (that is, not running at all times) or scale up and down dynamically based on the managed load profile across all applications. Properly factored, documented, and instrumented applications are the key to this new dimension of management which will enable us to optimize resources in ways we have not yet seen in distributed environments.

Microsoft's Investment in System Center

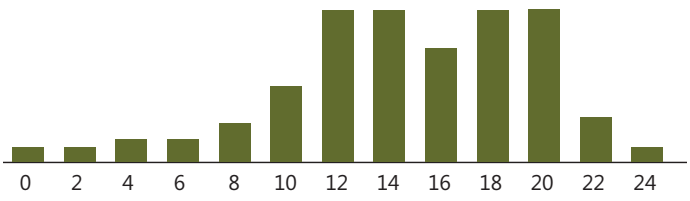
Just as we have begun componentization of the operating system, Microsoft has been pushing monitoring, software distribution, problem diagnosis, and virtualization strategies. A key investment is in management, infrastructure control, and automated operations, embodied in the Microsoft System Center product family. Today, Microsoft Operations Manager, Virtual Machine Manager, and Configuration Manager are already on a path to merge and enable dynamic and flexible control systems necessary to efficient use of resources. Looking forward, anticipate a rapid convergence on total system management solutions that let customers optimize to run more using fewer resource units — be they floor space, cooling, or compute capacity.

How does this fit with the building block approach? System Center will control the work within the dynamic operating environment. From a design and software development perspective, a Management Pack — the System Center control metadata — will define what to look for (monitoring data provided by application instrumentation that exposes key capacity and resource consumption information) and provide the formulae that determine what health or capacity impact these measures have on the critical system components. System Center will know what is running and how well and balance workloads over time.

Management packs will become a key control component for each of the compartmentalized elements of your systems. The management pack architecture is simple — an XML file that follows a particular set of schemata — but powerful because it is metadata that describes the condition and health of factored components. What works for large synchronous applications also works well for factored applications. In fact, the best management pack designs today — for example, those for Windows Server 2008, SharePoint, and Dynamics applications—exhibit layering and componentization out of the gate.

As System Center evolves, management packs will become core parts of business applications, infrastructure, and foundation elements in the stack. Since System Center is already capable of monitoring computer health for Linux and Unix servers, the foundation is laid for a revolution in application flexibility — one that will let you manage your systems with substantially fewer computing resources, but still lets you choose the operating system and software suppliers. No matter what technology you build on, as long as you enable the shift to fractional application design, you or someone else will be able to describe the monitoring and capacity utilization surface as a management pack. That will enable your applications to be managed economically and efficiently, unbound by the hardware constraints that limit application efficiency today.

Figure 7: Simplified Load Profile over 24 Hours



Conclusion

Application architects can design distributed systems for more effective resource management. First, we can align well-defined optimization goals with factorization. Scale-units enable us to group and align well-factored solutions with minimized deployments without compromising the overall deployment architecture. Scale-units combined with instrumentation will enable control software like System Center to manage a portfolio of applications based on the established optimization goal and approach. These capabilities are not far off. In a world with absolute limits on power consumption, the sooner we can bring them to bear, the better.

Resources

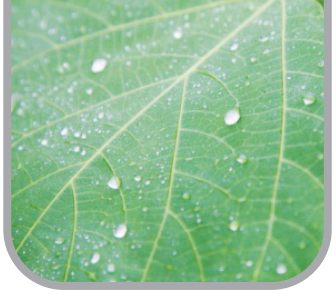
Measuring National Power in the Postindustrial Age, Ashley J. Tellis et. al, RAND Corporation, 2000
http://www.rand.org/pubs/monograph_reports/MR1110/

Wikipedia - Theory of Constraints
http://en.wikipedia.org/wiki/Theory_of_Constraints

About the Authors

Dan Rogers joined Microsoft in 1998 and has worked on a number of products including BizTalk, Visual Studio and MSN. He currently works in the System Center product team and focuses on monitoring strategies for enabling dynamic IT solutions. While in MSN, he helped design monitoring solutions for large scale dynamic properties such as popular storage and email services, and been a member of a number of projects involving lights out operations for a global data center initiative.

Ulrich Homann is the lead solutions architect in the strategic projects group of Microsoft worldwide services. As part of the office of the CTO, this group is responsible for the management and enterprise architecture of key strategic projects all over the world.



Architecture Journal Profile: Udi Dahan

The Architect Most Valuable Professional (MVP) program at Microsoft has recently been relaunched after a short recess of a few months. Our featured guest for this issue's profile has been recognized as a Microsoft Solutions Architect MVP many times over: Udi Dahan is a free thinker, SOA and IT authority, and celebrated "simplist"

AJ: Who are you, and what do you do?

UD: My name is Udi Dahan, and I'm an independent consultant focusing primarily on enterprise development and service-oriented architecture. While the majority of my work is with Fortune 500 enterprises on large, multi-year projects, I also help smaller companies in their technology choices and architectural decisions. On top of my regular consulting work, I speak at international developer conferences, lead an open source project (www.nServiceBus.com), and run a fairly popular blog.

AJ: Many of our readers know you as "The Software Simplist." Can you tell us how you went from Udi Dahan the individual to Udi Dahan the Software Simplist?

UD: When I was working at my last consulting firm, as a part of the rebranding effort, each senior consultant was to be given a unique title, something that would differentiate them from other consultants. I was pretty busy when all of this was going on, so I asked one of my friends in the marketing department to find something for me. My only request was that it be simple. When he came back after the weekend with "The Software Simplist," I kind of liked it and it grew on me. After that, it just stuck.

AJ: So tell us about your origins as an architect. Did you start your IT career as a developer under the supervision of a lead developer or a project leader? What factors determined your destiny as an "architect"?

UD: I guess I started my IT career at the age of 8, when I wrote my first program. My dad worked at the local university in the information systems department, so we had a computer at home, which wasn't very common at the time. When I asked him if he could get me some games for it, he promptly took me to the programming section at the library and gently explained that I'd be writing my own games. From that point on, I never really stopped programming.

After high school I got my first paid programming job working for one of the faculty at the university. There was no lead developer, just a professor telling me what he wanted the application to do, and me doing it. That direct connection to what users want, need, and think has permeated the rest of my career and helped me develop the business and communication skills that are critical to the success of an architect. Since then, I've always been working in close collaboration with end users looking for ways to make systems work for them, rather than the other way round. That strong connection to the business was definitely a defining factor.

AJ: What advice would you share to those who want to be recognized for his/her abilities as an architect?

UD: First and foremost, you have to know your stuff. Beyond reading endlessly on software architecture, design, patterns, technology, and development methodologies, you need to actually do it, and that means coding as well.

To be recognized, one needs to be successful, and that includes being perceived as successful. You can't escape from the organizational politics, and since architects rarely have any real organizational power, success means working with and through other people in the organization. Often, this means giving up some ego in order to create win-win scenarios that all stakeholders can buy in to.

AJ: Looks like it's not enough to be an architect but we must also make others take notice. In that sense, you have done an admirable job. You've long been publicly acknowledged — you are frequently enlisted to speak at conferences (Dr. Dobb's, TechEd's everywhere, QCon, just to mention a few). Do you have any advice to share with other architects who aspire to but don't know how to become a luminary?

UD: "Luminary" is quite the title to live up to and I'm not sure I'm there yet. I can say that if you want to be publicly recognized you have to do publicly visible activities like writing for magazines, blogging, and speaking at conferences. Interestingly enough, the written and oral communication skills that help architects be successful in their own organization are the same skills that will drive their recognition publicly.

AJ: The architect role requires understanding current and future technical trends to get the best for the business benefit — how do you stay up to date?

UD: On top of the voracious reading I mentioned, going to conferences is just as critical. Hearing from industry leaders about which trends

"I started my IT career at the age of 8, when I wrote my first program....We had a computer at home, which wasn't very common at the time. When I asked [my dad] if he could get me some games for it, he gently explained that I'd be writing my own games. From that point on, I never really stopped programming."

are important helps, but talking to other people and getting their perspective on what works and when is truly invaluable. As a speaker at many conferences, I'm in a better position to hear this from other speakers, experts in their field. Also, my consulting practice lets me see the direction many verticals are moving in and which technical trends will best serve them. Regardless, however far along one is in their career, many of us are smarter than any of us — your network will give you more knowledge and insight than you could possibly learn on your own.

AJ: That's true, however — does this happen to you? — it can be frustrating when you've tried to keep up-to-date on the latest technologies and trends, but after investing the considerable effort on the latest-and-greatest technologies (coding, watching webcasts, testing demos, and so forth), a dozen of new ones emerge from the pipeline (just look back to the last PDC). Do you have any recipe to efficiently deal with such avalanche?

UD: There is indeed a continuous flood of information coming at developers today. Never has so much information been so readily available and accessible to so many. It turns out, though, that much of that information is about how something works, less about what problem it is intended to solve, and much less about which problems it isn't suited for. As an architect looking at choosing the right tool for the right job, I focus a lot more on the last two. Often, I go digging to find out why a tool isn't a good fit for a job. So, I guess my recipe is to mostly ignore the avalanche and tap my network to find out the stuff the avalanche won't tell you.

AJ: Name the most important person you have ever met in this industry — What made him/her so important?

UD: I'm really in no position to judge relative importance, but I'd have to say the person that had the biggest impact on me was Charles Simonyi. I had the pleasure of cornering Charles over dinner at the Software Architecture Workshop in the northern mountains of Italy a couple of years ago. This was before he became an astronaut, but still, I wasn't exactly in the habit of eating dinner with billionaires. During those three hours, Charles shared with me his decades of experience and perspective, planting the seeds for much of my intention-focused design practices. Recently, I've started presenting on these practices at conferences, showing how well they handle diverse concerns like validation, persistence, and service layers. It never ceases to amaze me — how much practical value can grow from the seed of a quality idea.

AJ: Is there anything you did that you'd like to do differently if you could turn the clock back on your career?

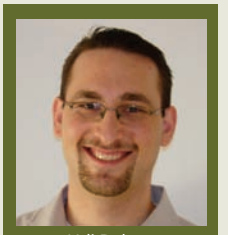
UD: One role that, in retrospect, I regret not taking was that of a tester. When I was just starting out, I had the choice of taking either a testing or development position. At the time, I viewed testing as something beneath development — something junior people had to do before being able to move into development. Naturally, I took the position as a dev. Today, my in-depth understanding of development serves me well, helping me optimize the work of the dozens and hundreds of developers on my projects. Had I spent more time in the testing trenches, I feel that I might have been able to do more for the testers I work with today.

AJ: What does Udi the Architect's future look like? What do you hope to accomplish in the next few years?

UD: As the Nobel Prize-winning physicist Niels Bohr said, "Prediction is very difficult, especially if it's about the future." I'll probably continue with my blog, speaking engagements, and the big three to five year projects I'm consulting on today. I hope to take my open source project, NServiceBus, to the level of its peers in the Java space. Also, I'm flirting with the idea of writing a book (after having contributed a couple of chapters in others), but that'll have to wait until my two-year-old lets me get a full night's sleep.

AJ: Thank you very much, Udi

UD: My pleasure.



Udi Dahan

Udi Dahan is a Connected Technologies Advisor working with Microsoft on WCF, WF, and Oslo. He also serves as a member of the Patterns & Practices Advisory Board for the CAB and Prism, and has received Microsoft's Most Valuable Professional award for four years running. He provides clients all over the world with training, mentoring, and high-end architecture consulting services, specializing in service-oriented, scalable, and secure .NET architecture design.

Udi is a member of the European Speakers Bureau of the International .NET Association, an author and trainer for the International Association of Software Architects and a founding member of its Architect Training Committee. He is also a Dr. Dobb's sponsored expert on Web Services, SOA, and XML. His blog covers service-oriented architecture and enterprise software development and can be found at http://www.UdiDahan.com.

Resources

NServiceBus
http://www.nservicebus.com/

Udi Dahan - the Software Simplist: a blog on Service-Oriented Architecture and Enterprise Software Development
http://www.udidahan.com

Wikipedia - Charles Simonyi
http://en.wikipedia.org/wiki/Charles_Simonyi

Profiling Energy Usage for Efficient Consumption

by Rajesh Chheda, Dan Shookowsky, Steve Stefanovich, and Joe Toscano

Summary

When measuring the effectiveness of application architecture, there are several key criteria by which it can be measured. First and foremost, is the architecture's ability to fulfill the requirements. These requirements often include specific, measurable items relating to cost, scalability, testability, and so forth. In the future, two new measurable criteria will become critical to evaluating application architectures: energy consumption/cost and environmental impact. But as we will see, the future is now.

The location: your corporate server room.

The time: present day.

The event: an unexpected visit from the "Software Architect of the Future."

There you sit, monitoring the performance of your latest line-of-business application freshly rolled out just 24 hours ago, when a strange visitor strolls in. She begins asking you about your enterprise architecture. Instead of calling security, you decide this is a chance to show off how cutting edge your organization is. You hit all the buzzwords: C#, SQL Server, WPF, WCF, and SilverLight. It scales up and out, is accessible from mobile devices, it slices and it dices. "Excellent," the Architect replies, "but what does your EUP look like?"

