Publisher
Richard Bowles

Managing Editor
Stuart Douglas

Content Architect
Tomm Aldridge
Bob Steigerwald

Program Manager
Stuart Douglas

Technical Editor
David Clark

Technical Illustrators
MPS Limited

Technical and Strategic Reviewers
Kamal Srinivasan
Jason Hendrickson
Mark Aggar
Ben Srour
Dale Juenemann
Joe Butler
Dave Boundy
Joe Daly
Mark Gross
German Monroy
Sherine Abdelhak
Tom Huff
Dr. Jon Weissman
Weijun Xiao

Intel Technology Journal

ISBN 978-1-934053-55-3, ISSN 1535-864X

Intel Technology Journal
Volume 16, Issue 3

Publisher: Richard Bowles
Managing Editor: Stuart Douglas

Library of Congress Cataloging in Publication Data:

Printed in China
10 9 8 7 6 5 4 3 2 1

First printing: July 2012

Notices and Disclaimers

INTEL® TECHNOLOGY JOURNAL
SUSTAINABLE INTELLIGENT SYSTEMS

Articles

FOREWORD

by Martin Curley
Vice President,
Director Intel Labs Europe and Senior Principal Engineer, Intel Corporation

Less is More, More Moore, and More than Moore

Welcome to the *Intel Technology Journal* Special Issue on Sustainable Intelligent Systems.

As we look forward there are two major paradigm shifts that we consider in the context of this special issue of the *Intel Technology Journal*. First, arguably the world is beginning a once-off transition to a system based primarily on electrical energy while in parallel there is increasing recognition of the need to adopt a global sustainability paradigm. Already forward-looking governments are embedding these paradigms into their policies and legislation. For example the European commission has set a 20/20/20 goal to achieve a 20-percent share of energy from renewable sources by 2020, reduce greenhouse gas emissions by 20 percent by 2020 compared to 1990, and achieve savings of 20 percent of the EU's energy consumption compared to projections for 2020. Indeed individual governments are going further with the Irish government having set a target of achieving 40 percent of Ireland's electricity needs from renewable energy sources by 2020 while both Germany and Ireland have set targets to have electric vehicle penetration at 10 percent by 2020.

Sustainable development as concept can be defined as "development that meets the needs of the present without compromising the ability of future generations to meet their own needs" (Brundtland Commission, 1987). The concept of "doing more with less" is a concise encapsulation of this concept. Many reports warn of the consequences of ignoring the need for a new sustainability paradigm. The Global Footprint Network has stated that if current consumption trends continue we would need two world's worth of resources to support us by 2050. E.F. Schumacher, the German economist who worked for the UK National Coal Board, far earlier than most was able to capture the essence of sustainability in his phrase "small is beautiful" and this is a good lens through which to consider Moore's law.

Moore's law is one of the few modern business phenomena that actually support the sustainability paradigm. We in the computer industry have certainly played our part. An ACEEE[1] study has shown computing energy efficiency improved by more than 2.8 million percent in the thirty years from 1978 to 2008 while other industries such as lighting, automotive, and agriculture achieved orders of magnitude less energy improvement. The introduction of multi-core indeed has accelerated this progress even further and the continuous shrinking of geometries has allowed Intel to achieve a "less is more" outcome where each successive generation of product is smaller, more powerful, and more energy efficient than the previous generation. As a consequence of this, one new server today routinely replaces ten older servers and consumes a fraction of the rack space in a data center.

Gartner estimates that the IT industry contributes roughly 2 percent of the world's energy green house gases but it is increasingly recognized that the big opportunity in the future is to substitute IT for other resources across the economy and indeed the broader society. Indeed another ACEEE report identified that the some of the most energy-intensive industries such as transportation were amongst the lowest in IT intensity indicating strong potential for significant energy savings through more deployment of IT. The Intel SAP co-Lab in Belfast recently showed that through smart scheduling algorithms that approximately 100,000 new electric vehicles (EVs) could be introduced onto the island of Ireland without any significant increase in electricity generation capacity. Imagine what also might be possible when each of the EVs acts also as an energy storage device in a smart grid.

[1] "A Smarter Shade of Green," ACEEE Report for the Technology CEO Council, 2008.

Through a continuous flow of innovations we see a future where Moore's law will continue to be upheld.

While acknowledging the ongoing contribution of future ultra low power IT devices and green software to directly improving energy efficiency, the broader impact of information and communications technology (ICT) to future sustainability will be through mechanisms such as

- energy and resource efficiency achieved through sophisticated sensing, analytics, and actuation
- dematerializing through substituting ICT resources for other inputs (such as labor and materials) in value chains and indeed broader societal activities
- enabling servitization, a shift from products to services, where the goal shifts from to attempting maximizing product consumption to optimizing service while optimizing product longevity and resource usage.

The combined effects of such mechanisms can help achieve "resource decoupling," whereby economic growth and quality of life can be improved while simultaneously reducing resource consumption and environment degradation. One glimpse of the impact of such mechanisms is the finding in another ACEEE report, which estimated that for every 1 kWHour of energy consumed in a US data center, an average of 10 kWHours were saved across the broader US economy. Some logistics companies have claimed fuel efficiencies of approximately 20 percent through using smart scheduling software to maximize the number of right-hand turns delivery drivers are able to make on their daily delivery runs. At the Intel founded Innovation Value Institute in Ireland we have defined a Sustainable Information and Communications Maturity Curve and Assessment instrument to act as a design pattern to help organizations assess and improve the impact of applying ICT for sustainability.

These mechanisms will and are already creating the potential to accelerate a transition to a new sustainability paradigm. We are seeing a virtuous circle where the cost of information is falling and the availability of information is dramatically increasing as wireless, LTE, and indeed fibre infrastructure continues to be built out. Almost unnoticeably the mobile Internet has emerged and we can envisage a world where if something has electrical power (which may be scavenged from the ambient environment), it will compute and communicate. This opens up the possibilities and power of ambient intelligence.

Year on year as information becomes cheaper, more of it can be substituted for other inputs such as natural resources, labor, and capital in both business and societal activity. EBook readers such as Amazons' Kindle* are a good but not well reported example of enabling sustainability. Fewer natural resources are consumed through production and distribution of eBooks than physical books and also readers have much more choice of what they want to consume and the books are cheaper. This is a good example of the sustainability paradigm at work almost unnoticeably. It is also a brilliant example of what McInerney and White describe in their book *Future Wealth*—"our ability to substitute information at will for every other resource, including capital and labor, is altering the economy (and indeed society) in a fundamental way." Skype* is another example of a disruptive innovation with a similar "sustainability" effect allowing relationships to be better sustained, business to be more efficient all at a lower cost and with more efficient use of resources. Furthermore professional video conferencing services can create remote meeting experiences that are as good as if not better than physically meeting saving time, money, and resources while improving productivity.

The shift from product to service, sometimes called servicizing, to meet dual goals of improved profitability and sustainability benefit is often enabled by ICT. In servicization, the focus is not on maximizing the consumption of products, but on the services the products deliver with one output being improved resource efficiency. The adoption of cloud computing and software as a service are contemporary examples of this. Rolls Royce with their "power by the hour" program whereby they sell hours of flying rather than aircraft engines, all enabled by remote telematics, was a pioneering example of this concept.

But energy is also getting personal and this is a concept that Intel's CTO Justin Rattner has championed. Increasingly policy and legislation is requiring that consumers have better information on their electricity consumption. A study on smart metering in Ireland showed that more than 80 percent of consumers who had better information on their consumption made a change to

their behavior causing a reduction in demand. Intel Labs Europe is pioneering a new approach called Wireless Energy Sensing Technology, which individualizes energy consumption to devices. Furthermore Intel Labs, through our Personal Office Energy Management software, is creating new infrastructure using personal and participative computing to create much richer building energy management information while helping users to monitor and change the comfort and energy consumption of their own environments. Intel Labs Europe is also extending its sensing and actuation through Wireless Environmental Sensing Technology (WEST), which spans the continuum from energy through environmental to sustainability sensing, and this is discussed in the article on advances in sensor technology to improve individual and indeed community wide contributions to sustainability.

In this issue we consider many topics pertinent to our collective futures. We of course explore concepts of green software and more energy efficient hardware. We explore power management improvements in both Windows* 8 and Android* platforms. We explore trusted personal energy clouds that provide seamless access to energy data but protect each individual's privacy and data. We discuss ways of standardizing measurement of the good effects of ICT deployment while also looking at methodologies to improve data center and storage efficiencies. We discuss how ambient sensing applied in cities can provide better energy efficiency while enabling new services.

More than 50 percent of the world's population live in cities and cities consume more than 75 percent of the world's energy as well as contributing about 80 percent of greenhouse gases. Cities are at the confluence of three megatrends—digital transformations, sustainability, and mass collaboration. The European Internet Foundation, a group of forward-looking European members of parliament have identified mass collaboration as a very significant emerging paradigm. We will see person-to-person collaboration, person-to-machine collaboration, and much machine-to-machine interaction in the Internet of things all enabling sustainable cities.

In London, together with Imperial College London and University College London, we are launching a new Intel Collaborative Research Institute for Sustainable Cities. Together we see the opportunity to collectively research and create an evolutionary leap for cities in terms of resource efficiency, new services and ease of living. Using a quadruple helix innovation approach involving government, industry, academia, and city dwellers, we hope to catalyze and drive a step change in the outcomes for innovation for energy efficiency and sustainability in cities. As part of the research, big data from an ambient intelligence platform supported by a dependable cloud will provide the opportunity to perform real-time optimizations and city visualizations as well as enabling a whole new set of services. We look forward to reporting in future on the research results and outcomes.

As I conclude I want to share the words of Denis and Donatello Meadows and others in their seminal report "The Limits to Growth."

- "Man Possesses, for a small moment in time, the most powerful combination of knowledge, tools and resources the world has ever known.
- He has all that is physically necessary to create a totally new form of human society—one that would be built to last for generations.
- The missing ingredients are a realistic long term goal that can guide mankind to the equilibrium society and the human will to achieve that goal."

Although these words were written forty years ago they seem even more accurate today. The technology is ready, are we? I encourage you to join us on our collective journey to a better future.

Martin Curley

GUEST FOREWORD

David E. Culler and Randy H. Katz
Computer Science Division, EECS Department
University of California, Berkeley

This issue of the *Intel Technical Journal* describes several cutting-edge research and development efforts focused on understanding the dual facets of energy-efficient information technology, to improve the energy consumption of modern processors and their related software systems, and the application of information technology to improve the energy efficiency of the physical world, particularly data centers and smart homes.

We see an opportunity to bring these two facets together, in terms of building the science and engineering knowledge necessary to create efficient, agile, model-driven, human-centered building systems. Buildings are a significant sector of the U.S. economy, and an important environment in which we live and work. Americans spend 90 percent of their lives in buildings, which are in turn responsible for 38 percent of CO_2 emissions, 71 percent of electricity consumption and 39 percent of energy use. Making buildings more efficient, across broad performance metrics, while keeping occupants comfortable, productive, and healthy, are important societal goals.

The twentieth century architect Le Corbusier stated "A house is a machine for living in." This essentially mechanistic view downplayed traditional architectural aesthetics in favor of occupant-centered functionality and comfort. A hundred years later, we still do not think of buildings as machines. Relatively little effort has been expended in making buildings quantifiable "better." Even LEED certification is little more than satisfying a checklist. Traditional building services, physically embedded within the building's structures, are expensive to extend or repurpose. Such changes usually require that walls be torn down, new pipes and conduits be run, and electricians and plumbers be hired. A true engineering approach is needed. Investments in new technologies and conceptual architectures are needed, but the industry's traditional metric is cost-to-build, not cost-to-operate, making it difficult to justify the return on investment to building owners for new extendable "future-proof" technologies. We believe that the only feasible approach is based on the premise that *a building is a machine, and one whose function (including its energy efficiency) can be extended through programming.*

Modern commercial buildings already incorporate increasingly sophisticated *Building Management Systems* (BMS), integrating more parameterized controls with improved sensors and better data collection and presentation capabilities. Web-based interfaces for status and management are now incorporated in many major building facilities. Yet these systems remain stovepipes with simple, decoupled control logic. Their architectures, services, and application programming interfaces (APIs) are not standardized, and often proprietary: only the BMS firm can add functionality. This slows the pace of innovation, buildings remain rigid in the functions and services they provide, and their quality and effectiveness remain difficult to quantify. The control scope is limited to a single building, and the instrumentation data it collects remains within it. These make it extremely challenging to achieve the societal goal of increasing building efficiency.

We believe that the overarching technical challenge is to understand how building functionality can be extended without costly "forklift upgrades," to quantitatively assess new functionality and its impact on building performance, and ultimately, to improve building efficiency. Borrowing inspiration from the rapidly evolving mobile device ecosystem, we believe there is a revolutionary opportunity based on the evolving concept of *Software-Defined Buildings (SDB)*. These buildings extend their function through programmability, implementing new controls, quantitative evaluations to assist in the assessment of building improvement, and modeling to understand and improve such building efficiencies as energy consumption. Buildings that communicate and cooperate could better share facilities, yielding further efficiencies. To accelerate innovation, a third-party developer ecosystem should be facilitated.

The underlying technology trends that are driving the information technology industry provide the essential foundation for the kind of enhanced building environment we envision: ubiquitous wireless communications, inexpensive mobile devices, cheap embedded processing, and essentially unlimited scalable processing and storage in the Cloud. A critical enabler is widely available location awareness with improved indoor accuracy, made possible by ubiquitous sensing and location awareness in modern smart phones that fuses GPS, Wi-Fi* fingerprinting, and other sensory inputs like ambient acoustics, magnetic signature, and motion detection. The phone is becoming the integral means of interaction between individuals, the building environment, its systems, and the ambient intelligent devices embedded therein.

More intelligent modes of occupant interaction would make it possible to deliver just the right amount of service to the spaces that are actually occupied. Co-located occupants could express their control desires, each providing their own possibly conflicting inputs, with the building adjudicating among them. We can exploit mobile devices (phones and tablets) that occupants own and carry as sensors/controllers, not just for location determination, but also to exploit these devices' inherent embedded sensing capabilities. New control methods, such as *Model Predictive Control (MPC),* offer the promise of replacing the difficult to design, tune, maintain and upgrade heuristics-driven decoupled local controllers with a systematic control architecture managing large-scale multi-input multi-output (MIMO) dynamically coupled systems. Methods like MPC can achieve performance guarantees while explicitly handling system constraints.

Software-defined buildings demand a new kind of operational software, a flexible, multi-service, and open *Building Integrated Operating System* (BIOS). Such a BIOS would allow third-party applications to run reliably in a sandboxed environment. It would support sensor and actuator access, access management, metadata, archiving, and discovery, as well as multiple simultaneously executing programs. Like a computer OS, the programs run at a variety of privilege levels, from critical to desirable but deferrable. Such programs access different resources, yet share the underlying physical resources. A BIOS would extend to the Cloud or to other buildings, outsourcing expensive or proprietary operations as well as load sharing, but does so safely with failover to local systems when connectivity is disrupted. Building operators retain supervisory management, controlling the separation *physically* (access different controls), *temporally* (change *controls* at different times), *informationally* (what information leaves the building), and *logically* (what actions or sequences are allowable).

We invite you to read and enjoy the following articles, illustrating Intel's commitment to advanced energy efficiency in processors, computer systems, mobile devices, data centers, and homes.

THE TRUSTED PERSONAL ENERGY CLOUD FOR THE SMART HOME

Contributors

Eve M. Schooler
Intel Labs

Jianqing Zhang
Intel Labs

Adedamola Omotosho
Intel Labs

Jessica McCarthy
Intel Labs

Meiyuan Zhao
Intel Labs

Qinghua Li
Pennsylvania State University

"The proliferation of intelligent energy devices and the corresponding tidal wave of energy-relevant data they generate creates a heightened need for data privacy, security, usability, and accessibility."

The proliferation of intelligent energy devices and the corresponding tidal wave of energy-relevant data they generate creates a heightened need for data privacy, security, usability, and accessibility. We examine these issues in the context of smart home energy management at the edges of the smart grid. We present the design and implementation of a trusted personal energy cloud, a secure information-centric network (ICN) architecture for personal energy data. The system leverages the "social life of devices" to help derive natural boundaries of trust, as well as ways to visualize and navigate the device data and device relationships that populate an energy-aware smart home. We demonstrate how such a layered architecture can be used as the foundation for collaborative analytics in the home and ultimately greater energy savings. Finally, we discuss future directions, including the applicability of the work beyond the home energy domain.

Introduction

With the proliferation of intelligent devices in our lives and in turn a tidal wave of data being generated by these devices, there is a heightened awareness of and need for data privacy, security, usability, and accessibility. We examine these issues in the context of smart home energy management at the edges of the smart grid, where whole-house energy monitoring already generates petabytes of data annually in the United States from smart meters alone. That represents one device per household collecting data every 10–15 minutes; imagine the chatter of tens or hundreds of devices in the home and the details about our lives revealed by that chatter.

We focus our work on energy management because climate change and population growth are societal problems of increasing urgency and because the current electrical infrastructure is inadequately provisioned to meet these challenges. As a result, there has been a global effort to transform the electrical grid into a smart grid, which could support distributed control, bidirectional information flow, the integration of renewables, among many other features, all of which offer greater efficiency and stability over the current grid and offer higher percentages of clean-energy technologies.

While the computing industry has worked hard to make computing devices and computing infrastructure more energy efficient, the energy consumed by these elements represents a mere 2 percent of the overall picture. Therefore, for greater impact, we focus our attention instead on the other 98 percent of the energy consumption, that is, using energy-efficient devices to intelligently

control the rest of the energy consumers. Thus an overarching goal is to enable technology-driven solutions to promote greater energy efficiency overall. And while energy efficiency is a critical first goal, it is sustainability that is the more compelling long-term objective. By sustainability we mean systems that consume no more energy than they are able to generate or store themselves. As closed-loop systems, they have a net-zero energy footprint. Realistically however, net-zero energy systems are rare, and when they do exist, they often qualify as net-zero only some percentage of the time.

"…while energy efficiency is a critical first goal, it is sustainability that is the more compelling long-term objective."

While there is growing concern about the amount of information generated by or stored on the devices that surround us, there is just as much concern about the migration of this data off premises due to its often confidential nature. Once the utilities have installed smart meters that sample whole-house energy usage data once every few minutes (versus what used to be once a month for billing purposes), a tremendous amount of private information can be mined from that data. For example, it is easy to discern when someone is at home or work, how many people are at home, if they are awake or asleep, when meals are served, using which appliances, if a medical device is in use, if the family is away, and so on. Thus there is a need to rethink the assumption that data services should be categorically outsourced to largely anonymous remote cloud services. On the other hand, it may be advantageous to data owners to negotiate with a third party who can monetize a version of that data within an agreed contract.

In this article, we present the design and implementation of a trusted personal energy cloud, a secure information-centric network (ICN) architecture for personal energy data. We describe the features of ICN that recommend it for consideration in a home energy management system (HEMS) environment and the trusted publish-subscribe (pub-sub) data bus we have built on top of ICN that serves as the virtual backplane for the ensemble of devices residing in the home. The system leverages "the social lives" of devices—their ability to identify and to interact with other familiar devices nearby—to help derive and form natural boundaries of trust, as well as ways to visualize and navigate the device data and device relationships in an energy-aware smart home at the edge of the smart grid. We demonstrate how such a layered architecture can be used as the foundation for collaborative analytics in the home and ultimately greater energy savings. In addition, we address the issues of usability and ease-of-use, particularly since novice users (versus IT experts) are typically the caretakers and not the managers of the devices in the "data center of the home." Finally, we discuss future directions, including the applicability of the work beyond the home energy domain.

Vision

In the sections below, we describe the justification for and momentum behind making energy "personal," how that relates to a similar phenomenon underway in the cloud community, and the challenges faced by introducing trust into the design of trusted personal energy clouds.

"Computing is woven into the fabric of our everyday lives, and empowers users by its ubiquity. Over time energy systems will evolve to be personalized and ubiquitous as well."

Personal Energy

More than thirty years ago, computing was centrally managed remotely from users and was far less personal; today, computing is woven into the fabric of our everyday lives, and empowers users by its ubiquity. Over time energy systems will evolve to be personalized and ubiquitous as well. The devices in such systems will be connected by the electrical infrastructure as well as cyberinfrastructure. We call such systems personal energy systems (PES).

Residing at the edges of the smart grid, example personal energy systems include: (1) the smart home, with a small number of unmanaged or self-managed devices; (2) the smart building, where, in contrast to the smart home, an order of magnitude more people operate IT-managed devices that often require integration with existing building energy management systems; and finally (3) communities/cities, collections of houses, buildings, and public infrastructures, where energy concerns have been aggregated, and the issue of private/public boundaries for information sharing is markedly pronounced. Personal energy systems are the last contexts in which user behaviors still have an impact on overall energy consumption.

Clouds

Personal computing is continuously evolving and in turn becoming more complex and more challenging as demand increases to manage, access, and share data seamlessly from any device or platform. As a result, cloud computing has emerged as a new paradigm to deliver computing, data management, and storage as a service to personal computing devices over the network, and it does not require users to know the location or other details of the cloud computing infrastructure.

"A personal cloud is an ensemble of smart devices that you might carry on your person with the intent to seamlessly and efficiently store, sync, stream, and share content across such devices in a trusted manner."

Usually, cloud computing concepts and technologies are associated with large data centers, whereas the personal cloud is an evolving concept that is intended to be a personal, individualized, but highly heterogeneous version of cloud computing. A personal cloud is an ensemble of smart devices that you might carry on your person with the intent to seamlessly and efficiently store, sync, stream, and share content across such devices in a trusted manner. The shift to the personal cloud will only accelerate with the explosion of smart personal devices and as users increasingly use cloud-based services. It is expected that 90 percent of web connected devices will have consumer cloud services by 2013[1] and it will be a USD 12 billion industry by 2016[2].

We call the integration of the personal cloud with personal energy systems the *personal energy cloud*. We argue that the personal energy cloud will substantially improve system manageability, efficiency, performance, and most importantly, energy savings, as compared to systems without more localized control, the resources of scoped clouds and finer-grain instrumentation for resource management. However, from the user's perspective the impact may be felt more significantly in other areas, such as convenience. The personal energy cloud does not exist in isolation. It is meant to nest inside of or intersect with multiple scopes of control within which one might belong: the smart home

cloud, the smart building cloud, the smart city cloud, and so on, as shown in Figure 1. These multiple trusted clouds enhances users' capabilities to monitor, measure, and control energy devices and share energy resources in a pervasive and ubiquitous manner. Supported by cloud computing technologies, collaborative analytics of large amounts of data across different devices and domains begins to look not only feasible but also effective at saving energy (see the section "Collaborative Analytics").

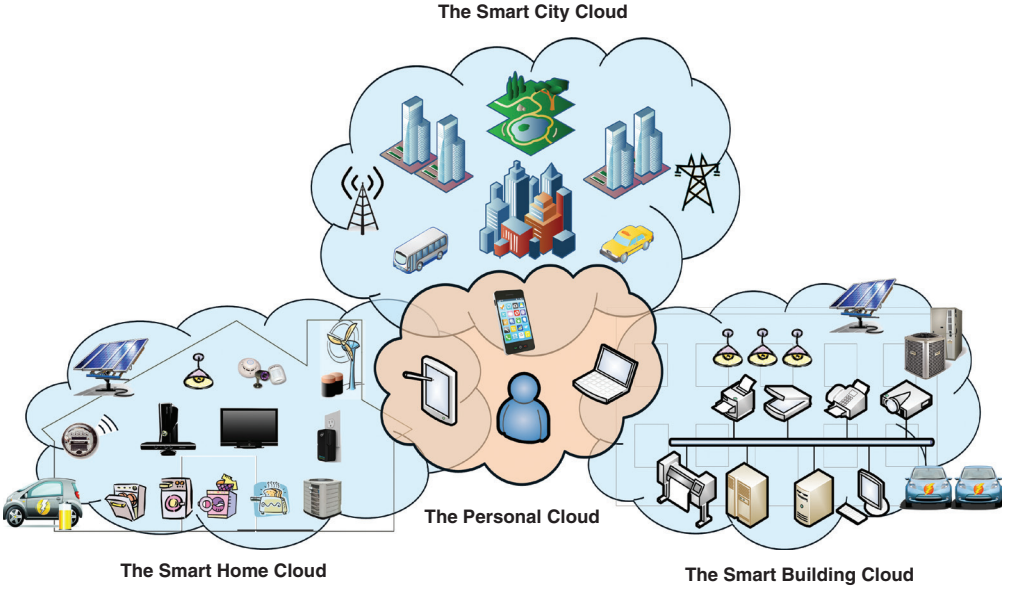


Figure 1: The personal energy cloud (Source: Intel Corporation, 2012)

Trust

In the cyber world, trust is based on the relationships developed between collaborating entities working together to achieve certain goals. Generally, trust is an act of faith; confidence and reliance in something that's expected to behave or deliver as promised[3]. Likewise, trust in a trusted personal energy cloud (TPEC) is a guarantee the whole system behaves as expected, including that information sharing preserves privacy, building actuation decisions are based on authentic energy usage data, and that data has not been tampered with between the time it was generated and the time it gets consumed by analytics.

"Trust in a trusted personal energy cloud (TPEC) is a guarantee the whole system behaves as expected, including that information sharing preserves privacy, building actuation decisions are based on authentic energy usage data…."

Users, devices, data, and/or applications are not equally trustworthy, despite the fact that when we set up our wireless networks in our homes and offices we treat them as such. They have different privileges and services, and have different capabilities to prove their behavior. As a result, there are varied obligations and constraints that really should be placed on data and device access, particularly since device connectivity and the ubiquity of data generation and collection are accelerating. Similarly, not all clouds are to be equally trusted. Thus in TPEC different logical trust "boundaries" should be

established. A boundary defines a group or cluster of devices that are expected to abide by the same set of rules or to satisfy the same set of requirements. We argue that energy services should reside at naturally-occurring public/private boundaries of the application domains. For energy those domains might exist as the personal domain (all the devices on one's person or that are typically mobile with the person), the domain of the home, the neighborhood, the office, public facilities, and so on. Sometimes, trust should be tied to a particular cluster of devices due to their behaviors, such as, for example, a group of devices exhibiting certain patterns of communication[4] (see the section "Derivation of Trust Boundaries"). Trust also needs to comply with the access control polices based on different privilege settings[5].

"Resources encompass not only information, but also energy, generation, storage and usage."

Note that resources encompass not only information, but also energy generation and storage, given the growing availability of the electric vehicle battery and photovoltaic cells, and the eventual bidirectionality of energy flowing to/from any device with the capability of generating, sharing, or storing energy. A personal energy management system should be able to verify the electrical energy flow under the support of information technologies, as well as personal cloud computing.

The Trusted Personal Energy Cloud Architecture

We propose a multi-layered architecture, shown in Figure 2, to implement the trusted personal energy cloud, which could be deployed in the smart home, the smart building, and the smart city. In this section we briefly introduce each layer and the motivations behind them, and then give a detailed presentation of a trusted data bus for the smart home, the fundamental component of the architecture.

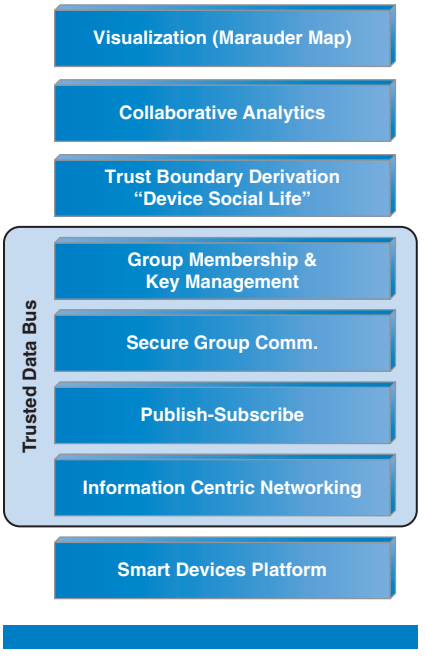


Figure 2: Trusted personal energy cloud architecture
(Source: Intel Corporation, 2012)

Overview

The trusted personal energy cloud faces a number of nontrivial challenges. To provide intelligent energy management, it must support the transmission of large amounts of ambient sensing data, measurements, and actuation commands among energy devices and smart personal computing platforms for monitoring and control. Due to the large number of heterogeneous devices, especially mobile devices that move between multiple network domains, dynamic data and service discovery become critical issues. The trusted personal energy cloud also aims to establish trustworthiness among devices as well as services, and to protect highly confidential and private information. Furthermore, the typical user is not an IT expert and therefore expects simplicity and usability from the system.

A foundational part of the TPEC, the *trusted data bus* is composed of a number of layers in the communication architecture and provides a virtual backplane across which data can be published and subscribed to by any device. Although we use it for energy management, it is generic enough to be repurposed for home automation, healthcare, entertainment applications, and so on. On top of the trusted data bus, we create a component with the ability to derive

the social life of devices that can be used to establish trust boundaries for groups of devices for secure group communication. We also develop a trusted collaborative analytics framework, which infers user presence and in turn how best to reduce energy usage when devices, rooms, and the house itself are unattended. The goals of trusted analytics are (1) to demonstrate building trusted analytics from trusted data, and (2) to deliver inference-based solutions to promote greater energy savings. Finally, to demonstrate the architecture's capabilities and usability, we design a "Marauders' Map*[6] for Personal Energy," a visualization tool to make the invisible visible—home energy usage, communication patterns, and smart device trust relationships. In the rest of this section, we will focus on the trusted data bus and defer the detailed introduction to other components to following sections.

Trusted Data Bus

We implement a proof-of-concept ICN-based system based on PARC's CCNx[7], creating a test bed to illustrate the trusted data bus for home energy management systems (HEMS) in an example smart home cloud shown in Figure 3.

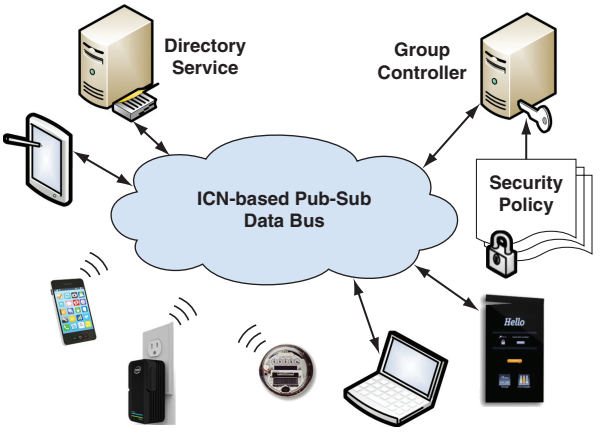


Figure 3: Example smart home cloud (Source: Intel Corporation, 2012)

Information Centric Networking

The trusted personal energy cloud presents a unique opportunity to experiment with "clean slate" Internet ideas, because the cyberinfastructure of the smart grid is largely in the early stages of development. Because these Internet clouds are smaller-scoped and administratively-independent, there is even greater flexibility to experiment with next-generation Internet ideas that would otherwise be difficult to deploy on a larger scale. In this work, we are particularly interested in information-centric networking (ICN)[8][9][10]. We argue its unique features can significantly benefit the design of the trusted data bus and HEMS context.

"The trusted personal energy cloud presents a unique opportunity to experiment with "clean slate" Internet ideas ... In this work, we are particularly interested in information-centric networking."

The principal idea behind ICN is to allow users to ask for data in an information-centric manner regardless of the data's physical location. A data

unit is requested, routed, and delivered via its name rather than its IP address. Secondly, ICN supports distributed content caching in the network, making the finding of data easier under conditions of device mobility and intermittent connectivity. Thirdly, ICN provides self-contained data security. Data is signed or encrypted upon creation. The security of the data does not rely on end-to-end security protocols that focus on the protection of the channel like IPsec or SSL.

These features strongly support secure and efficient data sharing in TPEC. A device can generate encrypted and/or signed data and publish it to the cloud independent of recipient locations. The data can be duplicated and stored in a distributed manner across the cloud with multiple copies, yet any data subscriber can request data by its name without knowing the exact storage locations. This mechanism maximizes the data reuse for varied applications and strongly facilitates one-to-many communications, which are needed by measurement-rich infrastructures like HEMS. Since the security of the data is tied to the data encryption rather than to the trust between data senders and receivers, ICN guarantees data privacy and access control even if the data is not hosted by the originator. (Later the section "Secure Group Communication and Group Key Management" describes how the veracity of data is managed.)

"Our interest in ICN is to explore its advantages (over TCP/IP) to provide more intuitive content naming and location, efficient data sharing, intrinsic security properties, and simple extensibility."

Thus our interest in ICN is to explore its advantages (over TCP/IP) to provide more intuitive content naming and location, efficient data sharing, intrinsic security properties, and simple extensibility.

Publish-Subscribe (Pub-Sub)

The example in Figure 3 of a trusted smart home cloud depicts a variety of personal handheld devices, appliances, and IT equipment typically found in a home, along with HEMS-oriented devices such as ambient sensors and a smart thermostat. In HEMS, devices publish data to and request data from other devices to perform energy management functions. For instance, a thermostat collects home temperature data from an ambient sensor and issues a "stop" command to a heater when the temperature reaches the resident's comfort setting. In ICN parlance, the thermostat expresses its *interest* in the room temperature data, the ICN cloud routes the *interest* packet to the sensor, and returns temperature data. If the temperature reaches the threshold, the thermostat publishes or pushes a "stop" command to the ICN cloud. Because the heater has previously expressed *interest* in commands from the thermostat, the cloud will deliver the "stop" command to the heater.

Note that some data will be requested by multiple devices in the home. For example, the room temperature can be subscribed to by a thermostat, the resident's smart phone, and a tablet that serves as a home hub where multiple residents go to check information. Another example is that the whole-house home energy usage data is subscribed to by a smart meter for an electric vehicle (EV) charging scheduler (to avoid transformer overload in the neighborhood), and possibly even a neighborhood-level energy data aggregator (to coordinate across homes' EV charging patterns). Similarly, a single device can subscribe

to a multitude of data; for instance a machine serving as a data archiver in the home might subscribe to all sensors generating data in the house. Therefore, pub-sub enables one-to-many and many-to-one communication found in HEMS. Although CCNx promotes the pub-sub paradigm at the packet-level, enhancements are required to support a range of HEMS applications. As a result, we design an upper-layer pub-sub API for HEMS, which extends the CCNx protocol with long-lived forwarding information (for both interest and data content) in the intermediate forwarding nodes and the publisher, such that persistent and periodic subscriptions are supported. As a result, the subscriber only needs to send *interest* once. After the data content is forwarded back, the forwarding table entries about the data and the pub-sub relationships, as well as the *interest*, are kept alive at intermediate forwarding nodes and the publisher, instead of being deleted as is done in the current CCNx. When new content is generated, the publisher can directly push the data to the network, and the data will be eventually delivered to the subscriber following the "alive" forwarding path.

"Pub-sub enables one-to-many and many-to-one communication found in HEMS."

We do this because it is very likely for a resident to request context updates regularly rather than by one-time subscription only. For example, in a power monitoring application, the resident would like to be notified of any major power event, like "the heater turned on." She subscribes to the power event data once and then receives notifications whenever something major happens. Such subscription is considered *persistent*. In addition, in an ambient sensing application, a home energy dashboard tracks temperature changes in the home. Since the thermometer is continually generating new readings, it is inefficient and undesirable to receive all new data. It might be sufficient to receive the temperature data at more spread out but regular intervals. In this case, the subscription is considered *periodic*.

Secure Group Communication and Group Key Management

The trusted data bus must address a number of security challenges[5][11]. In this work, we focus on confidentiality, integrity and authenticity. Energy data often contains sensitive information about people's lives. Leakage of such information may be extremely harmful and compromise people's privacy. Confidentiality aims to preserve data privacy and prevents unauthorized entities from accessing private energy data. Tampered or falsified HEMS data may cause unexpected or even dangerous side effects. For example, an air conditioner may not be turned on if the temperature reading is altered during transmission, and this may present a health risk during a severe heat wave. Integrity is intended to prevent unauthorized data modification and guarantee data authenticity.

Within the smart home cloud, the publisher and subscribers of the same data form a pub-sub group. However, to ensure that only group members gain access to data and that unauthorized members are prohibited, we designed a secure group communication scheme enhancing existing CCNx protocols. While vanilla CCNx protocols provide integrity and authenticity using digital signatures, our contribution is to provide data confidentiality through group

"Only group members gain access to data and … unauthorized members are prohibited."

key management. Figure 4 shows the full system protocol stack and the data format of the messages of the trusted data bus. Before the message payload is signed and encapsulated by CCNx, it is encrypted using the Blowfish[12] algorithm. If the payload is application data, like a room temperature reading, it is encrypted by a symmetric key shared by group members. If the payload is group key management data like a shared key, it will be encrypted by key encryption keys (KEK).

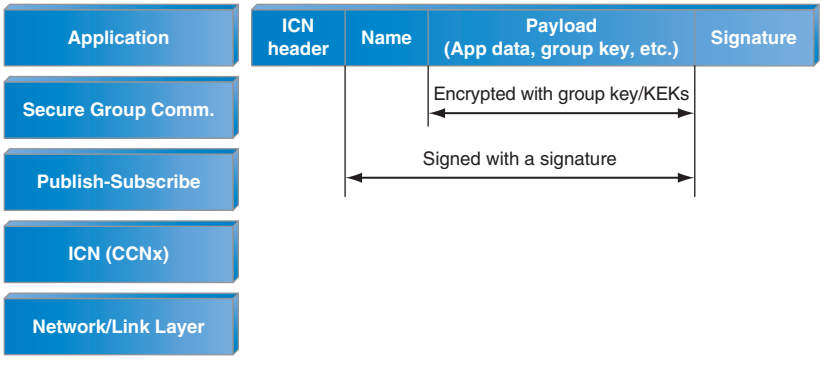


Figure 4: Trusted data bus: protocol stack and message data format (Source: Intel Corporation, 2012)

The approach relies on the existence of a group controller (Figure 3) that manages group memberships, enforces security policies, and distributes group keys. Before a device publishes (or subscribes to) data, it needs to create (or join) the group with which the data is associated and then to create (or obtain) a group key that essentially allows access to the data. The group controller also checks the security policies to verify if the requesting device has been authorized to access the data. If so, the group key is issued. The publisher encrypts data and the subscribers decrypt the data using the shared group key. Data privacy is preserved since unauthorized nodes cannot read the data content without the group key.

"Data privacy is preserved since unauthorized nodes cannot read the data content without the group key."

The group key must be updated when a device joins, leaves, or is evicted from the group. The generation, distribution, and refresh of group keys is considered group key management and is handled by the group controller. In this work, we designed an ICN-based group key management protocol based on the one-way function tree (OFT) algorithm[13], which provides a good balance between security and cost. Because home energy management systems are fairly static in terms of device dynamics, group key revisions are infrequent.

Another challenge for the secure group communication scheme is the management of access control policies, which may be different for data and commands. An example policy rule might be "Only the HEMS dashboard can subscribe to the living room temperature." Although there are many options for policy languages, they have varied degrees of complexity and sophistication. A simple language would support simple access control lists, identifying who can access which objects. A richer language might support distinct policies

for individual functions on an object and other conditions. Languages exist at all extremes from the simplest to the full-fledged programming languages, such as Ponder. The tradeoffs are among usability, utility, and specificity. In addition to using traditional policy languages, or even extending rules into a service level agreement, the autonomic derivation of trust boundaries (see the section "Derivation of Trust Boundaries") is another option that can be used to establish groups.

Directory Services

We also design a directory service, shown in Figure 3, for data registration and discovery. It organizes and provides access to available data by name. The directory service helps a device discover data in which it might be interested and registers data via its names for publishing. The directory service guarantees the uniqueness of data names in the domain it covers.

Derivation of Trust Boundaries

Previously we acknowledged trusted clouds as useful for privacy preservation of confidential data within the home. In the energy realm, the cloud is a construct that helps establish with which other devices you entrust sharing your data—for analytics, data migration, long-term storage, and so on. However there are other applications that could benefit from a trusted collection of devices that create a virtual boundary across which policies must be maintained. One is home, building, or city automation, where one device allows another to act as its proxy, for instance when a capability-challenged device pairs with another device to acquire that capability; the trusted cloud serves as a mechanism that vets proxies. Only devices that meet certain criteria qualify as trusted enough to gain membership in the "proxy" cloud of trust.

"…the cloud is a construct that helps establish with which other devices you entrust sharing your data—for analytics, data migration, long-term storage, and so on."

Given the importance of ease-of-use, the trusted cloud could also play an important role in simplifying the tedious task of bootstrapping new devices. When a new device brokers to join the trusted cloud in the home, it essentially trusts its peers in the cloud enough to borrow their default configuration, installation, and setup parameters. This use of the cloud would go a long way toward the objective to make home energy management usable by novice administrators, who, studies show, have serious difficulties deploying and managing their own home networks, due to the complexity of the tasks[14]. Furthermore, users have a limited attention span for configuring devices[15] and the continued level of interest in managing devices drops precipitously after a very short period of time[16].

"Given the importance of ease-of-use, the trusted cloud could also play an important role in simplifying the tedious task of bootstrapping new devices."

If we can agree that there are many roles for trusted clouds, the question then becomes: how to establish trust among unfamiliar devices that have no prior context and begin with no common domain of trust? We leverage the "the social life of devices," the emergent social relationships among devices, to mitigate this issue. The idea of devices having social lives goes beyond the social networks of their users. The social relationships between devices can be established via historical interactions and/or observations of similar temporal

"We believe the notion of device social lives could be used to reinforce trust among devices, much like what people do when they organically build trust with others."

and geographical behavioral patterns and properties. We believe the notion of device social lives could be used to reinforce trust among devices, much like what people do when they organically build trust with others. If devices naturally organize into groups—based on their observation and recognition of being regular co-travelers and exhibiting similarities, then the implication is that we can reuse these groupings to begin to establish trusted cloud boundaries.

To test this possibility, we implemented a Scavenger Hunter application that ran on devices to passively monitor the Wi-Fi* channel for other devices in its vicinity, in order to establish social contexts. This approach revealed which devices were regularly co-located, the percentage of time the devices were nearby to each other, how nearby, and so forth. We used spectral clustering techniques to identify social groups from the Wi-Fi data. Spectral clustering is one of the most widely used and theoretically reliable methods for data preprocessing and analysis[17]. At the high level, a spectral clustering algorithm takes the input of a data adjacency graph W, which is specified by the user and represents pairwise similarities w_{ij} between all data points. The output is the clustering of the data points that maximizes the within-cluster similarity and minimizes the similarity between the clusters.

In Figure 5, we display excerpted results from a four-month study, whereby a laptop regularly but passively scavenged the Wi-Fi channel for the MAC addresses of other devices on the network and from this sensed data built models of its social contexts and social life. The clustering algorithms discovered that the laptop regularly participated in 8 significant social contexts (or "social lives"), and denotes these contexts as the 8 social groups within which it participates. The two most prominent are home and work (RNB6), and several others that represent travel between Intel sites via air travel (SJ jet

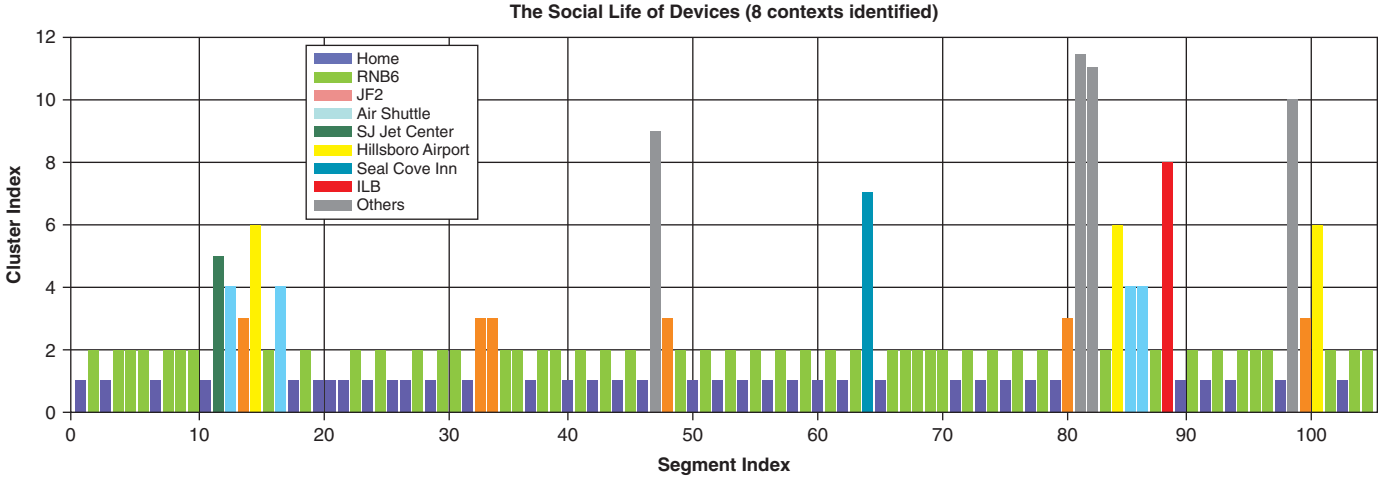


Figure 5: The social life of devices
(Source: Intel Corporation, 2012)

center, Air shuttle, Hillsboro airport), and another local Intel site (ILB). The importance of these results was the establishment of a simple criterion for trustworthiness: persistent proximity or familiarity.

The Wi-Fi Scavenger Hunter study implies that one could bootstrap trust of devices via a variety of sensing information. When no prior relationship exists, other sensed input provides the preliminary knowledge basis for an authorization policy that can be potentially created and enforced to manage secure pub-sub groups.

"When no prior relationship exists, other sensed input provides the preliminary knowledge basis for an authorization policy that can be potentially created and enforced to manage secure pub-sub groups."

There are many diverse criteria that could be used as candidate social properties for clustering devices as similar and/or socially related and therefore as having greater potential to trust each other:

- *Network behaviors.* Devices that commonly communicate with each other or with similar servers, use the same wired or wireless networks, run similar protocols, or meet certain network performance criteria (latency, connectedness, reliability), and so on.
- *Common hardware capabilities.* Devices with similar hardware security modules.
- *Shared physical environment.* Devices that are near each other, that is, are located in the same room.
- *Shared user/owner attributes.* Devices that are used or owned by the same user.
- *Application–driven associations.* Devices that are related by way of the application(s) they serve.

Collaborative Analytics

One of the critical goals of our trust-based architecture is to demonstrate trusted collaborative analytics. A first demonstration is collaborative energy management, the notion that we can correlate across trusted measurement data from multiple devices that populate the home to infer how best to reduce energy usage.

To accomplish this task, we explore the premise that up to 25 percent of energy consumed in residential buildings is due to devices that are unattended. Either the devices are on but unused, or they are off but are still plugged in and still consume energy (also known as vampire loads).

"Up to 25 percent of energy consumed in residential buildings is due to devices that are unattended."

Ideally, if we knew where users were located, then the home could be actuated to turn itself off when no users were present. In other words, do with homes what is done on the device platform; enact sleep states. For the smart home the sleep states exist at the device, room, and whole-house level, which are triggered by the presence or absence of users.

One approach is to infer user presence from household energy consumption data, as shown in Figure 6. We test this theory by analyzing a multiyear dataset

of energy usage data for a home in the southwestern United States. This dataset tracks energy usage for devices, appliances, IT equipment, as well as the home infrastructure itself (HVAC, heater, pool pump, and so forth). In addition, the dataset includes other sensor data, such as ambient temperature, humidity, cost, and CO_2 and light levels. Sampling rates range from every minute, to every 15 minutes, to every hour, and we explore the accuracy of the predictions as the fidelity of the data varies.

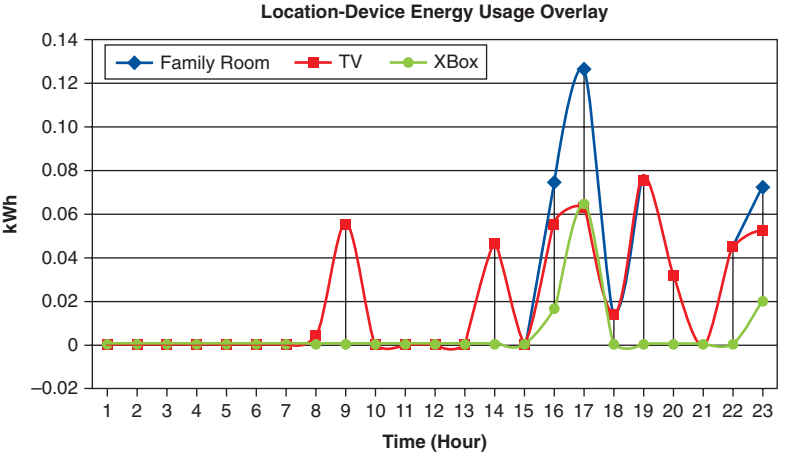


Figure 6: User presence. Correlated Xbox* and TV usage may indicate user presence in family room.
(Source: Intel Corporation, 2012)

"We posit that one can derive presence from historical data that reveals norms of the energy signatures of devices, rooms, and the whole house."

We posit that one can derive presence from historical data that reveals norms of the energy signatures of devices, rooms, and the whole house. These norms show when a device, is on, in use, sleeping, and/or off. For a high percentage of devices, if there is elevated energy consumption, it indicates usage by a user and thus user presence. Similarly, when a device is in a sleep state or off, it implies absence. Because we have a mapping of devices to physical locales, we can extrapolate further and use device level presence to deduce room level presence. Certain devices, such as an electric vehicle, might also imply whole house energy status; as the car comes and goes, user presence could be determined at a coarse granularity and the home could be put into the proper energy saving mode—more aggressive savings when no one is home, and less aggressive when users are present, but in known rooms within the home.

In Figure 7, we provide a real example that shows the norms for a whole-house energy signal, which reveals the absence of residents during a week-long stretch at the end of March 2010. This data was collected at a frequency of every 15 minutes.

"Once user presence is inferred, and recommendations for device, room, and whole-house actuation are similarly derived, we can also quantify the savings that would result from these changes…."

Once user presence is inferred, and recommendations for device, room, and whole-house actuation are similarly derived, we can also quantify the savings that would result from these changes; savings can take the form of cost, kWh,

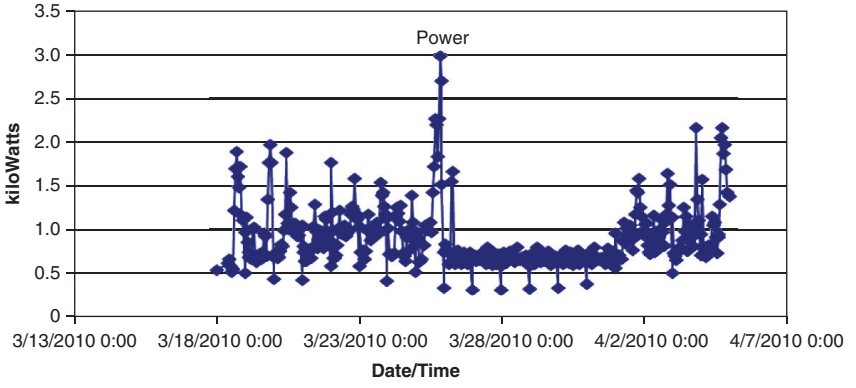


Figure 7: Whole-house power usage (Source: Intel Corporation, 2012)

CO_2 footprint, and so on. Moreover, these recommendations can be construed as a first step toward sustainability or net-zero energy usage. If we know how much energy is consumed, we know how much energy must be generated or stored to offset it.

Unlike a typical enterprise network that often supports centralized services (for example, the collection and analysis of data), the smart home offers the possibility that intelligence is distributed across the many devices and sensors that populate and are embedded in the space. As a result, analytics may be performed across a variety of devices, across a variety of data, and may be managed in a peer-wise fashion, which does not totally preclude centralization. Distributed analytics removes the existence of a single centralized point of failure or attack, it avoids transmission of data to a central (often remote) locale, and places decision making closer to the devices themselves. This benefits homeowners because greater control rests in their hands, and this benefits energy suppliers because it leads to a more highly instrumented infrastructure, with the potential for greater robustness.

The accuracy of derived user presence information can be improved if energy data is combined with other available data sources such as lighting data that might reveal when lights are turned on due to user activity, or perhaps audio data that might reveal when and which users are nearby (via footsteps or voices). Similarly, the actuation of the home states is not always a simple function of user presence. First, there is the reality that there are a range of device behaviors, some of which have strictly Boolean on/off characteristics, as shown in Figure 8, while there are others, shown in Figure 9, that exhibit a saw-tooth usage pattern (HVAC or refrigerators that have high and low temperature set points), or cycle, as shown in Figure 10, through multiple sleep states (IT equipment, game consoles), or never really revert to zero energy usage because they draw vampire loads when supposedly off, and so forth. Thus, to actuate these devices for energy savings means something different

"The accuracy of derived user presence information can be improved if energy data is combined with other available data sources such as lighting data that might reveal when lights are turned on due to user activity, or perhaps audio data that might reveal when and which users are near by (via footsteps or voices)."

"…consider presence as only one of several variables (comfort, cost, CO_2, footprint, and so on) to be prioritized by an optimization scheduling algorithm."

depending on the type of device it is. Second, more sophisticated techniques are under development that consider presence as only one of several variables (comfort, cost, CO_2 footprint, and so on) to be prioritized by an optimization scheduling algorithm. Furthermore, these techniques are likely a function of the preferences of each individual user or the combined preferences of several users present and are affected by decisions to turn on/off devices (such as heating or lighting).

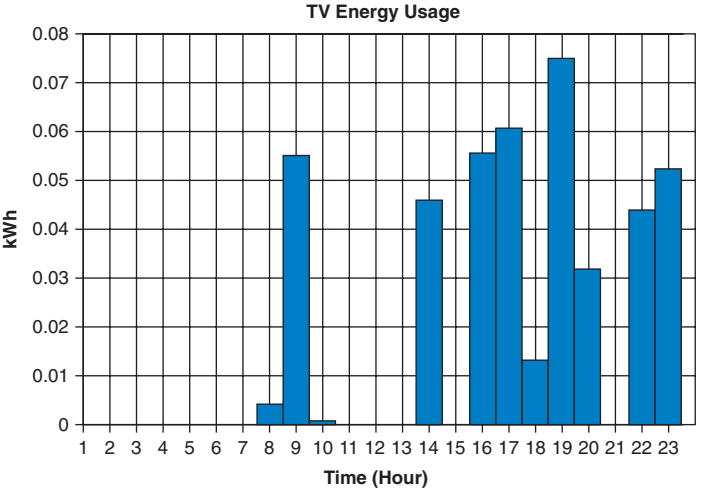


Figure 8: Boolean device usage
(Source: Intel Corporation, 2012)

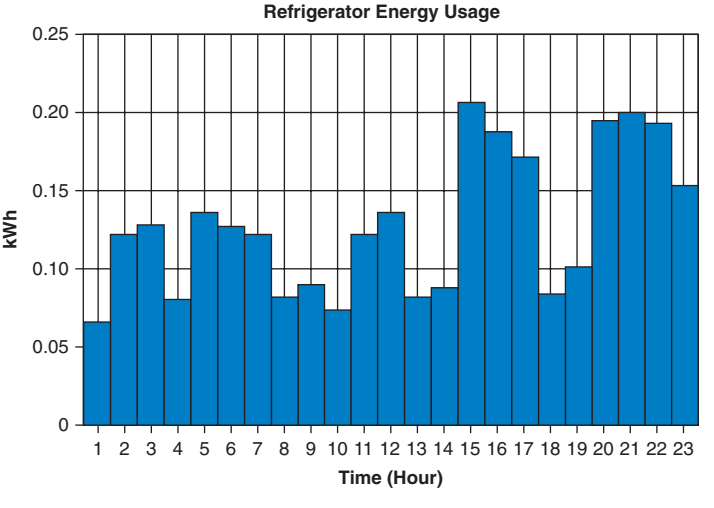


Figure 9: Always-on "sawtooth" device usage
(Source: Intel Corporation, 2012)

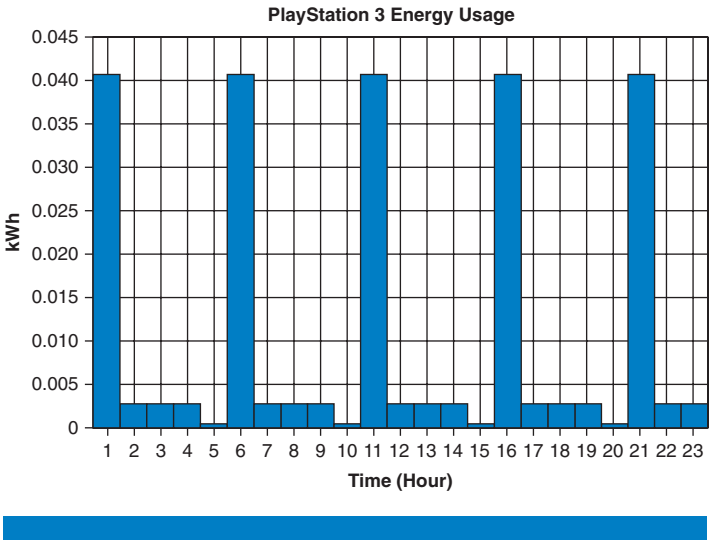


Figure 10: Cyclic device usage
(Source: Intel Corporation, 2012)

In addition to leveraging the availability of energy data for collaborative analytics for energy management, the same energy data can be used as input for collaborative analytics for the purpose of security—the security of the home itself, the security of the devices in the home, the network in the home, and/or the people who live in the home. Because of the availability of historical data, for each device that generates an energy signal we can derive its behavioral norm. When you combine presence and energy usage data and observe it over the course of a month, one can begin to learn things not only about the home's energy consumption but about the activities, trends, and day-to-day patterns of its inhabitants. Thus the energy data is but one of several data streams that can be used for anomaly detection and can be used to corroborate anomalies across the data streams. For instance an anomalous energy usage might indicate that a device has been compromised with malware or may indicate something benign like an HVAC sensor is misbehaving, causing the device to stay on instead of shutting off as designed. Comparing anomalies across data streams can reduce the incidence of false positives.

"…the same energy data can be used as input for collaborative analytics for the purpose of security—the security of the home itself, the security of the devices in the home, the network in the home, and/or the people who live in the home."

"Thus the energy data is but one of several data streams that can be used for anomaly detection and can be used to corroborate anomalies across the data streams."

Visualization

Whereas Intel traditionally cares about performance metrics such as MIPS or latency, inside the digital home a key metric is usability. Although this might be considered a "soft" metric (by comparison to CPU speed), it is a critical concern in the home context, given that it is the average home owner (as opposed to an IT department or expert) who will be maintaining the proliferating numbers of devices (15 billion devices connected to the Internet by 2015).[18] Toward this end, we have developed the idea of a personal energy map for energy data within the smart home. Alternatively we refer to this map

"…inside the digital home a key metric is usability."

as the Marauders' Map for Personal Energy, because it aspires to have certain properties reminiscent of the magical map in the Harry Potter books. We describe this further below.

"…make the invisible visible."

The motivation for a map has several dimensions. First and foremost is the desire to make the invisible visible. Many if not all of the smart home devices and data are invisible to the average user. Home owners rarely know just how many devices cohabitate the space with them, and furthermore how many of those are beaconing data to who knows where. The theory is that by making something visible (energy, security, device locations, data storage, communication), it becomes tangible, can be named, observed, can acquire a reputation, and users can begin to build relationships with it. Having a relationship also might be a first step toward motivating users to care about the devices and the data, and in turn begin to care about securing these resources as well as about reducing how much energy they consume.

"…make data easily navigable."

Secondly, there is a strong push to make data easily navigable. In the context of the smart home, make it self-evident to the user what measurements exist, at what granularities, over what durations of time. What metadata is accessible, to whom and under what conditions, that is, who owns the data, who has permissions to view or alter the data, who can share, store and/or analyze it? Can visualization help us understand the boundaries of accessibility of data? In other words, denote where data should not be allowed to flow.

Third, as devices are often proxies for the people that use them, devices begin to be anthropomorphic. They begin to exhibit patterns that reflect the people who own them and to have their own social lives. A map can serve as a visual IM or a visual directory service that notes the comings and goings of devices. It can reveal device status (on/off, asleep, away), a device's friends (with which other devices it communicates regularly, with which other devices it shares friends, with which other devices it has overlapping habits, et cetera), and the trustworthiness of devices (either based on direct observation or on the ratings collected from other devices).

"…the map is meant to amplify our understanding of energy, security, and cloud-related capabilities of the devices that populate the smart home."

From the perspective of a trusted personal energy cloud, the map is meant to amplify our understanding of energy, security, and cloud-related capabilities of the devices that populate the smart home. Figure 11 displays a first rendition of a personal energy map of the smart home, which reveals a golden dot for each of the energy-consuming devices in the house, and the closer view reveals the energy usage of the Xbox, TV, and PlayStation* in the family room over the course of a day in the life of the home. It is essentially a stylized blueprint of the home, on top of which is an overlay of the locations of the devices. The map is a helpful metaphor because too often users, devices, and data have a strong cyber-presence yet their actual locations in the physical world are unknown or highly imprecise.

In the future, we would like the map to illuminate resource sharing opportunities, such as, for example, for energy harvesting: does a device run off a battery? How quickly does the battery charges? How much of a device's

Figure 11: The Marauders' Map for Personal Energy (Source: Intel Corporation, 2012)

battery is presently charged? What percentage of the battery is available to share? Another form of resource sharing is the data itself; seeing what data lives where in the network and whether it lives redundantly on the network. More generally, the map could allow devices to reveal visually a range of their resources available for sharing (CPU, storage, battery, network, and display).

Given the potential of Scavenger Hunter type algorithms, the personal energy map is used to display not only the reputation of individual devices, but also the derivation of groupings of devices into trusted clouds, as described earlier. For instance, which devices are near each other for large chunks of the day, which devices have the same owner, which devices all run the same applications, and so on. Whatever the trusted cloud criteria, the map can be used to depict the group of devices that comprise the cloud and the trust criteria that leads to the association. At present a trusted cloud grouping is shown as a connected graph among the devices that are logically associated with each other.

"…the personal energy map is used to display not only the reputation of individual devices, but also the derivation of groups of devices into trusted clouds…."

"…the map can be used to depict the group of devices that comprise the cloud and the trust criteria that leads to the association."

For security, there is metadata that is often missing from data stores, such as who owns a device, where the data has been (its lineage over time), whether the data is firsthand information or derived, inferred, or processed, and so forth. Additionally, is there a trust rating that can be assigned to a device that indicates its trustworthiness, while at the same time providing a means to decide the riskiness of interacting with that device?[19] Visually this might

take the form of a device glowing green when it is reputable but red when it is suspect. Perhaps this visualization could be an extension to what is used by the presence/usage inference algorithm that lights up a device in deeper and deeper shades of yellow to orange to red to denote presence, usage, activity, or anomalous behavior, that is, how much a device's usage deviates from its norm. The relative trustworthiness of trusted clouds could be conveyed in the future by colorizing the connected graphs of the trusted clouds.

Conclusions and Future Work

"…we discuss the inevitable personalization of energy as energy resources and concerns become increasingly localized and pervasive."

"We present the design and implementation of a secure information-centric network architecture for the trusted personal energy cloud."

In this article, we discuss the inevitable personalization of energy as energy resources and concerns become increasingly localized and pervasive. We define the need for and coexistence of the trusted personal cloud with traditional clouds and argue that energy services should reside at naturally-occurring public/private boundaries, where privacy and security can be suitably managed.

We present the design and implementation of a secure information-centric network (ICN) architecture for the trusted personal energy cloud. We highlight the features of ICN that recommend it for consideration in a HEMS environment, and lobby that its ability to support data caching, name-based routing, and self-securing data enable a seamless and secure data model for intermittently connected mobile and/or low-power devices. We build a pub-sub data bus on top of ICN that serves as the virtual backplane for the ensemble of devices in the smart home, and discuss the secure communication protocol and group key mechanisms developed that ensure its security and privacy preservation. We describe the use of spectral clustering techniques to infer the "social life of devices" from passive sensing (of a device's Wi-Fi channel) and propose the use of these derived social groups to establish natural boundaries of trust. The output of the scavenger hunter serves as the input to the trust model and the group key management for the ICN-based pub-sub groups. We provide this as an example of usable security and privacy techniques at the edge of the smart grid that are designed for novice users who are more likely to be the system administrators of their homes than IT experts.

At the upper layers of the TPEC architecture, we provide an example of trusted collaborative analytics being built from a foundation of trusted measurement. We present early results of collaborative energy management, which infers user presence from energy signals and quantifies how much energy could be saved if a device, room, or whole-house could be actuated to be turned off or put to sleep when unattended. The algorithm calculates device norms, which in turn are helpful for the more general problem of energy anomaly detection and which could be repurposed for household security, home network manageability, and other applications. It is a first step on the path toward understanding what the house requires to achieve net-zero operation. Finally we describe our efforts to promote usability and ease-of-use, by designing a

personal energy map to help users visualize and navigate the device data and device relationships in an energy-aware smart home. The map is also meant to amplify our understanding of energy, security, and cloud-related capabilities of the devices that surround us. In the process, we hope to raise consumer awareness of privacy preservation.

There is much work to do to evaluate the effectiveness of the various techniques designed and the user experiences that they create. For example, it is critical to assess and deploy the policy languages that essentially "patrol" the boundaries of trusted clouds and to experiment with ways to bring them to life through policy visualization for average users and IT managers alike. There is the equally daunting task to enable seamless integration of devices into the personal energy cloud, such that an existing network and a set of policies can make sense of the new device and its inputs. We are investigating such standards as SEP 2.0, the smart energy profile, which enumerates the kinds of devices allowed to participate in the smart home cloud.

Going forward, we ask, what is the applicability of our research beyond the home energy domain? Over 50 percent of the world's population now live in cities and collectively these cities are responsible for over 70 percent of the world's energy consumption. This presents us with huge challenges and opportunities to use smart technologies to improve and enhance the daily lives of city inhabitants. The research challenges of the trusted personal energy cloud could be scaled and applied in an urban setting as the "trusted city energy cloud" and take into account the natural urban aggregation points. Immediate questions that come to mind: can ICN scale to support the urban-scale data bus and the numbers of participatory and embedded sensors? Do the inference algorithms for presence and for the social life of devices translate to the cityscape? How are algorithms complicated by the absence or sparseness of full information?

"Over 50 percent of the world's population now live in cities and collectively these cities are responsible for over 70 percent of the world's energy consumption."

"The research challenges of the trusted personal energy cloud could be scaled and applied in an urban setting as the 'trusted city energy cloud' and take into account the natural urban aggregation points."

Scaling up brings with it many complex scientific challenges that need to be addressed, including possibly the gathering, visualization, optimization, and securing of large-scale, real-time energy datasets in an urban setting. This aggregated energy data has the potential to present an "entire city view" to the various relevant stakeholders, such as city planners and managers, to make informed decisions and also provide citizens with the decision-making power to reduce not only their energy footprint but their carbon footprint.

Acknowledgment

We thank our many collaborators and co-travelers in Intel Labs IPR, IXR, and CSR, particularly Jesse Walker who has consistently guided our thinking on much of this research. In addition, we thank our external collaborators at Massachusetts Institute of Technology, Carnegie-Mellon University, and Trinity College Dublin. We are grateful to reviewers Dave Boundy and Joe Butler for their very helpful feedback.

References

[1] F. E. Gillett "The Personal Cloud: Transforming Personal Computing, Mobile, and Web Markets," Technical Report, Forrester Research, 2011.

[2] Gartner Report, "Consumers and the Personal Cloud," 2012.

[3] K. M., Khan and Q. Malluhi, "Establishing Trust in Cloud Computing", IT Professional, vol.12, no.5, pp.20–27, Sept.-Oct. 2010.

[4] M. Zhao, E. Schooler, J. Walker, B. Kveton, "The Social Life of Devices: A Wi-Fi Scavenger Hunter Case Study", Intel Technical Report, July 30, 2010.

[5] T. H. Kim, L. Bauer, J. Newsome, A. Perrig and J. Walker, "Challenges in access right assignment for secure home networks", In Proceedings of the 5th USENIX Workshop on Hot Topics in Security, August 2010.

[6] "Harry Potter and the Prisoner of Azkaban", J.K. Rowling, Scholastic Books, 1999.

[7] Project CCNx. http://www.ccnx.org.

[8] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. Proc. ACM CoNEXT, pages 1–12, 2009.

[9] D. Trossen, M. Sarela, and K. Sollins. Arguments for an information-centric internetworking architecture. ACM Sigcomm Computer Communications Review, April 2010.

[10] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. Proc. ACM SIGCOMM, 2007.

[11] M. Baugher and V. Lortz. Home-network threats and access controls. In Proceedings of the 4th international conference on Trust and trustworthy computing, TRUST'11, pages 217–230, Berlin, Heidelberg, 2011. Springer-Verlag.

[12] B. Schneier, Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish), Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993), Springer-Verlag, 1994, pp. 191–204.

[13] A. Sherman and D. McGrew. Key establishment in large dynamic groups using one-way function trees. IEEE Transactions on Software Engineering, 29(5):444–458, May 2003.

[14] R. Grinter, K. Edwards, M. Chetty, E. Poole, J. Sung, J. Yang, A. Crabtree, P. Tolmie, T. Rodden, C. Greenhalgh, and S. Benford, "The Ins and Outs of Home Networking: The Case for Useful and

Usable Domestic Networking", ACM Transactions on Computer-Human Interaction, Vol. 16, No. 2, Article 8, June 2009.

[15] C. Kuo, J. Walker, A. Perrig, "Low-cost Manufacturing, Usability, and Security: An Analysis of Bluetooth Simple Pairing and Wi-Fi Protected Setup", Usable Security (USEC), February 2007.

[16] Froehlich J. Froehlich, L. Findlater, "The Design of Eco-Feedback Technology", Proceedings of ACM CHI 2010, pages 62–75, April 2010.

[17] U. von Luxburg, "A Tutorial on Spectral Clustering", Journal of Statistics and Computing, Vol.17, No.4, page 395–416, 2007.

[18] Cisco Visual Networking Index: Forecast and Methodology (2010–2015), http://www.cisco.com/en/US/netsol/ns827/networking_solutions_white_papers_list.html

[19] S. Trifunovic, F. Legendre, C. Anastasiades, "Social Trust in Opportunistic Networks", In proceedings of IEEE INFOCOM Workshop on Network Science for Communication Networks (NetSciCom), pages 1–6, March 2010.

Author Biographies

Eve Schooler is a principal engineer in the Energy and Sustainability Lab inside Intel Lab and leads the Trusted Personal Energy Cloud project. Her interests lie at the intersection of networking, distributed systems, security, and scalable group algorithm design. Eve served on the Transport Directorate of the IETF, cofounded and cochaired the IETF MMUSIC working group for many years, and is a coauthor of the SIP protocol that is widely used for Internet telephony. Prior to Intel, she held positions at Apollo Computers, Information Sciences Institute, AT&T Labs-Research, and Pollere LLC. Eve obtained a BS from Yale, an MS from UCLA, and a PhD from Caltech, all in Computer Science.

Jianqing Zhang is a research scientist in the Energy and Sustainability Lab inside Intel Labs. His research areas include security, distributed systems, and networking. Currently he is working on trusted communication infrastructure for home/building energy management systems with the focus on energy efficiency and user experience. Prior to Intel, he worked at IBM China Software Development Laboratory as a software engineer for web applications development and testing. He received his BS and first MS from BeiHang University, his second MS from the University of Pennsylvania and his PhD from the University of Illinois at Urbana-Champaign.

Adedamola Omotosho is a rotation engineer in Intel's Rotational Engineering Program. He is currently completing his second rotation with Intel Labs Energy and Sustainability where he is researching collaborative analytics in the

Trusted Personal Energy Cloud. His work focuses on techniques to determine user presence, location, and activity from mining smart home energy data. Previous rotations and experiences have spanned working with Intel Software Quality, NAND Solutions Group, as well as design automation work with Intel's Fort Collins Design Center. Adedamola obtained a BS in Computer Engineering from Virginia Tech and Masters in Computer Science from Cornell University.

Jessica McCarthy is a research scientist with the Energy and Sustainability Lab at Intel Labs, where she is working on ESL's sustainable cities research agenda. Much of Jessica's focus is on the integration of urban solutions in smart cities and communities. Jessica is driving a future cities research program with Trinity College Dublin and she is also actively involved in European Framework Project (FP) research. She was the overall exploitation manager for SLA@SOI and is Intel's lead researcher on PLANTCockpit where Intel focuses in on improving energy monitoring and analysis in our manufacturing environment. She has been with Intel since 2005 and has over 12 years of IT industry experience, having held positions in companies such as Oracle and Alcatel-Lucent. Jessica's other areas of interest include cloud computing, infrastructure management, enterprise architecture, analytics and visualization. Jessica has a B.Sc. in Computer Science and Mathematics from the National University of Ireland Maynooth, an honours M.Sc. in Computing (IT) from the Dublin Institute of Technology, and is in the PhD program with Trinity College Dublin.

Qinghua Li is a PhD candidate in the Department of Computer Science and Engineering at Pennsylvania State University. His research interests include network security, mobile computing, and wireless networks. He was a graduate research intern in the Energy and Sustainability Lab inside Intel Labs in 2011 and worked on the Trusted Personal Energy Cloud project. He received a B.E. degree from Xi'an Jiaotong University, China, and an M.S. degree from Tsinghua University, China. He is a student member of the IEEE.

Meiyuan Zhao is a senior research scientist at Intel Labs working on improving the security and usability of the Intel next-generation platforms. She is currently focusing on enable security and trust management for machine-to-machine communication usage cases. Meiyuan received her PhD from Dartmouth College. Her research interests include network security, trust management, reputation systems, sensing and model building, sensor networks, swarm intelligence, peer-to-peer networks, routing protocols, and distributed systems.

ADVANCES IN SENSOR TECHNOLOGY TO IMPROVE INDIVIDUAL CONTRIBUTIONS TO SUSTAINABILITY

Contributors

Terrance O'Shea
Intel Corporation

Tomm Aldridge
Intel Energy and Sustainability Lab,
Intel Corporation

Bob Steigerwald
Intel Corporation

"70 percent of the world's population will occupy urban spaces by the year 2050. The megacities that are formed will have a significant impact on the local environments, particularly on energy and water resources, aquatic ecosystems, and air quality."

With urbanization increasing across the globe at an accelerating pace, it is estimated that 70 percent of the world's population will occupy urban spaces by the year 2050. The megacities that are formed will have a significant impact on the local environments, particularly on energy and water resources, aquatic ecosystems, and air quality. Making these megacities sustainable will require intelligent monitoring systems that provide real-time information to citizens and government. This article proposes a "Smart City Architecture" that employs a multitude of modular, sensor-based devices that fit seamlessly into the existing infrastructure. We describe the design principles of the sensor modules, aggregators, backend services, analytics, and data management. We then propose a variety of smart city sensor applications including microclimate sensing, pollution and structural monitoring, home management, traffic control, and water quality monitoring.

Introduction

In the course of human history, populations have historically been scattered throughout the landscape, but in more recent times there has been a rapid movement to urbanization. According to the Population Reference Bureau, 2008 brought about the first time the world's population was evenly split between urban and rural areas. Urbanization is increasing rapid to the point where 70 percent of the world population will be urban by 2050; furthermore most urban growth will occur in less developed countries. In 2012 the PRB estimates there are 16 megacities in the world where the population exceeds 10 million. They expect by 2025 that number will increase to 27, with 21 of them being in what are now less developed countries.

As these megacities grow their impact on the local environment will be profound. One of the most well documented cases is the urban heat-island effect, where cities tend to have higher air and surface temperatures than their rural surroundings, especially at night. The canyon-like buildings cause changes in airflow, trapping thermal energy in pavement and structures. Megacities also affect water resources, biodiversity, airflow, air quality, and the functioning of the natural ecosystem. Additionally since most cities are built along or near rivers, deltas, and coastlines to provide water sources, transportation routes, and power, there is often an impact on aquatic ecosystems. In these concentrated areas, the transportation and industry in megacities influence biogeochemical cycles, CO_2 levels, and other greenhouse gases and pollutants. In other words, with urbanization comes increasing problems with traffic, industrial pollution, structurally changing structural loads, energy demands, and water quality.

Within Intel Corporation, a small team of researchers are working on methods to make these megacities sustainable by intelligently monitoring the problems of traffic, industrial pollution, structurally changing structural loads, energy demands, and water quality while providing real-time information to the government and its citizens to aid in the sustainability of the city. To address the problem, low-cost wireless sensor platforms are being deployed in trials in what is called the *Smart City Architecture*.

"Low-cost wireless sensor platforms are being deployed in trials in what is called the Smart City Architecture."

Smart City Architecture

The Smart City Architecture began by considering the aforementioned problems that cities and particularly megacities are likely to face. An illustration of such a system is shown in the Figure 1.

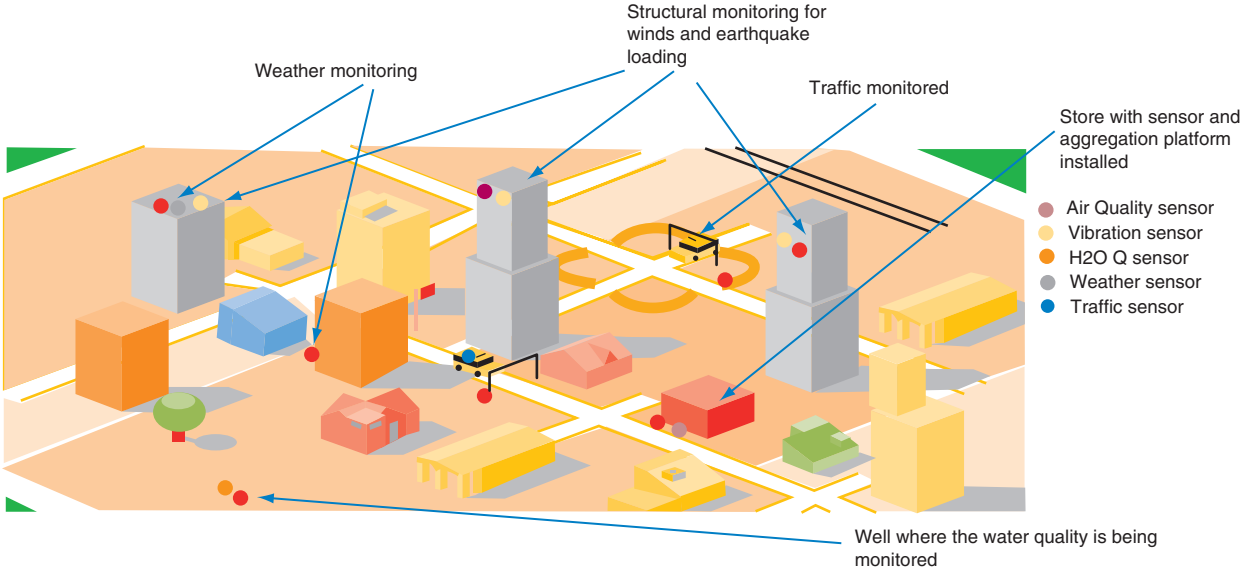


Figure 1: Prototypical smart city with sensors (Source: Intel Corporation, 2012)

Based on each of these problems, the stakeholders in the communities were identified and their needs and usages for each type of sensing were recorded from informational interviews. An example of part of the stakeholder table for the pollution sensor is shown in Table 1. In the development of the overall architecture the primary goal was to place the information in the hands of the community members while attempting to keep the total cost of ownership (TCO) of the system as low as possible. Keeping the TCO optimally low meant we needed considered not only the users of the system, but maintenance and installation issues as well.

Users	Needs	Usage
Urban Dwellers	Understand the air quality (AQ) in the area where they live and the impact on health (too much emissions = asthma attack).	Poll web pages periodically showing local, real emission and pollution data.
Commuters	Based on local weather patterns their traffic flow may change.	Poll Web pages periodically through smart devices to understand if their roads may be blocked.
Local Shop Owners	Impact traffic around his shop and needs to pull people in through advertising.	Local shop owner is where the system will be installed and may be the payer of the services for advertising.
Government Agencies	Watches traffic and weather patterns for controlling the flow of traffic and emissions impact on the environment.	Polling Web service for pollution impact around the environment, and weather microclimates
Installers	System integrators and installers	Stall device and initial confirmation of the systems. Looking for ease of installation and configuration. Time to Installation (TTI) > 20 minutes.

Table 1: Stakeholder Needs and Usages
(Source: Intel Corporation, 2012)

"One way to keep the system accessible and low cost is to use the existing Infrastructure."

One way to keep the system accessible and low cost is to use the existing infrastructure as much as possible, while utilizing commonly available components. These two considerations drove the design choice to use a wireless communications platform that fits within the existing infrastructure and 802.11 RF communication modules to keep with commonly available components.

Another requirement of the system is to keep the platform as modular as possible to enable a variety of sensing devices in the field. Since all devices would therefore have a common communications stack, local storage in case of connectivity loss, and an embedded processor, the decision was made to physically cleave the sensing device into two printed circuit boards. One circuit board contains the common components of embedded processor, power management elements, communication interfaces, and local memory, while the other contains physical sensing circuits or devices.

Due to different deployment models, there may be minor differences in number of sensors present in the collection device and how the user interaction is accomplished; however, where possible, hardware, firmware, software, and interface designs are shared across the system. Figure 2 shows the general data flow architecture. The sensing devices have wireless connectivity using 802.11, and multiple sensing devices connect to a local aggregator in the community. Multiple local aggregators then transmit information to a machine hosting a backend server. The backend server collects the information as it arrives and the data is stored in a database within the server. The server also supports HTTP access to the database so display devices such as a smartphone, tablet, or a laptop/PC can retrieve the data to be presented to the users.

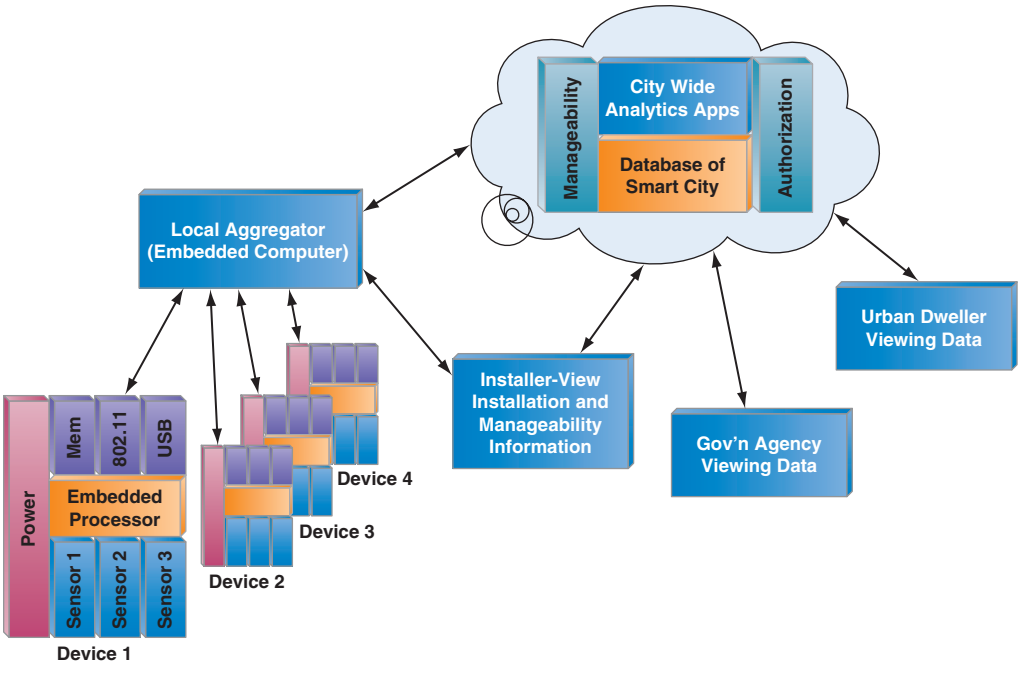


Figure 2: General data flow architecture of the smart city (Source: Intel Corporation, 2012)

Embedded Sensor Module

At a high level, the sensing module must support rapid modification and revision in a compact form factor. The embedded sensor module should contain an embedded processor, a communication module, memory for offline storage, power control elements, and an interface to the sensing boards.

"At a high level, the sensing module must support rapid modification and revision in a compact form factor."

In each application area, stakeholders reported tampering as a potential concern. To detect movement of the device such as an installation failure or an occupant tampering with sensor placement, two oppositely placed Omnidirectional passive MEMs vibration switches were used; specifically the Signal Quest* SQ-SEN.

The embedded processing needs led us to select a 32-bit low-power microcontroller with analog to digital converters, general-purpose input output pins, I2C, UART and SPI interfaces. The Atmel* family of processors (specifically the ATmega328P) enabled the implementation of all of the features including communication, micro SD storage capability, charging circuit to power from batteries, tilt sensors, and connectivity through on-board connectors. In strategic places several sets of LEDs of different colors were added to the design for debug, validation, and to facilitate rapid installation.

The form factor required was approximately the size of a matchbox (1.50″ × 1.75″) to ensure the device was unobtrusive. Battery charging and power monitoring circuits were added to the board to enable remote monitoring. The device was also designed to detect and transmit RF communication parameters

to assist in diagnostics, installation, and troubleshooting. The interface to the sensor board was implemented through two board-to-board connectors. The main board-to-board connector was a 30-pin I/O connector that included all the required interfaces to the sensor boards, while the other connector was a 20-pin subset of the first. The two connectors allowed us to make smaller boards when needed using the smaller connector for lower sensed applications. Figure 3 is a picture of the smart city sensor module.

Figure 3: Smart city sensor module (Source: Intel Corporation, 2012)

"A fundamental principle of a system built to promote sustainability is that the system itself is self-sufficient or in other words has a net-zero energy foot print."

"The hardware components should be designed to operate at low power and especially enter near-zero power states when idle."

Sensor Module Energy Efficiency

A fundamental principle of a system built to promote sustainability is that the system itself is self-sufficient or in other words has a net-zero energy foot print. The implications for the sensor modules is that they operate at very low power and have an accompanying mechanism to harvest enough energy to keep operating perpetually. With respect to low power operation, the design principles focus on:

- Low power hardware components
- Energy-efficient software
- Effective power management

The hardware components should be designed to operate at low power and especially enter near-zero power states when idle. The embedded processor operates at about 0.6 W, which operates on a duty cycle typically less than 2 percent on time. The accompanying Wi-Fi* module operates near 7 μA when idle and at 140 μA to 150 μA when transmitting. With a typical sensor message size of 36 bytes, the average time for transmission is 7.3 ms while the average receive time is 14.6 ms. The entire system at idle operates in the sub-100-mW range and when sending data operates at about 1.5 W.

Energy efficient software means that the software on the sensor module has been optimized to consume the least energy possible using techniques such as computational efficiency, data efficiency, idle efficiency, and context awareness.[1] Computational efficiency (*hurry up and get idle*) leads us to analyze the software to ensure that algorithms are the most efficient, use parallelism, and avoid wasteful activities such as spin-wait loops and privileged mode. Data efficiency techniques include effective use of the memory hierarchy, minimal I/O, and minimal data transfers. Idle efficiency means that the software is not performing unnecessary operations when it should be quiescent, such as polling or setting highly granular timer interrupts. Context awareness means that the software can and does react to environmental conditions, for example when the battery reaches a particular threshold the system may reduce the frequency of sensor data transmissions to conserve energy.