"My EUP? What's that?" The Architect explains that in the future, an Energy Usage Profile (EUP) is a fundamental part of every application design — through the use of a set of Energy Usage Profiles (EUPs), organizations of the future measure the ongoing energy costs and environmental impact of specific applications within the enterprise. "Sit down," she intones, "and I'll tell you all about it."

The Math

Before diving into specifics, a basic discussion of energy usage is in order. When measuring the electrical usage of any electronic component, simple math can be employed to assign a cost to that component. Let's begin with an example: the venerable 60 watt (W) incandescent lightbulb (common in your time but widely regarded as legacy hardware in the future). The electric company measures energy consumption in kilowatt-hours (kWh). Using our lightbulb as an example, if we were to leave it on for one hour, we would have consumed 60 watt-hours (or .06 kWh) of power (60/1000 = .06). Now say, for example, that the electric company charges $0.0897 per kilowatt-hour (excluding delivery charges, capacity charges, taxes, and all other fees), that means that the lightbulb costs roughly $0.0054 per hour to run (0.06 * 0.0897 = 0.0054). That may not seem like much but, if we were to leave that lightbulb on 24 hours a day, seven days a week, it would cost us $47.15 to run annually (0.0054 * 24 * 365 = 47.15). Now consider that each kilowatt-hour of electricity used also contributes approximately 2.3 pounds of CO_2 to the atmosphere. So, using a few simple calculations we can state an annual cost ($47.15) as well as an annual environmental impact (1,208 pounds of CO_2) for our little lightbulb. Going forward, we will be employing these formulae to calculate energy usage costs and environmental impact:

kWh = Watts / 1000
Annual Energy Cost = kWh * 24 * 365 * cost per kWh
Annual CO_2 Emissions = kWh * 24 * 365 * pounds of CO_2 per kWh

For the purposes of this discussion, we will use the values of $0.0897 for cost per kWh and 2.3 for pounds of CO_2 per kWh. These values may vary depending on where you live and how your electricity is generated.

Developing an Energy Usage Profile for the Hardware

Unfortunately we're not talking about lightbulbs, we're talking about servers. So how does the energy usage of the average server compare to that of a 60 W lightbulb? Well, unless you're running your company on laptops, it's much worse. According to research published by Dell and Principled Technologies, a single Dell PowerEdge M600 blade server consumes an average of 383.75 W when idle and 454.39 W under stress (see Table 1). Keep in mind that this figure can go up or down depending on workload. Using that baseline to recalculate our numbers, that one server costs $301.54 per year to run. It also produces 7,731.8 pounds of CO_2. The picture gets even bleaker when you consider how many servers

Table 1: Sample Hardware Energy Usage Profile (in watts)

Component	Idle Power Usage	Average Power Usage	Maximum Power Usage
Server 1	383.75	454.39	600 (Estimated)
CPU	40.8		130
HDD	14.35		17
DIMM1	3		3
DIMM2	3		3
Video	18.3		25.6
Network Card	4.95		4.95
CD/DVD	2		18

the typical application uses. For example, if you run SharePoint, are you running it on a farm? Maybe a farm with two Web front-end servers, an index server, and a database server? Suddenly the annual electricity cost increases to $1,206.16 and the CO_2 produced increases to over 30,927 pounds. The cost and impact calculations should be directly included in any calculations of return on investment for an application.

Using the previously mentioned formula for Annual Energy Cost, we can build an Energy Cost Profile for the hardware, as shown in Table 2. Finally, we can also build an Energy Impact Profile for the hardware, as shown in Table 3.

The numbers for specific components are not easily measured but can be obtained from manufacturers' Web sites and Web sites that perform independent hardware research. The server's energy consumption is much easier to measure. Several low-cost devices are available to monitor energy consumption. The important thing is to build an overall picture of how much energy the server uses at idle and under stress. The energy usage of individual components does not need to be exact, but is important because it provides a ranking system for energy consumers within the server. In the example above, the CPU is clearly the largest consumer of energy.

Using the EUP for the hardware, we can draw some general conclusions. Clearly, CPU usage is the most expensive resource in terms of actual cost and environmental impact. Memory usage has a minimal cost at best. Hard disk usage has minimal cost. The gist is that if we are going to attempt to optimize our infrastructure and application to minimize energy usage, the CPU should be the primary target.

Table 2: Sample Hardware Energy Cost Profile (in USD)

Component	Idle Annual Cost	Average Annual Cost	Peak Annual Cost
Server 1	$301.54	$357.41	$471.95
CPU	$32.06		$102.15
HDD	$11.28		$13.36
DIMM1	$2.36		$2.36
DIMM2	$2.36		$2.36
Video	$14.38		$20.12
Network Card	$3.89		$3.89
CD/DVD	$1.57		$14.14

Table 3: Sample Hardware Energy Impact Profile (in pounds of CO_2)

Component	Idle Annual Cost	Average Annual Cost	Peak Annual Cost
Server 1	7,731.8	9,164.46	12,101.22
CPU	822.04		2,619.24
HDD	289.12		342.52
DIMM1	60.44		60.44
DIMM2	60.44		60.44
Video	368.70		515.79
Network Card	99.73		99.73
CD/DVD	40.30		362.66

Developing an Energy Usage Profile for the Application

Dealing in averages may be fine if you're analyzing a large data center but what about the energy usage of a specific application's architecture? Average power consumption is not good enough. We want to know how much power a single application uses and, while we cannot get to the exact number, we can get pretty close. The first step is building an energy usage profile for the hardware.

As mentioned previously, the manufacturers' Web sites are a good place to start. While the manufacturer's figures can give you a good approximation for best and worst case scenarios for the hardware, it cannot tell you how much energy your particular application uses. In this case, we need to measure. Luckily, inexpensive products are available that will help measure the energy consumption of your application. These products, used with a good set of application load tests, can make the energy consumption picture much clearer. The key is to build a set of load tests that reflect both the average and peak loads for your application.

The idle consumption can be used to calculate a baseline cost for the application. When doing nothing, a three-server application has an idle energy consumption of 354 W (.345 kWh). The hourly cost is $0.025875 and the annual cost is $226.67. Through our EUP, we can thus calculate the baseline annual cost of the application. The measurements taken during average and peak application loads can then be used to calculate costs and impacts when the application is running at those levels (see Tables 4-6).

Once an application-level EUP has been created, specific areas of the application can be targeted for improvement.

Remember that what is high priority in terms of performance improvement may not always be high priority for energy usage reduction. For example, an operation that does a long write operation to a physical disk might be seen as a high priority performance issue. In terms of energy usage, however, this operation might be low priority because the physical disk consumes less energy than the CPU. An operation that utilizes a large amount of CPU time might be a higher priority. This is not to say that CPU-intensive operations should be replaced at the cost of application performance. It simply gives us a ranking system for determining which components of the application can be optimized to realize the largest cost savings. The hardware EUP will tell you which physical components of the hardware carry the most cost and impact. Using hardware EUP, we know that CPU utilization carries the largest

Table 4: Sample Application Energy Usage Profile (in watts)

Server	Idle Power Usage	Power Usage at Average Load	Power Usage at Peak Load
Server 1 – Application	304	425	525
Server 2 – Database	304	425	525
Server 3 – Web	304	425	525
Total Power Usage	912	1,275	1,575

Table 5: Sample Application Energy Usage Cost Profile (in USD)

Server	Idle Power Usage	Power Usage at Average Load	Power Usage at Peak Load
Server 1 – Application	$238.87	$333.95	$412.53
Server 2 – Database	$238.87	$333.95	$412.53
Server 3 – Web	$238.87	$333.95	$412.53
Total Cost	$716.61	$1,001.85	$1,237.59

Table 6: Sample Application Energy Usage Cost Profile (in pounds CO_2)

Server	Idle Power Usage	Power Usage at Average Load	Power Usage at Peak Load
Server 1 – Application	6,125	8,562.9	10,577.7
Server 2 – Database	6,125	8,562.9	10,577.7
Server 3 – Web	6,125	8,562.9	10,577.7
Total Impact	18,375	25,688.7	31,733.1

impact in terms of power usage, and should therefore be our first target.

Tools such as Visual Studio 2008 Profiler can be used to determine precise CPU usage for specific components of the application. These tracking statistics can then be used to attack the high cost portions of the application. In other words, find the most expensive operations and optimize them to reduce CPU usage. The ultimate goal is to lower resource usage to the point where multiple applications can be hosted on the same set of servers. Server sharing can reduce the energy footprint of all the applications involved.

Developing an Energy Usage Profile for the Operating System

Application-based optimizations are not the only way to save energy. Optimizing the operating system is just as important. The base hardware EUP also provides us with an idle operating system EUP. Looking at the data collected for physical servers when idle and at peak loads, we can see that a significant amount of energy is wasted on the system idle process. This wasted energy can be reclaimed through operating system virtualization. Virtualization allows the host machine to run at approximately 80 percent of peak processor utilization with fractionally increased power requirements. Table 7 translates these potential benefits into numbers.

Each physical server replaced with a virtual equivalent represents a significant savings, both in upfront costs as well as with respect to the environment. To put the carbon footprint in perspective, the average car produces 20 pounds of $C0_2$ per gallon of gas consumed. To maximize the number of virtual guest machines on a single host, optimization is a key requirement. This optimization includes minimizing disk and network utilization, as well as reducing the memory footprint.

The first step in tuning the virtual guest is to disable or deactivate unneeded services. Depending on the type of guest operating system and its purpose, there may be a number of unnecessary services running by default. Table 8 lists services which may be safely shutdown depending on your circumstances.

In addition to the services listed above, you should eliminate screen savers and evaluate your event logging requirements for each guest operating system. This will avoid wasted processor cycles and disk activity. Minimizing disk activity is essential to both performance and energy efficiency. For that reason, you should use fixed size virtual hard disks to avoid the need to resize the Virtual Hard Disk (VHD) file. If you don't require extensive logging, you can further reduce disk activity at the expense of data necessary for later analysis.

If you have the necessary RAM, you can use a RAM drive to cache frequently accessed data. A RAM drive trades increased memory utilization for improved performance. The advantage is that RAM drives don't have to hit the physical disk, providing energy efficiency in addition to the aforementioned performance.

Look closely at the network utilization of your applications with a network monitoring tool to locate and eliminate chatty protocols and unnecessary network communication. Wireshark is an effective and freely available tool for network analysis.

On the host server side, usage of the Windows Server 2008 Server Core installation will minimize the files and services used by the host operating system.

These optimizations reduce the amount of work done by the guest operating system as well as by the host. Less work translates into lower energy consumption.

Developing an External Energy Usage Profile

The impact of your application goes beyond CPU cycles and megabytes. The way end users interact with the application has a very real cost and impact. The actions taken within your application that are external to your design are measured in the External EUP. Think through the way end users will utilize a given application and try to anticipate the impact of those interactions. Following are examples of actions that should be tracked in an External EUP:

- Printing: Are users likely to print information? If so, how much? Are there ways to optimize screen layout to minimize the likelihood of printing?
- Information Retrieval: Are statements, invoices, or other materials mailed to users? Is there a way to present that information within the system?
- Updates: Do users need to leave their systems on to receive updates? If so, is there a way for the application to update itself while being used?
- Emailing: Purging bad email addresses eliminates energy wasted attempting to send messages.

While harder to quantify, these impacts are important because simple measures can be taken to mitigate them. For example, adding an on-screen print view might virtually eliminate the cost of paper, electrical power and toner usage for printers as well as the carbon production from both.

Table 7: Potential Benefits

	Annual Cost
Average Cost of Server:	$6,500
Average Power Cost:	$301
CO_2 Produced	7,731.8 pounds annually

Table 8: Services that may be disabled in a virtual machine

Name	Description
Automatic Updates	This service is not necessary if you have disabled Windows Update in favor of manual patch management.
Background Intelligent Transfer Service	This service is not necessary if you have disabled Windows Update in favor of manual patch management.
Clipbook	This has nothing to do with the clipboard or with the ability to cut and paste.
Error Reporting Service	This service can be safely disabled on a server.
Help and Support	
Indexing Service	This service may be required depending on the server's purpose.
Messenger	Messenger services are unnecessary on servers running in lights-out mode.
Portable Media Serial number	Music players are not necessary on a server.
Print Spooler	Unnecessary if you aren't printing from the server.
Smart Card	Not needed if smart cards aren't being used.
Task Scheduler	Unnecessary on most servers.
Themes	Themes are unnecessary on a server machine.
Uninterruptible Power Supply	The host server can manage UPS. This service is not needed in a virtual machine.
Web Client	
Windows Audio	Audio is unnecessary on a server.
Wireless Con guration	This service is unnecessary on both physical and virtual servers.