"Energy efficient software means that the software on the sensor module has been optimized to consume the least energy possible…"

The sensor module should also include effective power management, typically controlled by the OS, placing devices into low power states or turning them off when they are not needed. A typical power model for the sensor module is shown in Table 2.

SYSTEM SPECIFICATIONS					
Currents			**Duty Cycles**		
	value	**units**	**Model 1**	**Model 2**	**units**
Micro Processor (Atmega328P)					
current (full operation)	0.2	mA	0.05	2	%
current sleep	0.8	μA	99.95	93.4	%
Radio					
current in receive	0.14	mA	5.09091E-05	0	%
current transmit	0.15	mA	0.000305455	0.61	%
current sleep	7	μA	99.99964364	99.49	%
Logger					
write	15	mA	0	0	%
read	4	mA	0	0	%
sleep	0	μA	100	100	%
Sensor Board					
current (full operation)	10	mA	0.25	3	%
current sleep	5	μA	99.75	98	%
Battery Specifcations					
Capacity Loss/Year	3	%			

Table 2: Smart City Sensor Power Model
(Source: Intel Corporation, 2012)

"Using the surrounding environment for energy harvesting lends considerably to sustainability and reduces maintenance costs."

Sensor Module Power Sources

Ideally the sensor modules are powered from rechargeable batteries. Using the surrounding environment for energy harvesting lends considerably to sustainability and reduces maintenance costs. Where possible, the sensor batteries should be kept charged with one or more energy harvesting methods that may include:

- Solar or photovoltaic cells
- Micro wind turbines
- Wave energy converters
- Kinetic energy converters
- Piezoelectric crystals
- Ambient RF

Aggregators

The aggregators, usually embedded computers or mobile phones, receive data from all the sensors in the network, and may transmit them to an analysis engine or data visualization system. Alternatively, the aggregator may itself process the data and trigger responses by actuators or by communications with citizens or government officials. In general, actuators take the form of local PCs or mobile devices, such as PDAs and smartphones. We do not cover aggregator design in any detail in this article, because one typically uses mass-market devices such as phones and PCs or uses specific embedded computers.

"…our goal was to reuse as much code as possible while allowing different views based on which user was viewing the data and in what region and context."

Backend Services and Visualization

Since there was a high degree of commonality between the desired user experiences with reporting, our goal was to reuse as much code as possible while allowing different views based on which user was viewing the data and in what region and context. To accomplish this, we created an architecture that supported loosely coupled classes such that the final view over the data could be "re-skinned" to have a different look and feel while sharing large amounts of code "under the hood." To accomplish this, the Model, View, Presenter design pattern was adopted. Many articles describing this pattern, its merits, and how to use the design pattern are available on the Internet.[2][3][4]

The *model* holds the representation of the real-world events and inputs. In our implementation, the model objects are responsible for accessing the external database and periodically refreshing the data within the model to reflect the data from the database. As the data are updated, the model generates events that are delivered to any other objects that are listening for those events. For efficiency, events are only generated if the data actually changed from previous reports (for example, if the last known temperature was 70 degrees Fahrenheit and on the next query of information it is still 70, an event is not sent).

The *presenter* is responsible for tying the model to the *view.* To keep elements as loosely coupled as possible, the view and model are both structured to know nothing about any other objects in the system (beyond allowing listeners

to be registered for the events that they generate). The presenter holds the application/business logic that takes the real-world information from the model and determines how it should be represented to the user and then creates view objects that display the data in the desired fashion.

"The presenter holds the application/ business logic that takes the real-world information from the model and determines how it should be represented to the user…"

The view, in our case, is a passive view. It does not connect to the model for data, but sits passively and awaits information updates from the presenter.

This allows the view objects to concentrate only on the view aspect. It need not worry about where the information being displayed came from or how it was derived. In a very pure implementation of Model, View, Presenter, the view passes *all* user input events directly to the presenter and the presenter then informs the view how to react. In our implementation, this was relaxed to allow the view to internally handle any events that are specifically targeted to altering the view (for example, scroll events are intending only to move the presently displayed information to a new position and do not require interaction with the presenter object). However, any events that go across view objects (such as clicking a selection within the scroll field in order to pop up a dialog on the selected object) do flow through the presenter since the cross view object view behaviors may need to be changed based on the application or business needs.

Another technique used in this architecture is class inheritance. A number of base classes have been created that stub in core functionality and then additional classes are created that inherit the base functionality and tailor the behavior to the application or business needs. The framework provides hooks in the code to allow the presenter logic to instantiate the appropriate class for the application. This provides another mechanism to shape the applications views and behavior while using a majority of the framework as is.

User Interfaces

The user interface shows the user, metaphorically, what the readings are from the aforementioned variety of sensors. Each sensor reading may be temporally independent; thus, the UI must provide the ability to quickly scan through all the current and historical readings. The UI must also provide the state of the sensor, battery life, RF connectivity, and type of sensor reporting the data. With all of this data the user may be quickly overwhelmed, so metaphors (symbols) are used to represent the data, allowing users to rapidly understand the visualized data and the impacts. An overriding design principle has been to eliminate graphs/charts, translating data into loose visuals that give general indications of conditions at a glance, using size and color as conditional cues. In the smart city architecture the UI is driven by XML data from the cloud. Client devices use a web browser with content displayed via Adobe* Flash and Java* scripts.

"…metaphors (symbols) are used to represent the data, allowing users to rapidly understand the visualized data and the impacts."

For simplicity, each logged-in user has only two levels within the UI. The first level is the location choice, while the second level is the sensor readings at that location. Figure 4 shows an example of the location of the water quality sensors using color to indicate quick view status of sensors.

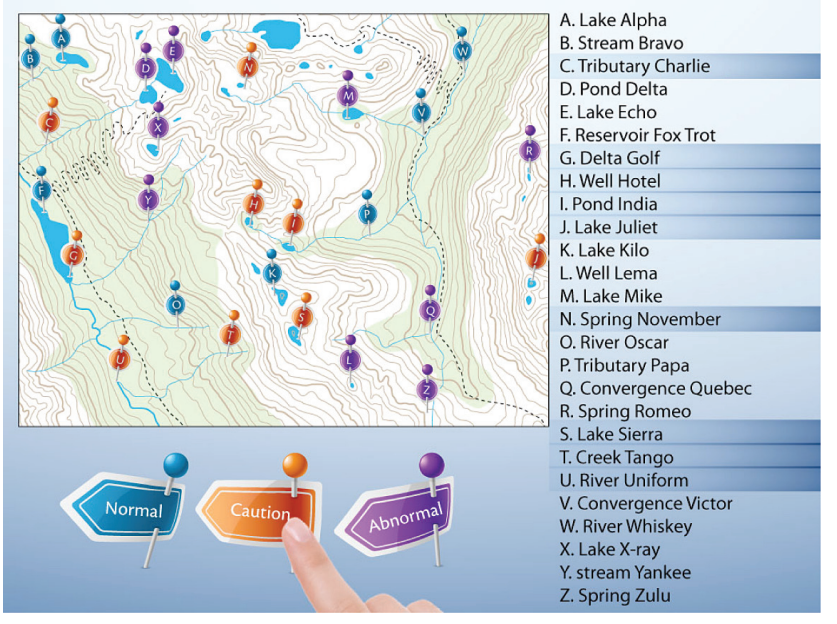


Figure 4: Sample UI showing the location of water sensors (Source: Intel Corporation, 2012)

"All detailed information can be obtained by clicking or highlighting a symbol on the map to reveal more granular information."

All detailed information can be obtained by clicking or highlighting a symbol on the map to reveal more granular information. Specifically, these detailed readings are:

- Colored black on 25 percent transparent white background.
- Represented by highlighted boxes that are initialized by "events" such as clicks, warnings/alerts, or gesture.
- Have overall status conditions represented as icons that may include: Wi-Fi strength, battery charge, overall status, and service needs.

Figure 5 shows a detail view of a chosen sensor. Bubble icons are used to show a high level indication of the condition (reading) of the sensor. New bubbles form at bottom and float upwards. The bubbles at the top are older readings and the bubbles at the bottom are the most recent. Bubbles form in different sizes to indicate high level overview of all sensors in a composite view, in this way showing or indicating trends among sensors.

Figure 6 shows one sensor highlighted (Temperature) but no granular data on the individual reading.

"With the proliferation of devices capable of generating a tidal wave of sensor data, there is an urgent need to protect data privacy..."

Data Management

With the proliferation of devices capable of generating a tidal wave of sensor data, there is an urgent need to protect data privacy, since even superficial data mining could leak highly confidential information. At the same time, more devices are connecting at the edges of the Internet, in unmanaged or self-managed networks (such as at home), where average users are responsible for their management, so ease of use and seamless integration are critical as well.

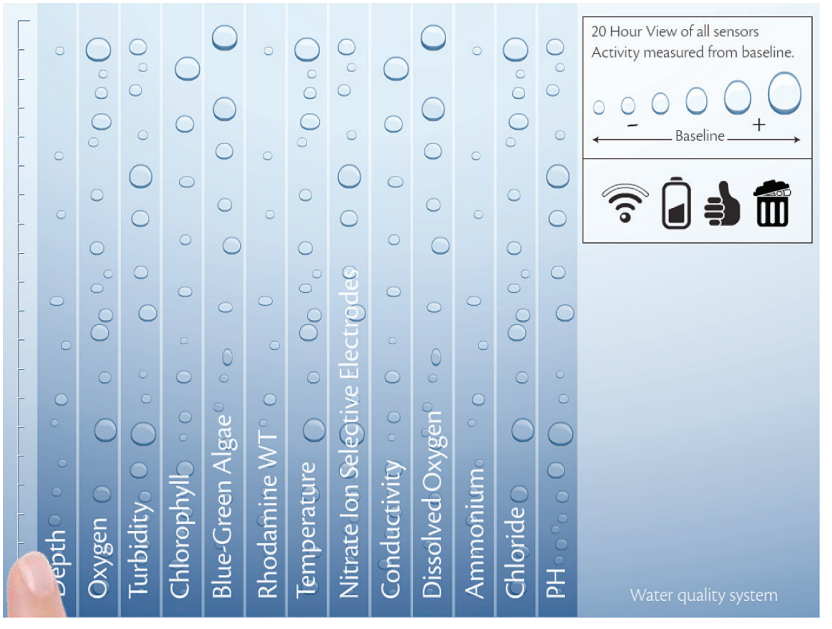


Figure 5: Detail view of a symbol selected from the map (Source: Intel Corporation, 2012)

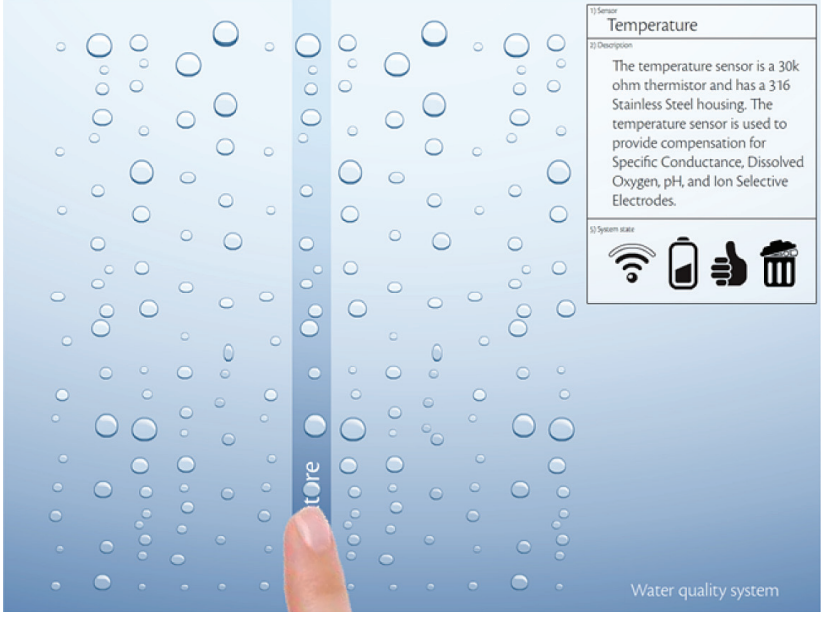


Figure 6: Selecting detailed information about a specific sensor (Source: Intel Corporation, 2012)

To address these challenges, Intel's Energy and Sustainability Lab (ESL) researchers are using technology that ensures secure and efficient communication. The technology allows trusted device groupings to communicate and collaborate, while thwarting unauthorized access to data,

"…Intel's Energy and Sustainability Lab (ESL) researchers are using technology that ensures secure and efficient communication."

thus protecting consumers' privacy. The device groupings generate collaborative analytics that are used to assess and improve energy efficiency.

"The technology also supports low-power communications by eliminating "senseless" sensing…"

The technology also supports low-power communications by eliminating "senseless" sensing (that is, sensing when it's not needed), enabling edge devices to remain asleep longer, employing context-aware adaptive protocols to reduce unnecessary network traffic and address Internet quiescence. The result is power savings through collaborative energy management.

Smart City Sensor Applications

"…increased urbanization leads to increasing problems with traffic, industrial pollution, changing structural loads, energy demands, and water quality."

As mentioned in the previous section, increased urbanization leads to increasing problems with traffic, industrial pollution, changing structural loads, energy demands, and water quality. In this section we identify specific opportunities where embedded sensor modules may be applied to monitor conditions, providing necessary data to keep thriving megacities sustainable.

Microclimate Sensing

Some of the applications for microclimate sensing include:

- Humidity sensor
- Temperature sensor
- Ambient light sensor
- 3-Axis Gravitometer
- Barometric pressure sensor
- Anemometer sensor

The board has been fitted with three vias to install wires to a Hall Effect switch (power plus two leads) or two wires to a magnetic reed sensor.

Micropollution

It is a well-known fact that polluted air is a major health hazard in urban areas. Although with improvements in pollution monitoring techniques during the last several decades that have steadily enhanced the ability to measure the health effects of air pollution the pollution is often measured in a few discrete locations in the cities. Often these discrete measurements are made on the tops of buildings, or at airport that often misrepresent the actual pollution witnessed by city dwellers at street level. Particulate matter (PM) is an additional concern to city dwellers. As PM rises there is a correlation with increases in the incidence of cardiopulmonary and respiratory diseases, coughing, bronchitis, and lung cancer, as well as premature deaths from these diseases.[5]

Researchers at Intel have developed a simple pollution sensing device that has integrated several sensors with the common embedded and RF stack. These sensors include:

- Particulate matter sensor
- Carbon monoxide sensor

- Air quality sensor
- Ammonia sensor
- Temperature sensor

Structural Monitoring

Structural monitoring in sustainable cities will be critical as loading has various concerns on the building's occupants. There has been a rise recently in sick building syndrome caused by machine loading of HVAC systems or other loading like winds or local traffic. Additionally as buildings become subject to large wind loads and earth movement, continuous monitoring of the loading of the building yields the life of the steel within the structure by monitoring the overall cumulative fatigue of the structure. Sensors for such measurement include:

"There has been a rise recently in sick building syndrome…"

- 3-axis gyroscope
- 3-axis accelerometer
- 3-Axis gravitometer
- Piezoelectric sensor for movement detection

The MMA8451Q is a smart low-power, three-axis, capacitive micromachined accelerometer with 14 bits of resolution. This accelerometer is packed with embedded functions with flexible user programmable options, configurable to two interrupt pins. Embedded interrupt functions allow for overall power savings relieving the host processor from continuously polling data. There is access to both low pass filtered data as well as high pass filtered data, which minimizes the data analysis required for jolt detection and faster transitions. The device can be configured to generate inertial wake-up interrupt signals from any combination of the configurable embedded functions allowing the MMA8451Q to monitor events and remain in a low power mode during periods of inactivity.

The ITG-3200 is a 3-axis MEMS gyroscope, with enhanced bias and sensitivity temperature stability, reducing the need for user calibration, it features three 16-bit ADCs for digitizing the gyro outputs, and a user-selectable internal low-pass filter bandwidth. Its sensitivity is 14.375 LBS per degree/sec, and a full scale range of ± 2000 degrees/sec. It generates an interrupt signal to let the processor know when new sampled data is available.

The HMC5843 is a 3-axis Digital Compass IC designed for low field magnetic sensing, using anisotropic magneto-resistive technology.

Home Management

Helping consumers to manage energy more efficiently is crucial to meeting the world's growing demand for power, as consumers account for a high percentage of energy consumption. For instance, in the US, 113 million households and 250 million automobiles consume 35 percent of total US energy.[6] In China, 360 million households and 76 million automobiles[7] account for 28 percent of the country's energy consumption.[8] ESL researchers are focused on finding

"Helping consumers to manage energy more efficiently is crucial to meeting the world's growing demand for power…"

ways to aggregate millions of small contributions from individual energy consumers to reap substantial energy savings and are exploring ways to improve energy efficiency at the community level.

Today, consumers and communities have limited visibility into and control over their energy usage. Monthly utility bills provide some insight but not real-time, actionable data. Current energy management systems can help to monitor aggregate power loads in homes and buildings, but not at the level of appliances and devices. Furthermore, the systems are expensive and require a considerable overhaul of the local electrical infrastructure.

"Intel's Wireless Energy Sensing Technology (WEST)… enables users to manage energy usage without the need for expensive or intrusive technologies."

ESL researchers are developing more effective, and cost-effective energy management tools called *personal energy systems* (PESs). These sensor-based systems will deliver detailed information about power consumption, including how much energy is being consumed, when, and by which devices and appliances. Such information will enable consumers and communities to better understand and manage their energy usage. A residential home, for example, might have its own PES that provides actionable, real-time data related to appliances, electrical devices and heating and cooling systems.

Figure 7: Residential device with Wireless Energy Sensing Technology (WEST) (Source: Intel Corporation, 2012)

Foundational Technologies

To implement PESs, the ESL has leveraged four foundational technologies: unobtrusive sensing; secure, efficient communications; energy management algorithms; and direct current distribution. These technologies have been combined in Intel's Wireless Energy Sensing Technology (WEST), which enables users to manage energy usage without the need for expensive or intrusive technologies. WEST is a low-cost sensor device that plugs into a standard wall outlet and monitors the energy usage of electrical devices in the home (shown in Figure 7).

Once plugged in, it establishes a secure wireless link to a home network to allow easy access (via laptop, smartphone, or television) to data about the current energy usage of every appliance in the home. WEST recognizes the unique signatures of major electrical devices and by utilizing this information can provide a detailed energy consumption profile to the user. Only one or two sensors are required for the typical home to compute detailed home appliance operations.

By empowering consumers to manage their energy usage, technology could help to reduce the energy consumption of the average US household by 15–31 percent. If just 1 percent of US households realized this level of savings, they would collectively reduce carbon emissions by 2.4 million metric tons—the equivalent of taking 535,000 cars off the road.

Buildings also represent fertile grounds for improving energy efficiency. According to the US Department of Energy, roughly 40 percent of total US energy is consumed in buildings.[9] PESs can be applied here as well, enabling the monitoring and control of lighting and plumbing as well as HVAC,

security, information technology (IT), and other systems. The research team will focus on managing devices used in buildings and integrating device management with building management systems.

In addition to utilizing intelligent control planes (ICPs) and simple sensors, buildings can take advantage of computing devices on the premises to improve energy efficiency. Technologies such as laptops, desktops, servers and printers have a rich array of sensors, including cameras, temperature sensors, and physical tools such as keyboards, all of which can be used to understand an environment and dynamically control power consumption.

Future technologies will be able to automate their power states and, when appropriate, go into deep sleep modes without being prompted. These technologies also will leverage virtualization, moving active loads to active servers and shutting down or reducing power for underutilized loads.

Intel estimates that efficiency gains on the order of 15-20 percent could be realized by taking advantage of computational sensors and aligning the IT and infrastructure systems within a building. Once aligned with a building's energy management system, such technologies could be used to monitor and dynamically control larger systems, such as lighting, HVAC, and plumbing.

"…efficiency gains on the order of 15-20 percent could be realized by taking advantage of computational sensors and aligning the IT and infrastructure systems within a building."

Finally, researchers will explore energy management solutions at the district level (microgrid, community, or campus). A business park, university, or corporate campus may have a number of PESs that connect and aggregate energy information (private and public) from various buildings. The goal is to employ PESs to optimize overall energy usage within a district.

Traffic Monitoring

A real-world situation requiring traffic monitoring has arisen in Brazil; the identification of any vehicle is planned using the National System for Automatic Vehicle Identification (or SINIAV). Brazil is the giant of Latin America. It has a population of 192 million, making it fifth in the world. Brazil's economy at 1.98 trillion US dollars (USD) is the world's ninth. The SINIAV system employs an RFID-based tag that is intended to be embedded in the national vehicle fleet by default in the coming years. Planning for the system has been proceeding since 2004. The tags began deployment mid-2011 and within three years all the approximately 50 million motor vehicles in the country are due to have them.

"The SINIAV system employs an RFID-based tag that is intended to be embedded in the national vehicle fleet by default in the coming years."

Under the Brazilian specifications, a unique serial number and manufacturer's number take up 96 bits programmed on manufacture leaving 928 programmable bits. 416 bits are used for registration/law enforcement purposes: serial number, vehicle plate, vehicle class, vehicle brand, chassis number, and the like.

The other 512 bits are available for any special data wanted in other applications such as tolling. Thus a trip toll system could write the entry place and time to memory for download along with place and time on exit

to compile a trip. Minimum performance standards are 99.9 percent fully accurate reads at 0 to 160 km/hr (100 mph).

"In a fleet of 50 million vehicles, one third are in some sort of illegal situation, and about 1,000 are stolen each day."

In a fleet of 50 million vehicles, one third are in some sort of illegal situation, and about 1,000 are stolen each day. Highway improvements in Brazil are concentrated on roads between major cities, and the lack of a wider repair or replacement plan increases the cost of highway freight transportation by 25 percent. Only recently, in 2008, Brazil announced USD 15 billion in highway infrastructure investment, equal to about 1 percent of GDP. The country's trucking industry also has a large black market component, and only 5 percent of insurance companies offer coverage for highway freight operations.

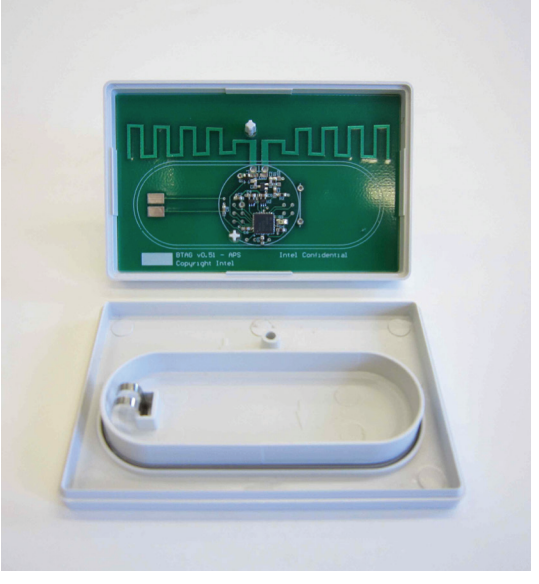


Figure 8: Brazil-Tag (BTAG) device (Source: Intel Corporation, 2012)

The system will permit authorities to conduct blitz inspections, using handheld antennae to single-out specific vehicles for attention. SINIAV is sensitive enough to identify the lane in which a vehicle is travelling, making it a useful tool for tolling and congestion management.

Three letters and four digits standardize Brazilian license plates. Plates are connected to vehicles by a small wire and a plastic seal, and the vehicle ID is matched with its RFID code, so that the system "creates a DNA for the vehicle, and the license plate is now encrypted." A secure, reliable system was essential to earn drivers' trust, so the von Braun Center spent four years studying the available data encryption and authentication technologies as well as developing functional reference platforms before opting for a system based on the AES-128 algorithm, then tested different antenna strengths on the license plate and in different positions on the vehicle. The solution for the traffic management, called BTAG, is shown in Figure 8.

BTAG is short for Brazil-Tag, and is a battery-assisted RFID Tag for use by the government of Brazil for Automatic Vehicle Identification (AVI). The tag supports a limited set of the EPC C1G2 RFID protocol, and the full SINIAV protocols; a superset to EPC C1G2 RFID standards.

"The vast majority of surface water on the planet is neither potable nor toxic."

Water Quality Monitoring

The vast majority of surface water on the planet is neither potable nor toxic. Industrial and commercial activity (such as manufacturing, mining, construction, transportation) is a major cause of water pollution, as well as runoff from agricultural areas, urban storm-water runoff and discharge of treated and untreated sewage (especially in developing countries). Contaminants that may be in untreated water include microorganisms such as viruses and bacteria; inorganic contaminants such as salts and metals; organic chemical contaminants from industrial processes and petroleum use; pesticides and herbicides; and radioactive contaminants. Water quality depends on the local geology and ecosystem, as well as human uses such as sewage dispersion, industrial pollution, use of water bodies as a heat sink, and overuse (which may lower the level of the water).

In the United States, the Environmental Protection Agency (EPA) limits the amounts of certain contaminants in tap water provided by public water

systems. The Safe Drinking Water Act authorizes EPA to issue two types of standards: primary standards regulate substances that potentially affect human health, and secondary standards prescribe aesthetic qualities, those that affect taste, odor, or appearance. The US Food and Drug Administration (FDA) regulations establish limits for contaminants in bottled water that must provide the same protection for public health. Drinking water, including bottled water, may reasonably be expected to contain at least small amounts of some contaminants. The presence of these contaminants does not necessarily indicate that the water poses a health risk.

To address the issues of understanding water quality the McPherson Daughter board was created.

"To address the issues of understanding water quality the McPherson Daughter board was created."

Figure 9 illustrates a typical setup for the McPherson system. In this system, the water sensors in the reservoir communicate through UARTs and ADCs to the code named McPherson daughter card. The daughter card is connected to the code named HighHill or code named CampHill main board to send RF communication of the sensor information to the router. The devices are power through battery, AC power or solar lines to the McPherson daughter card. The RF communications go to either a router or PC server which sends the information to an XML database on a server in the cloud. Any client with Flash capability can then access the XML database information for the site through the UI.

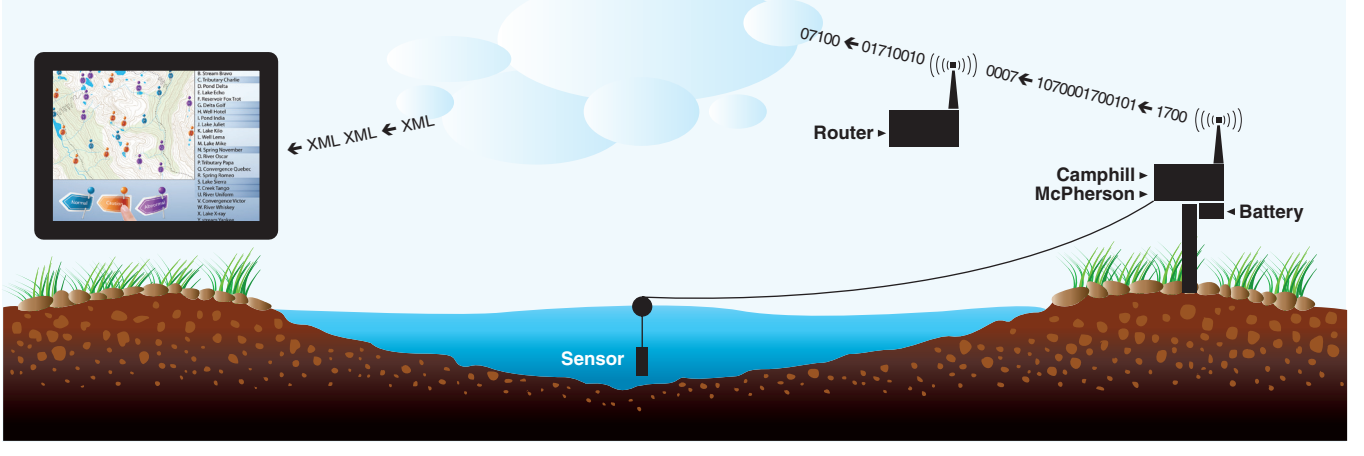


Figure 9: Example McPherson System
(Source: Intel Corporation, 2012)

The McPherson board is intended to be a daughter board for the code named Providencia Suite of sensors to add the capability of water quality sensing. Water quality is the physical, chemical and biological characteristics of water; a measure of the condition of water relative to the requirements of one or more biotic species and or to any human need or purpose.

Conclusion

The move towards urbanization across the globe leading to an increasing number of megacities gives rise to many new challenges. To achieve a sustainable megacity ecosystem, governments, organizations, businesses, and individuals will require information for decision-making that comes from intelligent monitoring systems for energy, water resources, air pollutants, traffic, and more. The proposed Smart City Architecture is a flexible and scalable approach designed to integrate easily into the existing infrastructure and provide a continuous flow of information about the environment from low-cost, low-power, wireless sensors. To avoid overwhelming decision-makers with a flood of data, we propose aggregators and advanced analytics coupled with intelligent, intuitive user interfaces. With the concepts and prototypes well understood, there are vast opportunities for additional research, empirical studies, and in particular the exciting prospects of applying the technologies and improving sustainability on a broad scale in real communities.

References

[1] Steigerwald, Bob, et al. Energy Aware Computing: Powerful Approaches to Green System Design. Intel Press, 2012.

[2] Wikipedia. 2012. Model-view-presenter. http://en.wikipedia.org/wiki/Model_View_Presenter

[3] Potel, Mike. 1996. MVP: Model-View-Presenter: The Taligent Programming Model for C++ and Java. Working Paper, Taligent Corp. http://www.wildcrest.com/Potel/Portfolio/mvp.pdf

[4] Newman, Rich. 2008. Model-View-Presenter Variations on the Basic Pattern. Personal blog. http://richnewman.wordpress.com/2008/02/26/model-view-presenter-variations-on-the-basic-pattern-introduction-to-cabscsf-part-24/

[5] Holgate, ST. "The epidemic of allergy and asthma," *Nature,* November 25, 1999.

[6] Energy Information Administration, International Energy Annual Review 2007. http://www.eia.doe.gov

[7] National Bureau of Statistics of China, February 25, 2010, htpp://www.stats.gov.cn)

[8] Energy information administration, International Energy Annual Review 2007. http://www.eia.doe.gov

[9] U.S. Energy Information Administration, http://tonto.eia.doe.gov/ask/generalenergy_faqs.asp#energy_use_person

Author Biographies

Tomm Aldridge is Intel Labs director of the Energy and Sustainability Lab within Intel Labs Europe. He has been with Intel since 1995 and in that time has served in numerous product and research roles including developing Itanium® server system architecture, leading several advanced power technology product path finding and research teams and leading platform efficiency research teams. His current role in Intel Labs brings his passion for efficiency and his broad technical knowledge to the challenge of creating a technology assisted sustainable and efficient society with current and future Intel products as a core ingredient. In addition to Tomm's Intel experience, he brings business creation and leadership experience from a successful HPC related startup and also a diverse career background including super computer system technology development and data center power architecture. Tomm holds a BS in Physics from New Mexico Institute of Mining and Technology and is a senior member of the IEEE.

Bob Steigerwald is an engineering manager at Intel Corporation. He has over 30 years of industry experience as a software engineer, Associate Professor of Computer Science, program manager, and engineering manager. He has spent the past four years leading an Intel team researching methods to improve software performance and energy efficiency. Bob is coauthor of the book *Energy Aware Computing* published by Intel Press. He earned his B.S. in Computer Science from the USAF Academy, Masters from University of Illinois, MBA from Rensselaer Polytechnic Institute, and PhD from the Naval Postgraduate School.

Terrance (Terry) J. O'Shea, PhD, O.E. is the senior principal engineer in Intel Labs IXR/ESI team. With over 18 years' experience, O'Shea is considered a pioneer in the field of sensor networks applications. In his current position, he is responsible for research and development of novel sensor-based technologies and ubiquitous computing applications as well as designing new radio technologies. During his tenure at Intel, O'Shea served on the faculty of the State University of New York at Buffalo and has coauthored two textbooks—most recently, Applications of *Wireless Sensor Networks for Healthcare,* published in 2009. He is the author of more than 80 other publications in electronic packaging, biomedical engineering, computer science, electrical engineering, sensor networks, sensing, and structural mechanics. His papers have been published in numerous IEEE symposia and in periodicals such as *IEEE Electronic Packaging, Circuit World, Journal of Biomedical Materials Research, Geriatric Medicine,* and *Journal of Applied Physics*. In addition, he holds 56 patents and has 63 patents pending. Prior to joining Intel, O'Shea was on the faculty of the University of Maryland. O'Shea holds a PhD in engineering science from the University of Arizona and master's and bachelor's degrees in engineering mechanics from the University of Tennessee.

A QUALITATIVE FRAMEWORK FOR ASSESSING ICT IMPACT ON ENERGY EFFICIENCY

Contributors

Keith Ellis
Intel Energy and Sustainability Lab, Intel Labs Europe

Farid, Fouchal
Lougborough University

Tarek Hassan
Loughborough University

Daniel Kuhn
Fraunhofer Institute for Production Systems and Design Technology

Charles Sheridan
Intel Energy and Sustainability Lab, Intel Labs Europe

Industrial sectors are faced with a "sustainability paradox" in maintaining economic growth while consuming fewer resources. Information communication technology (ICT) has proven central to the performance-driven development of modern industry in supporting systems at all levels. Given this pervasiveness, ICTs have a unique opportunity to address the sustainability paradox by enabling energy-efficient viable operations.

Yet demonstrating the enabling impact of ICT has proven somewhat arduous. Often the issue is not a lack of technological options, but rather a problem of interoperability and in understanding what changes will have the greatest impact.

This article introduces the REViSITE Framework used and posited as a common means of categorizing, comparing, and qualitatively estimating both the direct and enabling impact of ICTs on energy efficiency. The approach proved useful in offering a "lens" into the technologies, practices, and research of four target sectors, namely energy grids, the built environment, manufacturing, and lighting. The output of the framework was used in developing a multidisciplinary strategic research agenda with respect to ICT-enabled energy efficiency in the target sectors. The framework and the research agenda constitute the main deliverables of the REViSITE project, which was funded in part by the European Commission under the 7th Framework Programme.

Introduction: The REViSITE Project

In 2008, European Commission President José Manuel Barroso stated ". . . the real gains will come from ICT as an enabler to improve energy efficiency across the economy . . . especially in . . . energy intensive sectors[1]. In 2011, President Barroso suggested ". . . Since our best source of energy is in fact energy efficiency, and also considering the prices of energy, I think it is important from all points of view to achieve real progress of energy efficiency very soon . . . "[2].

"The enabling role of ICT is inherently apparent, understanding which ICTs are best positioned to deliver meaningful impact is less evident."

In short, energy efficiency is important for the energy security and sustainability of Europe and ICT has a paramount role to play in delivering energy efficiency. However, while the enabling role of ICT is inherently apparent, understanding which ICTs are best positioned to deliver meaningful impact is less evident.

REViSITE (Roadmap Enabling Vision and Strategy in ICT Enabled Energy Efficiency) is a coordinated action funded in part by the European Commission (EC) to understand which ICTs are best positioned to positively impact on sustainability goals and to promote cross-sectorial synergies.

The main project objectives were the development of:

- A common means of assessing the impact of ICT on energy efficiency.
- A cross-sectorial ICT for energy efficiency (ICT4EE) strategic research agenda (SRA).
- A multidisciplinary community to promote cross-sectorial ICT4EE.

In the sections that follow the REViSITE project and Framework are introduced. Impact-assessment best practice is discussed, as is the need for a common framework and taxonomy. How the framework and taxonomy were utilized to frame and prioritize relevant ICTs is described in the context of one target sector-manufacturing.

The REViSITE Framework

Progress in realizing ICT potential in terms of positively affecting energy efficiency objectives has been labored. Often technology adoption is less about the underline technology and more about understanding or demonstrating what technological choices will have the greatest impact. As identified by the EC there is a ". . . clear need to create a level playing field based on common ways of measuring energy performance . . . and on a common understanding of commitments, targets and methodology."[3] *Common* is an important word in the context of REViSITE as any adopted approach needed to be applicable across four target sectors.

"There is a . . . clear need to create a level playing field based on common ways of measuring energy performance…"

Assessment Best Practice

REViSITE research showed that assessment best practice, see Table 1, utilized some form of life cycle assessment (LCA) or life cycle thinking, and that the overall trend and need is towards a focus on quantifying the enabling effects of ICTs, not just their direct effects. Those methods and standards that focused on direct effects were not deemed suitable for REViSITE. Life cycle assessment

"Research showed that assessment best practice…utilized some form of life cycle assessment or life cycle thinking."

Body	Method	Direct effects	Enabling effects
ITU (International Telecoms Union)	Hybrid LCA	Yes	Yes
ETSI (European Telecoms Standards Institute)	Hybrid LCA (National level)	Yes	Yes
iNEMI (International Electronics Manufacturing Institute)	Process-LCA	Yes	No
IEC (International Electrotechnical Commission)	Process-LCA	Yes	No
Ericsson	Process-LCA	Yes	Yes
ATIS (Alliance for Telecom Industry Solutions)	Process-LCA	Yes	Yes
GeSI (Global e-Sustainability Initiative)	Hybrid	Yes	Yes
ISO standards 14040/44 & British standards Institute PAS-2050			

Table 1: Assessment Uses Some Form of Life Cycle Approach
(Source: Intel Corporation (REViSITE Partner))

methods, while generally accepted as the most accurate means of assessing sustainability impact, were typically product focused and taxing to complete.

What our review suggested is that while desirable, in practice, quantitatively assessing the impact of ICT on energy efficiency is often an arduous process.

Situations where an existing system and a replacement ICT-enabled system can be directly measured are not so common. Where feasible, the task is often complicated in that the replacement system rarely differs from the old with respect to just the ICT element, while in some cases the impact may reside in a different life cycle phase or sector.

Typically, in scenarios where opportunity for direct quantitative comparison is limited, some form of estimation is required based on heuristics, part measurement, secondary data, and specialist knowledge.

"REViSITE…developed a qualitatively … framework … based on life cycle thinking and an adapted Capability Maturity Model/ Framework."

In that vein REViSITE, similar to GeSI, developed a qualitatively based framework for identifying those ICTs most likely to positively affect energy efficiency (EE). The framework, shown in Figure 1, is based on life cycle thinking and an adapted Capability Maturity Model/Framework (CMM or CMF). It utilizes a triangulation approach in leveraging the heuristics of domain experts, together with available quantitative and qualitative sources[4].

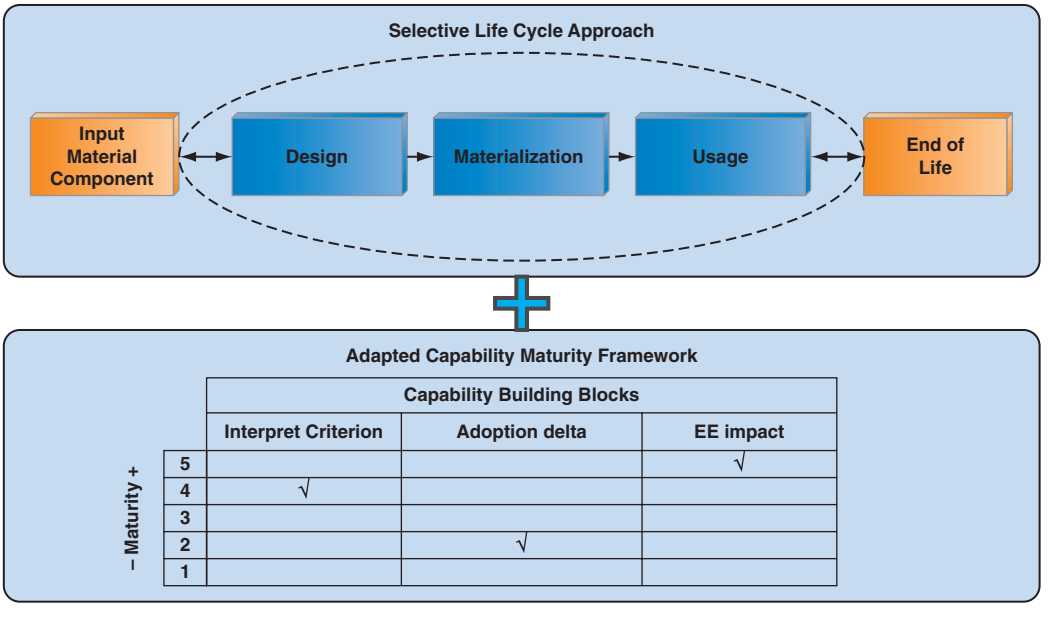


Figure 1: Combining life cycle thinking and CMF (Source: Intel Corporation (REViSITE Partner))

"Before comparing different sectors it was essential to first speak a common technical language so as to compare like with like…"

The SMARTT Taxonomy

Before comparing different sectors it was essential to first speak a common technical language so as to compare like with like and as such the first stage to methodology development was the creation of a common taxonomy, which, consistent with current best practice, was based on a life cycle approach.

The REViSITE developed SMARTT taxonomy has three levels, 6 high level categories and 23 subcategories. Both categories and subcategories were fixed and deemed to cover the scope of the ICT4EE domain. ICT themes at the third and lowest taxonomy level are domain user defined.

The six main categories follow the SMARTT acronym and are aligned to a bounded life cycle which includes the design, materialization, and usage phases, see Figure 2. Aligning main categories to a bounded life cycle offered an overarching structure the four target sectors could relate to and allowed for common categorization and cross-sectorial comparison.

"Aligning main categories to a bounded life cycle offered an overarching structure the four target sectors could relate to…"

Materialization was chosen as a non-sector-specific term understood by REViSITE partners to encompass construction, grid infrastructure, and production-system development.

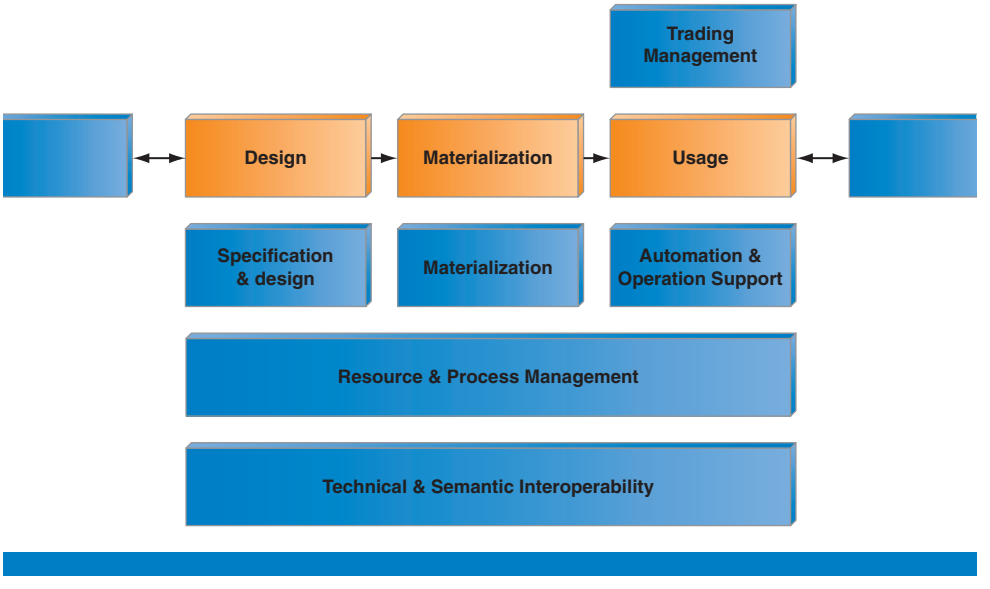


Figure 2: SMARTT taxonomy mapped to life cycle phases (Source: Intel Corporation (REViSITE Partner))

The categories "Specification and design ICTs," "Materialization ICTs," and "Automation and operation support ICTs" all vertically align to the bounded life cycle phases. "Resource and process management" together with "Technical semantic interoperability" are horizontal themes, while "Trading/transactional management ICTs" aligns primarily with the "usage" life cycle phase.

The 23 subcategories, shown in Figure 3, are nested within the 6 main categories. Subcategories allow for more granular/meaningful categorization. Individual, ICT research and technical development (RTD) themes sit at the lowest level nested within the subcategories. These RTD themes detail the specific areas of research giving existing or envisaged ICT exemplars.

"The … SMARTT taxonomy was utilized throughout the project as an integrative classification system and as an aid to cross-sector ICT4EE assessment."

The life cycle aligned SMARTT taxonomy was utilized throughout the project as an integrative classification system and as an aid to cross-sector ICT4EE assessment.

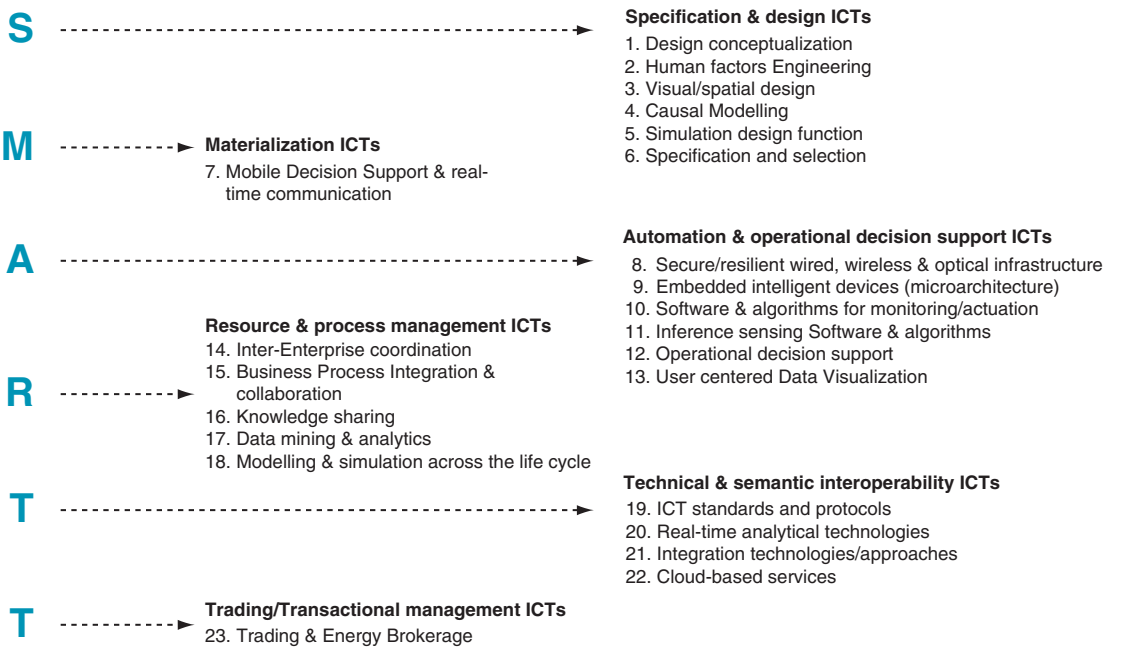


Figure 3: SMARTT subcategories
(Source: Intel Corporation (REViSITE Partner))

The SMARTT Taxonomy Applied to Manufacturing

The following six subsections serve two purposes. Firstly they offer a comprehensive overview of ICTs as they relate to energy-efficient production systems. Secondly they show how REViSITE research was framed using the SMARTT taxonomy. Again, the taxonomy offered a common language, a lens into the technologies and practices of other sectors, thus aiding cross-pollination in terms of energy efficiency best practice.

Specification and Design ICTs

The planning phase of a production system is well supported by various ICTs. Starting in the early definition phase with requirement management tools, followed by CAD tools for the detailed design of resources as well as computer-aided process planning (CAPP) and CAM tools for the planning of processes and machine operations, such as, for example, the generation of numerical control (NC) files, and finally various simulation tools for validation and verification of the future production systems (such as material flow simulations). All tools can be summarized under the concept of the "digital factory"[5] and all required information and data are managed in product life cycle management (PLM) systems, which also support the integration of other IT solutions used throughout the whole product life cycle, not only in the product development phase. However, especially in manufacturing planning, tools remain isolated. As such, fully integrated planning and validation of complete production systems and manufacturing process chains has yet to be achieved[6].

"Fully integrated planning and validation of complete production systems and manufacturing process chains has yet to be achieved."

Even though decisions in the design phase influence significantly the energy consumption in the usage phase of a production system, energy-related aspects are rarely considered due to absent methodical support in the factory planning phase[7][8]. Specifically, energy performance requirements are seldom captured in the specification phase and electronic catalogs of resources or process templates poorly cover energy consumption, while energy dependency aspects are typically not considered in early stages. The result is that holistic energy simulation is not supported and decisions are not verified by evaluating and accessing the energy consumption of the future production system.

"…energy-related aspects are rarely considered due to absent methodical support in the factory planning phase."

Materialization ICTs

The term *materialization* refers in this content to the formation of a production system; this encompasses the building and setup of the production equipment within the factory, but also implementation of control and automation ICTs, as well as any retrofitting. From an energy consumption perspective this phase is dominated by the construction of the building and the transport and erection of the building components including the production equipment (machines). This phase is supported by conventional logistical and project management software mainly from the construction sector. In terms of energy efficiency, ICTs tools should be used to define strategies and support decisions for onsite/offsite production of components of the building but also of large production equipment as well as the optimization of its transportation and the selection of ideally local suppliers.

Automation and Operational Decision Support ICTs

The usage phase of a production system is identical to the manufacturing phase of the product being produced and is mainly assisted by ICT in terms of automation, status, and process monitoring, as well as controlling and scheduling of production tasks. As in the design phase, ICTs for automation and control of manufacturing processes include methods and technologies on different levels. On the machine level, resources are controlled with programmable logic controllers (PLCs), which are responsible for controlling single-machine tasks on the lowest level. Machine data and operating data that can be acquired by means of supervisory control and data acquisition (SCADA) systems for example, are processed by manufacturing execution systems (MES) and used for controlling purposes. Production planning and control (PPC) systems assist the management and scheduling of the overall production processes. "The energy consumption [in the usage phase] of manufacturing facilities can be reduced by either using more efficient technologies and equipment, and/or through improved monitoring and control of energy used in infrastructure and technical services"[9]. In both cases ICT is essential.

ICT has a significant impact in the field of smart motor systems for the intelligent load adjusted control of electrical drives[10] and compressed air systems. Furthermore energy must be taken into account not only within the control of electrical drives but also for the control of the entire machine (for example, optimized NC programs or the selective switch off of components).

"ICT has a significant impact in the field of smart motor … and compressed air systems."

In the field of improved monitoring and control it is first essential to set up a wide network of sensors at the machine or even component level, in order to understand where and how much energy is being consumed. ICT can additionally support the analyses and visualization in terms of (automated) operational decision support. A further application is condition-based maintenance, whereby the condition of production equipment is monitored in order to launch preventative or even predictive maintenance activities.

Resource and Process Management ICTs

Resource and financial controlling in manufacturing is carried out with the assistance of enterprise resource planning (ERP), PPC systems, or MES. But energy efficiency is not in scope of scheduling or process planning and control yet[11]. ICT should be utilized in a way that takes a broader account of traditional business objectives by including energy efficiency. Energy-optimized production schedules directly affect energy efficiency, such as by reducing of standstill times with high base load power consumptions, cutting of energy-intensive retooling, rearrangements, or startups, or by avoiding unproductive consumptions during idle time. Furthermore energy-intensive tasks should be scheduled when the lowest economic and ecological effects are to be expected and macroeconomic energy consumption peak loads are avoided. Additionally, in terms of intralogistics planning, reduced inventory can assist in saving energy by reducing logistical efforts and/or be optimizing storage HVAC control. In the wider scope of production networks, supply chain management (SCM) tools are used for optimizing logistics and the flow of goods, which offers potential to optimize overall energy consumption of production networks.

In terms of knowledge sharing, ICT offers great potential by providing decision makers with energy-relevant information. Current tools for energy auditing assist professional consultants by covering data collection and analysis functions. More information platforms for a wider audience are required, platforms that visualize energy consumption of a factory on different levels, taking into account energy related key performance indicators (KPIs). These KPIs should also be used for benchmarking or even for energy efficiency labels for machines or processes. Guidelines and knowledge repositories are required for assisting in design of energy-efficient production equipment as well as processes. In this case, especially for the manufacturing process, a reliable database is required for listing statistical data for energy consumption and CO_2 emissions[12].

"Guidelines and knowledge repositories are required for assisting in design of energy-efficient production equipment as well as processes."

Technical and Semantic Interoperability ICTs

When it comes to the interaction of different ICTs integration and interoperability is the main issue. With regard to energy efficiency in planning and operation of production systems, the integration of energy consumption data in ERP and PLM systems is essential for the realization of ecological in addition to typical objectives (such as cost, time, and quality)[13]. For this reason interfaces with automation software and sensors are required to feedback energy consumption data from the shop floor into MES as well as back into ERP or PLM. In the design phase the integrated development and simulation

"…the integration of energy consumption data … is essential for the realization of ecological in addition to typical objectives."

of a product and its manufacturing processes has potential for increased energy efficiency in manufacturing[14]. Products can be designed for efficient manufacturing processes, whereas the constraints of the machine tools are taken into account in an early stage. ICT enables this kind of development and supports understanding with regarding the complex interdependencies between product design and manufacturing process planning. The integration of shop floor data in planning tools can also be used for real-time simulations of manufacturing factories to predict interdependencies, optimize manufacturing processes, and simulate the flexibility of the factory in the usage phase.

Trading/Transactional Management ICTs

Energy management systems are used for managing, monitoring, and reducing energy consumption of complex systems[15] and enable the integrated planning, monitoring, control, and optimization of energy usage at plant level. The main scope of these systems is the identification of energy losses and the prevention of load peaks. Even though there is a wide range of functionalities, the use of energy management is still not standard in industries. Apart from the independent energy management of production machines and processes, other subsystems of the factory building such as heating, lighting, and compressed air subsystems need to be integrated into a holistic energy management system, a fact supported by recent research[16]. However, holistic energy-efficient control of production sites extends beyond facility borders. It includes traditional factors such as production time, production rate, and quality, as well as energy efficiency, but it also takes into account the energy market dynamic, which is complex due to the growing number of deterministic energy sources such as wind turbines. This includes for instance aspects such as shifting workloads of secondary processes to avoid energy demand peaks or benefit from temporarily decreasing energy costs.

"…holistic energy-efficient control of production sites extends beyond facility borders."

Assessing ICT Impact on Energy Efficiency

The capability maturity element of the framework was designed to allow for common assessment of individual ICT offerings. Assessment followed the following three steps:

1. In the first step the ICT offering is defined and categorized in terms of the SMARTT taxonomy, the most relevant phases to consider and the likely first, second and third order effects are initially identified
2. In the second step we refine that which is to be subsequently assessed by identifying the most relevant phase(s)of both the ICT offering and the host system
3. In the last step domain expertise and secondary data is used to assess and estimate the *potential net impact*, *current adoption,* and *potential adoption* of the ICT offering with respect to energy efficiency, based on a five-point maturity scale. The five levels of maturity, as defined within REViSITE, are represented in Figure 4, with 1 being "inefficient" or "low" and 5 being "optimized" or "high."

"…domain expertise and secondary data is used to assess and estimate the potential net impact…"

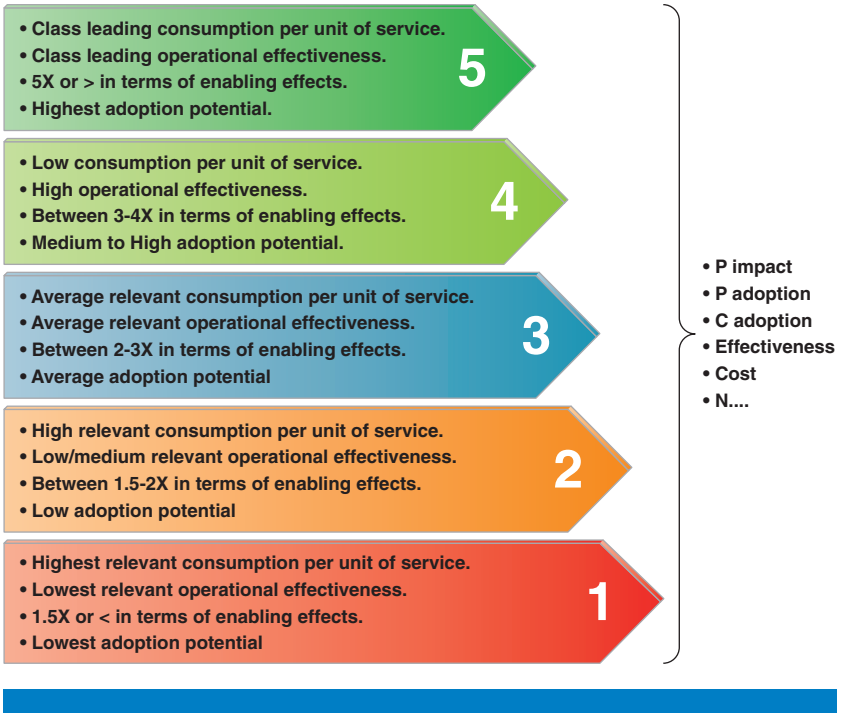


Figure 4: REViSITE defined five-point maturity scale (Source: Intel Corporation (REViSITE Partner))

However, ultimately the methodology was used to estimate the likely impact of the 23 ICT subcategories, as opposed to individual ICTs aligned to those subcategories. REViSITE community members, that is, experts in the four target sectors and the ICT sector, were invited to score the current adoption, potential adoption, and potential impact of the 23 SMARTT subcategories of Figure 5. An online survey medium was utilized and respondents scored the subcategories in the context of their own identified sector, based on a simple five-point scale. By doing so we were able to build sector-specific views regarding SRA relevance, where:

SRA Relevance = [Potential adoption − current adoption score] × Potential Impact

Assessment Applied to Manufacturing

A sector-specific ranking of the 23 themes identified in Figure 5 was developed utilizing the process described above. Figure 6 illustrates responses as they relate to manufacturing. Respondents rated current adoption, potential adoption, and potential impact based on a five-point scale. By scoring subcategories based on current adoption and potential adoption, we were able to use the adoption delta between the two in combination with potential impact scores to indicate relevance in terms of research priorities.

"We were able to use the adoption delta … in combination with potential impact scores to indicate relevance in terms of research priorities."

It was felt important to score both potential impact and SRA relevance, especially when it came to cross-sectorial comparison. For example subcategory 5—"Simulation—as part of the design function" and

S	1	Design conceptualisation ICTs for requirement engineering & ideation. E.G. Quality Function Deployment, Mind maps etc
S	2	Human factors Engineering ICTs to gather and model data describing the behaviour of end users/energy consumers
S	3	Visual/spatial design ICTs E.G. CAD (Autodesk, 3D studio max), Multimedia (e.g. Flash, Silverlight), Graphics (e.g. Photoshop, Illustrator) for digital mock-up etc
S	4	Causal Modelling ICTs used to describe/predict relationships in physical systems E.G. computer-aided diagramming (e.g. Sankey, Cause and effect, influence diagrams etc), Life cycle modelling, statistical packages such as JMP & MatLab etc
S	5	Simulation ICTs for predicting/estimating the dynamic behaviour of a system as part of the design function E.G. Computational Fluid dynamics, Finite element mode analysis, power system simulation etc
S	6	Product/component specification and selection ICTs E.G. material characteristic database specifying embedded energy, recyclability, thermal performance etc
M	7	Mobile Decision Support ICTs that utilise real-time communication to facilitate in the field decision making particularly in construction or civil engineering tasks
A	8	Secure/resilient wired, wireless and optical infrastructure for operational communication, monitoring & control
A	9	Embedded intelligent devices (micro architecture)for operational control, sensing & actuation at machine, plant or building level
A	10	Software & algorithms for operational monitoring & actuation of devices at machine, plant or building level
A	11	Inference sensing Software & algorithms for pattern & signal identification at machine, plant or building level
A	12	Operational decision support ICTs that integrate high level diverse systems such as safety, security, weather and energy etc at individual, building or district level for near real-time decision making
A	13	User Centred Data Visualisation ICTs to support system state awareness by human operators/users
R	14	Inter-Enterprise ICTs for supporting coordination e.g. contract & supply-network management in the context of reduced energy consumption
R	15	Business Process Integration & collaboration ICTs E.g. collaboration support, groupware tools, electronic conferencing, social-media,etc
R	16	Knowledge sharing ICTs, knowledge management, knowledge repositories, knowledge mining and semantic search, linked data, long-term data archival and recovery at enterprise or inter-enterprise level
R	17	ICTs for data mining & analytics in terms of energy consumption & optimisation, pattern identification, predictive diagnostics & analytics at enterprise or network level
R	18	Modelling & simulation ICTs e.g. What-if scenario planning continuous improvement across a sectors life cycle
Tr	19	Trading & Energy Brokerage ICTs e.g. Consumer/Producer forecasting algorithms, energy source tracking, consumption/price negotiation
Te	20	ICT standards and protocols for interoperability across heterogeneous devices at an enterprise, network or environmental level
Te	21	Real-time analytical technologies such as Complex Event Processing and in-memory databases for enhanced operational control and awareness
Te	22	Integration technologies/approaches such as service orientation and event driven architectures to facilitate heterogeneous device data interoperability at enterprise, network and environment level
Te	23	Use of cloud based services for tasks such as data management, monitoring and analysis

Figure 5: The 23 subcategories assessed by respondents (Source: Intel Corporation (REViSITE Partner))

subcategory 6—"product/component specification and selection" were scored comparably high with respect to potential impact on energy efficiency. However, the adoption delta of the two themes was very different. The indication was that subcategory 5, although important in the context of energy-efficient manufacturing, was already relatively sophisticated and therefore less relevant in

"This is the real value of the approach in that it offers a common lens into the technology, practices and research of other sectors."

terms of research priorities. However, this might not be the case in another sector and the manufacturing sector could therefore serve as a "best practice" use case. This is the real value of the approach in that it offers a common lens into the technology, practices and research of other sectors.

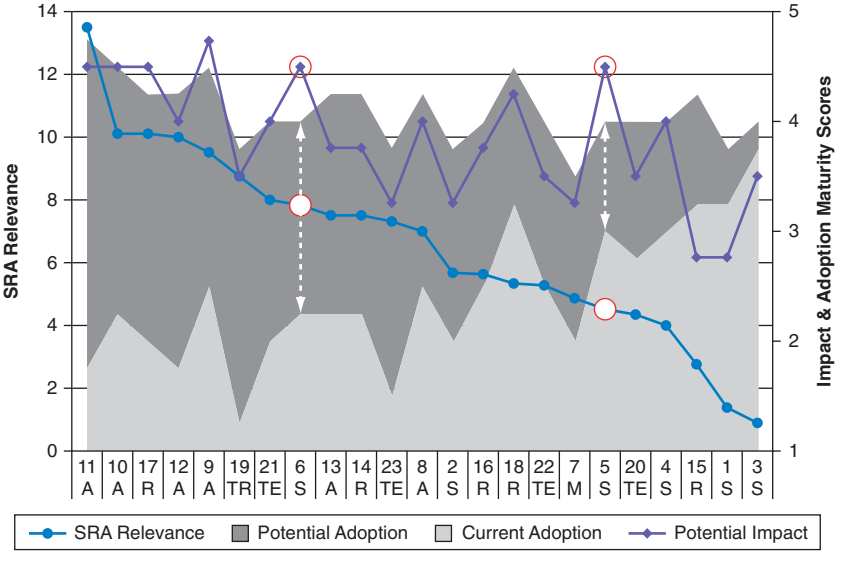


Figure 6: Manufacturing ranking graph
(Source: Intel Corporation (REViSITE Partner))

Figure 7 represents the same information as that of Figure 6 in the form of ranked list of the 23 ICT themes, highest to lowest in terms of SRA relevance. The top half of Figure 7 is highlighted to guide conversation in terms of prioritization but that is not to say other themes are to be ignored.

The column furthest right indicates the adoption delta, that is, *potential adoption* less *current adoption* and is also an indication of *potential impact* when the overall ranking position is taking into account.

The analysis for all four target sectors was subsequently analyzed with respect to common research priority trends.

How the Assessment Output Was Used

"The output together with subsequent cross-sectorial trending was used as reference material for workshop discussion, qualitative interviews, and analysis."

It must be stressed that the output was by no means seen as a definitive conclusion. Instead the output together with subsequent cross-sectorial trending was used as reference material for workshop discussion, qualitative interviews, and analysis.

Having completed a consultation process the following ICT themes were identified as the most important to consider in the context of energy-efficient production systems. This exercise was repeated for all sectors. Unsurprisingly there was duplication of themes and as such many of those listed below can be considered sector independent.

MFG: P - SRA [Relevance] = P Imp* (P Adopt - C Adopt)

SMART T cat.	ICT Theme no.	ICT Theme description	Adoption Delta
A	11	Inference sensing Software & algorithms for pattern & signal identification at machine, plant or building level	3.00
A	10	Software & algorithms for operational monitoring & actuation of devices at machine, plant or building level	2.25
R	17	ICTs for data mining & analytics in terms of energy consumption & optimisation, pattern identification, predictive diagnostics & analytics at enterprise or network level	2.25
A	12	Operational decision support ICTs that integrate high level diverse systems such as safety, security, weather and energy etc. at individual, building or district level for near real-time decision making	2.50
A	9	Embedded intelligent devices (micro architecture) for operational control, sensing & actuation at machine, plant or building level	2.00
TR	19	Trading & Energy Brokerage ICTs e.g. Consumer/Producer forecasting algorithms, energy source tracking, consumption/price negotiation	2.50
TE	21	Real-time analytical technologies such as Complex Event Processing and in-memory databases for enhanced operational control and awareness	2.00
S	6	Product/component specification and selection ICTs E.G. material characteristic database specifying embedded energy, recyclability, thermal performance etc.	1.75
A	13	User Centred Data Visualisation ICTs to support system state awareness by human operators/users	2.00
R	14	Inter-Enterprise ICTs for supporting coordination e.g. contract & supply-network management in the context of reduced energy consumption	2.00
TE	23	Use of cloud based services for tasks such as data management, monitoring and analysis	2.25
A	8	Secure/resilient wired, wireless and optical infrastructure for operational communication, monitoring & control	1.75
S	2	Human factors Engineering ICTs to gather and model data describing the behaviour of end users/ energy consumers	1.75
R	16	Knowledge sharing ICTs, knowledge management, knowledge repositories, knowledge mining and semantic search, linked data, long-term data archival and recovery at enterprise or inter-enterprise level	1.50
R	18	Modelling & simulation ICTs e.g. What-if scenario planning continuous improvement across a sectors life cycle	1.25
TE	22	Integration technologies/approaches such as service orientation and event driven architectures to facilitate heterogeneous device data interoperability at enterprise, network and environment level	1.50
M	7	Mobile Decision Support ICTs that utilise real-time communication to facilitate in the field decision making particularly in construction or civil engineering tasks	1.50
S	5	Simulation ICTs for predicting/estimating the dynamic behaviour of a system as part of the design function E.G. Computational Fluid dynamics, Finite element mode analysis, power system simulation etc.	1.00
TE	20	ICT standards and protocols for interoperability across heterogeneous devices at an enterprise, network or environmental level	1.25
S	4	Causal Modelling ICTs used to describe/predict relationships in physical systems E.G. computer-aided diagramming (e.g. Sankey, Cause and effect, influence diagrams etc.), Life cycle modelling, statistical packages such as JMP & MatLab etc.	1.00
R	15	Business Process Integration & collaboration ICTs E.g. collaboration support, groupware tools, electronic conferencing, social-media, etc.	1.00
S	1	Design conceptualisation ICTs for requirement engineering & ideation. E.G. Quality Function Deployment, Mind maps etc.	0.50
S	3	Visual/spatial design ICTs E.G. CAD (Autodesk, 3D studio max), Multimedia (e.g. Flash, Silverlight), Graphics (e.g. Photoshop, Illustrator) for digital mock-up etc.	0.25

Figure 7: Manufacturing SRA relevance
(Source: Intel Corporation (REViSITE Partner))

Full integration and validation of complete production systems and manufacturing process chains need to be achieved, and energy performance requirements need to be captured in the specification phase, with energy dependency aspects considered in early design/planning stages. Themes to consider are:

- The concepts of eco-design and of the digital factory
- ICT for rationalization/selection of components for better energy efficiency
- Electronic catalogs of resources/processes including relevant attributes of energy efficiency, that is, embedded energy, energy performance in use

"The importance of "automation and operational decision support ICTs" was unsurprisingly borne out in the research."

The importance of "automation and operational decision support ICTs" was unsurprisingly borne out in the research.

Themes to consider are:

- Embedded ICTs sensor and actuation devices with increased levels of autonomous diagnostics and machine learning, or understanding and controlling energy efficiency of the holistic production systems—targeting motor systems, compressed air, computer numerical control (CNC), and other energy-intensive processes
- Real-time monitoring and analytics; predictive controls algorithms for optimal production system performance including energy consumption, efficiency, and flexibility
- Intuitive, easily deployable, easily usable, dynamically adaptable visualizations incorporating streamed and asynchronous data

"From a horizontal life cycle perspective three words permeated the REViSITE research: integration, interoperability, and standards."

From a horizontal life cycle perspective three words permeated the REViSITE research: integration, interoperability, and standards. Themes to consider are:

- Agreed semantic ontologies/data models for describing energy flows that span production systems, the built environment that houses them, and the energy grids that power them
- The need to agree to and adopt data exchange protocol standards for system integration and interoperability
- Dynamic dependable extensible networking technologies and platforms essential to optimal operations and value chain performance
- Dynamic dependable and extensible local versus cloud-based control services for automated control and monitoring
- Real-/near-time analytics and machine learning, with the ability to manage large heterogeneous data volumes
- Common tools and metrics for energy performance estimation
- Green product life cycle management and holistic support tools for optimized production/supply networks and reduced embedded energy product profiles
- Enhanced value-driven business processes and ICT-enabled business models

- Regulations and market models that take into account the environmental aspects and ethical concerns of citizens
- ICTs to facilitate virtual enterprise business relationships
- Enhanced knowledge sharing including infrastructure, knowledge mining, semantic mapping, filtering, knowledge consolidation algorithms, distributed databases, and catalogs of reusable energy-efficient solutions

Conclusion

Industrial sectors are faced with a sustainable paradox in trying to deliver more with less. Renewable energy sources coupled with improved energy efficiency is central to achieving European energy security and sustainability targets.

ICT pervasiveness offers a unique opportunity to address these challenges by enabling a sustainable cross-sectorial energy network. However, assessing and demonstrating the enabling impact of ICT is often an arduous process.

The REViSITE Framework is proposed as a one means of common assessment. The framework combines "life cycle thinking," an adapted "Capability Maturity Framework," and the REViSITE developed SMARTT taxonomy.

The taxonomy was used throughout the project as an integrative classification system and an aid to cross-pollination in terms of energy efficiency best practice. The overall approach is posited as a useful lens into the technologies, practices, and research of other sectors.

"Industrial sectors are faced with a sustainable paradox… ICT pervasiveness offers a unique opportunity to address these challenges."

"The REViSITE Framework is… posited as a useful lens into the technologies, practices, and research of other sectors."

References

[1] Barroso, J.M.D., 2008, ICT Industry Has a Major Role to Play in the European Economy of the 21st Century, Reference: SPEECH/08/120, Date: 04/03/2008, http://europa.eu/rapid/pressReleasesAction.do.

[2] Barroso, J.M.D., 2011, Growth and economic governance—orientation debate on energy and innovation, Reference: SPEECH/11/1 Date: 05/01/2011, http://europa.eu/rapid/pressReleasesAction.do.

[3] EU commission 12/3/2009 http://eurlex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2009:0111:FIN:EN:PDF

[4] Ellis, K, Hannus, M, 2010, ICT4EE a common methodology to assess the impact of ICT developments, http://revisite.eu/del21.html

[5] VDI Directive 4499, Blatt 1, 2008, Digitale Fabrik Grundlagen.

[6] Stark, R. et al., 2010, The way forward of Virtual Product Creation—how to achieve robustness in using digital engineering technology?, Piracicaba, Proceedings of 15th International Seminar on High Technology.

[7] Reinema, C. et al., 2011, Energieeffiziente Fabriken: Ein Vorgehen zur integralen Gestaltung, wt Werkstattstechnik online 101/H.4, pp. 249–252.

[8] Engelmann, J. et al., 2008, Energieeffizienz als Planungsprämisse, Industrie Management 24/3, pp. 61–63.

[9] Seow, Y., Rahimifard, S., 2010, A Framework for Modelling Energy Consumption within Manufacturing Systems Proceedings of 43rd CIRP International Conference on Manufacturing Systems, pp. 222–227.

[10] GeSI, 2009, Smart 2020: Enabling the low carbon economy in the information age, http://www.GeSI.org

[11] Bunse, K., Vodicka, M., 2010, Managing Energy Efficiency in Manufacturing Processes—Implementing Energy Performance in Production Information Technology Systems, IFIP AICT 328, pp. 260–268.

[12] Duflou, J.R., Kellens, K., Dewulf, W., 2011, Unit process impact assessment for discrete part manufacturing: A state of the art, CIRPJ-129.

[13] Junge, M., 2007, Simulationsgestützte Entwicklung und Optimierung einer energieeffizienten Produktions-steuerung, Kassel University Press GmbH.

[14] Kaufmann, P.; Walker, M, 2009, Industrial Energy Optimization: Managing Energy Consumption for Higher Profitability, http://www.rockwellautomation.com.

[15] Hesselbach, J. et al, 2008, Energy Efficiency through optimized coordination of production and technical building services Conference Proceedings LCE2008—15th CIRP International Conference on Life Cycle Engineering, pp. 624–628.

[16] Ellis, K., Sheridan C., 2011, ICT4EE—Impact Assessment Model, http://revisite.eu/del23.html

Author Biographies

Keith A. Ellis is a research scientist with the Energy and Sustainability Lab, Intel Labs. Keith's prime research interests focus on ICT enablement in the context of energy/resource efficiency. He has participated in several EU funded and Intel projects in this space. Keith has worked with Intel for 17 years with roles in both manufacturing and IT innovation.

Farid Fouchal is a Post-doctoral Researcher in Digital Construction and ICT for Energy Efficiency at Loughborough University, UK. He has a physics background and a PhD in Computer Science and Engineering. His research experience includes work in several EU funded sustainable manufacturing and built environment research projects, he has over 20 publications.

Tarek Hassan is Professor of Construction Informatics at the Civil and Building Engineering Department of Loughborough University, UK. His academic experience is complemented by 10 years industrial experience with several international organizations. Professor Hassan has lead and participated in several EU funded projects in the ICT/built environment domain and has over 140 publications.

Daniel Kuhn is a Research Fellow in the division of Virtual Product Development at Fraunhofer Institute for Production Systems and Design Technologies. Daniel has been involved in several EU and nationally funded projects in the sustainable manufacturing domain. His research interests focus on digital design methods for energy-efficient production systems as well as product data management of reverse engineering data.

Charles G. Sheridan is the Associate Director of the Energy and Sustainability Lab, Intel Labs. He co-chairs a leading consortium focused on sustainable computing. Charlie has been involved in several EU, national, and Intel projects in the sustainability domain. Charlie has worked with Intel for 17 years with roles in both automation and IT innovation.

TAKING TURNS TO SAVE POWER

Contributor

Kim Shearer
Senior Program Manager,
Technology Policy Group,
Microsoft Corporation

Improving power management for PCs and devices offers value in three important areas: user experience (UX), operational expenses (OpEx), and responsibility. For any mobile device effective power management is key to providing long battery life and facilitating an instant-on, always-connected experience. As any manager of a large number of systems will attest, whether they are data center servers or corporate desktops, even a small reduction in power use per system can yield significant annual savings in energy.

The ICT industry consumes a significant amount of energy globally with the attendant emissions of greenhouse gases, mercury, and other poisons and pollutants. Keeping the energy use of the IT industry at a reasonable level will reduce the likelihood of regulation of energy use and emissions, allowing the IT industry to retain room to innovate.

In Windows 7*, Microsoft made significant improvements in efficiency and power management. Windows 8 builds on Windows 7 in three key areas: OS platform efficiency (idle hygiene), runtime device power management, and a Metro style application model.

Some of the improvements are at the OS level and benefit all customers and applications, but in order to realize the full potential of Windows 8 advances, developers will need to integrate their applications with the new power management capabilities.

Introduction

Effective power management offers customer value in a number of ways. Reduction in operational expenses (OpEx) is valuable for businesses. Even a small reduction in power used per system can yield significant savings in energy and costs when the number of systems is large. Effective power management in laptops and mobile devices improves battery life and allows reductions in the weight and size of devices. Power management is also a key factor in enabling the instant-on, always-connected experience that mobile device users' desire.

"The information and communications technology (ICT) industry is responsible for consumption of a significant amount of energy globally."

The information and communications technology (ICT) industry is responsible for consumption of a significant amount of energy globally (approximately 2.2 percent of US production and between 1.1 percent and 1.5 percent of global energy production[2]), with the attendant emissions of greenhouse gases, mercury, and other poisons and pollutants. As the use of ICT worldwide continues to grow, the demand for energy will also grow unless the industry as a whole increases the energy efficiency of PCs and other computing devices. Power management can help restrain the energy use of the

IT industry to a reasonable level, which can improve quality of life and reduce the likelihood of regulation that might limit the industry's ability to innovate.

In Windows 7 Microsoft made significant improvements in efficiency and power management, with features such as timer coalescing enabling extended idle periods and greater residency in low power states. Windows 8 builds on this and provides significant advances in three key areas:

- Idle hygiene
- Runtime device power management
- Metro style application model

Some of these improvements are at the OS level and benefit all customers and applications, but in order to get the most from Windows 8 improvements developers will need to understand and use the new application life cycle model. The new Metro style application model is right by design for application power management. Applications are suspended when they are not in the foreground, which makes it simpler for developers to write power-efficient applications. However, applications using the Metro style model can do much more by using the notification service, Live tiles, and by following principles of connected standby and good idle hygiene.

At a high level the goals for the Metro style application model are:

- Simplify writing applications that integrate well with power management.
- Provide power-efficient operating system managed infrastructure to perform common background tasks.
- Provide infrastructure to offload communication tasks so that most of the system can remain idle while still responding to real-time communications input.

By providing infrastructure such as the Windows notification service to offload updates and communications, Windows 8 allows much longer idle periods for most of the system. Providing infrastructure for common background tasks, such as file upload, to be passed to and managed by the operating system, Windows 8 makes it simpler for applications to perform these tasks. It also allows the operating system to use contextual data, such as power status and network cost, to manage these tasks in a resource-efficient way.

This article presents some of the important new power management concepts in Windows 8 and how they impact application development. Developers who use the new features can provide a great user experience while dramatically reducing power consumption of their application.

"Developers who use the new features can provide a great user experience while dramatically reducing power consumption of their application."

Idle Hygiene

One of the themes for developers of Windows 7 applications was to "get idle and stay idle." Modern PC platforms have processor and chipset idle states that can allow power to parts of the silicon to be turned off, or the clock to be

completely stopped. Devices such as hard disks and network interface cards also commonly have multiple power states. In order to make best use of these idle and low power states, the system should be allowed to enter these states for as long as possible, as soon as possible, when the system is "idle." The longer the system is in these low power states, the more energy can be saved. For processor and chip idle states, a minimal residency duration is required to save energy; the processor needs to be idle long enough to make it worth the transition into idle. This is because some power is consumed to enter and also to leave the idle state. Software should allow the system and hardware to spend as much time in power saving states as possible and have as few entries to and exits from idle state as possible. This allows the maximum energy savings over time for the system.

Microsoft added features such as timer coalescing in Windows 7 to help extend idle periods. Timer coalescing provides a way for applications to specify a tolerance parameter for timer expiration. An application that sets a timer to expire after one second might specify a tolerance of 0.1 seconds. When the timer is due to expire Windows can determine if there are any other timers due to expire in the same timer period. If there are no other timers Windows can delay servicing the timer till a later period, repeating this process if necessary up until 0.1 seconds later. While it is possible that the timer will still be serviced by itself, coalescing often allows the operating system to group a number of timers to be serviced together in one timer period. By grouping timers in this way Windows can allow several timer periods to pass without servicing a timer. During these idle periods the hardware gains maximum power savings from several periods that would otherwise have had small amounts of processing.

A fundamental change between Windows 7 and Windows 8 is that the concept of a regular timer period no longer exists and background tasks are coalesced by default. In Windows 7 the default timer period is 15.6 ms, and the maximum idle period was limited by this figure. In Windows 8 there is no longer a regular timer tick frequency. Results from testing pre-release suggest that Windows 8 can achieve much longer idle durations than Windows 7. The comparison of typical user scenario idle durations in Figure 1 shows that in Windows 8 the number of timer durations of 15.6 ms or less has been reduced to less than 40 percent with more than 35 percent of the idle durations being longer than 100 ms.

"…in Windows 8 the number of timer durations of 15.6 ms or less has been reduced to less than 40 percent…"

Instead of regularly checking for timer expiration, the system processor remains in an idle state until an event needs to be handled. For Windows 8 many of the operating system services have had their idle period increased. The work done to increase the length of time between periods of activity is called *idle hygiene*. As processors continue to improve power savings at idle, idle hygiene becomes more important. The work on idle hygiene in Windows 8 benefits any scenarios, as it reduces the power use on any system. These increased idle periods do not come at the expense of performance or responsiveness.

"The work on idle hygiene in Windows 8 benefits any scenarios, as it reduces the power use on any system."

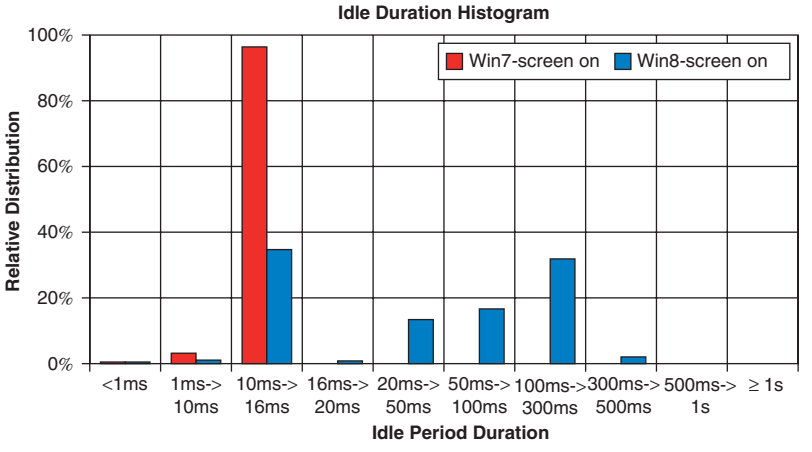


Figure 1: Idle duration distribution comparison for Windows 7 and Windows 8
(Source: Microsoft Corporation, 2012)

Runtime Device Power Management

In the last decade silicon manufacturers have continued not only to increase processor performance, but have also dramatically increased processor performance per watt. This has made it possible to create very low powered devices that still provide very useful processing performance. As the power per watt of processors has increased, OEMs have created very compact general-purpose computing devices. The usefulness of these devices depends heavily upon battery life.

To achieve great battery life requires not only a low power processor, but other components such as disk, memory, and screen must also be very power efficient. Even with energy-efficient hardware, the coordination between hardware and operating system is critical. Resources such as memory consumption, hard disk use, and processor cycles are closely managed by operating systems. In fact, a significant part of the operating system's job is to manage these core resources.

"Even with energy-efficient hardware, the coordination between hardware and operating system is critical."

Power as a First Class Resource

In Windows 8, power has been elevated to a first class resource. On devices with suitable capabilities, power use is managed along with other system resources to ensure the battery life and performance necessary for mobile devices. In order to enable this management while insulating applications from power features of specific hardware, Windows 8 has introduced a new device power framework.

"On devices with suitable capabilities, power use is managed along with other system resources."

This new framework allows all devices to advertise their power management capabilities and integrate them with a driver called the Power Engine Plug-In (PEP). PEP is designed for system-on-chip architectures and should be provided by the silicon manufacturer describing the capabilities of their specific

system. This makes it possible for system drivers such as USB controllers or pointing device controllers to deliver optimal power management for a system without extensive customization.

The architecture used to enable PEP is based on a port/mini-port scheme. PEP is a new mini-port that handles the power management interface to devices. In order to maintain good power management, devices are enabled to talk to PEP very quickly and efficiently.

Connected Standby

A device in connected standby appears to the user to be off, but remains in a state that allows for network traffic and other events to be processed and update application presence or wake the device if required. The ability of a device to remain in connected standby for many hours is an important part of the utility of mobile phones, tablets, and other portable devices. It should be clear that this requires not only low power hardware and great idle hygiene, but also good hardware and operating system coordination. Without these attributes, the battery life of a device in connected standby would be reduced.

While these features are necessary, they are not sufficient if the most of the system must be awake to service communications requests. In order to allow the system to manage power effectively while still providing an always-connected experience Windows 8 supports offloading of communications to the network interface card (NIC) hardware, usually either Wi-Fi* or mobile broadband.

"Communications modules can filter incoming requests and only wake the processor when necessary, and applications can use lightweight background tasks to update status."

This has two main advantages for power management: communications modules can filter incoming requests and only wake the processor when necessary, and applications can use lightweight background tasks to update status. By keeping status updated using mechanisms such as Live tiles, applications can reduce the need for users to unlock a device and activate full applications. This can allow much of the system to remain at idle for longer periods.

In Windows 8 on a capable system, the NIC filters incoming packets, only waking the system at necessary times, while the majority of the system remains in standby. When a communication is received, the NIC will apply filters to determine if this meets the requirements for alerting an application and waking the system. The systems that use this scheme are push notifications and real-time triggers. This offload technology is also applied to suspended Metro style applications when the screen is on.

One advantage of this is that an application may not need to be receiving CPU time, or even be in memory, for the infrastructure in Windows 8 to take suitable action. This saves the cost of swapping between applications and allocation of CPU cycles to applications that are waiting on input, and keeping most of the system awake to wait for communications. The mechanisms and application model for this are presented in the next sections.

Metro Style Application Model

As part of the Windows 8 release Microsoft has introduced a new application model called the Metro style application model. In part this is a new design language for application UI but the Metro style application model has much deeper implications, providing improvements to system and application power management.

In previous releases of Windows it has been important for applications to work well with power management. This largely meant using availability requests[4] correctly and facilitating system sleep and display power management. It has been unfortunately common to find applications that do not work well with power management. The new Metro style application model provides improvements to power management across a wide range of system usage. In part this is achieved by making it easy for developers to get power management right, and difficult for developers to get it wrong. The improvements to power management also come from allowing the operating system to manage background processing, and in some scenarios perform background tasks without the involvement of the application.

"The new Metro style application model provides improvements to power management across a wide range of system usage."

One important part of writing applications that work well with power management is to make sure that code is not run unless necessary. To this end the Metro style application model makes it easy for developers to be sure an application runs at the right time, does not run unless necessary, and uses resources in a power-friendly way.

For the majority of Metro style applications the general rule is, if the application is not on the screen, or the screen is not on, the application is not active. Any Metro style application that is not active will not be allocated CPU cycles.

"Any Metro style application that is not active will not be allocated CPU cycles."

Of course Metro style applications that are not active are not prevented from completing useful work. It is however enforced that Metro style applications use suitable mechanisms for both background tasks and event notification, rather than have processes or services that run continuously in the background. All background activity is managed by the operating system to guarantee that no application can consume excessive resources. In this way battery life and user responsiveness are not compromised by unnecessary or excessive background activity.

The three possible execution states for a Metro style application are:

- In the foreground actively running
- Suspended in the background
- Performing a defined background activity

By correctly using the notification and background task infrastructure Metro style applications can keep the user informed without continual consumption of system resources. For example, an application can use push notifications to update its Live tile at appropriate times. This keeps the user up to date and informed without the application having to run for an extended period.