Conclusions

It is not just the Architect of the Future that can use the methods described in this article. You can (and should) consider the energy usage when designing an application. Part of being an Architect is understanding how every aspect of your application functions. Energy usage is another key aspect and an energy usage profile is the tool for understanding it. Having a useful energy usage profile means:

- Understanding your hardware and its energy usage.
- Understanding your operating system and its energy usage.
- Understanding your application and its energy usage.
- Understanding the external impacts of your application.

Together, these areas of understanding combine to give you a total energy usage profile that you can act upon. More importantly, they give you a barometer for tracking the real cost of your application and the real gain of your optimization. The future of application architecture is looking greener.

Resources

Average Retail Price of Electricity to Ultimate Customers by End-Use Sector, by State, Energy Information Administration, October 3, 2008 http://www.eia.doe.gov/cneaf/electricity/epm/table5_6_a.html

Frequently Asked Global Change Questions, Carbon Dioxide Information Analysis Center http://cdiac.ornl.gov/pns/faq.html

Performance and Power Consumption on Dell, IBM, and HP Blade Servers, Principled Technologies, December 2007 http://www.dell.com/downloads/global/products/pedge/en/pe_blades_specjbb2005.pdf

About the Authors

Rajesh Chheda is a consultant at RDA with more than 10 years of experience designing and developing business solutions using Microsoft tools and technologies. He has worked on software solutions for real estate finance, lending, procurement. His current focus is on designing, customizing and deploying collaboration solutions for his clients using Microsoft SharePoint technology. He can be reached at chheda@rdacorp.com.

Dan Shookowsky is a senior software engineer with RDA. He has 15 years of experience in IT consulting with a significant focus on validated applications for the pharmaceutical industry running on Microsoft Windows platforms. He can be reached at shookowsky@rdacorp.com.

Steve Stefanovich has spent more than a decade with RDA as a developer and principal architect focusing on custom application development using Microsoft .Net and SharePoint technologies. Steve is a frequent contributor to RDA's Architecture and Collaboration blogs. He can be reached at stefanovich@rdacorp.com.

Joe Toscano currently is a SQL Server BI specialist with RDA and has worked with SQL Server since version 6.5 and with Sybase prior to that. Joe has both published materials and presented locally for the Philadelphia SQL Server User Group and numerous .Net Code Camps sessions, and internationally at The Professional Association for SQL

Project Genome: Wireless Sensor Network for Data Center Cooling

by Jie Liu, Feng Zhao, Jeff O'Reilly, Amaya Souarez, Michael Manos, Chieh-Jan Mike Liang, and Andreas Terzis

Summary

The IT industry is the one of the fastest growing sectors of the U.S. economy in terms of its energy consumption. According to a 2007 EPA report, U.S. data centers alone consumed 61 billion kWh in 2006 — enough energy to power 5.8 million average households. Even under conservative estimates, IT energy consumption is projected to double by 2011. Reducing data center energy consumption is a pressing issue for the entire IT industry now and into the future. In this article, we argue that dense and real-time environmental monitoring systems are needed to improve the energy efficiency of IT facilities.

Only a fraction of the electricity consumed by a data center actually powers IT equipment such as servers and networking devices. The rest is used by various environmental control systems such as Computer Room Air Conditioning (CRAC), water chillers, and (de-)humidifiers, or simply lost during the power delivery and conversion process. The data center Power Usage Effectiveness (PUE), defined as the ratio of the total facility power consumption over the power used by the IT equipment, is a metric used by The Green Grid to measure a data center's "overhead." A higher figure indicates greater energy "overhead" while a lower figure indicates a more efficient facility. For example, average data center has PUE ≈ 2, indicating that half of the total energy is used for IT equipment. However, the PUE can be as high as 3.5.

One key reason for the high PUE is the lack of visibility in the data center operating conditions. Conventional wisdom dictates that IT equipment need excessive cooling to operate reliably, so the AC systems in many data centers use very low set points and very high fan speed, to reduce the danger of creating any potential hot spot. Furthermore, when servers issue thermal alarms, data center operators have limited means to diagnose the problem and to make informed decisions other than further decreasing the CRAC's temperature settings.

Given the data centers' complex airflow and thermodynamics, dense and real-time environmental monitoring systems are necessary to improve their energy efficiency. The data these systems collect can help data center operators troubleshoot thermo-alarms, make intelligent decisions on rack layout and server deployments, and innovate on facility management. The data can be particularly useful as data centers start to require more sophisticated cooling control to accommodate environmental and workload changes. Air-side economizers bring in outside air for cooling. Dynamic server provisioning strategies, such as those presented by Chen et. al at NSDI 2008 (see Resources), can turn on or shut down a large number of servers following load fluctuation. Both techniques are effective ways to reduce data center total energy consumption, but variation in heat distribution across space and time may also cause thermo-instability.

Wireless sensor network (WSN) technology is an ideal candidate for this monitoring task as it is low-cost, nonintrusive, can provide wide coverage, and can be easily repurposed. Wireless sensors require no additional network and facility infrastructure in an already complicated data center IT environment. Compared to sensors on motherboards, external sensors are less sensitive to CPU or disk activities, thus the collected data is less noisy and is easier to understand.

At the same time, data center monitoring introduces new challenges to wireless sensor networks. A data center can have several adjacent server colocation rooms. Each colocation room can have several hundred racks. In practice, we have observed up to 5°C temperature variation across a couple of meters. Multiple sensing points per rack means thousands of sensors in the facility. Not only is the size of the network large, the network density can also be high. Dozens of sensors are within the one hop communication range of the radio (10 to 50 meters for IEEE 802.15.4 radios, for example), leading to high packet collision probabilities. In stark contrast to the scale and reliability requirements of the data center monitoring application, current wireless sensor network deployments comprise tens to hundreds of motes and achieve 20 ~ 60% data yields.

This paper presents the architecture and implementation of the Microsoft Research Data Center Genome (DC Genome) system, with a focus on RACNet, a large-scale sensor network for high-fidelity data center environmental monitoring. The overarching goal of the project is to

Figure 1: An illustration of the cross section of a data center. Cold air is blown from floor vents, while hot air rises from hot aisles. Mixed air eventually returns to the CRAC where it is cooled with the help of chilled water, and the cycle repeats.

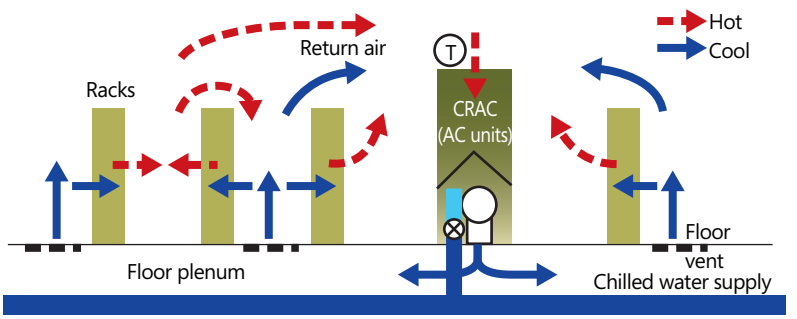
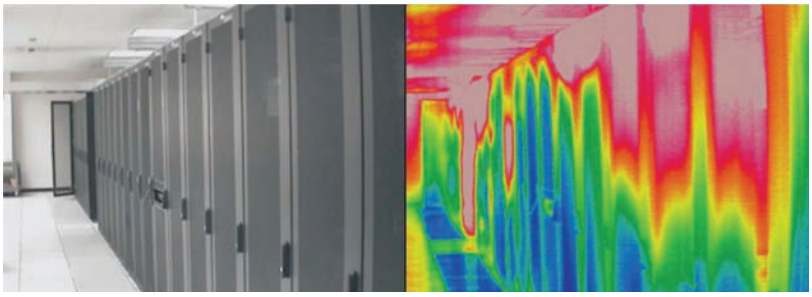


Figure 2: The thermal image of a cold aisle in a data center. The infrared thermal image shows significant variations on intake air temperature across racks and at different heights.



understand how energy is consumed in data centers as a function of facility design, cooling supply, server hardware, and workload distribution through data collection and analysis, and then to use this understanding to optimize and control data center resources. We report results from ~700 sensors deployed in a multi-mega-watt data center. This is one of the largest sensor networks in production use. Contrary to common belief that wireless sensor networks cannot maintain high data yield, RACNet provides over 99 percent data reliability, with 90 percent of the data being collected under the current soft real-time requirement of 30 seconds.

Data Center Cooling Management Background

There are many data center designs, from ad hoc server cabinets to dedicated containers. However, most professional data centers use a cold-aisle-hot-aisle cooling design. Figure 1 shows the cross section of a data center room that follows this design. Server racks are installed on a raised floor in aisles. Cool air is blown by the CRAC system into the sub-floor and vented back up to the servers through perforated floor tiles. The aisles with these vents are called cold aisles. Typically, servers in the racks draw cool air from the front, and blow hot exhaust air to the back in hot aisles. To effectively use the cool air, servers are arranged face-to-face in cold aisles. As Figure 1 illustrates, cool and hot air eventually mixes near the ceiling and is drawn back into the CRAC. In the CRAC, the mixed exhaust air exchanges heat with chilled water, supplied by a chilled water pipe from water chillers outside of the facility. Usually, there is a temperature sensor at the CRAC's air intake. The chill water valve opening and (sometimes) CRAC fan speed are controlled to regulate that temperature to a set point.

Data center operators have to balance two competing goals: minimizing the energy the CRAC consumes while at the same time ensuring that server operation is not negatively affected by high temperatures. However, setting the CRAC's parameters is a non-trivial task, because the airflow and thermodynamics of a data center can be fairly complicated. The underlying reason is that heat distribution depends on many factors such as chilled water temperature, CRAC fan speed, the distances between racks and the CRACs, rack layout, server types, and server workload. Figure 2 illustrates the end result of the complex interplay of all these factors for a particular data center cold aisle. The thermal image shows that the temperature across racks and across different heights of the same rack varies significantly. Heat distribution patterns also change over time. Without visibility into the patterns of heat distribution, data center operators have little choice but to over-cool the entire data center.

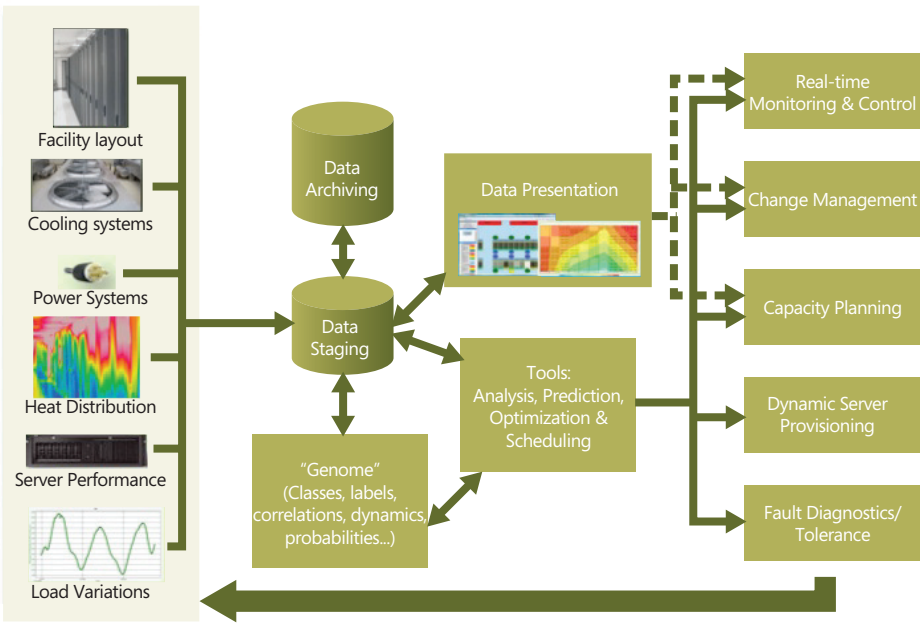
Some data centers use Computational Fluid Dynamics (CFD) simulations to estimate heat distribution and guide their cooling management strategies. Such CFD simulations are useful, particularly during a data center's design phase. They provide guidelines about room size, ceiling height, and equipment density. However, there are limitations to their usefulness: Accurate thermal models for computing devices are difficult to obtain; as soon as the rack layout or server types change, the current CFD model becomes obsolete; and updating CFD models is a time-consuming and expensive process.

DC Genome System Overview

Figure 3 depicts the architecture of the DC Genome system. Both the physical and cyber properties of a data center are measured to produce models and tools for facility management and performance optimization. Key components include:

- *Facility layout:* The rack, CRAC, and power distribution layout not only provide a basis for data presentation, but also affect cooling efficiency and, ultimately, data center capacity.
- *Cooling system:* The cooling system includes equipment such as the CRAC, water chillers, air economizers, and (de-)humidifier which are typically monitored by the building management system through a Supervisory Control and Data Acquisition (SCADA) system. The cooling

Figure 3: Overall architecture for the Data Center Genome system. Data collected from physical and cyber systems in data centers is correlated and analyzed to provide models and tools for data center management and performance optimization.



equipment consumes a majority of the non-critical electrical load (IT equipment is the critical load) of a data center. Other factors such as outside weather conditions can also affect cooling efficiency.