Application use of operating system managed infrastructure to perform all background activity allows for much tighter management of resources consumed by background tasks. This enables operating system management of power, battery life, and user responsiveness as first class resources.

Foreground Application

A Metro style application is active if it is full screen or is snapped to one side of the screen. For the application to be active the screen must also be on, so that it is reasonable to believe that the user is interacting with the system. In Windows 8 it is assumed that the foreground application is the one most important to the user and that application receives use of the system resources. When the user switches from one application to another, the current foreground application is suspended and cannot run any code. The new foreground application is then given access to system resources such as CPU, network, memory, and disk. In this way only one application is active at any time, and no applications are active when the user is not interacting with the system. This reduces background activity, which reduces power consumption by increasing idle periods.

It is intended that Metro style applications will do most of their work while in the active state. In order to provide an instant-on, always-connected experience to users, applications need to be able to receive notifications and process some important tasks while they are not active. Examples of this might be updating a Live tile to display changed status for an online auction bid, or periodic download of up-to-date articles from a news service.

"The way this background work is done is one of the more significant differences between the Metro style application model and the desktop application model."

The way this background work is done is one of the more significant differences between the Metro style application model and the desktop application model. Rather than use a service or other continuously running application as a desktop application would, a Metro style applications uses the Windows notification service (WNS), background transfer APIs, and the background task infrastructure of Windows 8. In this way Metro style applications can remain up to date, and perform important tasks such as printing and copying, without the resource drain of a background process. The Windows notification service and background transfer API are sufficient for most application scenarios and have been highly optimized for system performance and battery life.

Suspended Applications

When the user switches to a new foreground application, the previous foreground application is suspended and the Windows scheduler will no longer include the suspended process in CPU scheduling. In this way Windows prevents multiple applications from generating background activity and preventing the CPU from entering low power states, even if most of applications are not doing useful computation. Suspended applications are still memory resident, and can therefore be resumed very quickly. This gives fast switching between applications, without negative impact on battery life or system performance.

Unfortunately, system resources are finite. If a user keeps switching to Metro style applications that are not already active without returning to previous applications, eventually a system resource, most likely memory, will be exhausted. When a system resource is exhausted the system will have to terminate one of the currently suspended applications, perhaps to allow another application to become active. How often this happens will depend on the memory footprint of the suspended applications and the size allocated to the system swap file.

When the operating system must choose a suspended application to terminate, several factors are taken into account. Factors considered include when each application was last used and how much memory each application is using. The goal is to keep the number of terminated applications as small as possible.

Even in the case where an application must be terminated, developers can minimize the impact this has on the user experience. The Metro style application model allows developers to save incremental state while the application is active. When the user switches back to an application, the saved state can be used to resume at the point the application was suspended. This allows the system to suspend or terminate Metro style applications in a way that is power efficient and largely invisible to the user.

Performing Background Activity

The most significant change for developers in the Metro style application model may be in how applications undertake activity when they are not in the foreground. There are several reasons that Metro style applications would have activity that continues when the application is no longer in the foreground. Typical examples are transferring information or waiting for an incoming communication.

"The most significant change for developers in the Metro style application model may be in how applications undertake activity when they are not in the foreground."

There are several ways that a Metro style application can perform background activity:

- Background transfer API, Playback manager and Share contracts to continue routine tasks
- Windows push notifications
- Background tasks

Most Metro style applications should use the background transfer API and Windows push notifications to undertake any necessary background activity. These methods are extremely power efficient and offload some of the complexity from the developer to the operating system. Some applications will need to use background tasks but should only do so if the other options are insufficient. For a Metro style application to use background tasks it must be on the lock screen. As the number of applications on the lock screen is strictly limited only a small number of applications have the opportunity to undertake this type of background activity on one system.

"This allows the operating system to manage fulfillment of the request in a way that is efficient and compliant with good power management."

Background Transfer API

The background transfer API provides a mechanism for typical background transfer activities to be undertaken regardless of whether a Metro style application is in the foreground. The *BackgroundTransfer* namespace contains several classes that can be used to download or upload files and determine status of ongoing transfers.

When initiating file transfer, using the background transfer API applications can specify parameters that determine such things as whether the transfer should continue or suspend on a costed network such as mobile broadband.

The background transfer API is a fully brokered API, meaning that the application creates a request for file transfer that is given to the operating system. The operating system then performs all the actions required to complete the request. This allows the operating system to manage fulfillment of the request in a way that is efficient and compliant with good power management.

Playback Manager

The playback manager and the *msAudioCategory* assignment are intended for Metro style applications that need to play audio when the application is not in the foreground. In general Windows 8 will manage pausing and playing sound from applications as they are made foreground or suspended. If a Metro style application is intended to play audio when it is suspended, such as an integrated communications application, the playback manager provides APIs for developers to configure this behavior.

Share Contracts

The share contract in Windows 8 provides a mechanism for Metro style applications to share content with other Metro style applications. Frequently the data being shared will be small, such as a URL, and will be shared quickly. For applications that share larger content such as images or collections of files, the share contract provides an API for the receiving application to integrate with Windows 8 to manage the process.

Metro style applications can choose to implement the source sharing contract, the target sharing contract, or both. Users invoke sharing by the share charm in Windows 8.

If the foreground application supports the source share contract, when the user taps the share charm a dialog will open to choose the target for sharing the selected content. For content that is large, developers should use the methods for asynchronous transfer.

When a Metro style application that implements the target sharing contract is notified of a transfer, it should notify the system of transfer progress. These notifications allow the system to manage the source application correctly in the case that the user switches away from it before transfer is complete.

When an application receives a *shareTarget* event it can call the *reportStarted* method to inform the system that it is active and performing the transfer. Once

the application has retrieved all the data it should call the *reportDataReceived* method. This informs the system that it is safe to suspend or terminate the source application.

If the target application submits a task using the background transfer API to complete the sharing it can notify the system of this using the *reportSubmittedBackgroundTask* method. Finally, once the sharing is completed the target task can call the *reportCompleted* method.

Windows Push Notifications

A notification delivery system involves several pieces: logic for when to connect, authentication, local caching, rendering, error handling, back-off algorithms, throttling, and so on. The delivery system also has to deal with service side issues such as knowing when a device is connected, so it can cache undelivered content and handle retries. Typically much of this is left to the applications. Rather than require developers to write large parts of this and expect that they are both knowledgeable and motivated enough to do so in a way that works well with power management, Windows 8 provides the Windows Notification Service (WNS). This moves the management of notification infrastructure to a service that can be tightly managed by the operating system to keep power and other resource usage to a minimum.

There are two high-level design patterns for client server content delivery: polling and push. For the polling pattern the client must communicate with the service to see if there is new content. For the push pattern, when there is new content the service sends the data to the client. If a client using the polling pattern wishes to stay up to date then it must poll frequently. This is at odds with the goal of low power use. Frequent polling by numerous applications leads to reduced idle time, and therefore to increased power use.

Windows push notifications are managed by the Windows notification service (WNS), and are a way for a Metro style application to receive event and update information without polling or even remaining in the scheduling queue. Push notifications are therefore an extremely power-efficient way to provide updates and notifications for Metro style applications.

The Windows Notification Service (WNS) is a service run by Microsoft to deliver fresh content to Metro style applications, allowing them to take actions such as updating Live tiles. Applications do not need to be in the foreground to receive push notifications and users get the benefit of having Metro style applications provide up-to-date status and content without the application running or consuming resources.

"Applications do not need to be in the foreground to receive push notifications."

An example is having a weather application show the current outside temperature. This simple status is sufficient for many scenarios of a weather application. WNS can also be used to deliver onscreen notifications to users when important events happen, such as notification for an unusual breaking news event. Because every Windows-based device is always connected to WNS, these notifications are usually delivered as soon as they are sent.

"Applications provide up-to-date status and content without the application running or consuming resources."

Although WNS helps power the Live tiles and notifications on the Windows 8 start screen, it can also be used to enable real-time connectivity scenarios such as IM, VoIP, and email. If an application that uses WNS is added to the lock screen, that application can use push notifications to activate background tasks. A background task is part of your application code that executes in the background for some period of time. Background tasks are discussed in the next section.

WNS is intended for applications that have a cloud service that provides content, notifications, and updates to the Metro style application. The data flow for a Metro style application using push notifications is shown in Figure 2. Here the "Cloud Service" represents your business logic hosted on the Internet that can receive data from your Metro style application (such as the push notification channel) and send a push notification to WNS.

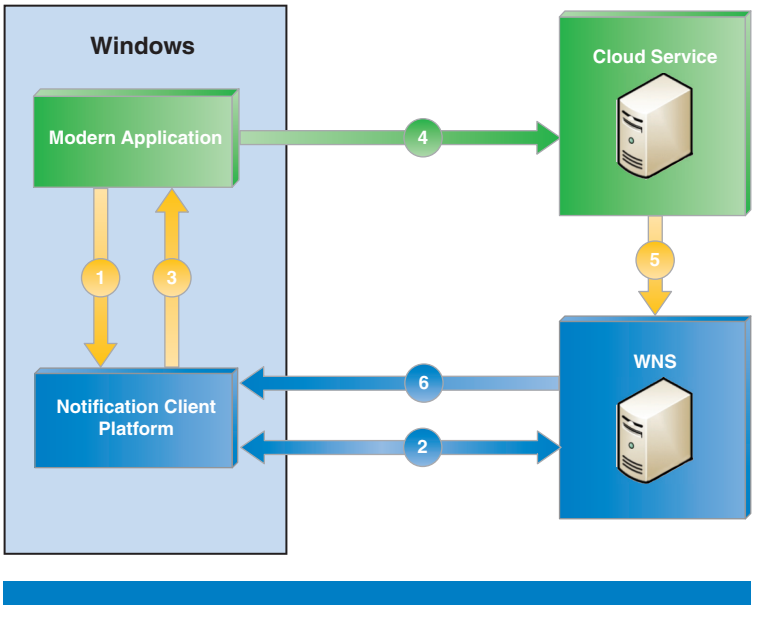


Figure 2: Data flow for the Windows notification service (Source: Microsoft Corporation, 2012)

For a Metro style application using WNS the data flow is:

1. The Metro style application requests a push notification channel from the Notification Client Platform (NCP).
2. The NCP asks WNS to create a notification channel. This channel is returned to the device in the form of a Uniform Resource Identifier (URI).
3. The notification channel URI is returned to the Metro style application.
4. The Metro style application sends the notification channel URI to its cloud service.

5. When the cloud service has an update or notification for the Metro style application, it authenticates with WNS and then posts the notification to WNS using the channel URI.
6. WNS receives the request and routes the notification to the appropriate device.

Once a Metro style application has completed the setup steps 1 through 4 it has no active part in the process. Whether the application is foreground, suspended, or even terminated, the push notification can still be delivered to the device. This allows the number of applications able to receiving push notifications to scale without preventing power management from achieving its goals of long idle times and low power use.

"Allows the number of applications able to receiving push notifications to scale without preventing power management from achieving its goals."

When a push notification is received by the system, the action taken will be one of the following:

1. Update a Live tile.
2. Update a badge icon for a lock screen application.
3. Cause a notification toast to pop for a lock screen application.
4. Cause a background task to execute for a lock screen application.

The only possibility for a Metro style application that is not on the lock screen is to update a Live tile; this is expected to be the main use case. The background task infrastructure and lock screen are described in the next section.

Live Tiles

For a typical Metro style application a push notification will be used to update to a Live tile. A Live tile is an application-defined object based on a template, which is displayed on the Start screen. Figure 3 shows an example of a Live tile. The Live tile can display an image, one or several lines of text, and a badge. The badge can be used for simple status updates by displaying an integer from 1 to 99, or one of a set of provided glyphs.

For example, an email application might choose to have its Live tile display an image of the person who sent the most recent email and the number of unread emails in the badge. A messaging application might choose to display user presence status rather than a number in the badge.

Figure 3: An example Live tile (Source: Microsoft Corporation, 2012)

When using WNS to update a Live tile, the application can be in any state including terminated. The tile can be updated regardless of application state as the application is not involved in the update process. Users can see up-to-date status for an application without that application having to be loaded into memory or needing to execute any code. The operating system manages all parts of the process, including caching notifications and discarding out-of-date notifications that arrived while the system was in connected standby. This allows the operating system to maintain a connected standby state and extend battery life by making context-aware decisions on how and when to process notifications.

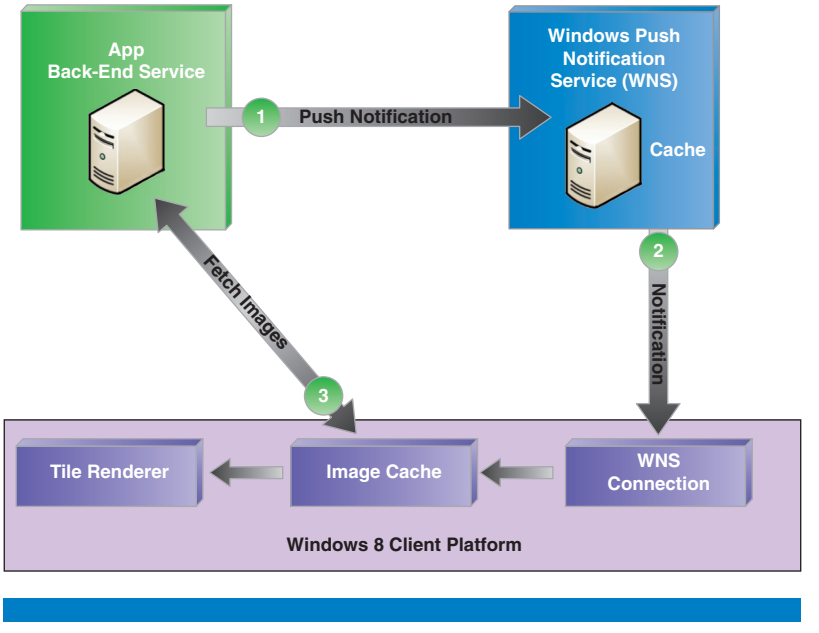


Figure 4: Push notifications platform
(Source: Microsoft Corporation, 2012)

The process for updating a Live tile when a push notification is received is shown in Figure 4. Step 1 in this figure is the same as step 5 from Figure 3, and the process proceeds as follows:

1. When the applications back-end service has an update or notification for the Metro style application, it authenticates with WNS and then posts the notification to WNS.
2. WNS receives a request and routes the notification to the appropriate device.
3. When it is time to show the Live tile update on the Start screen, the operating system fetches the image from the application service.
4. When the image is downloaded, the tile renderer presents the image on the start screen.

As the design the client platform components is focused on engineering for high performance and low power consumption, one of the key features is separating the notification payload from the image payload. A typical notification XML is less than 1 KB of data, but an image can be up to 150 KB. Separating these creates significant savings in network bandwidth for scenarios where there is duplication of the images. For example in the email scenario above, the image for a tile may be a profile picture of a friend that your PC can download once and cache locally to be reused. Separating the notification from the image also allowed the operating system to be smart about discarding unused notifications before it goes through the expense of downloading the image. If a device screen is off and is sitting at home while the user is elsewhere, there is no point in downloading images for tiles that will just be replaced by subsequent updates before the device is next used.

"Separating the notification from the image also allows the operating system to be smart about discarding unused notifications."

By using the Windows notification service to update Live tiles developers can provide sufficient functionality for many applications that are enhanced by up-to-date status or content, without many of the power-draining side effects of current methods. This also gives the operating system much greater opportunity to use its contextual knowledge of system state to improve the efficiency of updating application information.

"…gives the operating system much greater opportunity to use its contextual knowledge of system state…"

Background Tasks

If a Metro style application requires functionality that cannot be achieved using the features mentioned so far, it can choose to implement background tasks[1]. Before implementing background tasks the developer should consider the design implications. Background tasks will only fire for applications that are on the lock screen. If a Metro style application that is not a lock screen application attempts to use background tasks, the task will not be triggered.

Lock screen tasks are chosen by the user as the set of tasks that are permitted to display on the lock screen. They have a special status in the system because of this. An application may request the users' permission to be on the lock screen, but this may be refused as there are a limited number of lock screen slots available. Users may also remove an application from the lock screen if they need space for another application, or if an application on the lock screen behaves badly. An example of bad behavior would be excessive use of background tasks consuming power and reducing battery life.

Applications cannot use background tasks when they are not on the lock screen, so developers should only use this infrastructure if there is no other way to achieve acceptable functionality. Developers will also need to be aware that if their application is not on the lock screen the background tasks will not fire, and user experience may be degraded.

Scenarios for Background Tasks

Allowing applications to run code in the background when they are suspended is a powerful feature and is designed primarily for the real-time class of applications such as mail, VOIP, and chat applications. The background task execution environment is a restricted resource-managed environment and background tasks only receive a limited amount of CPU time. Background tasks should be used for small work items that have no interaction with the user and only provide limited service. Long running or expensive workloads in the background will deplete the user's battery and are not an appropriate use for background tasks.

"The background task execution environment is a restricted resource-managed environment and background tasks only receive a limited amount of CPU time."

Scenarios that are appropriate for background tasks include showing a toast notification for an incoming VOIP call or a chat message, or reacting to a change in system condition (for example, UserAway) and updating the server with this information. Scenarios that are not appropriate for background tasks are indexing mail, or anything that requires user interaction through displaying UI or audio.

Background Task Architecture

A Metro style application registers its background tasks with the background task infrastructure using the *BackgroundTaskBuilder* class. The background task is

implemented as a class that implements the *IBackgroundTask* interface. The class name is specified in the *TaskEntryPoint* property of the *BackgroundTaskBuilder* class. The background task class is hosted in an in-proc server DLL, which can be loaded in an application executable, as part of the application package, or in the system-provided *backgroundTaskHost.exe*. The background task can also be implemented as a JavaScript worker. The JavaScript worker is included in the application as a JavaScript file and the name of the JavaScript file is specified in the *TaskEntryPoint* property of the *BackgroundTaskBuilder* class. The JavaScript worker runs in the system-provided WWA host process similar to a web worker.

A background task must have exactly one trigger that describes the trigger event to launch the background task. The trigger is specified with the *SetTrigger* method of the *BackgroundTaskBuilder* class.

"Background conditions are used by an application to describe the set of conditions required to perform meaningful work."

A background task can optionally have one or more conditions that must be true for the task to launch. Background conditions are used by an application to describe the set of conditions required to perform meaningful work. For example if the background task requires the Internet, an *InternetAvailable* condition can be used to tell the system to wait for the Internet connection to be restored before launching the background task. Background conditions are specified with the *AddCondition* method of the *BackgroundTaskBuilder* class.

Background Task Infrastructure

When the trigger specified for a background task is fired, the background task infrastructure launches the class associated with the trigger regardless of the state of the application. The activation of the background task does not involve any UI and it does not bring the Metro style application to the foreground. There are triggers provided in the system for timer, system event, and real-time communication.

Figure 5 shows the process of registering and launching a background task. When an application uses the *BackgroundTaskBuilder* class to register a background task, the *BackgroundTaskBuilder* code gives the trigger details to the trigger implementations. It then registers the application class using the trigger with the system infrastructure. When the trigger is fired, the system infrastructure activates the class within the application container.

Background Tasks and Application Life Cycle

A background task can execute in either the system-provided *backgroundTaskHost.exe* or the application process to which it belongs. The application specifies the execution host in its manifest. When the background task is launched, an application can be in any of the states: running, suspended, or terminated.

If a background task executes in the system-provided *backgroundTaskHost.exe*, it is launched independently of the state of the application so it is not necessary for the system to change the application's state. This option is preferable because launching the task within *backgroundTaskHost.exe* is faster and provides better performance than launching the background task within the application. *BackgroundTaskHost.exe* is launched within the same application container as the application and it terminates after the background tasks finish.

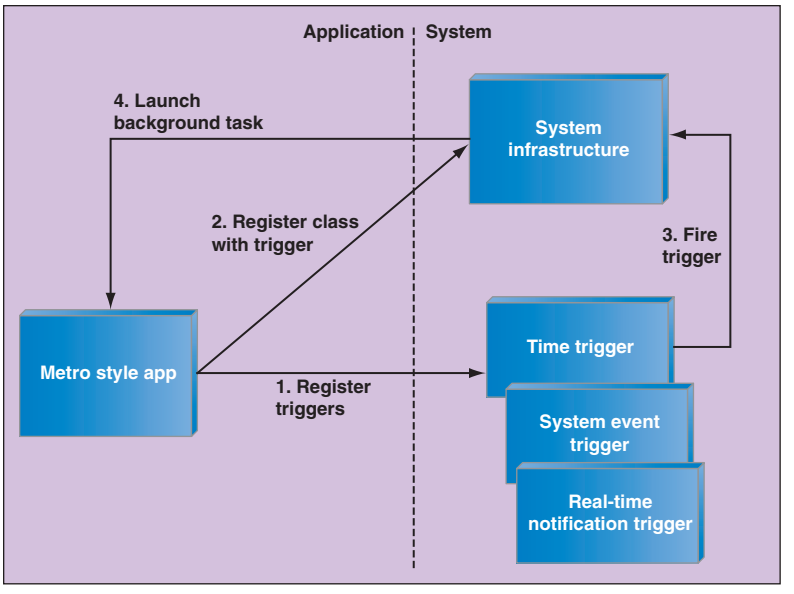


Figure 5: Background task registration and process flow (Source: Microsoft Corporation, 2012)

If a background task executes within the application process, the background task infrastructure might need to change the state of the application:

- Running: If the application is running, it is already in the foreground and the background task is simply launched within the application.
- Suspended: If the application is suspended, most of its threads are unfrozen and the background task is launched.
- Terminated: If the application is terminated, it is not running at the time of the trigger so the application is activated and the background task is run. The application activation does not involve UI and it does not bring the application to the foreground.

Background Tasks Implementation Concepts

Each background task has one trigger and zero or more conditions associated with it. All background task registrations are persistent; that is, the application doesn't have to be running for the background task to be triggered, and background tasks retain their association across the application lifecycle.

"Background task registrations are persistent; that is, the application doesn't have to be running for the background task to be triggered."

If a trigger fires for a background task but one or more of the conditions for executing the task are not satisfied, the trigger is latched. This means the background task infrastructure will monitor changes to conditions and execute the background task when they are all satisfied. The trigger need not fire again for the task to be run; it is considered latched in the fired state for that background task.

If an application wishes to display progress of a background task while the application is in the foreground, it can register a progress handler.

The *IBackgroundTaskInstance* that is passed to the *Run* method is used to communicate progress to the foreground application. The interface has a *Progress* property, which can be updated by the background task. The application can register a progress handler to receive progress updates from the background task when the application is in the foreground. The progress handler in the application does not receive the updates from the background task if the application is suspended or terminated.

The foreground application can also register a completion handler to be notified when the background task completes. The completion status or any exception thrown in the *Run* method of the background task is passed to the completion handler in the foreground application as part of the *BackgroundTaskCompletedEventArgs* input parameter. If the application is in the suspended or terminated state, it does not receive the completion notification.

When a Metro style application is launched from the terminated state, it can enumerate all background tasks it has registered with the system using the *BackgroundTaskRegistration.AllTasks* method. Because the application has just been launched, it must re-associate its progress and completion handlers with the background tasks. Even if the application has previously registered those associations, they are no longer valid because the application was terminated and removed from memory. Only the progress and completion handlers must be re-associated. The background task registrations themselves are persistent.

Background Task Resource Management

Background tasks are meant to be short-lived tasks that do not consume a lot of resources. This is a key contribution of background tasks to improved power management. If an application is running and the user is interacting with the application in the foreground, then no CPU quota or resource management policies are applied to the application's background tasks. If an application is suspended or terminated, its resources are constrained by the operating system to ensure good power management practices.

"If an application is suspended or terminated, its resources are constrained by the operating system to ensure good power management practices."

An application that is not on the lock screen does not receive any CPU time when it is not in the foreground. An application on the lock screen receives a limited amount of CPU time at regular intervals for its background tasks. If the lock screen application uses all of its available CPU time, its background tasks are suspended until the application's CPU quota is replenished at the next generation for CPU quota updates.

"In Windows 8, each application on the lock screen receives two seconds of CPU time every 15 minutes."

In Windows 8, each application on the lock screen receives two seconds of CPU time every 15 minutes, which can be utilized by all of the background tasks of the application. At the end of 15 minutes, each application on the lock screen receives another two seconds of CPU time for use by its background tasks. Any unused CPU time in the 15-minute interval is lost and not accumulated by the application. If a background task is suspended because it has used its two seconds of CPU time, a message is logged in the event viewer.

Threading Model for Background Tasks

If background tasks authored in C# or C++ are hosted within the application instead of the recommended *backgroundTaskHost.exe*, there are some threading model complexities to keep in mind.

For non-JavaScript applications, the background tasks are hosted in an in-process DLL, which is loaded in a multithreaded apartment (MTA) within the application. For JavaScript applications, background tasks are launched in a new single-threaded apartment (STA) within the WWA host process. The actual background task class can be STA or MTA. Because background tasks can run when an application is in a suspended or terminated state, they need to be decoupled from the foreground application. Loading the background task DLL in a separate apartment enforces separation of the background task from the application while allowing the background task to be controlled independently of the application.

"Background tasks can run when an application is in a suspended or terminated state, they need to be decoupled from the foreground application."

When an application is suspended, the UI STA thread is blocked in the Windows kernel. This thread is released only when the application transitions back to a running state in response to user interaction. When the application is suspended and a background task is triggered, all threads in the application except the UI STA thread are unfrozen and the background task is activated in an MTA thread. The UI STA thread remains blocked. If the background task tries to access an object that is owned by the blocked UI STA thread, then it will deadlock. To prevent this deadlock, the background task should not share data between the application and the background task.

"If the background task tries to access an object that is owned by the blocked UI STA thread, then it will deadlock."

As general best practices, if a background task is triggered for a Metro style application, the processing done must be as brief as possible. Less frequent background tasks with shorter processing duration will give better performance and power management. Keeping any background processing brief is a vital contribution developers can make. Using *backGroundTaskHost.exe* as the executable for background tasks means that the application does not need to be loaded into memory to execute the task. This is particularly important if the application was terminated due to resource constraints, as loading the application may cause the termination of other applications.

Desktop Applications

The Metro style application model supports most types of application in use today. However, some users will need to run applications that do not fit well into the Metro style application model. As an example a DVD-burning application is unlikely to be a Metro style application, and similarly Visual Studio or other development tools will be desktop applications. Once you start burning a DVD, you don't really want the process interrupted.

The advances in idle hygiene and runtime device power management will benefit all applications that run on Windows 8. Keeping idle periods as long as possible and allowing offload of communications to devices will reduce power use. In addition, the Windows sleep and hibernate models continue to work as

they do in Windows 7. When a system has active desktop applications they run as they always have, and Metro style applications are managed by the operating system as described in this article. When a system enters sleep both desktop and Metro style applications are completely suspended, just as all applications are suspended by sleep in Windows 7.

While system sleep provides great power savings it means that no updates are reaching the system. This is a disadvantage for Metro style applications that use Live tiles and the other background infrastructure to provide the always-on experience.

Desktop Activity Moderation

On devices that are capable of connected standby there is a new component in Windows 8 called the Desktop Activity Moderator. This component is designed to help reduce the resources utilization of desktop applications when a capable device enters the connected standby state. A typical desktop application expects to be in one of two states: full access to system resources, or no access because the system is asleep. The Desktop Activity Moderator helps prevent power drain by desktop applications by suspending all desktop applications when the device enters connected standby. To the desktop application this appears as though the system entered sleep, and all activity was suspended when the system is in connected standby. The operating system is then able to manage the Metro style applications and background processing as described, without the power use of desktop applications reducing battery life.

"The Desktop Activity Moderator helps prevent power drain by desktop applications by suspending all desktop applications when the device enters connected standby."

More details on the behavior of the Desktop Activity Moderator and desktop applications or services in connected standby are available in the Windows 8 Compatibility Cookbook[3].

Conclusion

The mechanisms discussed in this article facilitate a model of application execution that allows for improved operating system power management. By removing Metro style applications from the scheduler queue when they are not in the foreground much of the background activity observed on current systems can be avoided.

"The mechanisms discussed in this article facilitate a model of application execution that allows for improved operating system power management."

The Windows notification service provides an extremely power-efficient way for applications to receive status and content updates. When used in conjunction with Live tiles the operating system has control of the entire process and can make decisions based on its knowledge of the current system context to significantly enhance power management.

Applications that are not on the lock screen can perform background activity only through APIs that offload the work to the operating system. In this way the work can be managed by the operating system to use minimal power.

Lock screen applications have more freedom for performing background activity as they can use the background tasks infrastructure. While this gives

applications the ability to run tasks at any time, the operating system closely manages all background activity and strictly limits resource use for each application.

In this way the operating system provides developers with the tools required to write applications that enable great power management, without complexity. The operating system also has the ability to ensure that battery life is not compromised even by applications that are resource hungry. This is accomplished without undermining the perception of responsiveness and multitasking for the user.

Desktop applications will still be required on some systems for some tasks. While desktop applications do not fully participate in the new power management model, the numerous improvements to idle hygiene and activity moderation help improve the common desktop scenarios. For connected standby capable devices the desktop activity moderator limits the impact of desktop applications on battery life and power use.

Complete References

[1] Microsoft Corporation, "Introduction to background tasks: Guidelines for developers," Microsoft. http://www.microsoft.com/en-us/download/details.aspx?id=27411

[2] Jonathan G. Koomey, "Growth in Data Center Electricity Use 2005 to 2010," Analytics Press. http://www.analyticspress.com/datacenters.html

[3] Microsoft Corporation, "Windows 8 Consumer Preview and Windows Server 8 Beta Compatibility Cookbook," Microsoft. http://go.microsoft.com/fwlink/p/?LinkId=242534

[4] Microsoft Corporation, "Power Availability Requests," Microsoft. http://msdn.microsoft.com/en-us/windows/hardware/gg463205

Author Biography

Kim R. Shearer is part of the Environmental Sustainability (ES) team in the Technology Policy Group at Microsoft Corporation. The ES team manages many of the companywide sustainability efforts and creates policy guidance for Microsoft. Kim's focus is gathering and publishing best practices and guidance for developers both internal and external to Microsoft, and working with Microsoft product teams to continually improve energy and resource efficiency. Before joining Microsoft he pursued research into media indexing at Curtin University in Australia and IDIAP in Switzerland. Since joining Microsoft he has been part of several product teams, managed the sustainability portfolio for external research in Microsoft Research, and run software incident response for the Microsoft Security Response Center. He is currently active in investigating the part software plays in energy efficiency for Green IT, and IT for green.

ENERGY-AWARE COMPUTING FOR ANDROID* PLATFORMS

Contributors

Hung-Ching Chang
Department of Computer Science,
Virginia Tech

Abhishek R Agrawal
Software and Services Group,
Intel Corporation

Kirk W. Cameron
Department of Computer Science,
Virginia Tech

"Android smartphones and tablets are changing people's daily lives."*

Android* smartphones and tablets are changing people's daily lives. People can now perform tasks on the go that were previously impossible without a laptop or desktop system. The increasingly demanding applications combined with the mobile nature of such systems place heavy emphasis on battery life. Unfortunately, many end users are not satisfied with the battery life of their Android devices. One key challenge developer's face is to understand how their software impacts energy usage. Ideally, a resource within a device should only consume power when needed and otherwise remain inactive. In this survey, we study the aggressive power management techniques underpinning the Android operating system. To aid developer's understanding of application power consumption, we introduce system-wide, function-level power profiling tools and a methodology for energy-efficiency debugging. We demonstrate the application of these analysis tools on the Android platform for common usage scenarios on a shipping Motorola Atrix* smartphone and a production Motorola XOOM* tablet.

Introduction

Android mobile devices provide a balance of good hardware performance with a large user application market. According to Nielsen, a global marketing research company, Android is currently the most popular smartphone among United States subscribers reaching 31.2 percent of market share as of January, 2011[1][2]. Unfortunately, as the demands on smartphone usage increase, so do complaints about battery life. Many users would be satisfied if their Android smartphones simply lasted a full day on a single charge. Unfortunately, many users must carry their chargers with them and recharge several times a day under normal usage.

Popular online suggestions to extend battery lifetime include manually managing hardware components such as GPS, 3G, Wi-Fi*, and Bluetooth* and turning them off when they are not in use. However, this approach is wrought with inefficiency and frustration as users struggle to remember when to have different components on/off or they forget to enable a device they need.

"The Android operating system has some active power management techniques built into the software that conserve power."

The Android operating system has some active power management techniques built into the software that conserve power. However, users still complain heavily about battery life. In truth, little is known about the efficiency of the built-in techniques. The primary issue with power management in the PC and server domain has been the mismatch between the systems prediction of what devices are needed and when versus the actual usage of the system by the end user. This mismatch leads to power-performance inefficiencies that either

reduce a user's experience to the point that they disable power measurement or notice little benefit from enabling power management.

First and foremost, we need to better understand the potential effects of power management on Android devices. To this end, we suggest an optimization approach based upon direct measurement at the component level within an Android device. With the insight gained from this approach, we can evaluate and propose power efficiency methodologies and compare their effectiveness. With the resulting tool chain, we can assist designers in their goal of designing energy efficient applications.

In the remainder of this article, we discuss a methodology for analyzing the power-performance characteristics of Android devices. We use the methodology to compare and contrast existing approaches to energy efficient Android usage. We propose energy efficiency debugging tools to analyze software behavior in different usage scenarios. Lastly, we use our methodology to propose best practices for energy efficiency in Android devices.

Overview of Android Power Management

"Android is an open-source software stack, including operating system, middleware, and applications, for mobile devices."

Android is an open-source software stack, including operating system, middleware, and applications, for mobile devices[3]. Android software was initially developed by Android Inc., which was purchased by Google Inc., in 2005. Google published the entire Android source code in 2008 under the Apache License. The Android Open Source Project is currently the official site that maintains the Android source code and development cycle[4].

The Android mobile operating system is developed based on a modified Linux 2.6 kernel. The difference between the modified and standard Linux kernels is some of the libraries and drivers are either modified or completely rewritten to maximize the efficiency of running on mobile systems with limited resources. The Android community ultimately decided to implement its own C library (Bionic C) and Java virtual machine (Dalvik Virtual Machine, or DVM). However, since the Bionic library does not support the full GNU library and Android now is outside the typical Linux development cycle, it is challenging to port existing GNU/Linux applications and libraries to Android[4][5][6].

Standard Linux Power Management

"Power management (PM) is crucial in modern computer systems."

Power management (PM) is crucial in modern computer systems. PM helps to conserve power consumption by shutting down components as the system is at idle for some time. Linux-based systems have their own power management as a component in the kernel. The Linux PM manages power consumption via either Advanced Power Management (APM) or Advanced Configuration and Power Interface (ACPI)[7][8]. In APM mode, the BIOS controls the system's power management without the operating system's involvement. ACPI, the successor to APM, allows an operating system to directly control a system component's power states. In ACPI mode, the operating system plays a key role and is responsible for managing power consumption of the system[8].

"Android supports its own power management that focuses on reducing CPU power."

Android Power Management Architecture

Android supports its own power management that focuses on reducing CPU power when the system is idle. Android Power Manager (see Figure 1) is an aggressive, energy-saving policy tool that ensures applications and services request CPU resources with "wake locks" through the Android application framework. Native Linux libraries are used to carry out the policy settings. When awake, the CPU is powered on. When no wave locks are detected, the operating system places the hardware in deep sleep to save energy[9].

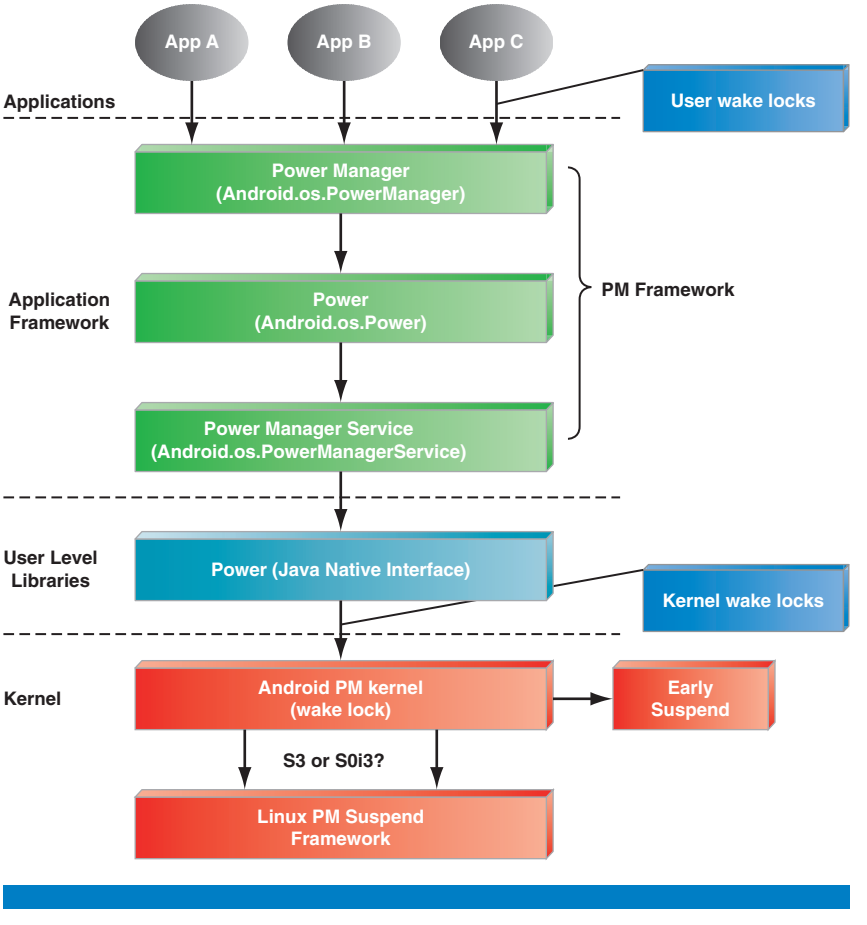


Figure 1: Android power management architecture and wake locks. (Source: Intel Corporation, 2012)

Figure 2 shows various types of wake locks. CPU, screen, and keyboard hardware components are powered on or off based upon observed application behavior. All power management calls to generate wake locks use the following five steps[9]:

1. Request a handle from the Power Manager service.
2. Create a wake lock and specify the power management flags for screen, timeout, and so on.
3. Acquire wake lock.

4. Perform operation (play MP3, open HTML page, and so on).
5. Release wake lock.

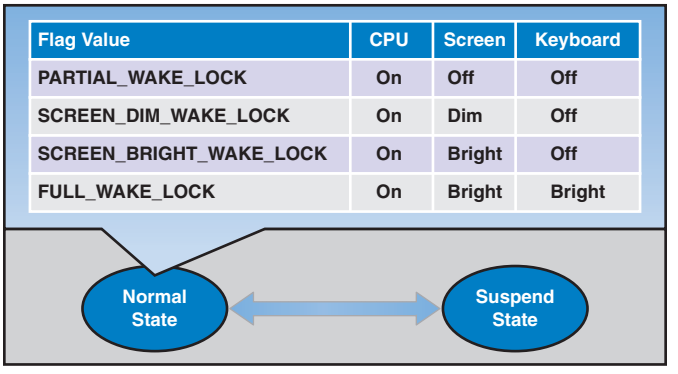


Figure 2: Wake lock settings[10]
(Source: Intel Corporation, 2012)

Android Power States

The standard Linux kernel has Suspend and Resume states. When entering the Suspend state, the system first freezes all processes, puts all the drivers into suspend mode, and suspends the CPU and hardware components. In the Resume state, the system is awoken by an interrupt and will then enable IRQ, resume the CPU, wake up the hardware components, resume past processes, and notify the Linux Power Management tool that the system has exited the suspend stage[11].

"An Android patched Linux kernel adds two additional power modes: early suspend mode and late resume mode."

An Android patched Linux kernel adds two additional power modes: early suspend mode and late resume mode. Figure 3 shows how these modes improve the efficiency of power saving design by allowing transitions between

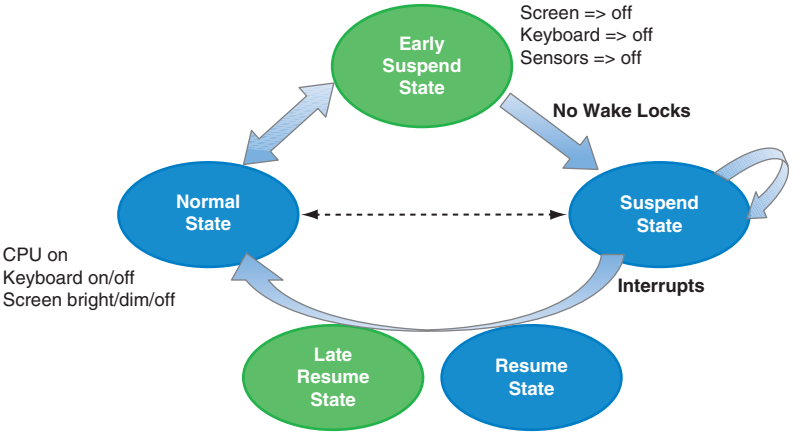


Figure 3: Android power states
(Source: Intel Corporation, 2012)

normal mode and suspend mode. Early suspend is the state between suspend state and normal state where the screen, keyboard, and sensors are affected. Late resume is executed after the system resume completes. In this state, the system enables and resumes the hardware components suspended during early suspend[3][11].

Related Work

Researchers have investigated the behavior and experiences of smartphone users[12][13][14]. The conclusion has been made that battery lifetime is the most important thing that end users care about[15][16][17]. K. Truong and et al.[18] implemented a task-centered battery interface for mobile users to help them have more insight as to how different applications affect the battery life of their smartphone. A. Shye et al.[19] observed user activities on the Android G1 smartphone for six months and concluded that battery life was the most significant factor to affect user experiences. D. Ferreira and et al.[20] presented a four-week study over more than 4,000 people on their charging behaviors, and showed that increased battery life will improve users' smartphone experience.

To increase the energy-efficiency of Android devices and applications, complete system power profiling or system power modeling can be used to improve our understanding of component/application power draw from the power outlet or battery[21][22][23]. L. Zhang et al. proposed PowerTutor[24][25], which is a power monitoring application for Android users to track whether their applications on mobile devices are running properly. Android developers can take advantage of the real-time power estimation feature from PowerTutor to make their code more energy-efficient. A. Pathak et al.[26] proposed a system-call-based power modeling approach for online power estimation for Android mobile and Windows® Phone. They further implemented eprof, which added their estimation model into gprof, for optimizing an application's energy efficiency. M. Dong et al. proposed Sesame[27], a self-modeling power estimation technique base on a regression model. Sesame does not need external assistance and is able to achieve reasonable accuracy on Nokia* N900 smartphones.

"A power optimization approach for software development on the Android platform to utilize the aggressive power saving features."

"… we identify the impact of various types of workloads for two Android mobile devices."

Most of this research focuses on identifying how battery lifetime impacts the end user experience or power modeling for online energy usage approximation for the device being used. In this article, we first propose a power optimization approach for software development on the Android platform to utilize the aggressive power saving features provided in the Android OS. Developing power-optimized software helps transfer the responsibility of energy-saving from end users to software developers, avoiding power-inefficient configuration inputs from end users. The CPU resource analysis tools we suggest can be used for online monitoring and to directly access software power events. In addition, our measurements not only confirm that the CPU is the greatest power hog and significantly reduces battery lifetime but, more importantly, we identify the impact of various types of workloads for two Android mobile devices.

Software Power Optimization on Android

Software plays an important role in platform energy efficiency. The Android mobile operating system has energy-oriented power management, and mobile platform hardware components are being aggressively designed to reduce power consumption. Software must not preclude residency in low power states. An energy-efficient application should have minimum or no impact on platform power consumption at idle and request as few CPU resources as possible to complete its task.

"Software plays an important role in platform energy efficiency."