- *Power system:* Besides non-critical power consumed by the cooling and power distribution system, detailed monitoring of the power consumed by various IT equipment is essential.
- *Server performance:* Server activities are typically represented by the utilization of key components such as processors, disks, memories, and network card. Measuring these performance counters is key to understanding how heat is generated by various servers.
- *Load variation:* Server and network load can usually be measured by the network activities for online service hosting. With application-level knowledge, more meaningful indicators of system load, such as queries per second or concurrent users, can be derived.
- *Environmental conditions:* Physical properties, such as temperature distribution, have traditionally been difficult to collect at a fine granularity. The RACNet system tackles this key challenge.

Data collected from various sources can be used to build models that correlate the physical and performance parameters. Thus derived, the "Genome" of data centers is a rich family of models, being useful for various purposes. Deriving and applying these models relies on building algorithms and tools for analysis, classification, prediction, optimization, and scheduling. The data and tools can be used by data center operators, facility managers, and decision makers to perform various tasks, such as:

- *Real-time monitoring and control:* Examples include resolving thermal alarms, discovering and mitigating hot spots, and adaptive cooling control.
- *Change management:* Given a small number of servers to be deployed, a data center operator can make informed decisions about their placement depending on the available space, extra power, and sufficient cooling.
- *Capacity planning:* Given a load growth model and an understanding of resource dependencies, one can analyze the capacity utilization over various dimensions to decide whether to install more servers into existing data centers, to upgrade server hardware, or to build new data centers to meet future business need.
- *Dynamic server provisioning and load distribution:* Server load can vary significantly over time. The traditional philosophy of static server provisioning and even load distribution will cause the data center to run at the worst-case scenario. Recent studies show significant energy saving benefits by consolidating servers and load. Controlling air cooling precisely to meet dynamic critical power variations is difficult, but the inverse strategy of distributing load according to cooling efficiency is promising.
- *Fault diagnostics and fault tolerance:* Many hardware faults in data centers are caused by either long term stressing or an abrupt changes in operating conditions. On the other hand, modern software architecture can tolerate significant hardware failures without sacrificing software reliability or user experience. This is changing the game of data center reliability. One should consider the total cost of ownership including both acquiring hardware and maintaining their operating conditions.

Figure 4: MSR Genomotes with the relative size of a U.S. quarter; master mote (left) and slave sensor (right).

In the rest of the article, we focus on RACNet, the wireless sensor network aspect of the DC Genome system. It fills in a missing piece in holistic data center management.

RACNet Sensor Network

The design of RACNet faces several technical challenges that must be resolved in order to achieve high-fidelity environmental monitoring across an entire data center.

- *Low cost of ownership:* There may be thousands of sensing points in a data center. The cheaper we can implement the system — in terms of hardware, infrastructure, installation labor, and maintenance — the more likely the technology will be adopted.
- *High data fidelity:* The DC Genome system relies on continuous data streams from the sensors for high-level modeling, analysis, and decision making. We set 30-second sampling rates on temperature and humidity sensing, and we require 99 percent data yield. To facilitate real-time monitoring and control, we also require over 90 percent data to be received by the deadline (when the next samples are taken).
- *Seamless integration:* Environmental sensors are organic parts of the overall data center management system. The sensor network should be integrated with the rest of the infrastructure, such as facility management, asset management, and performance management. This requires us to have an open interface for the sensor data, while hiding the complexity of managing thousands of devices.

We tackle these challenges by innovative hardware, protocol, and system designs.

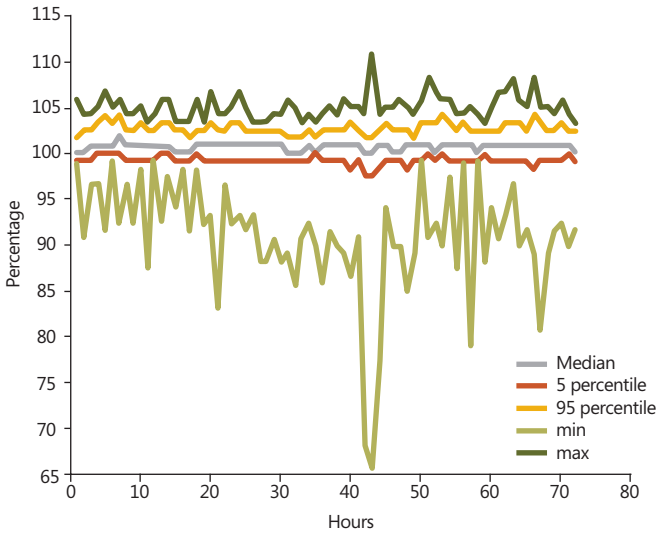
Genomotes

Genomotes are sensor motes we specifically developed for the DC Genome project. To meet the requirements of low cost of ownership, we chose IEEE 802.15.4 wireless technology over wired and WiFi (see Resources). Wireless nodes give the advantage of easy installation and ammunition to network administrative boundaries. Compared to WiFi, 802.15.4 radio is lower power, has a simpler network stack, and require fewer processing cycles. Thus we are able to reduce total cost by using simpler microcontrollers.

Although other similar wireless sensor designs are available on the market (such as SUNSPOT, Tmote, and SynapSense nodes), Genomotes are customized to simplify the installation process and reduce the number of wireless nodes in the network without sacrificing flexibility and scalability. Specifically, as shown in Figure 4, we designed two classes of Genomotes, master motes and slave sensors.

A Genomote master has a CC2420 802.15.4 radio, 1 MB flash memory, and a rechargeable battery. It is typically installed at the top of a rack. In addition to the radio, it has a RS232 socket. Each slave node has two serial ports, which are used to connect multiple slaves to the same head, forming a daisy chain that spans the height of a rack. Slave sensors are equipped with various sensors, such as temperature, humidity, and so on. Since they share the same serial communication

Figure 5: Data yield percentage from 174 wireless nodes (696 sensors) in a production data center. It shows the minimum, 5th percentile, median, 95th percentile, and maximum hourly yield from all sensors.



protocol, different kinds of sensors can be mixed and matched on the same chain.

Without a radio, external memory, nor battery, slaves are about half the cost of the master motes. The master periodically collects the slaves' measurements using a simple polling protocol and stores them in its local flash. DC Genome gateways then periodically retrieve stored measurements from each master using a reliable Data Collection Protocol (rDCP). A chain of four nodes can be powered at any mote via a single server's USB port, thanks to the low power circuit design.

This hierarchical design has several benefits. First, separating data acquisition and forwarding means that the master can work with slaves covering different sensing modalities. Second, because the ratio of slaves to masters is high, simplifying the slave's design minimizes the overall deployment cost, especially for large-scale networks. Finally, the design reduces the number of wireless nodes in the network that compete for limited bandwidth, while allowing individual racks to be moved without tangling wires.

Reliable Data Collection

Our system faces several challenges for reliable data collection. Low power wireless radios like IEEE 802.15.4 are known to have high bit-error rates compared to other wireless technologies. At the same time, data centers impose a tough RF environment due to the high metal contents of servers, racks, cables, railings, and so on. Furthermore, the high density of wireless nodes in RACNet — several dozen within the same communication hop — increases the likelihood of packet collisions.

RACNet's innovative rDCP data collection protocol achieves high throughput and high reliability using three key technologies:

- *Channel diversity:* IEEE 802.15.4 defines 16 concurrent channels in the 2.4GHz ISM band. Although the radio chip can only tune into one channel at any given time, rDCP can coordinate among multiple base stations to use multiple channels concurrently. The number of nodes on each channel is dynamically balanced to adapt to channel quality changes. Using multiple channels reduces the number of nodes on each channel, reducing the chances for packet collision.
- *Adaptive bidirectional collection tree:* On each wireless channel, a collection tree is dynamically built to adapt to link quality changes. Due to the large number of nodes in a one-hop communication range, viable links are abundant. Choosing the right links for high quality communication is key to improving hop-to-hop success rates.
- *Coordinated data retrieval:* Thanks to the flash memory on board, each master node caches data locally before it is retrieved. To avoid losing data due to packet collision, data is polled by the base station, rather than pushed by the sensors. Only one data retrieval stream exists on an active channel at any given time.

"Our system faces several challenges for reliable data collection. Low power wireless radios like IEEE 802.15.4 are known to have high bit-error rates compared to other wireless technologies. At the same time, data centers impose a tough RF environment due to the high metal contents of servers, racks, cables, railings, and so on. Furthermore, the high density of wireless nodes in RACNet — several dozen within the same communication hop — increases the likelihood of packet collisions."

Figure 5 shows the data collection yield over three days from 174 wireless master motes, driving 522 additional sensors. The data yield is computed as the ratio between the actual collected data entries and the theoretical result of 120 samples per sensor per hour. It is shown that over 95 percent of the sensors give higher than 99 percent data yield constantly. (The over 100 percent data yield is an artifact of in-network time synchronization: When local time proceeds faster and must occasionally be adjusted back to the global time, multiple samples are taken at roughly the same time stamp.) Figure 6 further shows the data collection latency, defined as the time differences between when the data is sampled and when they are

Figure 6: Data collection latency distribution of 10,000 data samples using three wireless channels.

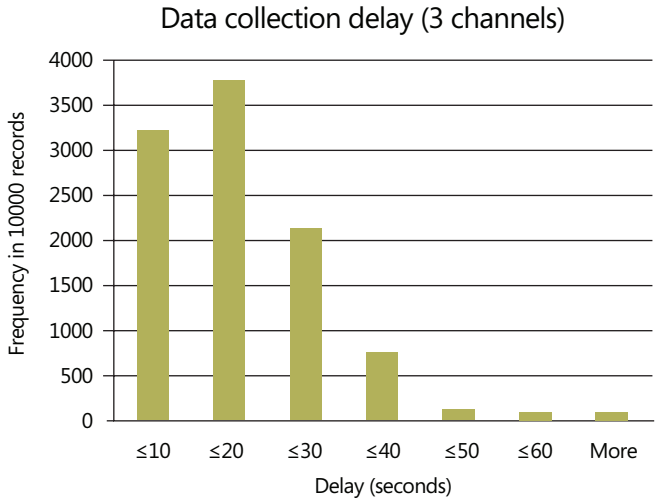
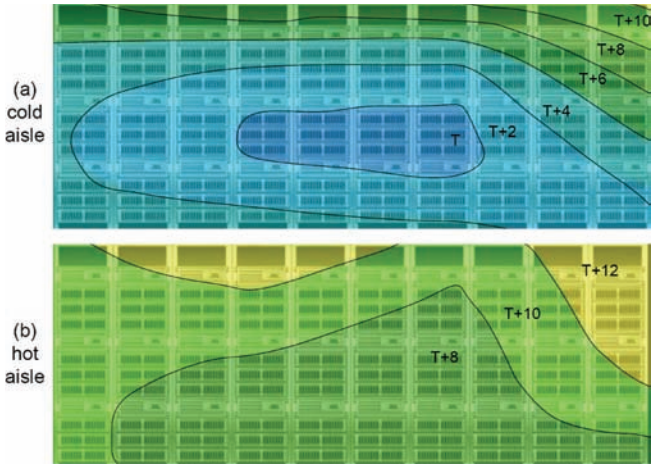


Figure 7: Heat map of a cold aisle and a hot aisle, generated from sensor data.



entered into the DC Genome central database. When using three wireless channels concurrently, over 90 percent of sensor data is collected before the 30 second deadline.

These unprecedented results show that a wireless sensor network can be used to reliably collect environmental data in data centers with low hardware cost and easy installation and maintenance.

Some Findings from Sensor Data

We have deployed thousands of Genomotes in multiple production data centers. In this section, we present some results that provide new insights to data center operations and workload management.

Heat distribution

Figure 7 presents heat maps generated from 24 sensors in the front and back of a row. In the cold aisle, the temperature difference between the hottest and coldest spots is as much as 10°C. It is evident that the racks' mid sections, rather than their bottoms, are the coolest areas, even though cool air blows up from the floor. This counter-intuitive heat distribution is observed in almost all data centers and is driven by Bernoulli's principle. This principle states that an increase in fluid (e.g. air flow) speed decreases its pressure. Fast moving cold air near the floor creates low pressure pockets which draw warm air from the back of the rack. The high temperature at the top right corner is due to uneven air flow which prevents cool air from reaching that area. As a consequence, hot air from the back of the rack flows to the front.

Heat maps like these can be useful in many ways. For example, if cool air can reach the top right corner by slightly increasing the CRAC's fan speed, then the overall temperature of the supplied air can be increased. Moreover, these measurements can guide the CRAC control system. Instead of using the temperature at the CRAC's return air point to control the amount of cooling, we can regulate the chill water valve opening based on the maximum air intake from all active servers. However, designing optimal control laws remains a significant challenge, as changes at the single cooling supply point can affect different data center locations disproportionally.

Thermal runaway

Thermal runaway is a critical operation parameter, which refers to the temperature changes when a data center loses cool air supply. Predicting

Figure 8: Intake air temperature from a row of ten racks, labeled from 1 to 10, during a thermal runaway event. Each rack has three sensors at the top, middle, and bottom, respectively. Temperature changes depend on locations.

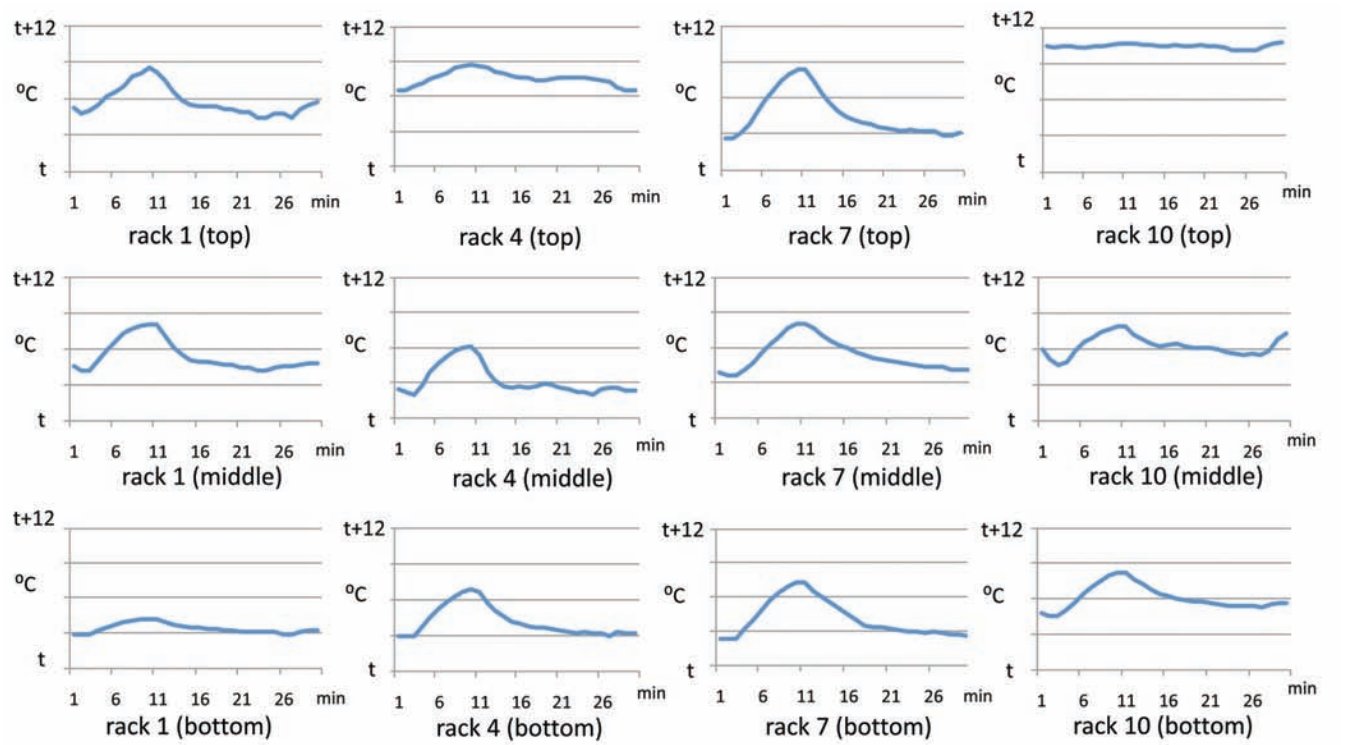
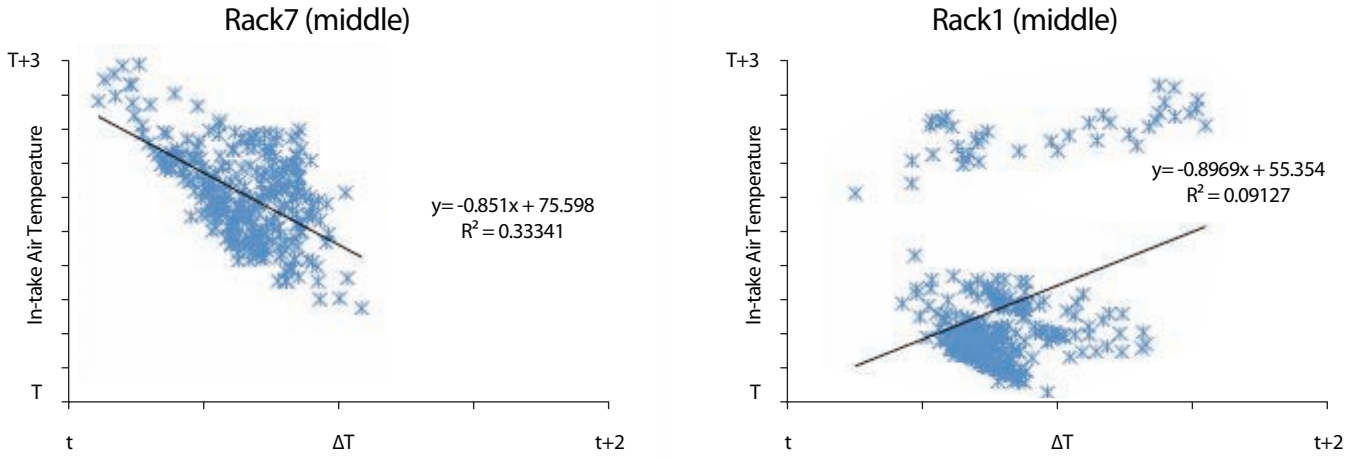


Figure 9: Sensitivity analysis across the middle sections of racks 1 and 7. The CRAC provides cooler air when rack 7 generates more heat, compared to rack 1.



thermal runaway transients through simulations is difficult because their accuracy depends on the thermal properties of IT equipment, which are difficult to obtain. On the other hand, RACNet collected actual thermal runaway data during an instance when a CRAC was temporarily shut down for maintenance.

Figure 8 plots the temperature evolution at various locations across a row of ten racks during the maintenance interval. The CRAC was turned off for 12 minutes. The midsections — normally the coolest regions — experienced rapid temperature increases when the CRAC stopped. In contrast, temperature changed moderately at the two ends of the row, especially at the top and bottom of the rack. This is because those racks have better access to room air, which serves as a cooling reserve. This is an important finding because large temperature changes in a short period of time can be fatal to hard drives. For example, 20°C/hr is the maximum safe rate of temperature change for the Seagate SAS 300GB 15K RPM hard drive, according to its specifications. Notice that, in the middle of rack 7, the rate of temperature change is almost 40°C/hr in the first 15 minutes of CRAC shutdown. This implies that storage intensive servers need to be placed carefully if the data center has a high risk of losing CRAC power.

Thermal stability challenges in dynamic server provisioning

Dynamic server provisioning and virtualization can effectively adjust the number of active servers based on server work load, reducing the total energy consumption during periods of low utilization. Given that many servers are functionally equivalent, which servers should be shut down to minimize energy consumption? Moreover, can turning off servers result in uneven heat generation and cause thermal instability?

We answer these questions by performing sensitivity analysis over the collected measurements. Assume — as is true for many commodity servers — that a server's fan speed is constant over time, independent of the server's workload. Then, the difference, ΔT, between the exhaust temperature and the intake temperature is proportional to the amount of heat a server generates. Ideally, a CRAC responds to the generated heat and provides cold air to the rack intake. So, the greater ΔT is, the lower the in-take air temperature should be. Figure 9 presents the results from one such example. The figures show scatter plots between ΔT and intake air temperatures at middle sections of racks 1 and 7, as well as linear trend lines. The corresponding R^2 metrics show how well the linear regressions are. We observe that the CRAC responds to the temperature changes at rack 7 much better than those at rack 1. In fact, an increase of ΔT at rack 1 is uncorrelated with its in-take air temperature. Such CRAC sensitivity variations create additional challenges for dynamically shutting down servers. Consider a scenario in which locations A and B rely on the same CRAC to provide cool air. However, the CRAC is extremely sensitive to servers at location A, while not sensitive to servers at locations B. Consider now that we migrate load from servers at location A to servers at location B and shut down the servers at A, making $\Delta T_A = 0$. The CRAC then believes that there is not much heat generated in its effective zone and thus increases the temperature of the cooling air. However, because the CRAC is not sensitive to ΔT_B at location B, the active servers despite having extra workload have insufficient cool air supply. Servers at B are then at risk of generating thermal alarms and shutting down.

Conclusion

The RACNets presented in this paper is among the first attempts to provide fine-grained and real-time visibility into data center cooling behaviors. Such visibility is becoming increasingly important as cooling accounts for a significant portion of total data center energy consumptions.

This practical application challenges existing sensor network technologies in terms of reliability and scalability. The rDCP protocol tackles these challenges with three key technologies: channel diversity, bi-directional collection trees, and coordinated data downloading. This is the first reported result of maintaining higher than 99 percent data yield in production sensor networks.

Collecting cooling data is a first step toward understanding the energy usage patterns of data centers. To reduce the total data center energy consumption without sacrificing user performance or device life, we need an understanding of key operation and performance parameters — power consumption, device utilizations, network traffic, application behaviors, and so forth — that is both holistic and fine-grained. With such knowledge, we will be able to close the loop between physical resources and application performance.

Acknowledgement

DC Genome is a multi-year collaboration between Microsoft Research and Microsoft Global Foundation Services. The authors would like to thank Darrell Amundson, Martin Anthony Cruz, Sean James, Liqian Luo, Buddha Manandhar, Suman Nath, Bodhi Priyantha, Rasyamond Raihan, Kelly Roark, Michael Sosebee, and Qiang Wang, for their direct and indirect contributions to the project.

Resources

2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver, Texas Instruments
http://focus.ti.com/lit/ds/symlink/cc2420.pdf

"An analysis of a large scale habitat monitoring application," Robert Szewczyk, 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys 2004), Baltimore, Md., November, 2004
http://www.eecs.harvard.edu/~mdw/course/cs263/papers/gdi-sensys04.pdf

"Energy-aware server provisioning and load dispatching for connection-intensive internet services," Gong Chen et. al, 5th USENIX Symposium on Networked Systems Design & Implementation (NSDI 2008), San Francisco, Calif., April 2008
http://www.usenix.org/event/nsdi08/tech/chen.html

EPA Report on Server and Data Center Energy Efficiency, U.S. Environmental Protection Agency, ENERGY STAR Program, 2007
http://www.energystar.gov/index.cfm?c=prod_development.server_efficiency_study

"Fidelity and yield in a volcano monitoring sensor network," Geoff Werner-Allen et. al, 7th Symposium on Operating Systems Design and Implementation (OSDI '06), Seattle, Wash., November 2006
http://www.eecs.harvard.edu/~konrad/papers/volcano-osdi06.pdf

"FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments," Carl Hartung et. al, 4th International Conference on Mobile Systems, Applications, and Services (MobiSys 2006), Uppsala, Sweden, June 2006
http://www.cs.colorado.edu/~rhan/Papers/p28-hartung.pdf

"The green grid data center power efficiency metrics: PUE and DCiE," The Green Grid
http://www.thegreengrid.org/gg_content/TGG_Data_Center_Power_Efficiency_Metrics_PUE_and_DCiE.pdf

IEEE 802.15.4
http://www.ieee802.org/15/pub/TG4.html

"In the data center, power and cooling costs more than the it equipment it supports," Christian L. Belady, ElectronicsCooling, February 2007
http://electronics-cooling.com/articles/2007/feb/a3/

Sentilla
http://www.sentilla.com/

"RACNet: Reliable ACquisition Network for high-fidelity data center sensing," Chieh-Jan Liang et. al, Microsoft Research Technical Report (MSR-TR-2008-145), 2008
http://research.microsoft.com/research/pubs/view.aspx?type=Technical%20Report&id=1562
"Smart cooling of data centers," C. D. Patel et. al, Proceedings of International Electronic Packaging Technical Conference and Exhibition (Maui, Hawaii, June 2003)

Smart Works
http://www.smart-works.com.Project Sun Spot
http://www.sunspotworld.com/

SynapSense
http://www.synapsense.com

"Wireless sensor networks for structural health monitoring," S. Kim et. al, 4th ACM Conference on Embedded Networked Sensor Systems (SenSys 2006), Boulder, Colo., November 2006
http://portal.acm.org/citation.cfm?id=1182889

About the Authors

Dr. Jie Liu is a senior researcher in the Networked Embedded Computing Group of Microsoft Research, whose work focuses on understanding and managing the physical properties of computing. His contributions have generally been in modeling frameworks, simulation technologies, program/protocol design, resource control, and novel applications of these systems. He has published extensively in these areas and filed a number of patents. His recent work has been in large-scale networked embedded systems such as sensor networks to large-scale networked computing infrastructures such as data centers.