In Figure 4, we suggest a step-by-step approach to optimize software energy-efficiency on Android platforms for a wide range of usage scenarios. This software-power-debugging approach does not involve end users in manually turning devices off and on for the battery lifetime. In addition, the usage scenarios selected in the process widely cover an end user's daily activities from streaming media playback, browsing, messaging, to gaming.

"… we suggest a step-by-step approach to optimize software energy-efficiency on Android platforms for a wide range of usage scenarios."

Using this energy optimization process, Android software on mobile hardware platforms is first investigated by analysis tools based on preselected

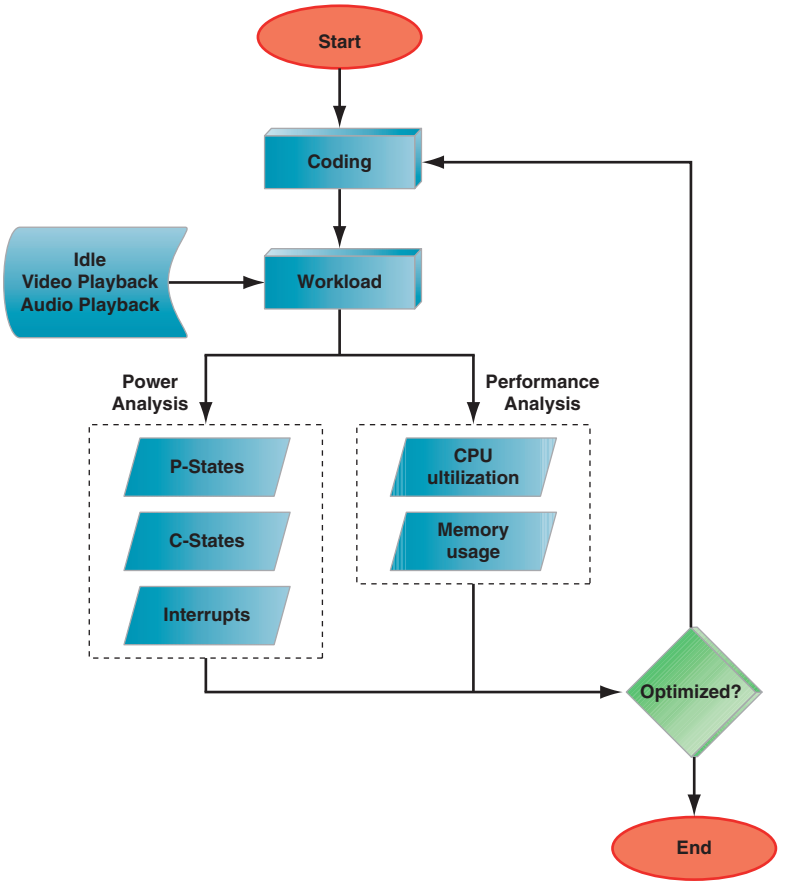


Figure 4: CPU power optimization (Source: Intel Corporation, 2012)

usage scenarios, and the tools generate reports with suggestions for possible additional energy savings and potential issues. Software developers can then take the suggestions and look to improve their code. After the software is tuned for Android, smartphone devices are scheduled to stay in low power states to conserve energy and only wake up as needed. This methodology is designed for application developers to debug power issues and improve software energy-efficiency in a standard and convenient way.

Software Power Analysis Tools

"The power analysis tools that can be used for energy-efficiency debugging to the applications on the Android platform."

This section briefly introduces the power analysis tools that can be used for energy-efficiency debugging to the applications on the Android platform.

CPUFreq

The CPUFreq driver is provided by Linux kernel for users to control processor performance states (P-states) on the fly at runtime. Most of the processor power management on a Linux system focuses on changing the processor frequency and/or voltage to manage processor power consumption depending on the workloads. By choosing the proper frequency according to system load, this mechanism saves power. The number of P-states is processor dependent. The higher the number of P-states, the slower the processor frequency runs with lower power consumption.

CPUIdle

CPUIdle is another technology supported by the Linux kernel to enable processor operating states (C-states) support. A processor supports multiple operating states and has the capability to turn off unused components to save power. Higher C-states result in more components in CPU switched off to reduce power consumption with higher penalties (such as longer wakeup time) to bring the processor back to a working state (C0).

System Interrupts

The proc/interrupts file in the Android kernel is used to record the number of interrupts per CPU, which can be used as a reference to track back the sources that wake CPUs up. The file includes both external and internal interrupts to the system. External interrupts come from I/O devices. Internal interrupts are requested mainly from NMI (nonmaskable interrupt), LOC (local timer interrupt), TLB (TLB flush interrupt), RES (rescheduling interrupt), and CAL (remote function call interrupt).

Procrank

The procrank program[28] provides more detailed memory usage per process with Vss (Virtual set size), Rss (Resident set size), Pss (Proportional set size), and Uss (Unique set size). In an Android system, Pss and Uss memory utilizations are more meaningful than Vss and Rss. The procrank program also shows the PID of the processes and the command line of how each process is launched.

Uss indicates the set of pages allocated uniquely to a process. It also means the amount of memory will be freed after this process is halted. Pss refers to the amount of memory shared with other processes assuming memory is divided evenly among the processes. Memory owned by multiple processes will not be freed unless all processes that share this memory are terminated. Pss indicates the amount of memory this process is contributing.

Dalvik Debug Monitor Server (DDMS)

Dalvik Debug Monitor Server (DDMS)[29][30] is a debugging tool that comes with the Android development SDK. It is integrated into the Eclipse developing tool, and all the DDMS features can be accessed within the Eclipse IDE. DDMS offers features such as Threads, Heaps, and Allocation Tracker to track detailed memory usages within a process. These features are valuable for tracking memory usage that may impact application performance.

To access memory tracking features, users can just use the tab in DDMS Perspective from within the Eclipse IDE. Threads featured in DDMS show the active threads within a VM. The Heaps feature shows how much heap memory a process is using at runtime. The heap usage information can be retrieved at any time during the execution of the application. Also, through the Allocation Tracker tab, DDMS lets users track the memory allocation of objects. Therefore, users are able to track objects being allocated to memory and to see which classes and threads are creating objects. The Allocation Tracer feature allows users to see how objects are being created in real time when certain actions are performed on the application.

Case Study

This section demonstrates the application of these analysis tools on the Android platform for common usage scenarios on a shipping Motorola Atrix smartphone and a production Motorola XOOM tablet.

"… demonstrates the application of these analysis tools on the Android platform for common usage scenarios."

NVIDIA Tegra 2* SoC

The NVIDIA Tegra 2 SoC is integrated with a dual-core ARM Coretex-A9* CPU and an NVIDIA GPU. ARM's dual-core Coretex-A9 is capable of operating in synchronize/asynchronize mode, which turns off the other CPU core to save power and brings it back up when necessary.

ARM Coretex-A9 MPCore offers three operation modes: busy mode, CPU flow-control mode (LP3), and power-gated mode (LP3). While flow-control mode turns off most of the components except the memory controller, the power-gated mode shuts down an idle CPU core to save power.

Usage Scenarios

The usage scenarios studied span idle, local video playback without hardware acceleration, photo editing, streaming video, streaming audio, social networking, eBook reading, and Web browsing. The test duration is

"We collect CPU idle-states residency, P-states residency, interrupts/sec, and wakeups/sec for each usage scenario under test."

120 seconds for the workloads. We collect CPU idle-states residency, P-states residency, interrupts/sec, and wakeups/sec for each usage scenario under test.

Idle
This test represents the baseline for the Android device not running any user triggered workload. The activity on the system is solely from the Android operating system, and, therefore, we use this baseline to understand the energy efficiency of an Android system without our proposed techniques.

Local Video Software/Hardware Playback
In this benchmark, we analyze the energy efficiency of Android devices playing a video file and showcase the tradeoffs between hardware acceleration and the software approach by offloading video stream to hardware decoder. The MX Video Player, which offers the option to turn hardware acceleration on or off, is used during the benchmark for software decoding playback and for hardware decoding playback.

Photo Editing
In this test, we collect software power events trace CPU workload as the user edits a photo on his/her smartphone. The photo under use is captured from the camera on the Android devices and then Adobe Photoshop Express* is used to edit the photo by manually applying the effects to the picture at the rate of five seconds per effect. The effects include saturation, adjusting the saturation, tint, and contrast.

Reading an eBook
This test is designed to capture when CPU cores remain in idle states as a user generates periodic events on the Android devices using the Internet over Wi-Fi. The Kindle* app from the Android Marketplace is downloaded and installed on the Android devices. The workload is generated by swiping the screen to advance through the pages at the rate of one page per second.

Streaming Video
In this benchmark, we profile Android devices for playing a video clip over Wi-Fi. We use the YouTube* app that comes with the device to playback the video clip from YouTube Web site. The clip is identical to the one used in the first local video software/hardware playback benchmark so we can see the impact of the same video clip being played locally versus streaming over Wi-Fi. Notably, the YouTube app utilizes the hardware decoder for streaming Video playback, which will give an indication of decoder efficiency.

Streaming Audio
This benchmark is used to study the time CPUs spend at idle during audio streaming by measuring idle-state and P-state residency as well as the interrupts being measured. The Pandora* app is used and during playback the screen backlight is turned off.

Social Networking
This benchmark is used to study CPUs woken up for social networking by measuring idle-state and P-state residency and interrupts. The Facebook* app

for Android is selected for the social networking workload. The workload is generated by scrolling up and down to see the update messages on the News Feed board in the Facebook app. The workload is designed to capture unpredictable demand from the end user with the overhead of a Wi-Fi connection.

Web Browsing

The last benchmark tests browser performance on Android devices particularly for HTML5 and JavaScript. Two benchmarks, PeaceKeeper and SunSpider, are used to conduct these tests. PeaceKeeper measures the browser's speed for HTML5 while SunSpider tests the core JavaScript language only. The workloads are launched directly online through a browser. After the tests are conducted scores are generated and captured. These scores as well as the software power events from the devices under test are collected and compared.

Motorola Atrix Smartphone

The device under test is the Motorola Atrix 4G smartphone running Android Froyo 2.2. The Atrix smartphone features NVIDIA dual-core 1-GHz Tegra 2 processor with a four-inch LCD screen and 1 GB RAM. (The device is rooted.) Table 1 provides system details.

"The device under test is the Motorola Atrix 4G smartphone running Android Froyo 2.2."

Component	Specification
SoC	Tegra 2 AP20H
CPU	1 GHz dual-core ARM Cortex-A9
GPU	ULP GeForce GPU 300 MHz
RAM	1 GB LPDDR2 RAM
FLASH	16 GB NAND
Cellular Radio	Qualcomm MSM6200
Wi-Fi/Bluetooth/FM	Broadcom BCM4329
GPS	uPC8236T6N
Display	4.0-inch TFT LCD 960x540
Battery	1930 mAh Li-Po Battery

Table 1: Motorola Atrix Hardware Specification[31]
(Source: Motorola, Inc., 2012)

Test Methodology and Results

For each usage scenario, we collect the idle-state residency, P-state residency, and interrupts per CPU for software power events. The experiments are done with the Atrix smartphone in airplane mode, turning off Bluetooth* and GPS. The Wi-Fi option is turned on for streaming audio/video, social networking, reading an eBook, and Web browsing usage scenarios.

The following sections discuss the software power results in detail from each usage scenario on the Motorola Atrix smartphone under test, as shown on Figures 5 through 8. The usage scenarios cover idle, video playback, photo editing, reading an eBook, streaming video/audio, social networking, and Web browsing.

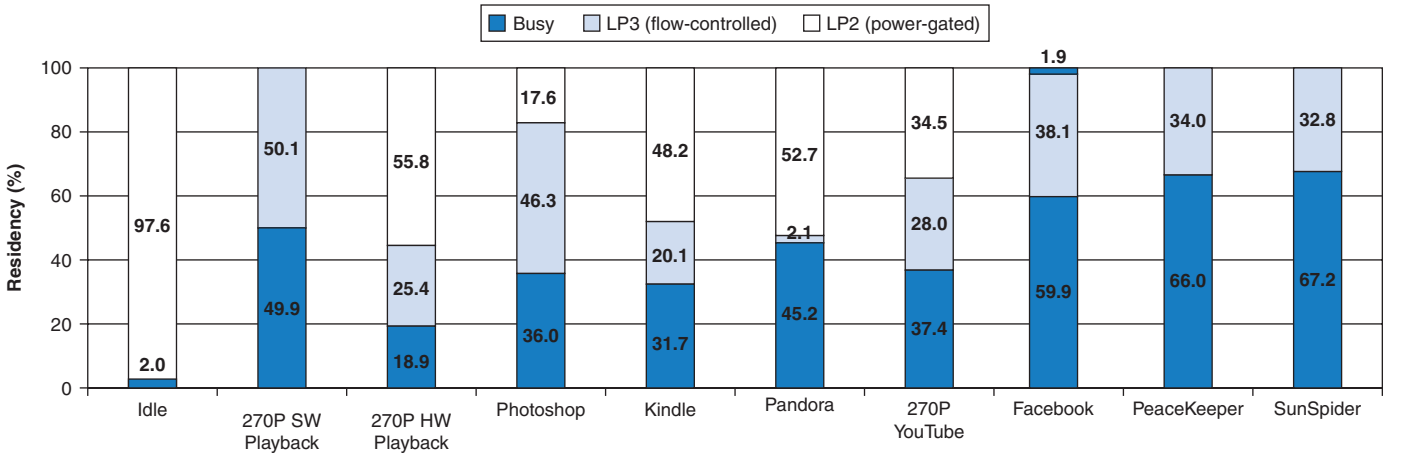


Figure 5: Idle-states residency on CPU0 from Motorola Atrix Smartphone (Source: Intel Corporation, 2012)

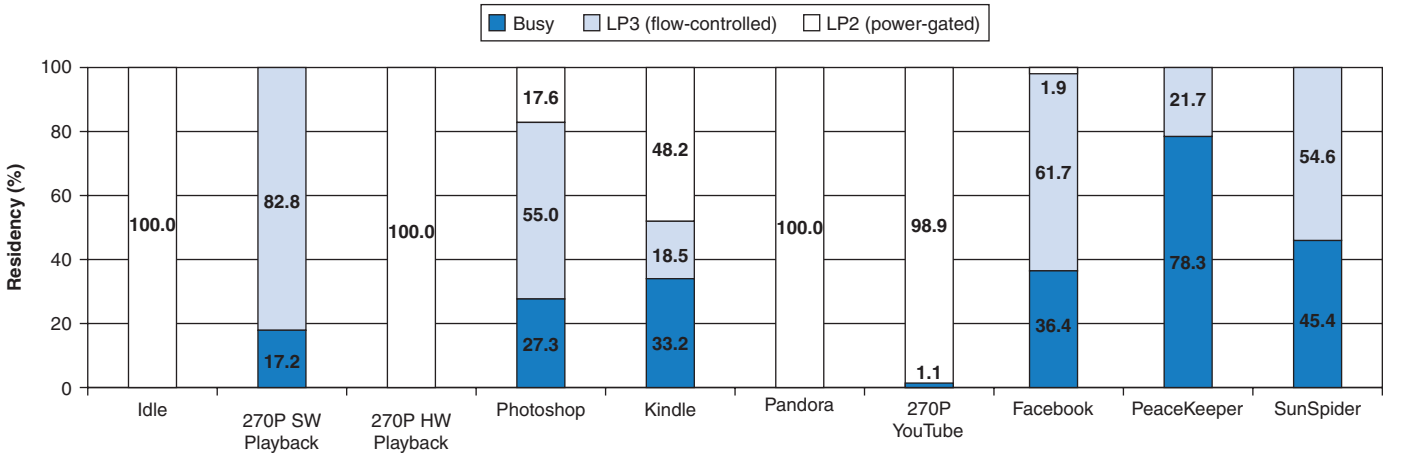


Figure 6: Idle-states residency on CPU1 from Motorola Atrix Smartphone (Source: Intel Corporation, 2012)

"… captures the baseline of the Atrix smartphone while not performing any active tasks."

Idle

This test captures the baseline of the Atrix smartphone while not performing any active tasks. 97.6 percent of the time CPU0 is at the idle state, and CPU1 is completely shut down. CPU0 has 51.3 interrupts/sec at idle while CPU1 has interrupt events. For the duration of the test, the CPU is running at 312 MHz 98.6 percent of the time. The CPU frequency never drops to 216 MHz although we observe the opportunity in the kernel.

Local Video Software/Hardware Playback

We use an H.264-encoded video clip with 480x270 resolution and 569 kbps bitrate for the video stream with 104 kbps for audio stream.

The results show that the software video decoding schedules workload on both CPU cores for decompressing video data, while in hardware accelerated

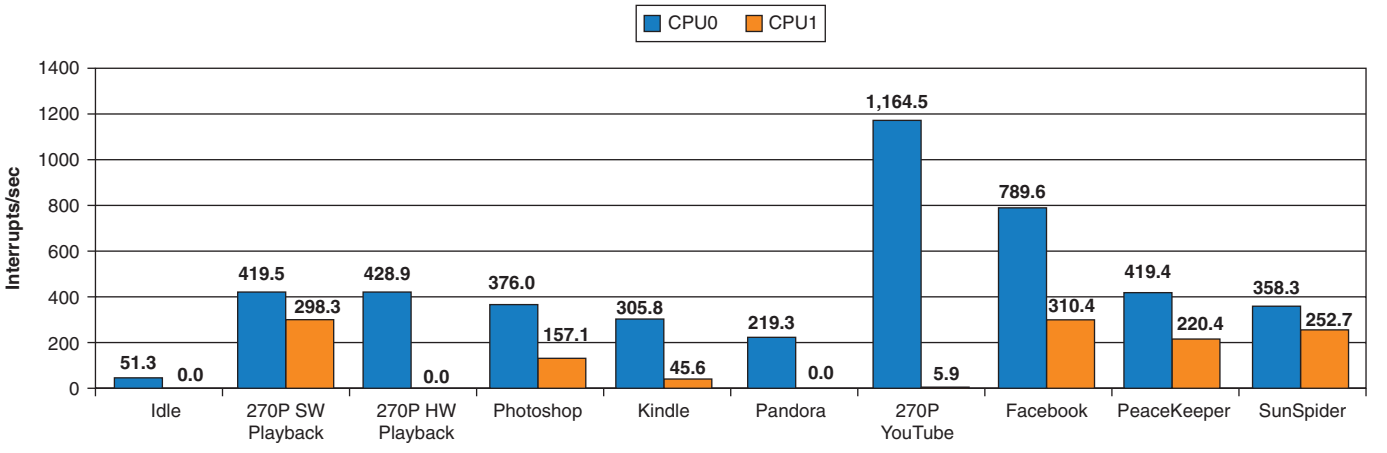


Figure 7: Interrupts/sec from Motorola Atrix Smartphone (Source: Intel Corporation, 2012)

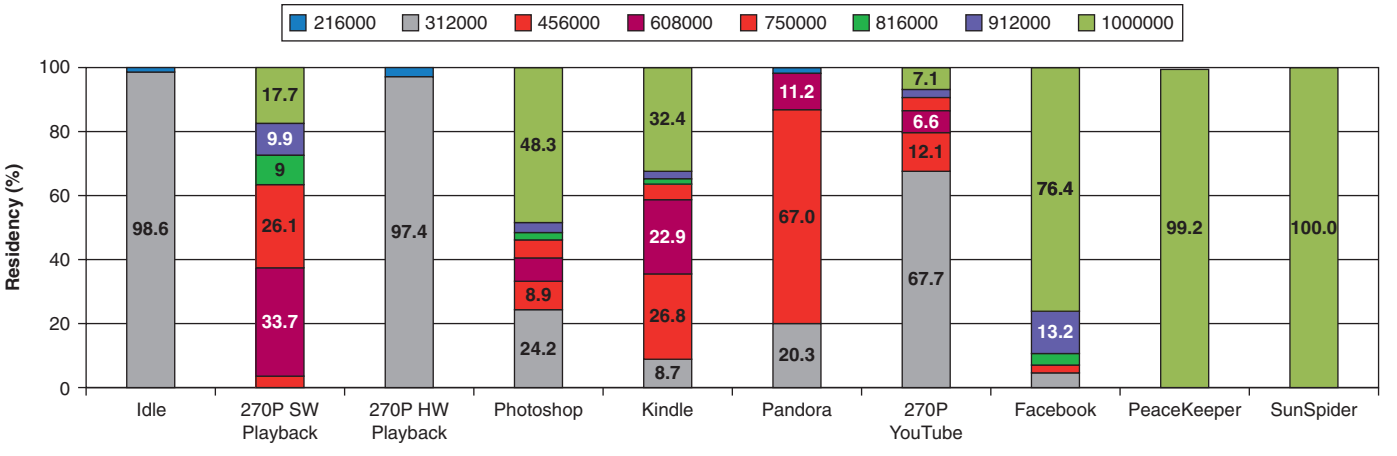


Figure 8: P-states residency on CPU0 and CPU1 from Motorola Atrix Smartphone (Source: Intel Corporation, 2012)

decoding the workload is offloaded to the GPU so only one CPU core is running to offload video data to the GPU. During software playback, CPU0 and CPU1 are 50.1 percent and 17.2 percent busy respectively during playback and the hardware accelerated decoding reduces load on CPU0 to 18.9 percent while CPU1 completely shuts down.

Moreover, looking at the P-state residency, the software approach runs CPU0 and CPU1 with frequencies higher than 700MHz 63 percent of the time while the hardware approach reduces CPU0 to 300MHz for 97 percent of the time. We additionally observe that during software playback CPU1 gets interrupted more frequently, which indicates workload is continuously scheduled to both CPU cores. During software playback, CPU1 is at 298.3 interrupts/sec while, during hardware playback, CPU1 has no interrupt events.

"During software playback CPU1 gets interrupted more frequently, which indicates workload is continuously scheduled to both CPU cores."

Photo Editing

This is the experiment to demonstrate how the Atrix smartphone behaves when a user generates unpredictable workload and the system is not able to offload work to the GPU for acceleration. Both CPU0 and CPU1 are busier than during the software video playback, since busy state residency is 36 percent and 27 percent respectively on CPU0 and CPU1. CPU0 sees 376 interrupts/sec while CPU1 sees 157.1 interrupts/sec. Both CPU0 and CPU1 spend 48 percent of their time at the 1 GHz highest frequency.

Reading an eBook

While conducting the eBook reading benchmark roughly 68 percent of the time CPU0 and CPU1 are at idle state, and about 32 percent of the time during this test CPU0 and CPU1 are in the busy state. CPU0 gets 305.8 interrupt/sec while CPU1 gets 45.6 interrupts/sec. For the most of the test the CPU is running at three frequencies: 456 MHz, 608 MHz, and 1 GHz for 26.8 percent, 22.9 percent, and 32.4 percent (respectively) of the time.

"For the streaming video test, the busy state of CPU0 is 37 percent, an 18 percent increase over hardware playback."

Streaming Video

For the streaming video test, the busy state of CPU0 is 37 percent, an 18 percent increase over hardware playback. Interrupts/sec reaches 1170 interrupts/sec or is 2.7 times local video playback. Moreover, the P-state residency statistics show that the time spent at the lowest frequency of 216 MHz is 67.7 percent or a 29.7 percent drop compared to local hardware playback.

"Streaming audio is basically a lightweight version of streaming video where data is streaming and the playback is audio only."

Streaming Audio

Streaming audio is basically a lightweight version of streaming video where data is streaming and the playback is audio only. For this benchmark, 55 percent of the time CPU0 is in the idle state and CPU1 is completely shut down. CPU0 sees 219.3 interrupt/sec while CPU1 sees no interrupt events and shuts off. CPU0 and CPU1 run less than 456 MHz 87.3 percent of the test duration.

Social Networking

For the social networking benchmark, CPU resources are heavily used at 57 percent for CPU0 and 36 percent for CPU1. CPU0 sees 789.6 interrupts/sec while CPU1 sees 310.4 interrupts/sec. Both CPU0 and CPU1 spend 78 percent of their time at the 1 GHz frequency level.

Web Browsing

During the PeaceKeeper HTML5 benchmark the workload is intensive as CPU0 and CPU1 spend 66 percent and 78.3 percent of their time, respectively, in busy states. CPU0 sees 419.4 interrupts/sec while CPU1 sees 220.4 interrupts/sec. Both CPU0/CPU1 spend 99.2 percent of their time at the 1 GHz frequency level.

In the SunSpider JavaScript benchmark the workload is again intensive as CPU0 and CPU1 spend 67.2 percent and 45.4 percent of their time, respectively, in the busy state. CPU0 sees 358.3 interrupts/sec while CPU1 sees

252.7 interrupts/sec. Both CPU0 and CPU1 spend 100 percent of their time at the 1 GHz frequency.

Discussion

On the Android platform for the Motorola Atrix, software engineers can optimize software to maximize CPU idle time by offloading workload to the GPU as a highest priority. The second priority would be to schedule the workload on only one CPU core as much as possible so the other CPU core gets shut down to save power. As a last resort if these options are not suitable, the system should schedule workloads on both CPU cores in Tegra 2 SoC to complete the job earlier and then move to an idle state. This approach, however, will increase CPU power consumption in the short term and may ultimately reduce battery life.

"… maximize CPU idle time by offloading workload to the GPU."

Motorola XOOM Tablet

The device under test is the Motorola XOOM tablet running Android 3.0 Honeycomb. The XOOM tablet features an NVIDIA dual-core 1 GHz Tegra 2 SoC with a 10.1-inch LCD screen and 1 GB DDR2 SDRAM. (The device is rooted.) Table 2 lists system detail.

"The device under test is the Motorola XOOM tablet running Android 3.0 Honeycomb."

Component	Specification
SoC	Tegra 2 T20
CPU	1 GHz dual-core ARM Cortex-A9
GPU	ULP GeForce GPU 333 MHz
RAM	1 GB DDR2 SDRAM
FLASH	32 GB NAND
Wi-Fi/Bluetooth/FM	Broadcom BCM4329
GPS	Broadcom BCM4750
Display	10.1-inch LCD 1280x800
Battery	6500 mAh Li-Po Battery

Table 2: Motorola XOOM Hardware Specification[32]
(Source: Motorola, Inc., 2012)

Test Methodology and Results

For each usage scenario, we collect the idle-state residency, P-state residency, and interrupts per CPU. In addition, wakeup events per second are also gathered. The experiment is done with the XOOM tablet in airplane mode with the Bluetooth and GPS turned off. Wi-Fi is turned on for streaming audio/video, social networking, reading an eBook and Web browsing.

The following sections discuss the software power results in detail from each usage scenario on the Motorola XOOM tablet under test, as shown in Figures 9 through 12. The usage scenarios include idle, video playback, photo editing, reading an eBook, streaming video/audio, social networking, and web browsing.

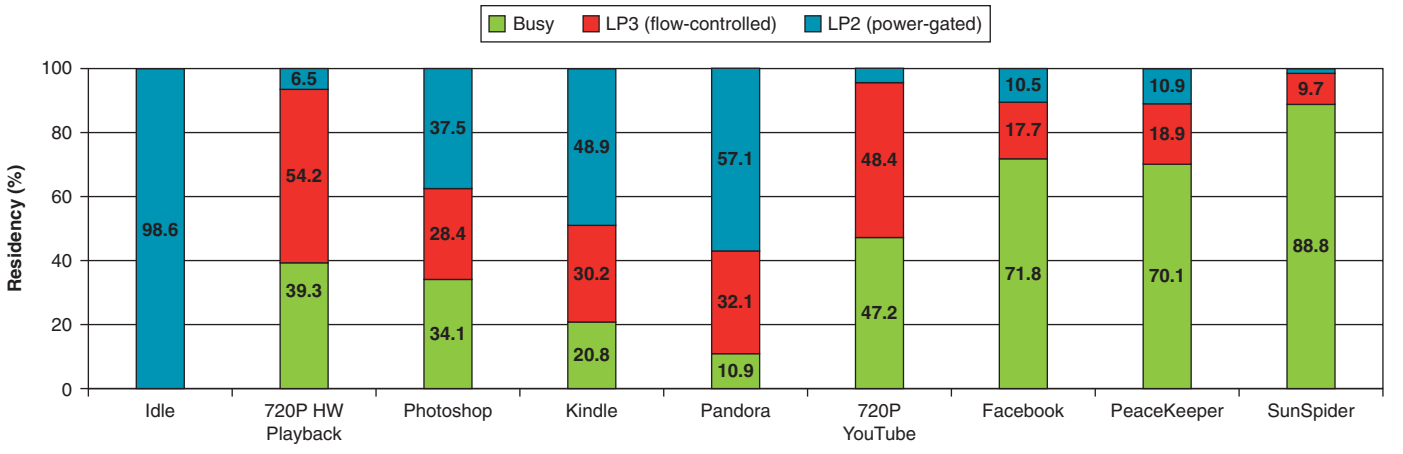


Figure 9: Idle-states residency on CPU0 from Motorola XOOM tablet (Source: Intel Corporation, 2012)

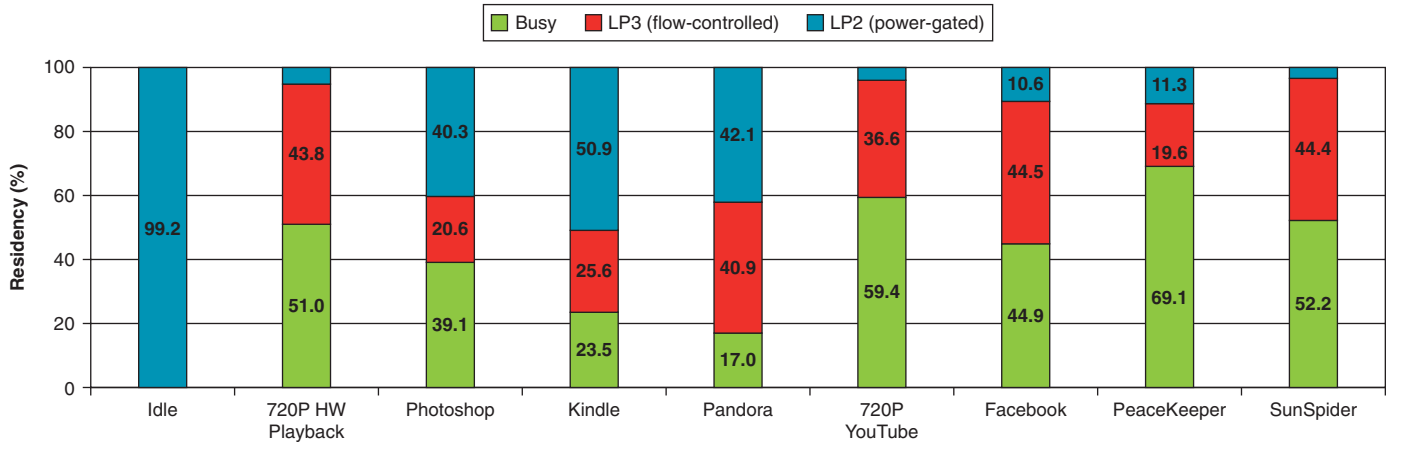


Figure 10: Idle-states residency on CPU1 from Motorola XOOM tablet (Source: Intel Corporation, 2012)

This test represents the baseline of the XOOM tablet when not performing any active task.

"The hardware video decoding schedules workload on both CPU cores."

Idle

This test represents the baseline of the XOOM tablet when not performing any active task. We observe 98.6 percent of the time CPU0 is in the idle state and 99.2 percent of the time CPU1 is in the idle state. CPU0 sees 19.7 interrupt/sec at idle while CPU1 has just 2.8 interrupts/sec. For the duration of this test the CPU is running at 216 MHz 99.3 percent of the time.

Local Video Hardware Playback

We use an H.264-encoded video clip with 1280x720 resolution and 2075 kbps bitrate for video streaming with 125 kbps for audio streaming.

The hardware video decoding schedules workload on both CPU cores. This is somewhat surprising since the majority of the work needed is to feed workload to the GPU.

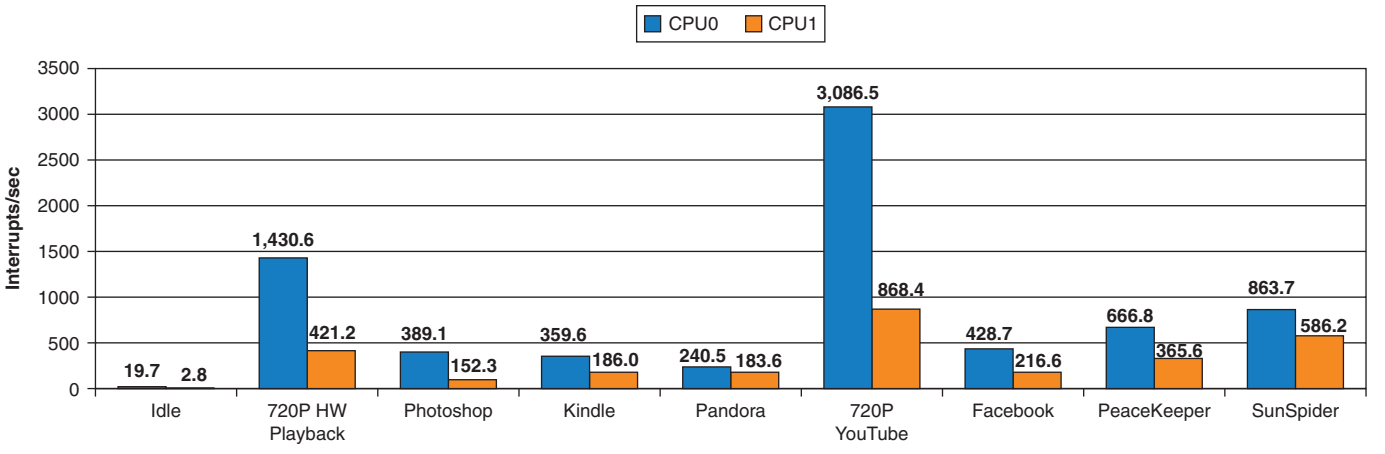


Figure 11: Interrupts/sec from Motorola XOOM tablet (Source: Intel Corporation, 2012)

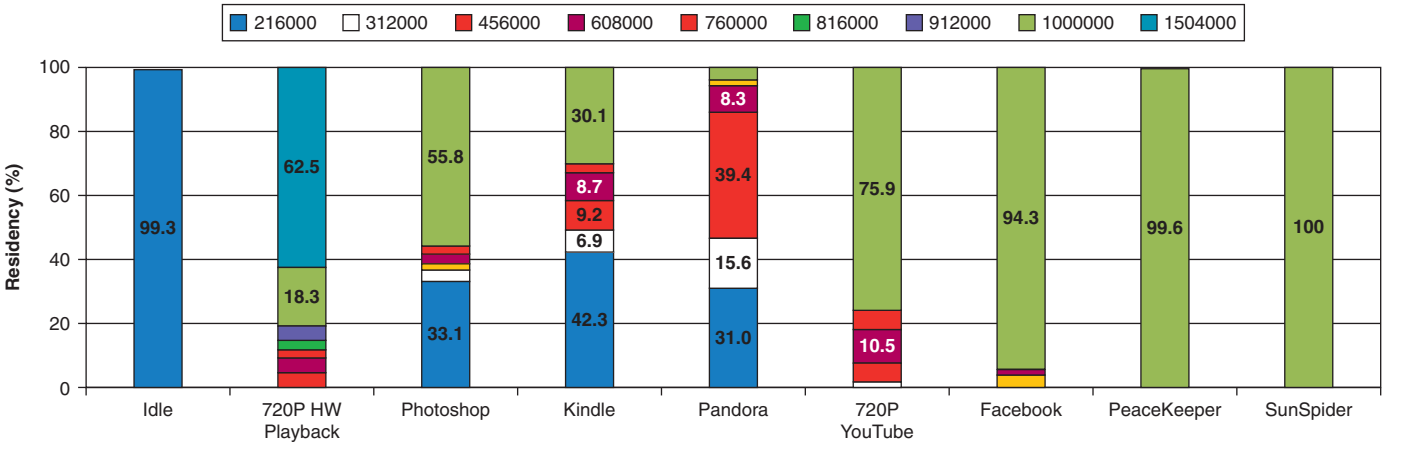


Figure 12: P-states residency on CPU0/CPU1 from Motorola XOOM tablet (Source: Intel Corporation, 2012)

During hardware playback CPU0 and CPU1 are 39.3 percent and 51 percent busy respectively. In the busy state, the hardware supported decoding approach runs CPU0 and CPU1 at 1.5 GHz speed 62.5 percent of the time. We also observe that the hardware playback on CPU0 and CPU1 is interrupted frequently at 1430.6 interrupts/sec and 421.2 interrupts/sec respectively. This confirms workload is continuously scheduled across both CPU cores.

Photo Editing

The photo editing experiments capture random workload behavior on the XOOM tablet that cannot be offloaded to the GPU for acceleration. The stress on CPU0 and CPU1 are similar to the hardware video playback, which keeps the cores busy state residencies at 34.1 percent and 39.1 percent respectively. Both CPU0 and CPU1 spend 55.8 percent of their

"The photo editing experiments capture random workload behavior on the XOOM tablet that cannot be offloaded to the GPU for acceleration."

time at 1 GHz frequency. CPU0 sees 389.1 interrupts/sec and CPU1 sees 152.3 interrupts/sec during the workload.

Reading an eBook

For the eBook reading benchmark 70 percent of the test duration CPU0 and CPU1 spend in the idle state; 22 percent of the time CPU0 and CPU1 are in the busy state. CPU0 sees 359.6 interrupt/sec while CPU1 sees 186 interrupts/sec. For the duration of the test each CPU is running at two frequencies: 216 MHz and 1 GHz for 42.3 percent and 30.1 percent respectively.

Streaming Video

"7.9 percent increase over hardware playback."

Figures 9 and 10 show busy state for CPU0 is 47.2 percent, which is a 7.9 percent increase over hardware playback. The interrupts/sec for CPU0 reaches 3086.5 interrupts/sec or 1655.9 interrupts/sec more than hardware video playback locally. In addition, CPU1 is in the busy state 59.4 percent of the test or an 8.4 percent increase over hardware playback. The interrupts account for 868.4 interrupts/sec or 447.2 interrupts/sec over hardware playback locally. CPU0 and CPU1 run at 1 GHz 75.9 percent of the time during the streaming video playback.

Streaming Audio

"We observe fewer interrupts as compared to the streaming video workload."

Streaming audio is similar to the streaming video test without the video portion. 10.9 percent of the time CPU0 is in an idle state while CPU1 is in idle 17 percent of the time. In this case, we observe fewer interrupts as compared to the streaming video workload. CPU0 sees 240.5 interrupts/sec and CPU1 sees 183.6 interrupts/sec. 86 percent of the time CPU0 and CPU1 run at less than 450 MHz frequency for this workload.

Social Networking

For this CPU intensive workload both CPU0 and CPU1 are busy 71.8 percent and 44.9 percent of the time. Interrupts on CPU0 and CPU1 are 428.7 interrupts/sec and 216.6 interrupts/sec, respectively. Both CPU0 and CPU1 spend 94.3 percent of the time at the 1 GHz frequency level.

Web Browsing

In the HTML5 benchmark we observe the workload requires CPU resource CPU0 and CPU1 in busy states 70.1 percent and 69.1 percent of the time, respectively. CPU0 sees 666.8 interrupts/sec while CPU1 sees 365.6 interrupts/sec. Both CPU0 and CPU1 spend 99.6 percent of their time at the 1 GHz frequency.

During the JavaScript benchmark, we observe CPU0 and CPU1 spending 88.8 percent and 52.2 percent of their time in the busy state, respectively. CPU0 sees 863.7 interrupts/sec while CPU1 sees 586.2 interrupts/sec. Both CPU0 and CPU1 spend 100 percent of their time at the 1 GHz frequency.

Discussion

On the Android platform for the Motorola Xoom tablet, device applications tend to schedule workload on both CPU cores while seeking the opportunity

to offload to the GPU. The scheduling appears more performance centric where the primary goal is to finish a task fast and go to idle state to save power.

Conclusions and Comparisons

In this case study, we have revealed the relevant system-level software power events that CPU resource analysis tools can provide on the Motorola Atrix smartphone and the Motorola XOOM tablet using workloads based upon common usage scenarios. The Atrix smartphone and the XOOM tablet both feature aggressive energy-saving scheduling within the Android operating system with little practical difference between the functionality.

The Atrix smartphone and XOOM tablet both come with NVIDIA Tegra 2 SoC and 1 GB memory but the Atrix smartphone has low-power versions of the Tegra SoC and memory. Both the devices try to offload their workload from the ARM CPU to the NVIDIA GPU as much as possible. For workloads that cannot be offloaded, Atrix tends to schedule only one CPU core to run the task while XOOM schedules the workload on both of its dual-core ARM CPU cores. We also observe that both Android devices prioritize running at low frequencies and never use full speed capabilities of the CPU resources even under intensive benchmark workloads. Finally, the Atrix smartphone never runs at the 216 MHz frequency, which is available in the Android kernel. The XOOM tablet only uses the 1.5 GHz setting for the MX player; in all other instances the tablet runs only as fast as the 1 GHz frequency.

"Both the devices try to offload their workload from the ARM CPU to the NVIDIA GPU."

Summary

The Android-based smartphone has become the most popular in the United States while smartphones are changing people's daily lives. To make best use of such smartphone's battery efficiency is critical. Although Android devices have aggressive power management capabilities derived from the standard Linux kernel, software applications and services are optimized to run in an energy-efficient way.

CPU power optimization approaches are needed to tune software on Android mobile systems. In this study, we have demonstrated the promise of CPU resource analysis tools for system-wide platform diagnosis. Tuned software applications and services request CPU resources only when necessary so devices remain at low power states as much as possible thereby increasing energy efficiency and ultimately battery life.

"CPU power optimization approaches are needed to tune software on Android mobile systems."

References

[1] S. Schroeder, "Android Overtakes BlackBerry As the Top U.S. Smartphone Platform," Mashable Tech, retrieved March 1, 2011. Available: http://mashable.com/2011/03/08/android-smartphone-platform-report/

[2] J. O'Dell, "Android Wins in the U.S. Smartphone Wars," Mashable Tech, retrieved March 1, 2012. Available: http://mashable.com/2011/03/03/nielsen-android-winner/

[3] Android Open Source Project, retrieved March 1, 2012. Available: http://source.android.com/

[4] D. A. Heger, "Mobile Devices – An Introduction to the Android Operating Environment, Design, Architecture, and Performance Implications," DHTechnologies (DHT), retrieved March 1, 2012. Available: http://www.cmg.org/measureit/issues/mit78/m_78_3.pdf

[5] R. Darell, "The Complete Android History Timeline," Bit Rebels, retrieved April 13, 2012. Available: http://www.bitrebels.com/technology/the-complete-android-history-timeline-infographic/

[6] F. Maker and Y.-H. Chan, "A Survey on Android vs. Linux," University of California, Davis, 2009.

[7] "Battery Powered Linux Mini-HOWTO," The Linux Documentation Project (TLDP), retrieved March 1, 2012. Available: http://tldp.org/HOWTO/Battery-Powered/powermgm.html

[8] S. Vaddagiri, A. K. Santhanam, V. Sukthankar, and M. Iyer, "Power Management in Linux-Based Systems," *Linux Journal*, retrieved March 1, 2012. Available: http://www.linuxjournal.com/article/6699

[9] Power Management, Android Platform Development Kit, retrieved March 1, 2012. Available: http://www.netmite.com/android/mydroid/development/pdk/docs/power_management.html

[10] PowerManager Class, Android Developers, retrieved March 1, 2012. Available: http://developer.android.com/reference/android/os/PowerManager.html

[11] J. Zhang, "Linux Kernel and Android Suspend/Resume," retrieved March 1, 2012. Available: http://kzjblog.appspot.com/2010/11/20/suspend-en.html#sec-6

[12] E. Oliver, "The challenges in large-scale smartphone user studies," in *Proceedings of the 2nd ACM International Workshop on Hot Topics in Planet-scale Measurement (HotPlanet'10)*, pp. 1–5, 2010.