Feng Zhao (http://research.microsoft.com/~zhao) is a principal researcher at Microsoft Research, where he manages the Networked Embedded Computing Group. He received a Ph.D. in electrical engineering and computer science from MIT. Feng was a principal scientist at Xerox PARC and has taught at Ohio State and Stanford. He serves as the founding editor-in-chief of ACM Transactions on Sensor Networks, and has written and contributed to over 100 technical papers and books, including a recent book, Wireless Sensor Networks: An information processing approach, with Leo Guibas (Morgan Kaufmann). He has received a number of awards, and his work has been featured in news media such as BBC World News, BusinessWeek, and Technology Review.

Jeff O'Reilly is a senior program manager in Microsoft's Data Center Solutions Research and Engineering group. He has worked in the data center industry for the past 10 years in management and operations. Jeff's recent focus is on implementation and integration of SCADA systems for the purposes of optimizing data center operations.

Amaya Souarez is a group manager within the Datacenter Solutions Group of Microsoft Global Foundation Services, leading the DC Automation & Tools team.

Michael Manos is a seasoned information systems management executive with over 15 years of industry experience and technical certifications. In his current role, Michael is responsible for the world-wide operations and construction efforts of all Internet and enterprise data centers for Microsoft Corporation. In addition to his responsibility for the ongoing administrative and technical support of servers, network equipment, and data center equipment residing within these facilities, his role includes the design, construction, and facility-related technology research as it relates to data center architecture.

Chieh-Jan Mike Liang is a computer science Ph.D. student at Johns Hopkins University. He is a member of the HiNRG group. His current research focuses on large-scale sensor networks—specifically, problems related to efficient network architecture and robust sensor network platform.

Andreas Terzis is an assistant professor in the Department of Computer Science at Johns Hopkins University, where he heads the Hopkins InterNetworking Research (HiNRG) Group. His research interests are in the broad area of wireless sensor networks, including protocol design, system support, and data management. Andreas is a recipient of the NSF CAREER award.

Green IT in Practice: SQL Server Consolidation in Microsoft IT

by Mark Pohto

Summary

The Microsoft environmental sustainability effort impacts all phases of the software development life cycle and operations. Several teams have committed to optimizing resource utilization. Current efforts include data gathering and analysis to identify areas for improvement, platform standardization, capacity management, consolidation and provisioning of energy efficient facilities and hardware. This article describes how Microsoft IT SQL Server consolidation activities are contributing to effective resource utilization and environmental sustainability.

Data Center Energy Consumption Trends

According to a recent report to Congress provided by the Environmental Protection Agency and Energy Star, data centers consumed about 1.5 percent of the total U.S. electricity consumption or 61 billion kilowatt-hours (kWh). Estimates indicate that by 2011 data center energy consumption could nearly double. The report recommends several strategies for reversing this trend, such as financial incentive programs, best practice guidance, and energy star ratings for data center hardware.

Strategies for lowering overall energy consumption include a broad range of activities that, when combined, could reverse the historical consumption trend. The study indicated that, if implemented, such activities could reduce data center electricity consumption by amounts equivalent to the power produced by as many as 15 new power plants. The savings correspond to carbon dioxide emission reductions of 15 to 47 million metric tons. (For more on data center energy consumption trends and recommendations, see the Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431, listed in Resources

Opportunities

When it comes to protecting the environment, everyone must demonstrate social responsibility and initiative. Microsoft IT has been engaged in related activities for several years. The efforts have targeted data centers, various layers of the application stack as well as specific server workloads. Major consolidation efforts have included RightSizing, Storage Utility, Compute Utility, File Server Utility, and recently, the SQL Server consolidation initiative. Each effort has significantly reduced the data center footprint and has contributed to environmental sustainability. Areas of opportunity targeted by the SQL Server consolidation initiative include:

- Energy efficient hardware
- Hardware consolidation
- Intelligent allocation of hardware resources
- Energy efficient data center design
- Intelligent placement of data center facilities
- Elimination of unnecessary hardware
- Enabling power management where practical.

In addition to environmental sustainability benefits, SQL Server consolidation also presents clear business benefits, such as:

- *Reduce operating and capital expenses.* New hardware has far greater computing power than hardware that is nearing end-of-life. Newer hardware generally requires less power and cooling.
- *Environmental sustainability.* For example, lower power and cooling requirements are the primary areas where the SQL Server Utility addresses environmental concerns, but an overall reduction in data center space will also contribute over the long term.
- *Provide business continuity, scalability, and availability.* Requirements do not change for the SQL Server Utility; the goal is to find opportunities to standardize and to improve best practices in these areas.
- *Provide a standardized server build library.* New technologies such as Hyper-V open new opportunities for standardization. Part of the vision of the SQL Server Utility is to eliminate or streamline many of the manual steps needed to build an application environment. This can be achieved by establishing Hyper-V guests which are built with standard software configuration, including the operating system, SQL Server, tools, and approved configurations which can be provided for use in different phases of the software development life cycle.

Initial Situation

Currently, the Microsoft IT application portfolio consists of about 2,700 applications. There are approximately 100,000 databases on 5,000 SQL Server Instances, most of which are on dedicated hosts. Approximately 20 percent of those hosts reach end-of-life each year and are replaced. Average CPU utilization across these hosts is below 10 percent, indicating significant opportunity for host consolidation.

Fortunately, the landscape for SQL Server consolidation has changed dramatically over the past few months. New technologies such as Windows Server 2008, SQL Server 2008, Hyper-V, System Center Virtual Machine Manager, System Center Operations Manager, improved storage technologies and more powerful servers provide greater opportunities for consolidation than ever before.

In addition to server consolidation, other virtualization opportunities exist. Those benefits are not the focus of this article but are described under General Benefits later in this article.

Desired Situation

The objective is to design and deploy a SQL Server Utility service to reduce operating and capital expenses through consolidation and multi-tenancy. The SQL Server consolidation initiative will be based on the already

Table 1: Server Temperatures

Server Temperature	Mean CPU%	Maximum CPU%
Permafrost	<=1	<=5
Cold	<=5	<=20
Warm	<=20	<=50
Hot	>20	>50

successful shared service models of Storage Utility, Compute Utility, and File Server Utility. Each of these utilities has contributed to standardization of the Microsoft IT infrastructure, and each provides a more predictable and reliable platform for other applications and services. The Storage Utility and Compute Utility will be discussed in more detail later but are both important to the SQL Server Utility design.

Not all SQL Server instances will be good candidates for consolidation. Initially, this solution will be developed with the most typical OLTP databases in mind. Multi-tenancy scenarios will also be supported for those customers who only want a single database. We expect that consolidation will be appropriate for thousands of instances and that multi-tenancy will be appropriate for hundreds of databases.

In fiscal year 2009, the goal is to engineer and provide a SQL Server Utility that will reduce dedicated single instance SQL Server hosts by 10 percent and position Microsoft IT for continued consolidation.

Solution Mindset

The solution mindset requires executive sponsorship, data-driven discussion, coherent storage and computing foundations, and an articulated consolidation strategy.

Executive sponsorship. Executive commitment to maximize data center resource utilization and promote environmental sustainability is critical. Several activities described in this section provided the foundation needed for consolidation. None of these activities would have been likely to succeed without executive sponsorship and investment.

A data-driven discussion. Microsoft IT developed the RightSizing initiative to ensure effective utilization of servers in the data center and in managed labs. Because significant underutilization occurs, one of the initiative's first tasks was for Microsoft IT to identify underutilized servers that might be good candidates for virtualization (for more information on RightSizing, see Resources). The Capacity Management team relies on RightSizing data.

To accurately compare the performance of server platforms of varying architectures, Microsoft IT has developed a Compute Unit (CU) formula for each server platform that utilizes an industry standard, architecture-agnostic, benchmark suite from the Standard Performance Evaluation Corporation (SPEC). The SPEC benchmarks are developed in such a way to allow a repeatable test with strict result submission requirements. The Microsoft IT CU formula uses a baseline (not peak) benchmark that measures the rate of integer calculation work a server platform can perform in a given amount of time.

The servers available for purchase today represent a massive increase in performance over systems available in the past. Today's 2-way server provides the equivalent computing power of a 4-way from 12 to 18 months ago and even matches the Compute Unit capabilities of a four-year-old 8-way server. By collecting information about current hardware and processor utilization, the RightSizing team can make recommendations on how to maximize server utilization.

Data center servers are underutilized, with an overall average CPU utilization of ~9.75 percent. As new server platforms are introduced with performance capabilities far surpassing their predecessors, this already low CPU utilization number will continue to fall.

Table 1 and Figure 1 depict current processor utilization for a sample set of servers in Microsoft IT. A "temperature" was assigned to each consumption category for reference in RightSizing discussions. These numbers indicate substantial opportunity for host consolidation.

Storage foundation. The Storage Utility service provides shared or dedicated SAN storage to which data center servers can connect. The service provides the SAN storage and all hardware required to connect a server to the SAN, as well as all maintenance and management functions, but does not provide the server itself.

An effect of the Storage Utility was that instead of engineering and purchasing small, medium, large servers with anticipated DAS for SQL Server and other server types, Microsoft IT was able to modify the standard builds so that minimal DAS was included in the purchase. This means that application support teams no longer had to anticipate disk capacity over the life of the server which usually resulted in underutilization.

Computing foundation. The Compute Utility strategy abstracts the services provided by hardware at Microsoft data centers. Rather than having a business unit address its computing requirements by purchasing a server, in this approach, a business group provides its computing capacity requirements, and Microsoft IT then determines whether a virtual or physical server can meet those requirements and provides the service. The Compute Utility strategy sought to create this level of abstraction for business groups to encourage them to purchase computing power and storage without worrying about the server hardware.

Other utilities such as the File Server utility rely on RightSizing data and reside on Storage and Compute Utilities. SQL Server consolidation will further contribute to environmental sustainability and will also rely on these utilities.

Consolidation strategy. There are multiple approaches to database consolidation. More common approaches include instance consolidation and host consolidation. In instance consolidation, databases from multiple SQL Server instances are consolidated under fewer instances, and considerations range from CPU, memory, and I/O subsystem management to sort/collation sequences to endpoint usage. In host consolidation, the host is partitioned (typically with Hyper-V or Windows System Resource

Figure 1: Server Temperatures Sample Set

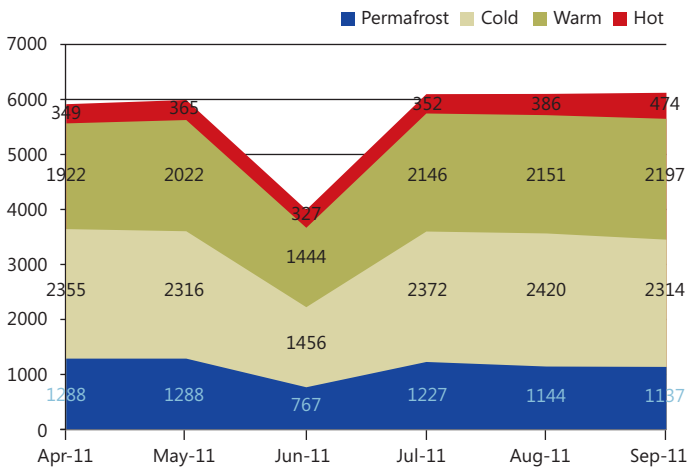
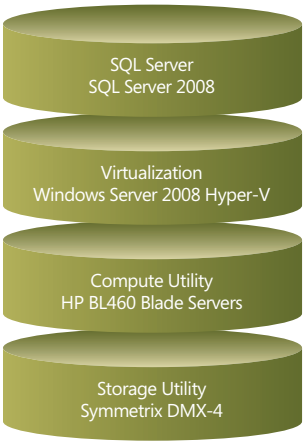


Figure 2: MSIT SQL Server Consolidation Stack



Manager) and a larger number of instances are placed on each host. Each method of consolidation has its own considerations but many areas are much simpler to manage in a host consolidation approach.

Microsoft IT has compared consolidation approaches and has selected host consolidation for its ability to meet our consolidation objectives while introducing minimal risk. Since host consolidation still requires shared resources such as CPU, memory, IO, and network, selection of a manageable and flexible host partitioning method will have a significant impact on day-to-day operations. The General Benefits section of this article describes some important advantages of using Hyper-V compared to Windows System Resource Manager and multiple named instances.

All SQL Server Utility offerings will leverage layers of the existing Microsoft IT utility stack (Figure 2). Later phases of the consolidation initiative will consider other forms of consolidation as well as multi-tenancy.

Host consolidation requires that resources on a single host be managed to ensure that each instance receives predictable and reliable memory, processor, network, and I/O. Windows System Resource Manager and Hyper-V host partitioning technologies were evaluated; Hyper-V was selected for its manageability, scalability, and standardization benefits. Since Hyper-V on Windows 2008 scales to a maximum of four processors per guest (two processors for Windows 2003), consolidation of physical SQL Server instances will also be used to consolidate instances that require more than four processors. Physical SQL Server instances will be less manageable than virtual instances but will provide a consistent, proven approach to consolidation until Hyper-V scales to more than four processors.