[13] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *Proceedings of the 8th international conference on Mobile systems, applications, and services (MobiSys'10)*, pp. 179–194, 2010.

[14] A. Shye, B. Scholbrock, and G. Memik, "Into the wild: Studying real user activity patterns to guide power optimizations for mobile architectures" in *42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO'42)*, pp. 168–178, 2009.

[15] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, et al., "MAUI: making smartphones last longer with code offload," in *Proceeedings of the 8th international conference on Mobile systems, applications, and services (MobiSys'10)*, pp. 49–62, 2010.

[16] A. Rahmati, A. Qian, and L. Zhong, "Understanding human-battery interaction on mobile phones," in *Proceedings of the 9th international conference on Human computer interaction with mobile devices and services (MobileHCI'07)*, pp. 265–272, 2007.

[17] N. Banerjee, A. Rahmati, M. D. Corner, S. Rollins, and L. Zhong, "Users and batteries: interactions and adaptive energy management in mobile systems (UbiComp'07)," in *Proceedings of the 9th international conference on Ubiquitous computing*, pp. 217–234, 2007.

[18] K. N. Truong, J. A. Kientz, T. Sohn, A. Rosenzweig, A. Fonville, and T. Smith, "The design and evaluation of a task-centered battery interface," in *Proceedings of the 12th ACM international conference on Ubiquitous computing (UBICOMP'10)*, pp. 341–350, 2010.

[19] A. Shye, B. Scholbrock, G. Memik, and P. A. Dinda, "Characterizing and Modeling User Activity on Smartphones: Summary," *SIGMETRICS Performance Evaluation Review*, vol. 38, no. 1, pp. 375–376, June 2010.

[20] D. Ferreira, A. K. Dey, and V. Kostakos, "Understanding Human-Smartphone Concerns: A Study of Battery Life," in *Proceedings of the 9th international conference on Pervasive (Pervasive'11)*, pp. 19–33, 2011.

[21] R. Joseph and M. Martonosi, "Run-time power estimation in high performance microprocessors," in *Proceedings of the 2001 international symposium on Low power electronics and design (ISLPED'01)*, pp. 135–140, 2001.

[22] C. Isci and M. Martonosi, "Runtime Power Monitoring in High-End Processors: Methodology and Empirical Data," in *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture (MICRO'36)*, p. 93, 2003.

[23] R. Ge, X. Feng, S. Song, H.-C. Chang, D. Li, and K. W. Cameron, "PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 5, pp. 658–671, 2010.

[24] Z. Yang, "PowerTutor – A Power Monitor for Android-Based Mobile Platforms" EECS, University of Michigan, retrieved March 1, 2012. Available: http://ziyang.eecs.umich.edu/projects/powertutor/

[25] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, et al., "Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones," in *Proceedings of the*

eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis (CODES/ISSS '10), pp. 105–114, 2010.

[26] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang, "Fine-Grained Power Modeling for Smartphones Using System Call Tracing," presented at the Proceedings of the sixth conference on Computer systems (EuroSys'11), Salzburg, Austria, 2011.

[27] M. Dong and L. Zhong, "Sesame: Self-Constructive Energy Modeling for Battery-Powered Mobile Systems," in *Proceedings of the 9th international conference on Mobile systems, applications, and services (MobiSys'11)*, pp. 335–348, 2011.

[28] Android Memory Usage, Embedded Linux Wiki, retrieved March 1, 2012. Available: http://elinux.org/Android_Memory_Usage

[29] Debbugging: Using DDMS, Android Developers, retrieved March 1, 2012. Available: http://developer.android.com/guide/developing/debugging/ddms.html

[30] Debugging from Eclipse with ADT, Android Developers, retrieved March 1, 2012. Available: http://developer.android.com/guide/developing/debugging/debugging-projects.html

[31] Motorola Atrix 4G MB865 Specifications Table, Motorola Developer, retrieved March 1, 2012. Available: http://developer.motorola.com/products/atrix-2-mb865/

[32] Motorola Xoom Specifications Table, Motorola Developer, retrieved March 1, 2012. Available: http://developer.motorola.com/products/xoom-mz601/

Author Biographies

Hung-Ching Chang received a BS degree in electronic engineering from Huafan University, Taiwan. He has been working towards a PhD in computer science at Virginia Polytechnic Institute and State University since 2007. He has worked as an intern in the Software and Services Group at Intel. His major research interests include high performance parallel computing, power-aware computing, and computer architecture.

Abhishek Agrawal has over 10 years of industry and research experience. He is currently a senior technical lead in the Software and Services Group at Intel and leads software power enabling for Intel's client and Intel® Atom™ based platforms. He chairs the industry-wide power management working group for the Climate Savers Computing Initiative and has authored several industry whitepapers and technical papers on energy efficiency in refereed international conferences and journals as well as coauthored a book entitled *Energy Aware Computing*. Abhishek is a member of IEEE and ACM and is on many industry panels and IEEE/ACM conference committees on Green Computing.

Kirk W. Cameron received his BS degree in mathematics from the University of Florida in 1994 and a PhD degree in computer science from Louisiana State University in 2000. He is currently an associate professor of computer science at Virginia Polytechnic Institute and State University. He directs the SCAPE Laboratory at Virginia Tech, where he pioneered the area of high-performance, power-aware computing to improve the efficiency of high-end systems. He has received numerous awards and accolades for his research and publications including the National Science Foundation Career Award in 2004, the Department of Energy Career Award in 2004, the USC COE Young Investigator Research Award in 2005, the Best Paper Nominee SC06, the VT COE Fellow in 2007, the IBM Faculty Award in 2007, the Uptime Institute Fellow in 2008, and was invited to the 2008 National Academy of Engineering Symposium. He is on the editorial board and the editor for the IEEE Computer "Green IT" column. He is a member of the IEEE and the IEEE Computer Society.

IMPROVING USER EXPERIENCE BY MAKING SOFTWARE GREEN

Contributors

Jun De Vega
Developer Relations Division,
Software and Systems Group,
Intel Corporation

Manuj Sabharwal
Developer Relations Division,
Software and Systems Group,
Intel Corporation

Rajshree Chabukswar
Developer Relations Division,
Software and Systems Group,
Intel Corporation

"Improved energy efficiency is combination of hardware capability and well-designed software."

"Software needs to pay equal attention to prolonging battery life."

Be it Ultrabooks™ or tablets, longer battery life is one of the essential pillars of enhanced user experience. Improved energy efficiency is a combination of hardware capability and well-designed software. A single badly behaving software application can impact an entire system's energy efficiency and hence provide a bad user experience. This article provides an overview of various hardware features introduced by Intel to enhance energy efficiency in recent years. Since in a typical usage scenario the system is idle 80 percent of the time, application behavior in idle mode has been very critical to extending battery life. Even if a single application has services waking up and causing the CPU to come out of sleep states, the battery life of the system decreases significantly. The article presents some of the case studies on how to analyze idle applications for energy efficiency, mainly focusing on Windows*. Along with idle, active applications have a high impact the battery life by poor performance and/or by doing wasted/extra work. We show how to identify and fix such performance issues and to write efficient software, which can yield better performance per watt. We also discuss power benefits of offloading active work to other specialized components as compared to executing them on the CPU. The article details software analysis methods/tools that Intel has been using to investigate power consumption issues.

Introduction

Longer battery life is one of the main ways to improve the user experience with portable devices such as laptops or tablets. In recent years hardware has evolved significantly to provide power saving modes so that the system battery life would increase with the introduction of processor frequency states and sleep states. But in conjunction with hardware, software needs to pay equal attention to prolonging battery life. If a background application is constantly waking up the CPU, battery life can be impacted greatly. Likewise, during active runtime, if an application is doing wasted or unnecessary work, that impacts both power and performance. This article discusses tools, methods, and case studies to improve energy efficiency of the software to make software "green" and by doing so improve user experience. The more efficient the software is the less power is consumed, providing longer battery time. The case studies and tools discussed in this article are focused on Intel Core architecture running with Microsoft Windows*, but can be applied on other architectures and operating systems as well. The case studies discussed in the article provide a foundation for building green software. Moreover, the contribution of this work is proposing several software optimizations and presenting the benefits gained. One of the optimizations presented offered more than 2W power savings in

idle, and another optimization for active workload case reduced CPU usage by 15 percent, resulting in lowering the processor frequency states and saving power.

Hardware Features Overview

This section discusses some of the hardware features offered by recent processors to improve power efficiency. Subsequent sections will discuss software analysis and optimizations.

Processor Power Management

The power of microprocessors can be managed at the level of the whole processor or in specific areas. With dynamic voltage scaling and dynamic frequency scaling, the CPU core voltage, clock rate, or both, can be altered to decrease power consumption at the price of potentially lower performance. This is sometimes done in real time to optimize the power-performance tradeoff. Intel has three technologies for power management: P-states (Enhanced Intel SpeedStep® Technology or EIST), Turbo Boost, and C-states Demotion and Un-demotion.

"With dynamic voltage scaling and dynamic frequency scaling, the CPU core voltage, clock rate, or both, can be altered to decrease power consumption at the price of potentially lower performance."

P-States (Performance States, Processor Active)

Modern processors enable power savings by dynamically switching between multiple frequency and voltage points (P-states). P-states provide a way to scale the frequency and voltage at which the processor runs so as to reduce the power consumption of the CPU:

- P0 – Turbo frequency
- P1 – High Frequency Mode (HFM), Guaranteed or Base Frequency
- P2 . . . Pn-1 (Several steps of P-states increments)
- Pn – Lowest Frequency Mode (LFM)

Enhanced Intel SpeedStep® Technology (EIST)

EIST is the official name of P-state in Intel processors. It is the mechanism that allows a processor to change its voltage and frequency operating point (OP), either to save power or get more performance. It is co-managed by software and hardware. Software is responsible for determining performance needs and making the appropriate request to hardware. The hardware is responsible for setting internal limitations (such as power consumption, voltage requirements, number of cores active, and temperature). Software "expects" to at least get the performance level and frequency it requested from hardware under normal conditions. New Intel processors like the 2nd Generation Intel® Core™ processors include a power control unit (PCU) dedicated to making the chip's power management more efficient. Code that doesn't scale with frequency is more efficient to run at lower P-states in order to save power. Developers don't control P-states, hence it is recommended to optimize for processor usage from application side in order to reduce unnecessary work (see the case studies later in this article).

"Developers don't control P-states, hence it is recommended to optimize for processor usage from application side in order to reduce unnecessary work."

"Generally, higher C-states turn off more parts of the CPU, which significantly reduce power consumption."

C-States (Processor Idle Sleep States)

With the exception of C0, C-states are sleep or idle states; there is no processing done. Different processors support different numbers of C-states in which various parts of the CPU are turned off. Generally, higher C-states turn off more parts of the CPU, which significantly reduce power consumption. Processors may have deeper C-states that are not exposed to the operating system. The ACPI standard only defines four CPU power states from C0-C3 as shown in Table 1:

- C0, the CPU is processing and P-state transition happens
- C1 halts the CPU and there is no processing but the CPU's own hardware management determines whether there will be any significant power savings (required by ACPI).
- C2 is optional; most clocks are stopped
- C3 is also known as sleep or completely stop all clocks in the CPU

ACPI/OS	Hardware C-State
C1	C1
C2	C3
C3	C6 or C7 (AC, DC)

Table 1: Mapping of ACPI to Hardware C-States (Source: Intel Corporation, 2012[1])

Intel® Turbo Boost Technology

Intel® Turbo Boost Technology is a feature in newer Intel processors that automatically enables the processor cores to run faster than the guaranteed operating frequency as long as it operates below power, current, and temperature specification limits.

When the operating system requests P0 state, the processor sets core frequencies between P1 to P0 range. A P0 state with only one core busy achieves the maximum possible Intel Turbo Boost Technology frequency, whereas when the processor is running two to four cores the frequency is constrained by processor limitations. Under normal conditions the frequency does not go below P1, even when all cores are running.

"The 2nd Generation Intel Core processors have energy MSRs (model-specific registers) that provide the processor energy/power status."

Processor Energy Counters

The 2nd Generation Intel Core processors have energy MSRs (model-specific registers) that provide the processor energy/power status. The RAPL (Running Average Power Limit) interfaces provide mechanisms to enforce power consumption limit.

RAPL interfaces consist of non-architectural MSRs. Each RAPL domain supports the following set of capabilities, some of which are optional:

- Power limit – MSR interfaces to specify power limit, time window; lock bit, clamp bit, and so on.

- Energy Status – Power metering interface providing energy consumption information.
- Perf Status (Optional) – Interface providing information on the performance effects (regression) due to power limits. It is defined as a duration metric that measures the power limit effect in the respective domain. The meaning of duration is domain specific.
- Power Info (Optional) – Interface providing information on the range of parameters for a given domain, minimum power, maximum power, and so on.
- Policy (Optional) – Four-bit priority information, which is a hint to hardware for dividing budget between sub-domains in a parent domain.

"Energy Status – Power metering interface providing energy consumption information."

Each of the above capabilities requires specific units in order to describe them. Power is expressed in watts, time is expressed in seconds and energy is expressed in joules. Scaling factors are supplied to each unit to make the information presented meaningful in a finite number of bits. Units for power, energy, and time are exposed in the read-only MSR_RAPL_POWER_UNIT MSR. Figure 1 shows a representation of the MSR.

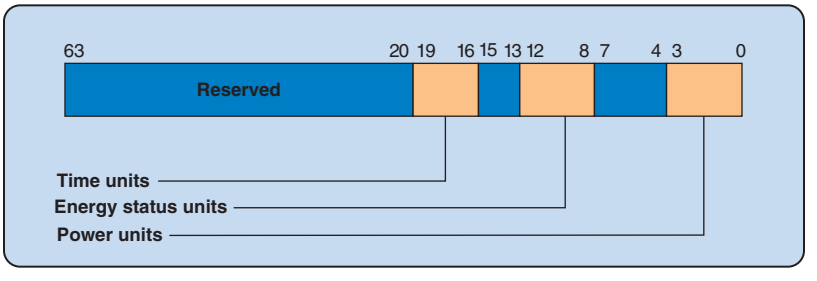


Figure 1: MSR_RAPL_POWER_UNIT Register
(Source: Intel Software Developer Manual, 2011[1])

Total Energy Consumed (bits 31:0): The unsigned integer value represents the total amount of energy consumed since that last time this register is cleared. The unit of this field is specified by the "Energy Status Units" field of MSR_RAPL_POWER_UNIT. MSR_PKG_ENERGY_STATUS is part of the RAPL domain and reports measured actual energy usage. See details in Table 2.

Register Address	Register Name	Bit Description
611H	MSR_PKG_ENERGY_ STATUS	PKG Energy Status (R/O) See Section 14.7.3, "Package RAPL Domain"

Table 2: MSR and the description
(Source: Intel Corporation 2012[1])

Tools

This section presents tools that can help you analyze the impact of software on system power. It gives an overview of the tools used by many Intel engineers when analyzing software power/performance issues.

Windows Performance Monitor

Windows Performance Monitor (perfmon) is a higher level analysis tool used to understand application behavior and define system bottlenecks. It offers various statistics such as user and privilege mode time, to understand whether the application is spending most of the time in user space or in kernel components. Statistics like system call rate and context switch rate can help determine if the application is making excessive system call entries and if the context switch rate is affecting application performance. A high page fault rate is typically an indication of memory management issues that might be affecting application performance. Typeperf is a command line wrapper for Windows Performance Monitor where data can be logged into a CSV file.

Intel® VTune™ Amplifier XE

Intel® VTune™ Amplifier provides a way to get to next level of analysis to identify hot modules or processes and drill down into each component for further analysis. It also offers a way to get call stack information to understand the flow between various application level modules.

This is one of the primary tools used at Intel for any power/performance analysis.

Intel® Power Gadget

Determining the power consumption of a device requires some tools that can provide real-time power usage of the device or some of its components. Intel® Power Gadget is a software-based power monitoring tool enabled for 2nd Generation Intel Core processors. It includes a Microsoft Windows sidebar gadget, driver, and libraries to monitor real-time processor package power information in watts using the energy counters in the processor.

"Intel Power Gadget is a simple and easy way to monitor energy usage of the processor."

Common use of Intel Power Gadget is via a Windows 7 sidebar gadget. It is a simple and easy way to monitor energy usage of the processor. The gadget displays the processor power consumption (watts) and frequency (MHz) in real-time as graphs, as shown in Figure 2. It allows the user to log the power and frequency measurements and has a command line utility called PwrlogSample.exe to collect the same power information.

Intel® Performance Bottleneck Analyzer Power Correlation

"Intel Performance Bottleneck Analyzer (Intel PBA) is a toolset built to parse data collected by VTune Amplifier to automatically find power/performance bottlenecks."

Intel Performance Bottleneck Analyzer (Intel PBA) is a toolset built to parse data collected by VTune Amplifier to automatically find power/performance bottlenecks. The toolset creates a report of issues identified and ranks them in order of impact for developers to take a look at it. The power analysis capability

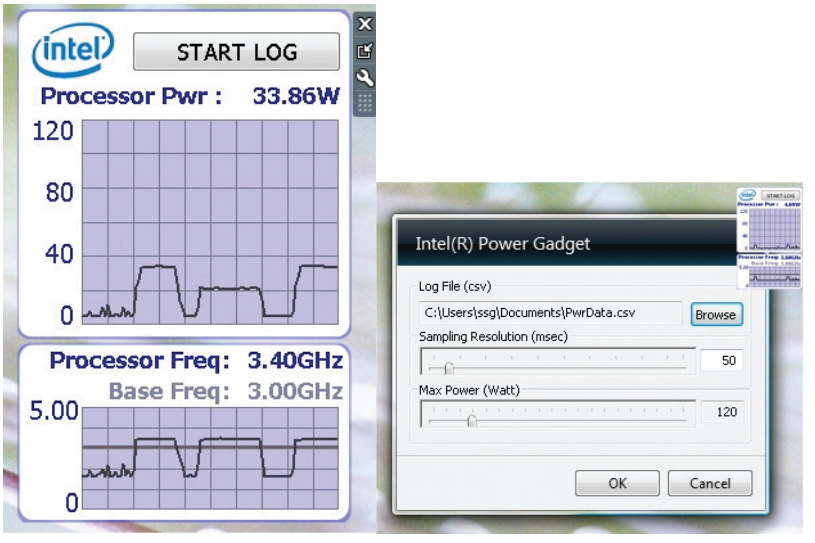


Figure 2: Intel® Power Gadget utility
(Source: Intel Corporation, 2012)

offered by this tool is currently applicable to active analysis. The current version correlates power data collected using NetDAQ* with performance data to create two bins: high and low power consumption areas. The tool then creates reports showing modules/processes/threads and functions distribution in these two bins. A new version that uses energy MSRs will soon be available publically.

Figure 3 shows that when correlating high and low power data, we get a distribution of clocks in a high power bucket and low power bucket.

Module_Name	HighPowerClocks%	LowPowerClocks%
App.DLL	64.04	33.59
ntkrnlpa.exe	17.53	20.62
igdumd32.dll	4.81	24.08
kernel32.dll	4.8	1.78
ntdll.dll	3.65	2.14

Figure 3: Module view of high-low power buckets
(Source: Data obtained by running Intel PBA analysis on an application workload on Intel® Core™ i5 based system)

Drilling down further indicates that a spin-wait loop without pause instruction was causing high power buckets in the application module, as shown in Figure 4.

Function	HighPowerClock%	LowPowerClock%
SpinWait() with no PAUSE	3.58	1.64
RtlUnwind	2.94	3.21
RasterPathPointsPaintServer::GetRasterSegAAType2	1.77	1.19
ReduceCurve	1.75	1.15
memcpy	1.13	3.45
Transition::Add2	1.11	0.77
OffsetComputerLoop::computeTangent	1.08	0.72

Figure 4: Function view of high-low power buckets
(Source: Data obtained by running Intel PBA analysis on an application workload on Intel® Core™ i5 based system)

Microsoft Windows Performance Analyzer

Windows Performance Analyzer, also known as xPerf, is a tool from Microsoft to analyze a wide range of power and performance problems including operating system scheduling, I/O activities, interrupts, call stacks to Win32 API, usage, and responsiveness issues. Windows Performance Analyzer leverages the event tracing for windows (ETW) mechanism to collect the events and save them in one or more files. Several events such as context switches, ISRs, and DPCs are used to optimize applications for power efficiency. On Windows 8 xPerf extends the profiling to identify problems related to C/P-states lifetime, CPU starvation issues, and slow on/off transition. On Windows 8 xPerf is supported on different architectures. One of the important features of ETW available on Windows is symbol decoding, which provide stacks on kernel events.

Hardware Instrumentation

"Software can have an impact on any small peripherals connected to the platform."

Measuring platform components power such as LAN, chipset, and storage currently requires external instruments. Software can have an impact on any small peripherals connected to the platform. In our idle study we used hardware instrumentation to collect the power numbers in watts for different components on the platform. We measure the drop in current across the register for voltage change at instrumented components.

Intel® Media Software Development Kit (SDK)

"It is a software development library that exposes Intel platforms' media acceleration capabilities."

Intel® Media SDK is a tool specifically designed for software developers of media applications for video playback and encoding and enables developers to take advantage of hardware acceleration in Intel platforms. It is a software development library that exposes Intel platforms' media acceleration capabilities (encoding, decoding, and transcoding). The library API is cross-platform and works across operating systems with software fallback.

Client Application Power Analysis

After a quick look at tools, this section presents idle and active case studies and their impact on system power. Studies discussed here showcase how the tools mentioned in the previous subsection can be used to identify primary

bottlenecks. Along with CPU optimizations, we provide a case study to demonstrate the benefit of using GPU hardware offload in order to save power.

Idle Power Analysis

An application at idle must consume negligible platform power over the baseline. Our definition of idle software: when no user tasks are being executed by an application. The scenario of idle software includes a consumer chat application running in the background with friends online but not doing any active chat. Hardware and operating system vendors have optimized their operating system, firmware, drivers, and all the small components on the consumer platform to achieve better energy efficiency. In addition, as hardware components' power states become more sensitive, software must be well behaved at idle so it doesn't needlessly wake components, which would limit battery life.

"Software must be well behaved at idle so it doesn't needlessly wake components, which would limit battery life."

In this section, we study the power benefits achieved from many aggressive and traditional optimizations. The case study here is focused on a media application that has significant impact on system power even in idle state. Optimization techniques discussed here can be extended to any client- or server-based idle applications to extend battery life on a Windows operating system.

To perform the analysis, we downloaded a media application onto a system with a clean installation of Windows 8 (BUILD version). We simulated a real-world scenario by opening the application in default windowed mode and let it be idle for 10 to 15 minutes. An application at "idle" must not act as an "active" workload; it should affect power usage only minimally, relative to system idle. We measured the impact of application on total platform with hardware instrumentation. Figure 5 shows the impact of power on different components when the application is at idle. There is significant increase in CPU (Core + UnCore), LAN, and chipset power while the display and I/O have negligible increase in total power.

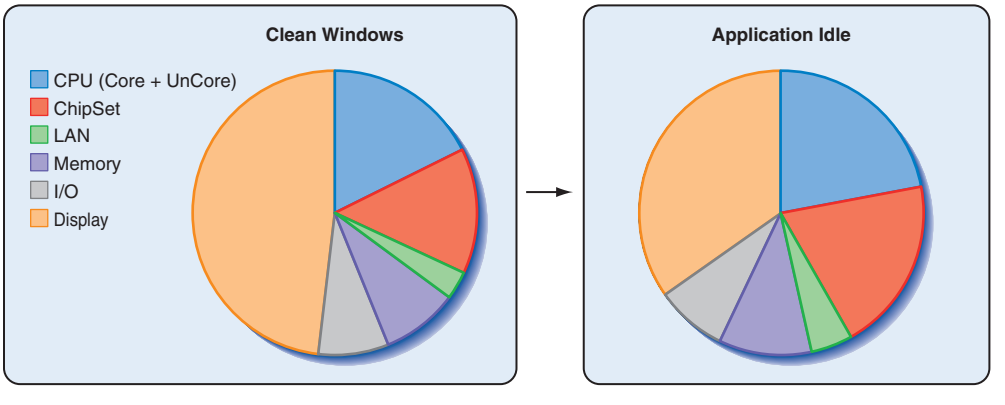


Figure 5: Power profile difference comparing clean OS build with application installed (Source: Data obtained by running Fluke* NetDAQ instrument on an application workload on Intel® Core™ i5 based system)

To identify the reason for the increase in power usage of different components, we used xPerf for root-cause analysis by running *xPerf –on digaeasy –stackwalk CSwitch* command line in administrator mode for 60 seconds.

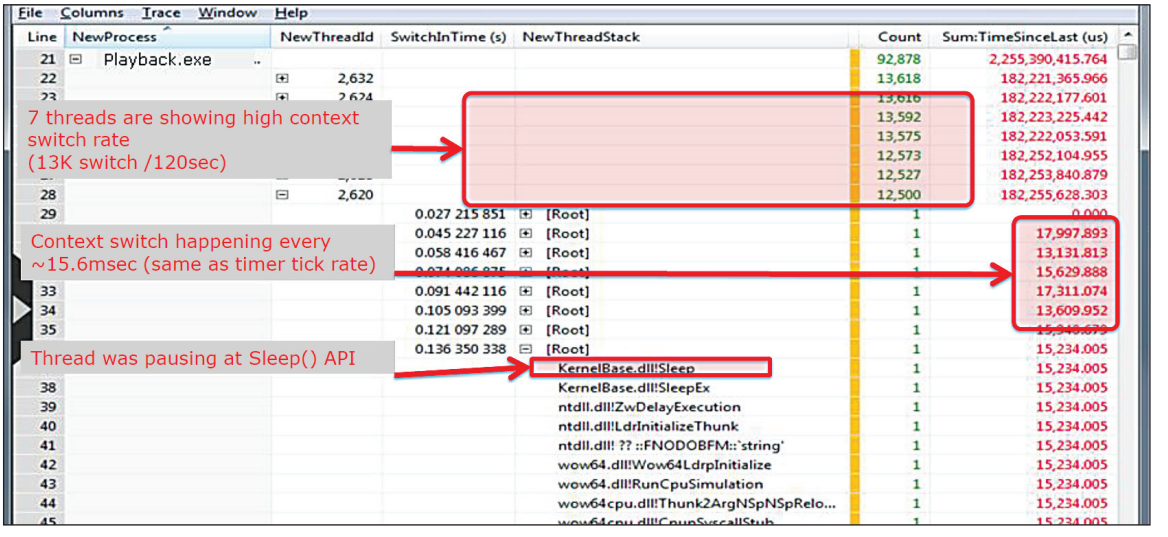


Figure 6: XPerf trace showing call stacks
(Source: Data obtained by running Windows* Performance Analyzer on an application workload on Intel® Core™ i5 based system)

Figure 6 shows the Win32 calls made by playback applications when it was idle. The Count column provides the number of context switches due to the application thread mechanism. Context switches counts for the playback application reaches approximately 13K switches/120 seconds. The NewThreadId column shows multiple threads forked by the playback application and the NewThreadStack column shows the call stacks triggered by kernel events. Tracing shows that the application is having a high context switches count even at idle state, and Sleep() Win32 API is called from playback application threads every 15 milliseconds. The application thread is calling a function by calling Sleep(1). The TimeSinceLast column shows the time of context switch every 15.6 milliseconds, but at the default timer tick rate of Windows at 15.6 milliseconds, the call is actually equivalent to a sleep that is much larger than 1 millisecond and a wait for 15.6 milliseconds, since a thread can only wake up when the kernel wakes it. Such a Sleep call means the thread is inactive for a very long time while the lock could become available or work could be placed in the queue.

"Such a Sleep call means the thread is inactive for a very long time while the lock could become available or work could be placed in the queue."

Visualizing the code, we see that this media application had several sleep functions in a loop:

```
for (;;) {
if ( . . check some condition) {
. . do something . . .
SetEvent()
```

```
}
Sleep(1); //wait for next tick
}
A better implementation would be the following:
for (;;) { WaitForSingleObject();
. . . . Do something #1 . . .
}
for(;;) { WaitForSingleObject();
. . . . Do something #2 . . .
}
```

Consolidating the activities using the SetWaitableTime32Ex Win32 API with the maximum tolerable delay can be another alternative to reduce the frequency of periodic activities.

Another issue we found was related to network activity. Media playback was frequently doing updates on movie titles at idle. The platform power breakdown shows the media application has significant impact on the network component even when the application is at idle.

"The media application has significant impact on the network component even when the application is at idle."

SwitchInTime (s)	NewProcess	Cpu	NewThreadStack
6.623 126 695	Playback.exe (2112)	1	⊟ [Root]
			sechost.dll!OpenServiceA
			sechost.dll!ROpenServiceA
			rpcrt4.dll!NdrpSendReceive
			rpcrt4.dll!NdrSendReceive
			rpcrt4.dll!I_RpcSendReceive

Figure 7: Call stacks for Playback.exe
(Source: Data obtained by running Windows* Performance Analyzer on an application workload on Intel® Core™ i5 based system)

Figure 7 shows other issues causing frequent wakeups of the platform at application idle. The Playback application at idle is invoking frequent remote procedure calls (RPCs). The application wakes the CPU by frequently calling OpenServiceA and closeServiceA. Invoking RPCs causes an inter-process interrupt (IPI) storm, which is a problem because IPIs are used to implement inter-process communication. The recommended way is by using a simpler Win32 API instead of doing Windows Managed Instrumentation (WMI).

"Invoking RPCs causes an inter-process interrupt (IPI) storm, which is a problem because IPIs are used to implement inter-process communication. The recommended way is by using a simpler Win32 API instead of doing Windows Managed Instrumentation (WMI)."

As a result of these optimizations, we achieved power saving of 2.2 watts through software optimization and reduction in package C0 residency by 33.2 percent.

Real-World Applications Analysis on a Semi-idle System

Users install many applications on their systems during daily usage. During long PC usage these applications run in the background but only use a small percent of processing resources. Such applications, typically described as being in a *semi-idle state,* cause significant impact on platform performance and power due to frequent but low level activity.

"Encryption software running in the background wakes up at the sub-millisecond range, causing the CPU to come out of deep sleep states and affecting total platform power consumption."

To understand the impact of various applications running in the background, we collected hardware performance monitoring events on a normal user usage scenario (browsing, productivity, or writing email) for 15 minutes duration using VTune Amplifier. Later, we post-processed the collected data using Intel PBA. Figure 8 shows that encryption software running in the background wakes up at the sub-millisecond range, causing the CPU to come out of deep sleep states and affecting total platform power consumption.

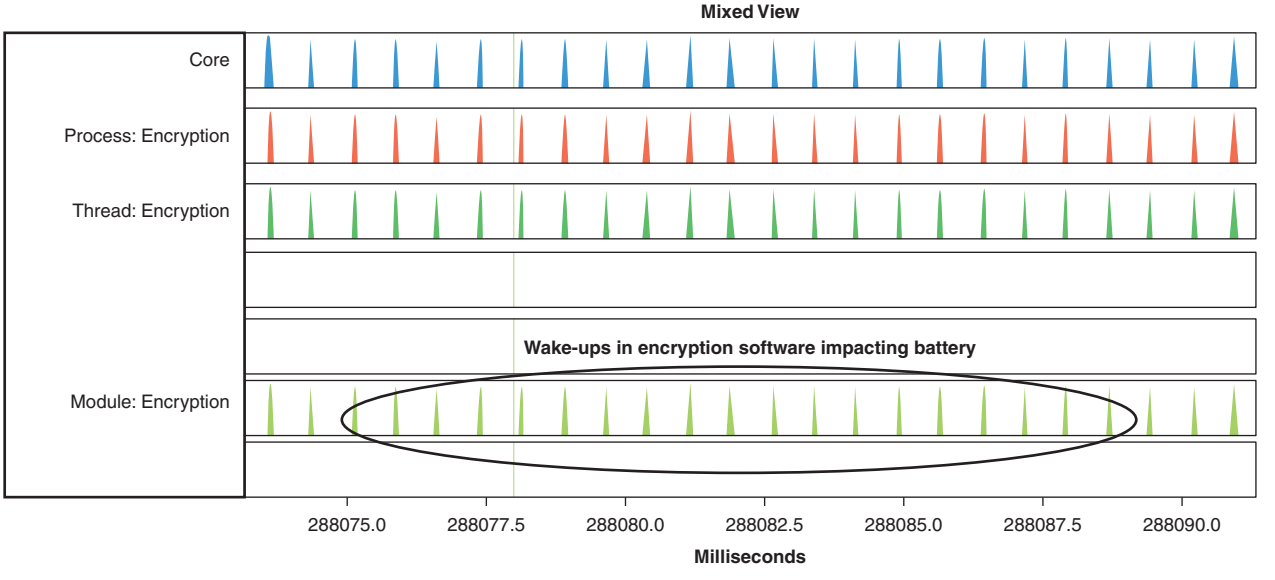


Figure 8: Overtime behavior of wakeups seen in encryption software
(Source: Intel® VTune™ Amplifier data post-processed with Intel® PBA on Intel® Core™ i7 system)

The X-axis shows the overtime value in milliseconds while the Y-axis shows the utilization for total core, process, modules, and functions. Since such activities are active for a short duration, it would be impossible to analyze these without looking at them over time. Aggregated time spent in such short activities is very low and hence won't show up in the top five modules.

"Activities that last for a very small duration and have high frequency have a huge impact on total platform power."

Activities that last for a very small duration and have high frequency have a huge impact on total platform power, because the system cannot be in deep C-state for longer duration. Such activities can be identified by using the analysis tools and optimized by using timer coalescing and by increasing the idle time between each wakeup. It is recommended always to avoid frequent short wakeups as much as possible and to use the largest granularity for sleep that the application can tolerate. Timer coalescing should be used where applicable as well.

Active Power Analysis

In many real-world usage scenarios, our laptops or tablets do some amount of work either in foreground or background. Hence software power efficiency in active state becomes very essential. In many cases, we have seen that in

performance optimizations, those that can help finish the task at hand faster can yield power savings as well. When analyzing software for active power efficiency, it is important to identify the basic performance characteristics of the application under analysis. Some of the key areas to look for include:

"When analyzing software for active power efficiency, it is important to identify the basic performance characteristics of the application under analysis."

- How much CPU is being used? (user and kernel mode)
- How many threads are actively doing work?
- Is the application disk bound?
- What is the CPU usage distribution across cores over time?
- What is the P- and C-state residency during the application runtime?
- What is the actual power for individual components using NetDAQ*? For package power, we now have energy MSRs introduced in Intel Core architecture (codenamed Sandy Bridge)

To demonstrate how such analysis and optimization for power can be achieved, we examine a case study for a security application. In this scenario, application developers reported an increased CPU usage issue (hence consuming more power) when the background application is installed.

Initial analysis with Windows Performance Monitor showed about 40 percent user mode time and about 15 percent kernel mode time (time spent in executing ring 0 operations). See figure 9 below. This data indicated an issue where a long time is spent in ring 0 compared to user space. Overall, combing these two usages, the operation took about 55 percent of the CPU with the background application installed (whereas without the application, the same operation only took about 30 percent of the CPU with less than 5 percent ring0 time).

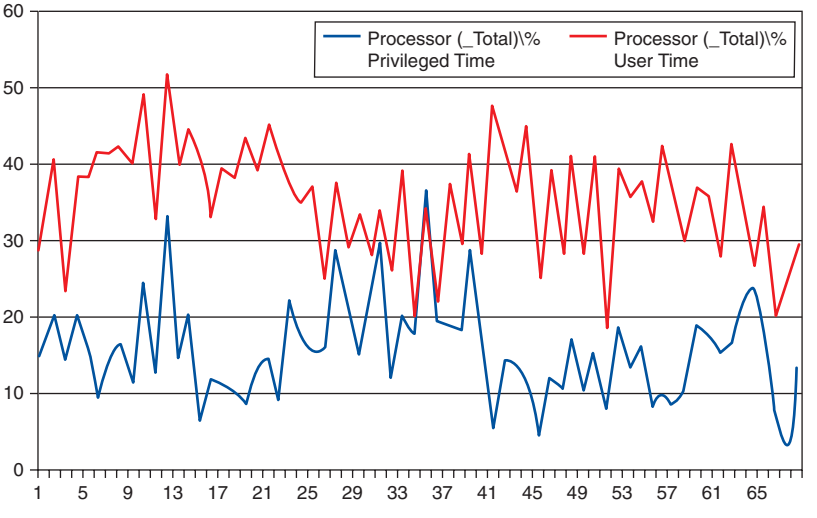


Figure 9: User mode and kernel mode CPU usage from Windows Performance Monitor
(Source: Data collected in lab using Windows Performance Monitor on Intel® Core™ i7 system)

While looking at Windows Performance Monitor data, along with high time in ring 0 mode, we observed that when the application is installed, a high interrupt rate is observed as well as shown in figure 10. The threshold for a high interrupt rate is ~5K/sec.

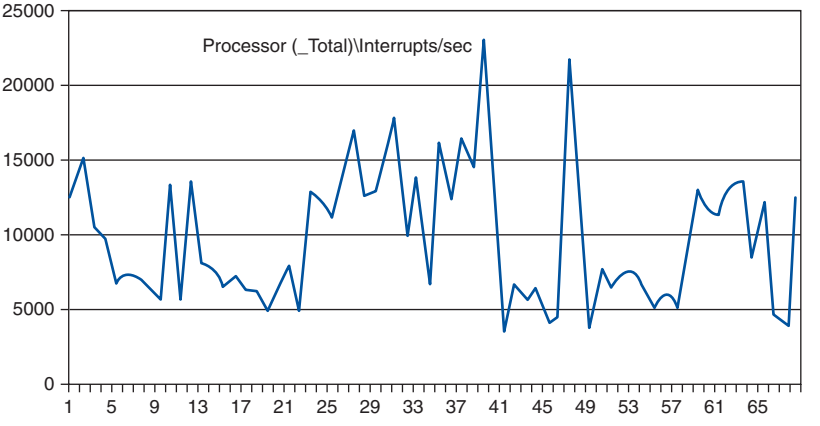


Figure 10: Interrupt rate per sec from Windows Perfmon (Source: Data collected in lab using Windows Performance Monitor on Intel® Core™ i7 system)

The next step was to identify the most time-consuming modules on the system. For that, further analysis with VTune Amplifier showed that the module with the highest CPU utilization was from the OS (X.dll and Y.dll were from the application).

Module_Name	Clockticks%	CPI
ntoskrnl.exe	25.05	1.59
X.dll	17.87	0.75
Y.dll	11.99	0.89
ntdll.dll	7.58	1.37
ntfs.sys	5.02	1.72
fltmgr.sys	3.43	1.84
msvcr90.dll	2.38	0.81

Table 3: Hot Modules from Intel® VTune™ Amplifier (Source: Data collected in lab using Intel® VTune™ Amplifier on Intel® Core™ i7 system)

As shown in Table 3, since a significant amount of time is spent in the Windows OS component, the next step would be to trace these calls back to application software.

Using XPerf, we were able to determine the root cause to be high time in kernel (stacks going to ntoskrnl.exe) along with a high interrupt rate observed.

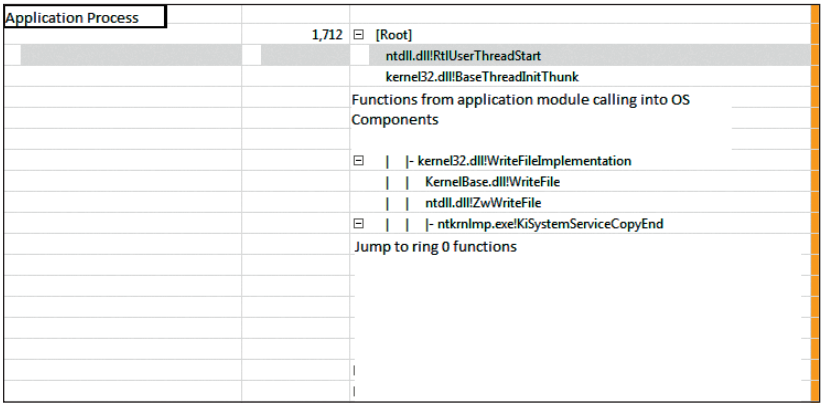


Figure 11: Call stacks from Application to OS (Source: Data collected in lab using Windows xPerf on Intel® Core™ i7 system)

As indicated in Figure 11, the application had a class that was eventually calling into the *WriteFile()* function, which jumps into the kernel. Discussion with application developers revealed that the writes weren't supposed to happen as frequently as seen in this data. This was then re-architected to take out frequent writes. As a result of this optimization, CPU usage went down to 35 percent total as compared to 55 percent seen before, saving power and freeing up CPU resources.

Hardware Offload (GPU) Graphics Decode/Encode

Some tasks in video transcode are handled more efficiently in hardware offloading of video rendering and encoding in terms of performance and power usage. In this section we look at a couple of applications and their power/performance impact using offload. The test system has an Intel HD graphics capability on a mobile/laptop platform. Two workloads used were encoding an H.264 video file and running an HTML5 test with hardware acceleration in the application and browser enabled and disabled. The baseline used was using software and the hardware offload shows good scaling in performance and power. Figure 12 shows the scaling over baseline, hence higher numbers mean better performance and lower power.

"Offloading certain graphics tasks to the GPU instead of the CPU may be beneficial to the overall user experience."

Offloading certain graphics tasks to the GPU instead of the CPU may be beneficial to the overall user experience. Intel Media SDK offers solutions for developers to offload the processing to hardware as applicable.

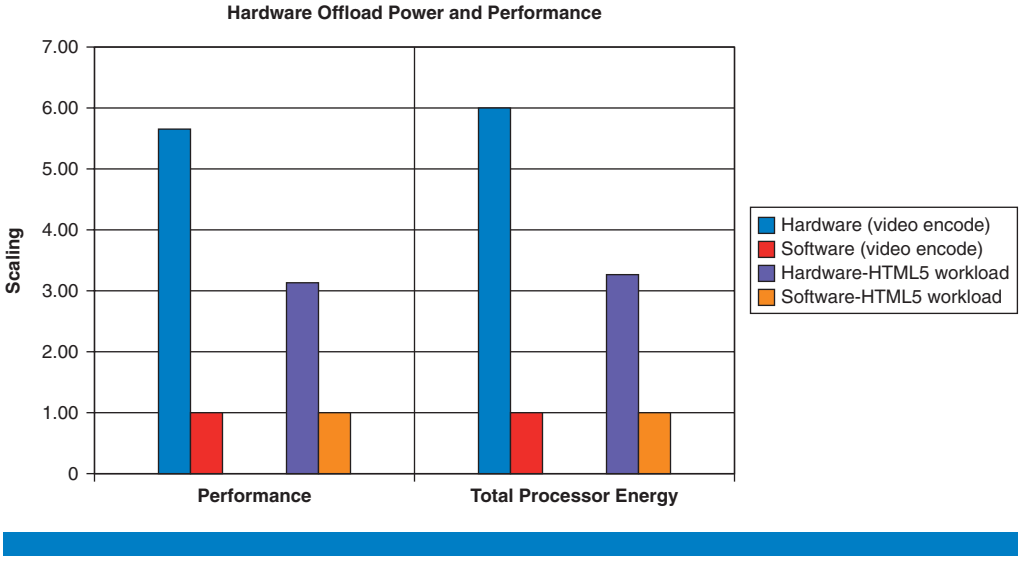


Figure 12: Power and energy comparison with and without hardware offload (Source: Data collected from Intel® Core™ i7 based system collected in lab)

Summary

"Software has a significant impact on system battery life even when hardware is optimized."

The battery life of a device is a key factor in user experience. If an end user has to carry the power adapter for portable devices everywhere or have to recharge their mobile device often then it significantly affects usability for that device. There are several components that affect the battery life of a device, from hardware components like the processor, display, and so on, to the software like the operating system, drivers, and applications. Software has a significant impact on system battery life even when hardware is optimized. As shown in the case studies, semi-idle systems with many preinstalled software applications are key usages for 95 percent of the users. Badly behaving software in that case can reduce the battery life in half if not properly optimized. Even if gains from software optimizations are limited to only a few minutes, each minute adds up to extend battery life and can extend the device use without the need to recharge. For both ultrabook and tablet devices, longer battery life is one of the essential factors in enhancing the user experience. By using the tools and methods described in this article, the software ecosystem can help achieve the goal of extended battery life and provide improved user experience with "green" computing.