The approach to host consolidation primarily targets servers that are approaching end-of-life. This provides minimal service disruption (since server replacement would occur with or without consolidation).

Multi-tenancy instances will also be deployed on Hyper-V and will be scaled to maximize host resources. As server and virtualization technologies improve, the Hyper-V approach to multi-tenancy will provide increasingly improved manageability and scalability.

Figure 3 depicts six end-of-life servers with 30 compute units each being replaced by equivalent Hyper-V guests on a single server. Figure 4 depicts SQL Server Utility offerings that cover different tenant needs. Hyper-V will be used to partition host resources for those tenants who need up to four processors and named instances either on a shared or dedicated server will be used for those who require additional processor or memory resources. As Hyper-V, hardware, and other technologies evolve, more and more tenant needs will be met using Hyper-V. Table 2 shows the standard Hyper-V guest configurations that will be used in the SQL Server Utility.

Solution Implementation

The primary requirements of the SQL Consolidation are to reduce operating and capital expenses by more effectively utilizing data center resources. System qualities, such as availability, that existed in the non-consolidated legacy environment are still requirements for the new consolidated environment. This architecture design followed basic IEEE-1471 guidance in identifying stakeholders, gathering and understanding requirements, and selecting and designing for specific architectural viewpoints. Architectural viewpoints were selected from traditional IT service management functions and were evaluated against Microsoft Operations Manager 4.0 IT service life cycle and current technology trends to anticipate future effectiveness of the design and to identify opportunities for improvement. Environmental sustainability has become an architectural consideration for business/home construction, automotive, and other industries; it is also a relevant architectural framework viewpoint for IT.

Availability and Business Continuity requirements provide the application development teams and operations teams with the flexibility to employ any SQL Server feature that is appropriate for an application. At the time this article is being written, SQL Server clustering on Hyper-V has not yet been approved as a supported implementation. However, database mirroring and log shipping are supported. SQL Server clustering will be introduced to the SQL Server Utility service as soon as that support is available. This project will be executed in phases, deploying implementations as support becomes available, so non-clustered deployments will be completed ahead of those that require clustering.

Figure 5 depicts Hyper-V SQL Server Guests Deployed from a standard build library with optional business continuity and high availability options. Tenant instances will be colocated and will be placed to distribute workload based on application business cycle requirements.

Manageability in the consolidated environment is improved but very similar to what it was in the non-consolidated environment. New and future solutions like System Center Virtual Machine Manager, Data Protection Manager, and System Center Operations Manager will help ensure continuous improvement in the area of manageability.

Provisioning. One key benefit of this consolidation architecture will be the ability to quickly provision new SQL Server guests. Since storage and host servers will be preconfigured, the turnaround time for providing a new SQL Server guest will be reduced by weeks when compared to the acquisition process for dedicated physical hosts.

Figure 3: Host Partitioning with Virtual Instances on Hyper-V or Physical Instances on WSRM

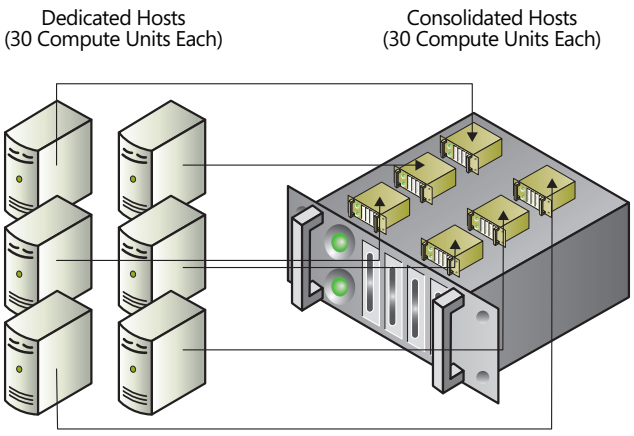
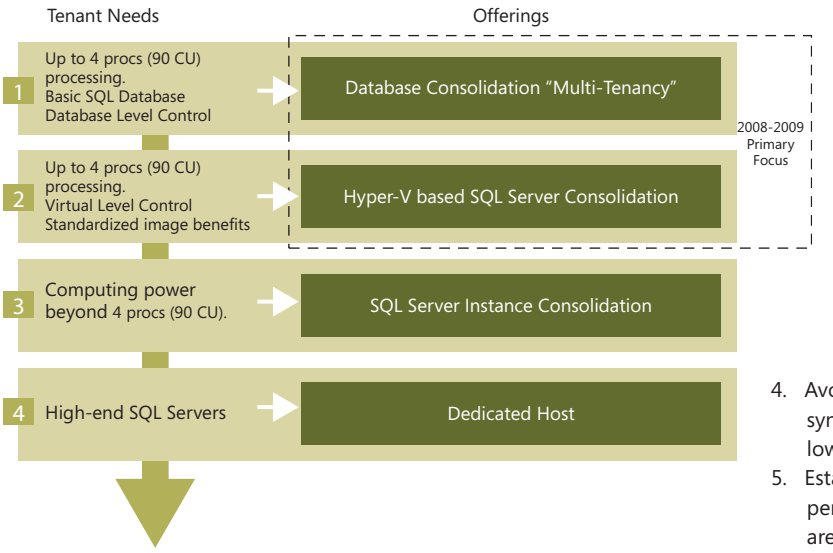


Figure 4: MSIT SQL Server Utility Offerings



Virtual Machine Manager Library, a standard build library, developed by Microsoft IT, with consistent Windows Server and SQL Server installation and configuration is another key manageability improvement for consolidation with Hyper-V which will allow IT developers, testers, and production support teams to ensure a consistent experience across phases of the Software Development Life Cycle.

Imagine replacing less efficient build processes with a standardized and possibly automated deployment process. Colleagues with many years of experience in development and production support express excitement when contemplating the increased environmental consistency and stability offered by this approach.

Relocation of a virtual SQL Server instance is simpler and lower risk than the method of building a server, installing Windows Server and SQL Server then migrating databases to the new physical SQL Server instance.

Basic database administration tasks for the consolidated SQL Server environment will continue to leverage the backup utilities that were written by Microsoft IT (see the September 2007 article on SQL Server Automation Scripts in SQL Server Magazine). Data Protection Manager has been deployed within Microsoft IT and adoption for SQL Server backups is on the roadmap for this initiative.

Finally, the snapshot feature of Hyper-V will improve the ability to deploy and roll back host changes. You simply take snapshots at key points in your change deployment so that, instead of having to start from scratch and rebuild a server, you have the option to roll back. While there is overhead associated with taking guest snapshots, as a tactical deployment tool, Hyper-V snapshots have advantages over a manual roll back.

Performance requirements and abilities in the consolidated environment are also very similar to the previous environment. Since the Storage Utility had been implemented prior to SQL Server consolidation, a track record of performance from the storage layer already exists. In the consolidated environment, SQL Server operations teams will still be able to provide I/O performance expectations to the Storage Utility team and will be able to obtain Hyper-V guests with sufficient processor and memory from the Compute Utility team. Here are a few performance guidelines that are used in this consolidation effort:

1. Crawl, walk, run. Don't go for the maximum host consolidation ratio right away. Begin with the smaller workloads, validate your deployment, and refine the plan. Maximize your resource utilization in phases after establishing and evaluating your actual utilization.
2. Use Hyper-V pass-through disk or fixed Virtual Hard Disks (VHDs) for storage (Figure 6). Fixed VHDs offer some manageability benefits but provide slightly lower performance. Moving a guest to another host, for example, is simplified when using VHDs.
3. Do not overcommit processors for SQL Server guests. Begin with one logical processor for one physical processor. Verify your performance and refine your configuration as needed. At some point, this may include overcommitment, but begin without overcommitment to manage risks.
4. Avoid the use of emulated devices in Hyper-V. Favor synthetic devices which provide better performance and lower processor overhead.
5. Establish an operating level agreement (OLA) with performance requirements for your storage provider if they are a separate service. Microsoft IT SQL Server Utility has requirements for 1ms average disk/second read/write for log and 8ms for OLTP data.

Since Windows 2008 Hyper-V guests may use a maximum of four processors, native SQL instance consolidation is planned for workloads that require more than four processors. Native SQL instance resource management can be effectively accomplished using both SQL Server configuration and Windows System Resource Manager. Once Hyper-V scales to more than four processors, SQL instances that require more processing power will be deployed and managed on Hyper-V. (The recently published "Running SQL Server 2008 in a Hyper-V Environment - Best Practices and Performance Recommendations" offers more detailed Hyper-V guidance; see Resources.)

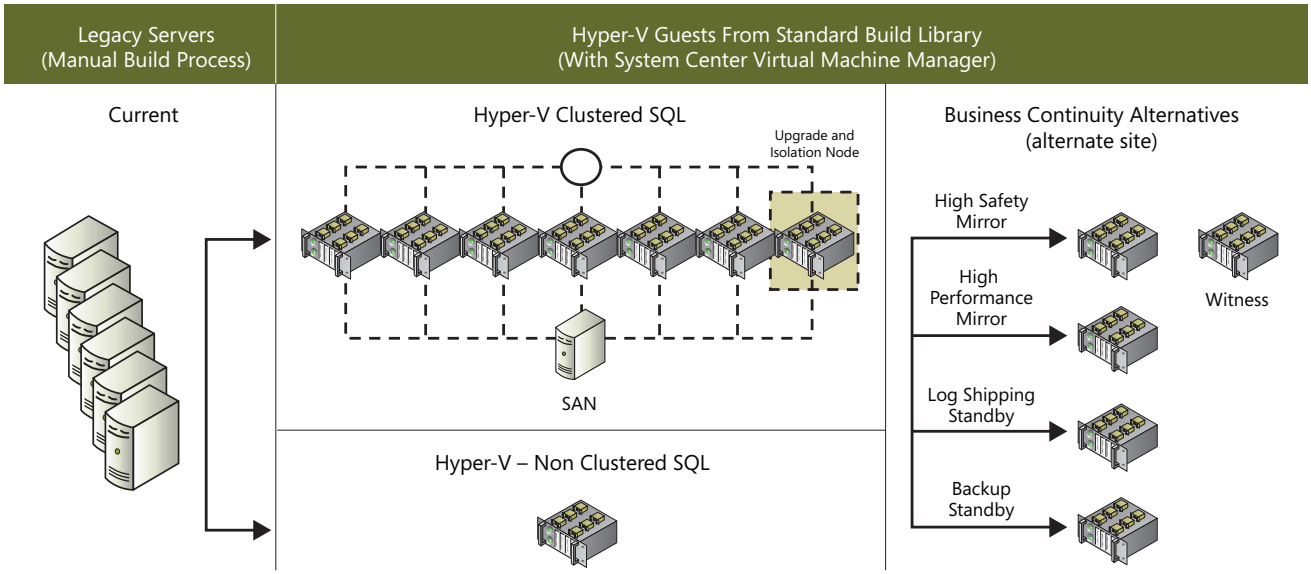
Predictability/repeatability can be improved by developing and adopting a configuration and deployment strategy that spans multiple phases of your software development life cycle. In today's environment developers, testers and production support teams build servers using standard install bits for the operating system, SQL Server, and additional tools. This time-consuming approach can sometimes result in inconsistencies between environments. These inconsistencies can ultimately result in unanticipated or unpredictable behavior once an application is deployed into production environments. Using Hyper-V guests that are preconfigured with standard builds that the SQL Server Utility team expects will reduce or eliminate inconsistencies across the software development life cycle.

Reliability was a key concern for SQL Server consolidation tenants. The concern was that consolidated workloads and configurations would interfere with one another and that changes made for one application/tenant would impact others and cause additional application downtime.

Table 2: MSIT SQL Server Utility Offerings

Offering	Memory (GB)	Processors
Virtual Instance Low	2 - 4	1
Virtual Instance Medium	4 - 8	2
Virtual Instance High	16	4
Physical Instance	variable	>4

Figure 5: Hyper-V SQL Server Guests with Business Continuity and High Availability

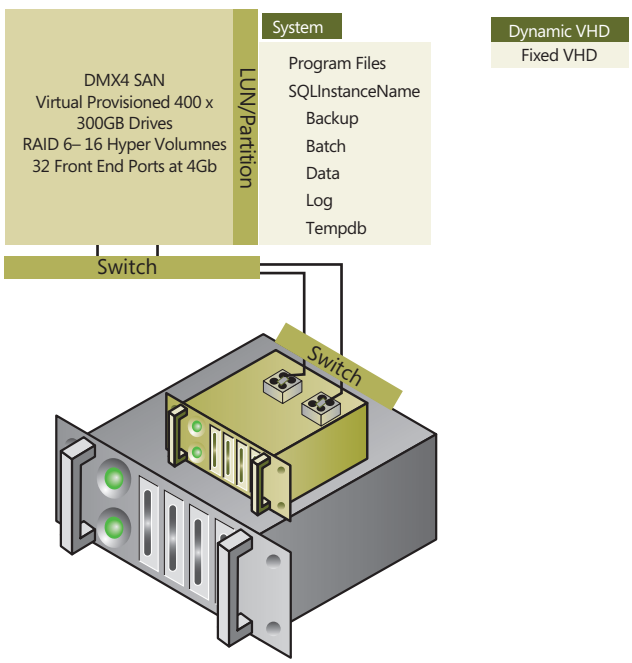


Hyper-V provides operating system isolation so production support teams can prioritize their upgrade activities without dependencies on other application support teams. This greatly improves flexibility when it comes to test, change, and release; and improves overall reliability for applications due to fewer conflicts.

Scalability and Capacity Management are easier with virtualization. RightSizing teams and capacity management teams have greater agility when it comes to sizing up or down. If a tenant requests a medium-sized two-processor guest with 4 GB of memory, but after deployment, learns that they underestimated their resource requirements, it is a simple matter to add memory and processors to the guest, even if it means relocating the guest to a host with available resources. This flexibility to reconfigure and relocate Hyper-V guests means that IT teams no longer have to over-purchase hardware resources which will result in less underutilization of data center resources.

SQL Server instances that require more than four processors will be in scope for consolidation but will be consolidated using physical rather than virtual instances. Windows System Resource Manager will be used in cases where multiple physical instances share a single host. Physical hosts will be dedicated to specific tenants for improved SLA management, security and customer satisfaction. As Hyper-V supports increased numbers of processors, more databases will be appropriately hosted in virtual instances.

Figure 6: Hyper-V SQL Server Guest VHDs



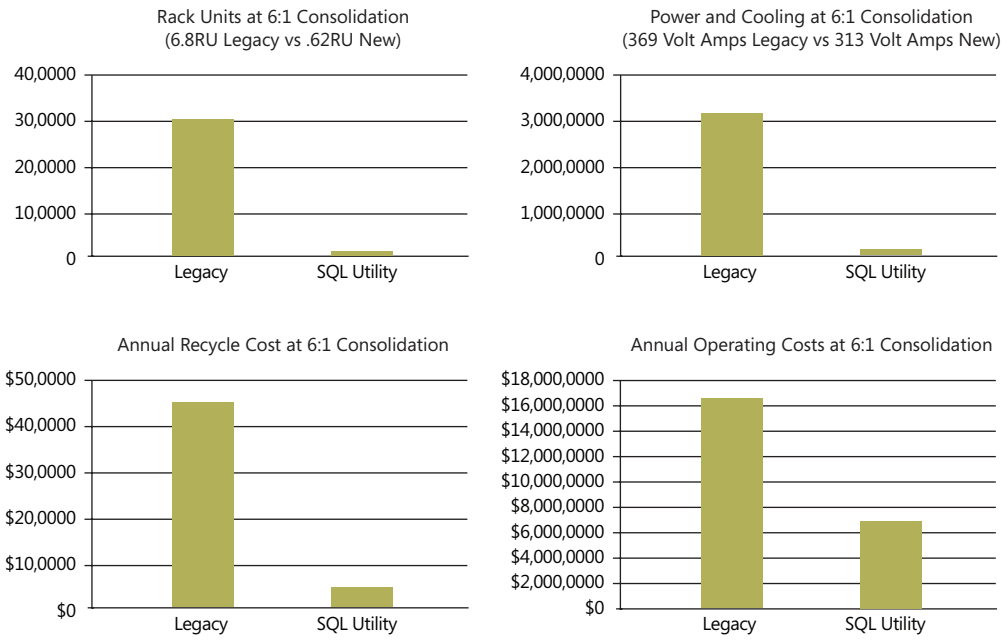
Physical Resource Optimization (PRO). Using System Center Virtual Machine Manager 2008 and System Center Operations Manager 2007, administrators can assess historical performance data and intelligently place new SQL Server guests to optimize physical resource utilization and distribute workloads.

Security presents one of the greatest concerns. Microsoft IT production support teams have traditionally had access to host configurations. They are responsible and accountable for changes made to hosts and though changes are made through established change control processes, these teams wish to maintain the level of access and control that they've had historically. The Hyper-V approach to host partitioning provides production support teams with the flexibility to schedule, test, and apply changes and security patches in a timely manner.

Other methods of host partitioning, such as Windows System Resource Manager, meant that each instance on any given host would need to be on the same build of Windows Server and may need to coexist with other tenants who would likely have different service level agreements, maintenance windows, priorities, and application business cycles. Other partitioning methods also introduce security questions for data access, service account usage, certificates, logins/roles, and access to the operating system and hardware.

Service Monitoring and Control changes for this project are minimal in early phases. Monitoring requirements are somewhat increased by introducing an additional layer for virtualization, but since virtualization has been broadly adopted by other platform layers such as for file servers and Web servers, service monitoring and control is expected to

Figure 7: Impacts of Consolidation



On average, end-of-life servers use 369 volt amps while new servers use 313 volt amps and can run at slightly higher temperatures. Similar power requirements exist for cooling. This means that there will be a dramatic reduction in power requirements, over 3 million volt amps, and eventually, there may be opportunities to modify data center cooling requirements to further reduce power consumption.

Recycle costs for this project were estimated, but it is clear that deploying fewer servers has an impact on recycling and the environment. Figure 7 depicts the impacts of a 6:1 consolidation ratio for 5,000 SQL Server hosts (recycle costs were estimated).

Consolidation Across the Software Development Life Cycle

Figure 8 depicts Hyper-V streamlined processes across the software development life cycle and well into production operations environments.

become a standardized and effective part of our environment. Monitoring services currently provided using System Center Operations Manager 2007 by the Microsoft IT Enterprise Monitoring Team for the preconsolidation environment will also be leveraged by the SQL Server Utility. The SQL Server Utility team believes that availability monitoring for databases and SQL Server services are a fundamental requirement for this service offering. Therefore, adoption of a database availability reporting/scorecard system and for improving the System Center Operations Manager SQL Server management pack rules are in scope for early project phases.

Another product that will be used to ensure consistent and optimal configuration is System Center Configuration Manager. Using this product with Desired Configuration Management, detection of configurations which have deviated or "drifted" from the known good and approved configurations will be much easier.

Supportability is important to application owners. The SQL Server Utility project will not deploy unsupported configurations. We will, however, anticipate future supported configuration scenarios and posture accordingly.

Results

Consolidation in the Data Center

Operating costs are expected to drop sharply but not as dramatically as power and space. That is primarily because, even though we will have fewer physical servers, there will still be a cost associated with managing the Hyper-V guests. Annual operating costs for the SQL Server utility are expected to be $11 million/year lower than previously.

A typical end-of-life SQL Server host in the Microsoft data center occupies 6.8 rack units. Servers provided by the Compute Utility team occupy less than 1 rack unit and provide enough computing power to host five or six instances of SQL Server. This comes to a savings of over 33,000 rack units or about 700 racks! This number does not even take into account that the DAS being replaced by SAN also comes with significant savings.

General Benefits

The Table 3 shows the benefits of consolidation on Hyper-V guests compared to using native instances of SQL Server.

Conclusion

Effective resource utilization within the data center has a significant impact on the global efforts to protect our environment. Beginning with the server build, and on through transportation to the data center, rack space

Figure 8 Streamlining Processes with Hyper-V

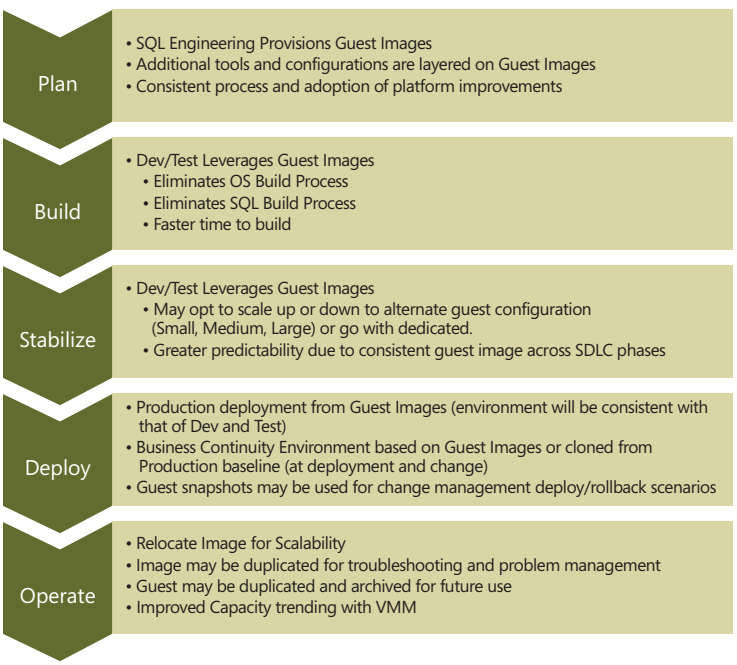


Table 3: SQL Server Utility System Qualities – Hyper-V and Physical SQL Server Instances

System Quality	Feature	Hyper-V	SQL Inst.
Manageability	Ability to build and provide canned environment	YES	NO
Manageability	Deploy/Rollback Benefits	YES	NO
Manageability	End-to-End (development through production) use	YES	NO
Manageability	Simple migration to new host during server retire/replacement	YES	NO
Manageability	Simplicity for Instance scale up	YES	NO
Manageability	Simplicity for cloning a production environment (e.g. to Test)	YES	NO
Security	Transparent to accomplish same level of security as with a dedicated host?	YES	NO
Scalability	Dynamic sharing of processor resources	YES	YES
Scalability	Processors Supported per environment	4	>32
Performance	Acceptable Performance	YES	YES
Availability	Clustering Option (Future ability for SQL Server on Hyper-V)	YES	YES
Business Continuity	Supports SQL Business Continuity features?	YES	YES
Supportability	SQL 2005 and 2008 CSS Support (not yet with SQL Server clustering)	YES	YES

utilization, power, cooling, and finally, to the eventual end-of-life recycle, each server deployed or not deployed has an impact on efficient resource utilization and ultimately on the environment.

Executive sponsorship is crucial to enterprise-wide consolidation projects. It is not difficult to make a business case for consolidation, particularly when you consider data center power and rack space benefits. The socialization, evangelism, and budget controls that executive sponsorship provides are vital to deployment and adoption.

Today, finding opportunities to consolidate various IT layers and functions is easier than ever before. Windows Server 2008, Hyper-V, and SQL 2008 are a few of the products that are truly game changers when it comes to effective resource utilization and consolidation.

The Microsoft IT SQL Server Utility project is currently in progress and is expected to make a substantial impact on environmental sustainability while achieving many other virtualization and consolidation benefits.

Resources

Data Center Energy Forecast
http://svlg.net/campaigns/datacenter/docs/DCEFR_report.pdf

Microsoft IT RightSizing - Identifying Server Candidates for Virtualization
http://technet.microsoft.com/en-us/library/cc700692.aspx

Microsoft IT Showcase Virtualization Whitepaper
http://technet.microsoft.com/en-us/library/cc713312.aspx

Report to Congress on Server and Data Center Energy Efficiency (Public Law 109-431)
http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf

Running SQL Server 2008 in a Hyper-V Environment – Best Practices and Performance Recommendations
http://sqlcat.com/whitepapers/archive/2008/10/03/running-sql-server-2008-in-a-hyper-v-environment-best-practices-and-performance-recommendations.aspx

SQL Server Magazine SQL Server Automation Scripts (September 2007)
http://www.sqlmag.com/Article/ArticleID/96463/SQL_Server_Automation_Scripts.html

Standard Performance Evaluation Corporation
http://spec.org

About the Author

Mark Pohto is a senior systems engineer in Microsoft IT. Mark has worked in IT since 1987 and has a strong passion for leadership and database technologies. He joined Microsoft's Redmond Platform Services team in 1999. His diverse experience and education includes language training at the DOD Defense Language Institute, several years in the intelligence community, U.S. Army Officer training, and MIS training at Our Lady of the Lake University in San Antonio, Texas. At Microsoft, he has worked as a senior DBA in the Database Operations Group and has managed the team responsible for application monitoring using Microsoft Operations Manager. He developed and published the automation tools used for database administration and has contributed to the books SQL Server 2000 High Availability and Pro SQL Server 2005 High Availability by Allan Hirt. As the group manager for the Microsoft SQL Server Center of Excellence, Mark managed the team that developed SQL Server solutions such as the Risk Assessment Program and Service Level Monitoring and the SQL Server Microsoft Certified Architect program. He has taught architecture in programs for Exchange Server, Windows Server and SQL Server. Mark recently returned to Microsoft IT to lead an IT-wide SQL Server consolidation project.

Jimmy May, Microsoft IT senior performance engineer, and Ward Pond, Microsoft Services technology architect, also contributed to this article.

Platform
Architecture Team

Microsoft

098-110485

18

subscribe at
www.architecturejournal.net