Acknowledgements

The authors would like to thank the reviewers, Tom Huff and Sherine Abdelhak of Intel Corporation.

References

[1] Intel Corporation 2011, Intel optimization guide and developer manuals.
http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-3b-part-2-manual.html
http://www.intel.com/content/dam/doc/manual/64-ia-32-architectures-software-developer-system-programming-manual-325384.pdf

[2] Microsoft Windows Perfmon documentation. http://msdn.microsoft.com/en-us/library/aa645516(v=vs.71).aspx

[3] Microsoft XPerf Tool. http://msdn.microsoft.com/en-us/performance/cc709422

[4] Chabukswar Rajshree 2008, DVD Playback Power Consumption. http://software.intel.com/en-us/articles/dvd-playback-power-consumption-analysis/

[5] Chabukswar Rajshree 2008, Effect of multi-threading on power consumption. http://software.intel.com/en-us/articles/maximizing-power-savings-on-mobile-platforms/

[6] De Vega, Krishnan 2008, Disk I/O Power Consumption. http://softwarecommunity.intel.com/articles/eng/1091.htm

[7] De Vega, Chabukswar, 2008 WLAN Power Consumption. http://www3.intel.com/cd/ids/developer/asmo-na/eng/333927.htm

[8] Intel® Energy-Efficient Performance. http://www.intel.com/technology/eep/index.htm

[9] Krishnan Karthikeyan, 2008 Timer interrupt impact on CPU power. http://softwarecommunity.intel.com/articles/eng/1086.htm

[10] Intel® VTune™ Amplifier. http://software.intel.com/en-us/articles/intel-vtune-amplifier-xe/

[11] De Vega, Kim, Krishnan 2011 Intel® Power Gadget. http://software.intel.com/en-us/articles/intel-power-gadget/

[12] Intel® Performance Bottleneck Analyzer. http://software.intel.com/en-us/articles/intel-performance-bottleneck-analyzer/

[13] Intel® Media SDK. http://software.intel.com/en-us/articles/vcsource-tools-media-sdk/

Author Biographies

Jun De Vega is an Application Engineer in Intel's Software and Services Group, working on application power and performance optimization for Intel® architecture. He supports enabling of ISV applications on Intel® Mobile and Desktop Platforms.

Manuj Sabharwal is a Software Engineer in the Software Solutions Group at Intel. Manuj has been involved in exploring power enhancement opportunities for idle software workloads. He also works on enabling client platforms through software optimization techniques. Manuj holds a Master's degree in Computer Science from California State University, Sacramento and a Bachelor's in Computer Engineering from Visvesvaraya Technological University, India

Rajshree Chabukswar is a software engineer in Intel's Software solutions group. Rajshree has been with Intel for 10 years mainly focusing on power and performance optimizations on client software. Her primary focus has been Intel Core and Atom architectures for software optimizations. She holds a master's degree in computer engineering from Syracuse University and bachelor's degree in computer engineering from university of Mumbai, India

SLA-GUIDED ENERGY SAVINGS FOR ENTERPRISE SERVERS

Contributors

Vlasia Anagnostopoulou
University of California Santa Barbara

Martin Dimitrov
Intel Corporation

Kshitij A. Doshi
Software and Services Group,
Intel Corporation

"In continuous conversation with software, this solution puts into effect power consumption changes that converge software performance with that expected by an SLA."

Current hardware and operating system power management mechanisms for servers and data center machines employ control schemes that take little or no guidance from application software. On these machines, the hardware and OS take responsibility for controlling frequencies of cores according to indications of resource utilization. Sophisticated as these schemes are, they miss the cogent participation of application logic (such as transaction processing, media, business intelligence, and other services) in determining precisely how quickly programs must execute to meet service level agreements (SLAs) or when to defer optional activities. For decades now, server software designers have innovated and tuned their offerings for best performance, driven by a competitive landscape that rewarded highest performance with market success. In practice this has meant software architectures in which complex and compound purposes are achieved by concurrent progressions of computations, where external logic cannot slow an isolated resource without risking unintended consequences to widely mapped computations. We devised and prototyped one answer to this quandary: a solution we call an SLA Governor, that engages software in the management of its own power-performance tradeoffs. In continuous conversation with software, this solution puts into effect power consumption changes that converge software performance with that expected by an SLA. Compared to a well-tuned baseline system with a default OS–based power management, the SLA Governor provides 8.3 percent better energy savings for an online transaction processing workload and 7.3 percent better power normalized performance for the SPECpower benchmark. These gains are obtained on two recent generation machines with Intel E5-5500 (codename Nehalem) and Intel E5-5600 (codename Westmere) series processors, while executing the same load and maintaining the application's SLA.

Introduction

Fuelled by exponentially growing data volumes[13], enterprise systems face growing information processing rates and correspondingly escalating electrical energy demands for powering and cooling the computing hardware[6][26]. The resulting economical and ecological concerns have driven the topic of energy efficiency to the forefront of systems research. Mechanisms that drive hardware into more restful states under lighter load conditions are becoming commonly available in hardware, firmware, and operating system (OS) layers[7], and are complemented by practices such as concentrating work on fewer systems during periods of sub-peak demand[23].

With the exception of personal computing devices (such as laptops, tablets, handhelds) where exigencies of battery driven execution have driven OS and applications to work closely to save device power, the common power management approaches on most computers leave application software largely out of sway. One popular approach for power management[24] employs CPU utilization as a predictor of demand; the approach keeps applications agnostic about hardware but cannot take into account the effects on response time or service rate realized at the application level. Performance delivered at the granularity of a complex transaction depends not just on CPU usage but also on speeds of devices and on an application's ability to parallelize well. Relying entirely on processor utilization has other pitfalls: one is that far less than the ideal amount of energy can be saved when an OS policy is too conservative, and another is that a machine can fail to deliver SLA-mandated performance (since that metric is invisible to an OS-driven power manager). Vertically integrated power-performance management, such as in single purpose devices, is particularly rare in server systems, in line with a longstanding software engineering practice of layered abstraction in which runtimes present hardware capabilities in one standard way for applications to consume. Another reason why it is uncommon for application software to implement power control is that doing it well requires application logic to comprehend the shifting relationships between speeds of different hardware components and the performance it needs to realize. In the face of these engineering issues, an approach in which ongoing application level performance feedback is used to alter frequencies of hardware components presents a viable alternative for shaping power management decisions. In particular, when a service level agreement (SLA) sets in place an expectation for performance, and the actual performance is better than that mandated by the SLA, a governance mechanism can take advantage of the available slack to reduce power. The rate at which requests arrive at an application server is rarely fixed; thus, during intervals of low demand a server can conceivably meet an SLA expectation of some threshold response time or a threshold service rate with fewer CPUs, slower CPUs, or, fewer *and* slower CPUs. It could similarly employ much less memory capacity or bandwidth[18] when such reductions nevertheless maintain the application performance within the parameters of its SLA. Equally, when a server faces the prospect of exceeding thermal or power limits, application-guided power management can prioritize critical tasks over those that are deferrable, elect to trade performance for power savings, provide graduated quality of service, and so on, until normal operating conditions return. Many server applications, such as database management systems (DBMS), web applications servers, and Java virtual machines (JVMs), have self-regulating capabilities built into them. They commonly fine-tune their operations on the basis of various event metrics and diagnostics. Thus it is possible for them to recognize phases in their own execution and to identify and prioritize tasks upon which other tasks depend. Depending upon how an SLA is structured, the subset of requests that they choose to handle promptly can vary over time as well. The challenge that must be addressed is how to interlace the awareness

"During intervals of low demand a server can conceivably meet an SLA expectation of some threshold response time or a threshold service rate with fewer CPUs, slower CPUs, or, fewer and *slower CPUs."*

of hardware operating conditions with application-aligned performance requirements, and have applications guide resource scheduling without eroding device abstractions that an OS furnishes.

"In order to drive coordinated decisions across multiple layers of software without surrendering the longstanding benefits of hardware abstraction, it is critical that the software layers including the OS can interact through a middleware layer for power-performance adjustments."

In order to drive coordinated decisions across multiple layers of software without surrendering the longstanding benefits of hardware abstraction, it is critical that the software layers including the OS can interact through a middleware layer for power-performance adjustments. We introduce one such OS-agnostic liaison module in this article and describe how it preserves a separation of concerns between an enterprise application and the OS while making decisions that are fine-tuned and tailored to the specific needs of the application. The module is called *SLA Governor*. The SLA Governor exploits the slack between the SLA and the observed applications' performance to reduce system-level energy consumption by directing hardware course corrections.

For simplicity, in the first implementation of SLA Governor, just two types of course-corrective actions are fully implemented: it adjusts frequency of CPU cores and it activates and deactivates CPU cores. Further refinements are being added, as described in the "Future Work" section. We explore its effectiveness by conducting a study on two recent generation Intel platforms from the E5-5500 (previously codenamed Nehalem) and E5-5600 (previously codenamed Westmere) series, using two enterprise workloads for evaluation: (a) a time varying version of TPoX (an OLTP workload)[3] and (b) the SPECpower[2] benchmark. We use the default Linux OnDemand[24] power governor as the baseline in our evaluation. Compared with that baseline, the SLA Governor obtains 8.3 percent additional savings in system energy for the TPoX workload and 7.3 percent higher power normalized performance score for the SPECpower benchmark. This article makes the following contributions:

- We propose active participation in power management decisions from server applications and show how to achieve the communication that is needed.
- We develop an OS-independent, SLA-aware power governor that attempts to maintain an SLA while directing power or performance favoring course corrections.
- We evaluate its operation under two enterprise workloads at full system scale in two recent generation machines, and demonstrate 8.3 percent energy savings on TPOX (OLTP) and 7.3 percent improvement of the SPECpower score.

The rest of the article is organized as follows: in the next section we present related work, in the third section we provide background on current operating system and hardware power management approaches. In the section "SLA Governor: An SLA-Aware Energy Optimization Framework" we describe in detail the design and functionality of our governor. "Experimental Methodology" describes our experimental infrastructure and "Evaluation of the SLA Governor" presents results. This is followed by a section on future work and a conclusion.

Related Work

Service-level agreements (or other metrics of quality of service) have been widely explored as a mechanism to drive power management in various types of systems, from real-time[11][12][16][17][21][25] and web-servers[5], to parallel processing systems[19]. Flautner et al.[12] is such a state-of-the-art real-time system, where they extended the Linux kernel to delay task execution based on CPU utilization, while saving energy, subject to not delaying any task deadlines. The implementation of such a mechanism, however, is based upon a-priori knowledge of task executions (the deadlines are considered to be an SLA proxy), while our implementation is dynamic. Other approaches enhanced the SLA approach by including cache miss rates (or metrics to track memory behavior) and used them to build the statistical dependence between frequency operating points and memory performance and power consumption[11][21][25]. Our work is different in concept, since our power management decisions depend directly on the application's performance input and SLA, though our approach can be easily extended to also track hardware metrics (in addition to OS- and application-level metrics) and use them to fine-tune our SLA-driven algorithm.

Application-driven power management has also been explored as a mechanism for power management in real-time systems[14][20]. The closest to our work is the work of Hayamizu et al.[14], in which they implement SLA-based hardware correction. This approach results in significant performance oscillations and SLA violations at the application level, as it is not sufficiently nuanced or reactive in adjusting its policies. The resulting performance degradation in their work counteracts the value proposition of application and SLA-based power-performance management. By comparison, we pursue a software-based approach that allows more flexible and refined consideration of when and when not to exploit an SLA slack. In this article we delve into some of the subtleties of managing power-performance tradeoffs in practical scenarios. This article also explores the consolidation of threads to fewer cores in addition to frequency scaling. Lastly, this work builds on a short, earlier version of our work, Anagnostopoulou et al. [28]

Background

In this section we give an overview of power management controls that the Linux OS uses, and which we too employ in the SLA Governor. The *Advanced Configuration and Power Interface (ACPI)*[1] specification provides an open standard by which devices can be power managed. We exercise ACPI power states implemented in the Intel® Xeon® processors in our experimental setup using various OS commands described in this section. A summary of power governance schemes commonly furnished in most Linux distributions is also included in this section.

"P-states denote the different frequencies of operation and are often called active power states. Modern server processors incorporate a number of P-states."

Core P- and C-States

P-states denote the different frequencies of operation and are often called *active* power states. Modern server processors incorporate a number of P-states. For

instance, the Intel processors in our experiments have 10 different P-states, where in each P-state frequency and voltage is scaled together. Variation of power with frequency and voltage is given by:

$$P = \propto f * V^2 \quad (1)$$

"C-states denote different levels of restfulness in the micro architecture and are often called idle *power states. The processors we use support C1, C3, and C6 states for cores, with higher numbers denoting greater power savings."*

C-states denote different levels of restfulness in the micro architecture and are often called *idle* power states. The processors we use support C1, C3, and C6 states for cores, with higher numbers denoting greater power savings. Power consumption in the C6 state is extremely low, and thus there is strong motivation to transition to C6 whenever operating conditions permit; at the same time, transitioning back from C6 to an active state can incur significant latency penalties from having to reestablish contextual and cache state.

OS Interfaces

The Linux OS exposes various filesystem-style interfaces for active management of core power states. For managing P-states, Linux features the interface at */sys/devices/system/cpu*, described in Brodowski[8]. Using this interface, one can subscribe to a frequency as long as it is within the hardware's supported values. The SLA Governor uses an available system command in order to direct P states:

```
cpufreq-set -c [thr-id] -d [minfreq] -u [maxfreq]
```

Linux also exposes an interface[10] describing the idle states, but this interface does not allow the user to explicitly transition cores to C-states. Instead, another interface offered by Linux at */sys/devices/system/cpu/cpu[id]/online* allows one to add or remove cores (the operations are respectively called *onlining* and *offlining*) from the list of schedulable cores. On recent Linux versions (2.6.37 onward) we verified that offlined cores transition to the desired C6 (the lowest power) state. Unfortunately, some enterprise workloads rely on pinning their threads to certain cores and thus offlining cores with the above method created system instabilities. As an alternative approach, we use the taskset interface and migrate work away from the cores, which we want to idle. Then, we rely on the OS to transition the idle cores to C6 sleep state. The Linux command that we use is:

```
taskset -p [pid] -c [thread-list]
```

Linux Scheduler

The load balancer in the Linux scheduler features a power-preserving mode, where at times when the number of threads is small, the load balancer consolidates the threads onto fewer cores/packages so that the rest of the cores/packages can be idled[27]. This behavior changes when the number of runnable threads increases, and the balancer tries to balance the threads among the cores as much as possible (performance mode).

Idle Driver

The above behavior is exploited via a driver which transitions cores that have nothing runnable to idle states. The default driver in Linux performs these C-state transitions quickly[9].

OS Governors

Linux features a set of power governors, which are summarized in Table 1. Three of the governors, PowerSave, Performance, and Userspace, set the frequency of the processor to a fixed value. PowerSave and Performance set the processor frequency to the lowest/highest available value respectively. Userspace locks the processor frequency to a user-specified value. Once specified, the processor frequency remains fixed and no dynamic frequency scaling is performed. The OnDemand governor performs dynamic frequency scaling in response to variations in load (as estimated by CPU utilization). The OnDemand manager is the default Linux power manager, and the most flexible, which is why we use it as the baseline for comparison against the SLA Governor.

"The OnDemand manager is the default Linux power manager, and the most flexible, which is why we use it as the baseline for comparison against the SLA Governor."

Governor	Frequency setting
PowerSave	Lowest allowable
Performance	Highest allowable
OnDemand	Proportional to the CPU utilization, as long as the utilization has been over a user-defined upthreshold in the past iteration
Userspace	User defined

Table 1: Overview of OS Governors
(Source: Intel Corporation, 2012)

SLA Governor: An SLA-Aware Energy Optimization Framework

At a high level, the goal of the SLA Governor is to minimize the power consumption of the platform while shaping the performance of the application so that the application's SLA in not violated. In the following section, we describe in detail our implementation of the SLA Governor as well as some of the design principles that guided our decision making during the development of this framework.

"The goal of the SLA Governor is to minimize the power consumption of the platform while shaping the performance of the application so that the application's SLA in not violated."

Figure 1 shows a block diagram of the SLA Governor and its interaction with the system. The SLA Governor is a user-level daemon, which runs in the

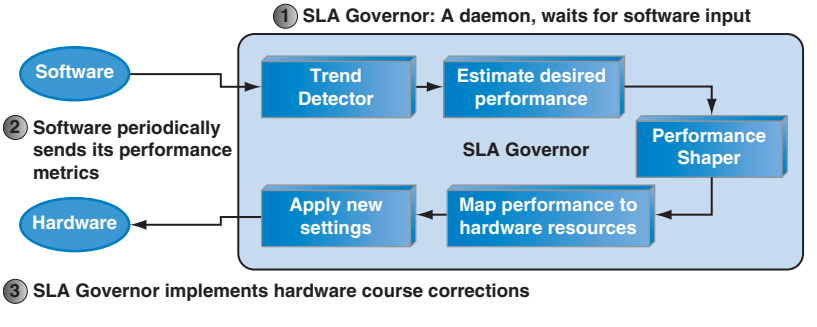


Figure 1: Architecture of the SLA Governor
(Source: Intel Corporation, 2012)

background and waits for input from the higher-level application by listening on a socket. The application periodically communicates its performance metrics to the SLA Governor. The SLA Governor processes the application input, makes power optimization decisions while trying to preserve the application's SLA, and then implements those decisions in the form of hardware course corrections. The SLA Governor consists of a number of different modules, which we describe in more detail.

The Trend Detector

This module processes a number of response-time samples over a moving window and smoothens sample-to-sample variations to identify the current performance trend. Its effectiveness is subject to two key parameters: the length of the window and the averaging method over the window; our experimentation suggested that either arithmetic mean or median taken over a moving window of between four and ten samples provided a good balance between responsiveness and noise reduction. Our findings are in line with the work from Andreolini and Casolari[4], where they examine different models for tracking application performance.

Estimation of Desired Performance

The processed performance data from the trend detector is used in order to compute a new desired performance delta, which brings application performance closer to the SLA without exceeding it. In practice, in order to allow the SLA Governor sufficient time to react to sudden increases in performance demand, we actually use an SLA target that is more conservative than the actual required application SLA by a small epsilon margin. This way, if the conservative SLA target is violated, the SLA Governor still has time to react and prevent violations to the actual application SLA.

The Performance Shaper

The performance shaper module is a collection of heuristics that adjust the new, desired performance delta, so that dramatic performance oscillations or unnecessary hardware course corrections are avoided. For example, if the desired performance delta is within a small fraction (such as 2 percent) of the targeted performance, then course corrections are avoided. On the other hand, large performance delta requests (such as 70 percent) are capped and enforced gradually. Additional heuristics are applied when the workload is to be consolidated to a different number of cores, due to the nonlinear performance effects, which we observed (see the section "Evaluation of the SLA Governor"). Some of those heuristics include providing a frequency boost to remaining cores during consolidation and using conservative triggers, which allow for core consolidation only when we are well below the required performance SLA.

Mapping of Performance to Hardware Resources

Once the new desired performance delta is determined, the SLA Governor needs to decide how to allocate hardware resources to the application so that it achieves its desired performance while minimizing the system's power

consumption. At this time, we only consider managing core P-states (frequency states) and C-states (sleep states). In order to understand how a desired performance delta is mapped to actual CPU resources, consider the following example. Consider a situation in which all the processor cores are currently running at 2.4 GHz and the desired performance delta is −10 percent (the application requires 10 percent less performance). We can now calculate that the application wants to run at 2.4 GHz − 0.1 × 2.4 GHz = 2.16 GHz. Since there may not be a 2.16 GHz frequency available on the machine, we select a combination of frequencies (such as 2.2 GHz on some cores and 2.1 GHz on other cores) so that we approximate this desired frequency as closely as possible. If all the cores are already running at the lowest possible frequency, we may choose to consolidate work to fewer cores. In this case, the mapping to hardware resources is performed similarly, with idle cores being represented by a frequency of 0 GHz. In the case of core consolidation, we may perform additional frequency adjustments due to the nonlinear effects on performance. Note that in the above example, we assume that adjusting the processor frequency will result in a proportional adjustment in observed application performance. This may not be true for some applications, especially if they are mostly memory- or I/O-bound applications. However, since the process is continuous and iterative, future adjustments requested by the application are likely to lead the SLA Governor to converge, as suggested by our experimental results. In addition to choosing frequencies and number of cores to match the desired application performance, the mapping function must also select the setting, which consumes the least amount of power. Because of performance issues related to core consolidation, we give priority to voltage/frequency scaling before attempting to perform core consolidation. Similarly, we choose configurations that raise processor frequency by a small amount on several processor cores over configurations that raise processor frequency by a large amount on few processor cores.

"Because of performance issues related to core consolidation, we give priority to voltage/frequency scaling before attempting to perform core consolidation."

Applying the New Settings

In this module, once the desired P- and C-state settings have been determined, we enforce them on the system.

Experimental Methodology

In this work, we experiment with two enterprise workloads, TPoX and a SPECpower-like workload. In the following section we describe our system and workload setup, as well as the low level instrumentation required for collecting power and performance metrics.

TPoX Setup

Transaction Processing over XML (TPoX) is an XML database benchmark based on a financial application scenario, such as a brokerage company for online trading of stocks[3]. The TPoX benchmark aims to evaluate the performance of XML-only databases, an important emerging use-case. TPoX generates significant load to all system components: CPU, memory, I/O, and

network, which is representative of many server workloads and desirable for the evaluation of system-level energy optimization approaches. In addition, TPoX is an open source benchmark, which allowed us to make some important modifications to the workload driver. We describe these next. TPoX is typically deployed as a two-tier workload, where a client machine is used to initiate a number of concurrent database requests, and a database server machine is used to service those requests. Unfortunately, the TPoX workload is a steady state workload by design, which means that the system load remains constant for the duration of the workload. In order to emulate a more realistic scenario, in which server load varies throughout the day, we modified the workload driver (executing on the client machine) to introduce variable think times between transaction requests. By controlling the think times between transactions, we were able to generate different system loads. We also modified the TPoX driver to ensure that the arrival time of transactions remains constant (and independent of the service rate at the server). In this way, we ensure that the same amount of work was completed in our baseline and in our SLA Governor experiments over the same period of time. In addition to modifying the transaction think times, we also modified the workload driver to open a socket connection to the SLA Governor and supply average transaction response times at intervals of 1 second. (Note, in this work we propose that the application, for example, a DBMS, communicate its performance with the SLA Governor. Unfortunately, we do not have source code access to the DBMS and thus retrieve the performance statistics from the workload driver).

"We ensure that the same amount of work was completed in our baseline and in our SLA Governor experiments over the same period of time."

After applying the necessary changes to the workload driver, we set up the workload as follows. For the database server we use SUSE* Enterprise Linux 11.1 with upgraded kernel version 2.6.37, due to its tickless scheduler and advanced capabilities of transitioning the processors into low power sleep states (C-states). In addition, we patched the kernel to prevent processor C-state autodemotion (with autodemotion the processor may ignore OS requests to enter deep C-states). This patch ensures that an idle or an offlined core will correctly transition to the deepest sleep state. As a database management system (DBMS), we use IBM's DB2* Enterprise Edition version 9.7. For the TPoX database we use the *small-scale* database, which is 100 GB of XML files. As our hardware platform, we use an Intel Server Platform with two Intel Xeon X5680 (codename Westmere EP) processors at 3.33 GHz. Each processor has 6 cores with 2 threads per core (with multithreading), for a total of 24 hardware threads. The system is populated with 48 GB of memory. In order to support the high I/O demands of the workload, we use 3 × 160-GB direct-attached solid state drives (SSDs), configured in a software RAID 0 for holding the database tables and indexes. In addition, we use 1 × 64-GB SSD for holding the database log. The database server was connected to a client platform (workload driver) over a 1 Gb/s Ethernet connection.

SPECpower Setup: The SPECpower

The SPECpower workload was specifically designed to evaluate the energy efficiency characteristics of server platforms[2], and is used extensively in the industry as the de-facto energy efficiency benchmark. SPECpower evaluates the

performance of server-side Java and exercises mainly the CPUs and memory. SPECpower is designed to exercise the system at different load levels, with the default workload stressing the system from 100 percent load all the way down to idle at staircase intervals of 10 percent. In addition, SPECpower has configuration files, which allow the user to easily modify the load line and create different load patterns. Note that the SPECpower runs that we perform are not compliant with the rules for publishing the results, because we do not comply with the rules regarding the type of power meters to be used, the number of load calibrations phases, and an idle phase. Therefore, we refer to our workload as SPECpower-like.

SPECpower is a throughput-oriented workload, in which a number of client requests are bundled together in batches and the workload driver tries to execute a certain number of batches for a given time interval in order to achieve the desired load level. The response time of individual batches does not matter and may vary significantly, as long as the throughput requirements are met. Similarly to our TPoX setup, we opened a socket connection from the SPECpower driver to the SLA Governor and used it to communicate the benchmarks observed performance at one-second intervals.

The SPECpower server is configured with SUSE Enerprise Linux with upgraded kernel 2.6.37 and OpenJDK 64-bit Server VM Java 1.6.0-20. We configure both the server and the client emulator/workload driver to be on the same system. For our hardware setup, we use an Intel Server Platform with two Intel Xeon X5560 (codename Nehalem EP) processors at 2.8 GHz. Each processor has 4 dual threaded cores, for a total of 16 hardware threads. The system is populated with 24 GB of memory and an SSD drive for I/O storage. We purposely use different platforms, with different generation processors, for our TPoX and SPECpower experiments, because this allows us to gain additional confidence in the generality of our experimental results.

Low-Level Instrumentation

During workload execution, we configured our systems to collect a number of performance and power metrics. Both machines collect total system power. The SPECpower server collects total system (at the wall) power using a WT210 Yokogawa power meter. The TPoX server collects total system power using instrumented power supplies and a power management engine called Intel® Intelligent Power Node Manager[15]. This engine enables both total system power measurement and management capabilities, however, in our setup, we only use the measurement capability. In addition, in both systems, we developed scripts to accumulate statistics about the P- and C-states residencies of the processor cores.

In addition to the basic statistics collected at both systems, the TPoX server was instrumented to collect detailed power samples of CPU power (for each socket) and DRAM power (for two of the memory DIMMs). We empirically verified that due to memory address interleaving, memory DIMMs are usually exercised and thus we could extrapolate total DRAM power based on the two DIMMs that we measure.

"The TPoX server was instrumented to collect detailed power samples of CPU power (for each socket) and DRAM power (for two of the memory DIMMs)."

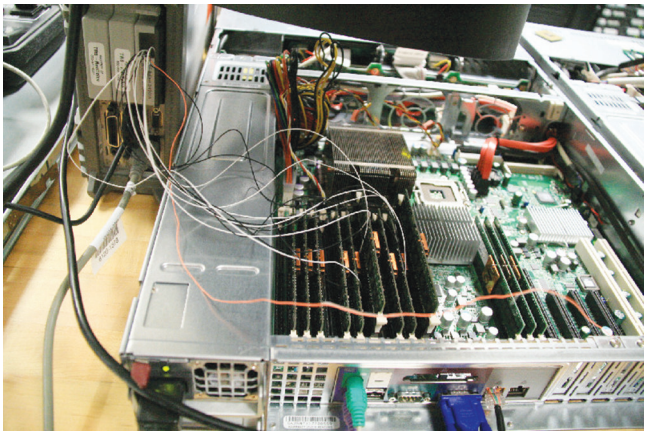


Figure 2: Instrumented motherboard, with probes connected to the Agilent* DAQ
(Source: Intel Corporation, 2012)

"To measure DRAM power, we use riser cards, which separate the memory DIMM from the board and include sense resistors for capturing voltage and current."

"To measure processor power, we modified the motherboard and placed sense resistors under key VRMs in order to measure processor power at the voltage regulator modules (VRMs)."

Figure 2 shows our instrumented server and the probes/wires sensing CPU and DRAM current and voltage. The probes are being aggregated using an Agilent Data Acquisition system. To measure DRAM power, we use riser cards, which separate the memory DIMM from the board and include sense resistors for capturing voltage and current. Measuring processor power directly is more challenging, because the processor has multiple power sources, supplied through many individual pins. Power to those pins is supplied by voltage regulator modules (VRM). The purpose of the VRMs is to provide a specific output voltage derived from the +12V system power plane. Thus, to measure processor power, we modified the motherboard and placed sense resistors under key VRMs in order to measure processor power at the voltage regulator modules (VRMs). We also used an internal tool on the TPoX server to collect detailed microarchitectural events, such as cache misses/hits, TLB, branch predictor, and so on.

Evaluation of the SLA Governor

In this section, we quantify the energy savings achieved by incorporating application-level metrics (in our SLA Governor approach) against the savings of an OS-managed approach (baseline). We also quantify the effects on Service Level Agreement violations. In both TPoX and SPECpower, the duration of the experiment and the amount of work completed are the same, as described in the previous section. We first look at the energy savings achieved when frequency scaling is employed. We evaluate the impact of additional core consolidation in the SLA Governor in the section "Extending SLA Governor with Core Consolidation."

TPoX Experimental Results

In this section we present out findings for the TPoX workload.

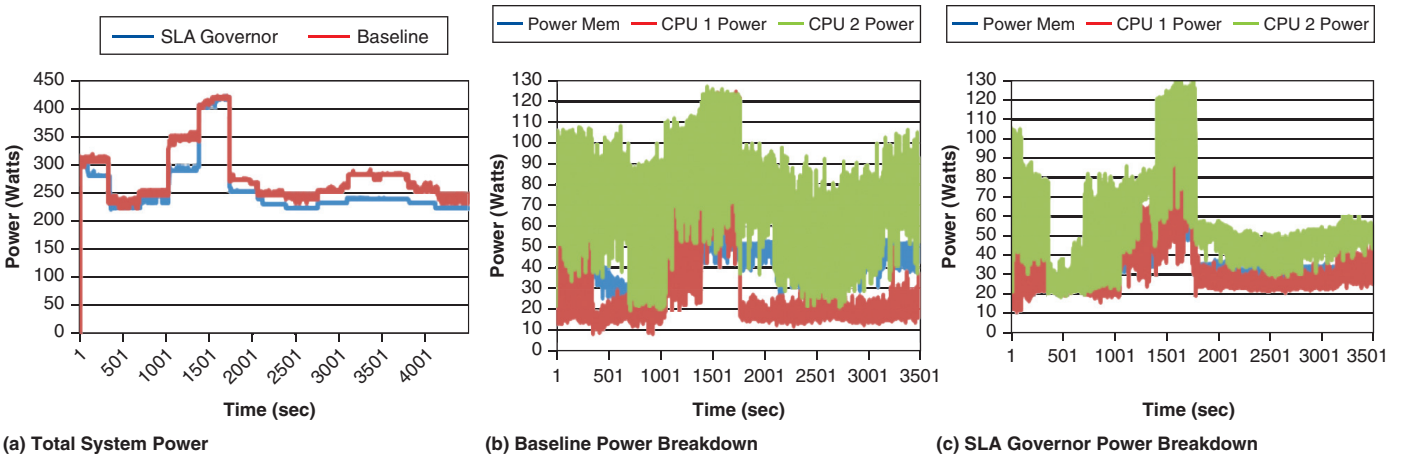


Figure 3: TPoX system power and power breakdown for the baseline and the SLA Governor (Source: Intel Corporation, 2012)

TPoX Energy Savings

Figures 3(a) through 3(c) show the total system power (at the wall) over the duration of the TPoX run for both baseline and the SLA Governor. Overall, we achieve 8.3 percent of system-level energy savings over the duration of this run, while maintaining or improving the SLA. In addition, we also show the power breakdown among system components: CPU per socket power and memory power. Comparing Figure 3(b) baseline with Figure 3(c) SLA Governor, we can observe even more significant energy reduction at the component level. Overall, we achieve 11 percent reduction in CPU energy and 21 percent reduction in memory energy. Here, we would like to highlight that these energy savings are observed on what is already a well-tuned baseline system with a tickless kernel. For instance, the cores in the baseline/SLA Governor spent only 41 percent/42 percent respectively of the execution time in C0 (or active execution state) and were at C1/C3/C6 sleep states for the remaining 59 percent/58 percent of the time. Since frequency scaling only reduces active power (during C0), it is beneficial only for about 40 percent of time in this experiment.

"We achieve 8.3 percent of system-level energy savings over the duration of this run, while maintaining or improving the SLA."

"We achieve 11 percent reduction in CPU energy and 21 percent reduction in memory energy."

Interestingly, the memory savings achieved are the deepest among all other parts of the system, even though we do not directly control the memory low-power states. While further analysis is needed, we believe that the memory energy savings are a secondary effect, due to the decreased memory activity levels at lower CPU frequencies, which allow for memory DIMMs to transition to low-power states more often. This intuition has been explored in depth by Fan et al.[11]

In order to gain a deeper understanding of how the SLA Governor is able to achieve such energy savings, we collected CPU P/C-state residency statistics, as well as system utilization statistics. From Figure 4, we can see that CPU utilization under the SLA Governor (top curve) is generally higher compared

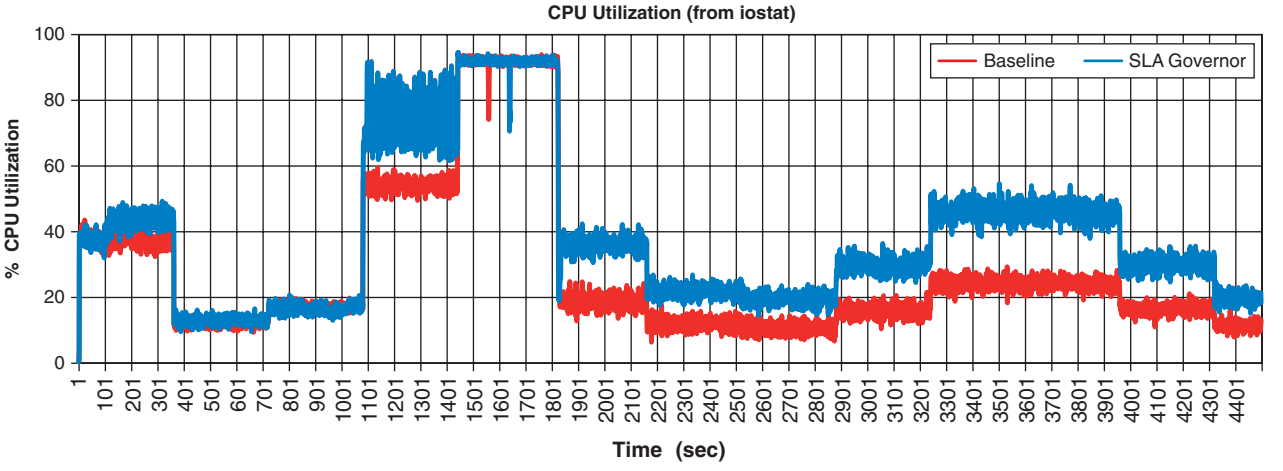


Figure 4: TPoX CPU utilization
(Source: Intel Corporation, 2012)

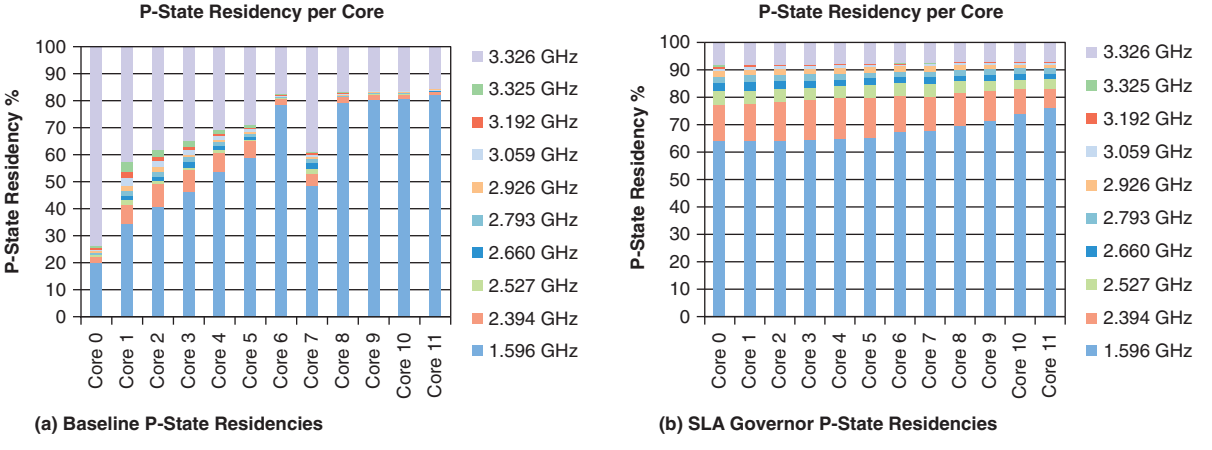


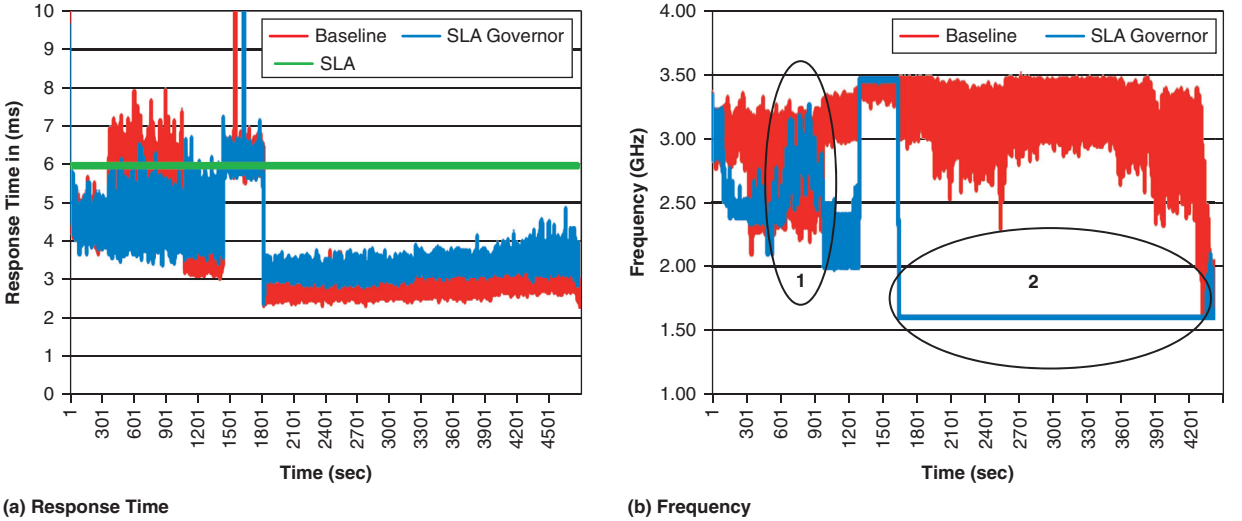
Figure 5: TPoX P-state residencies
(Source: Intel Corporation, 2012)

to the baseline (bottom curve). This is because under the SLA Governor, we force the cores to execute at lower frequencies more often, as we show in Figures 5(a) and 5(b). These figures show the percentage of execution time that each core spends at a given P-state (frequency state), for the baseline and the SLA Governor (respectively). We can see that under the SLA Governor, the cores spend significantly lower fraction of their time, only about 5 percent, in the highest CPU frequency (3.3 GHz) as compared to upwards of 20 percent in the baseline case. In Figure 5(a), we can also observe that P-state residency is nonuniform among the processor cores; that is, some cores spend more time in higher frequencies than others. Our per-socket power measurements correlate with this observation, since we can observe that Socket 0 power is higher than Socket 1 power in Figures 3(b) and 3(c). This behavior is due to the Linux

scheduler, which tries to consolidate work to the cores of one of the two sockets, in the case of a two-socket platform, in order to allow for the cores in the other socket to transition to the deepest sleep state. Comparing the C-state residencies between the baseline and the SLA Governor, we observed that cores spend approximately the same amount of time in deep sleep states (C3 and C6), with the baseline spending slightly more time in C3/6 on some occasions, due to executing generally at higher frequencies, that is, rush to sleep.

TPoX Response Time and SLA

In this section, we look at how well the baseline and the SLA Governor performed with respect to response time and with satisfying the service level agreement. We combine the response time of individual TPoX transactions into a single average response time (using a harmonic mean). We also configured the SLA Governor to maintain a service level agreement on the response time of 6 milliseconds or less. Note that it is the average response time over the one-second sample intervals that we evaluate against the SLA, and not individual transaction times. The 6-millisecond limit on the response time was selected such that, at very high CPU utilizations (above 90 percent), the server will start to violate the response time agreement even if all the cores are running at the highest frequency, that is, we start to exceed the capacity of the server. Note that the baseline OnDemand governor is not aware of the service level agreement of the application and only observes CPU utilization. Our experimental results are shown in Figure 6(a) and in Figure 6(b). From Figure 6(a) we can see that the SLA Governor is very competitive in terms of maintaining the service level agreement on response time. In particular, the SLA Governor incurred 291 SLA violations (average response time over one-second interval was above 6 milliseconds) compared to 479 SLA violations in the baseline case, 39 percent fewer violations. The reason is that the SLA



(a) Response Time

(b) Frequency

Figure 6: TPoX response time and frequency (Source: Intel Corporation, 2012)

Governor is strictly guided by the demands of the application and not CPU utilization. Figure 6(b) compares the baseline and SLA Governor average CPU frequencies over the duration of the TPoX run. From Figure 6(b) we can see that the SLA Governor maintains high CPU frequency when there is pressure on response time (area 1 in the figure) and maintains very low CPU frequency when response time is significantly below the service level agreement (area 2 in the figure). Interestingly, during the initial phases of the TPoX run, the workload is mostly I/O bound and has relatively low CPU utilization, as we can cross-reference with Figure 6(a). Since the SLA Governor is guided by the application demands and not CPU utilization, it maintains a high frequency and is able to avoid many service level agreement violations compared to the baseline.

SPECpower Experimental Results

In this section we present our experimental results for the SPECpower workload. As mentioned earlier, the SPECpower workload was designed to evaluate the energy efficiency of a server system. The workload has 10 different load levels with the requirement that a certain number of transaction batches are executed during each load level. By analyzing the response-time behavior of the workload at the different load levels, we set an SLA target of 100-millisecond response time on transaction batches. The SPECpower workload driver obtains the power readings from the Yokogawa power meter and computes the overall performance/watt score for the benchmark.

"The spiky load line is more challenging to the SLA Governor, since it is more difficult for the trend detector to quickly establish the required performance level of the application."

In Figures 7(a) through 7(d), we present the SPECpower generated statistics comparing baseline to SLA Governor for two different load lines, staircase and spiky, respectively. The horizontal bars correspond to the performance/watt number at a given load level. The line on the chart corresponds to the average active power for the given load level. Notice that the spiky load line is more challenging to the SLA Governor, since it is more difficult for the trend detector to quickly establish the required performance level of the application. While this workload was executed on a different machine than the TPoX workload, we observe similar trends in terms of utilization at low-frequency states: see Figure 8(a) and Figure 8(b). From these two figures we can see that the SLA governor takes much better advantage of the wide spectrum of P-states available on the machine, while the OnDemand governor tends to choose only between the highest and the lowest P-state. Interestingly, the response-time behavior of SPECpower transaction batches was significantly different that that of TPoX transactions. In particular, in SPECpower most transaction batches complete in under 100 milliseconds; however, even under light loads we observe some outlier transaction batches taking as much as two seconds to complete. In order to filter out those outliers and prevent unnecessary hardware corrections at the SLA Governor, we employed a different filter in the trend detector, which uses a median instead of a mean average smoothing function.

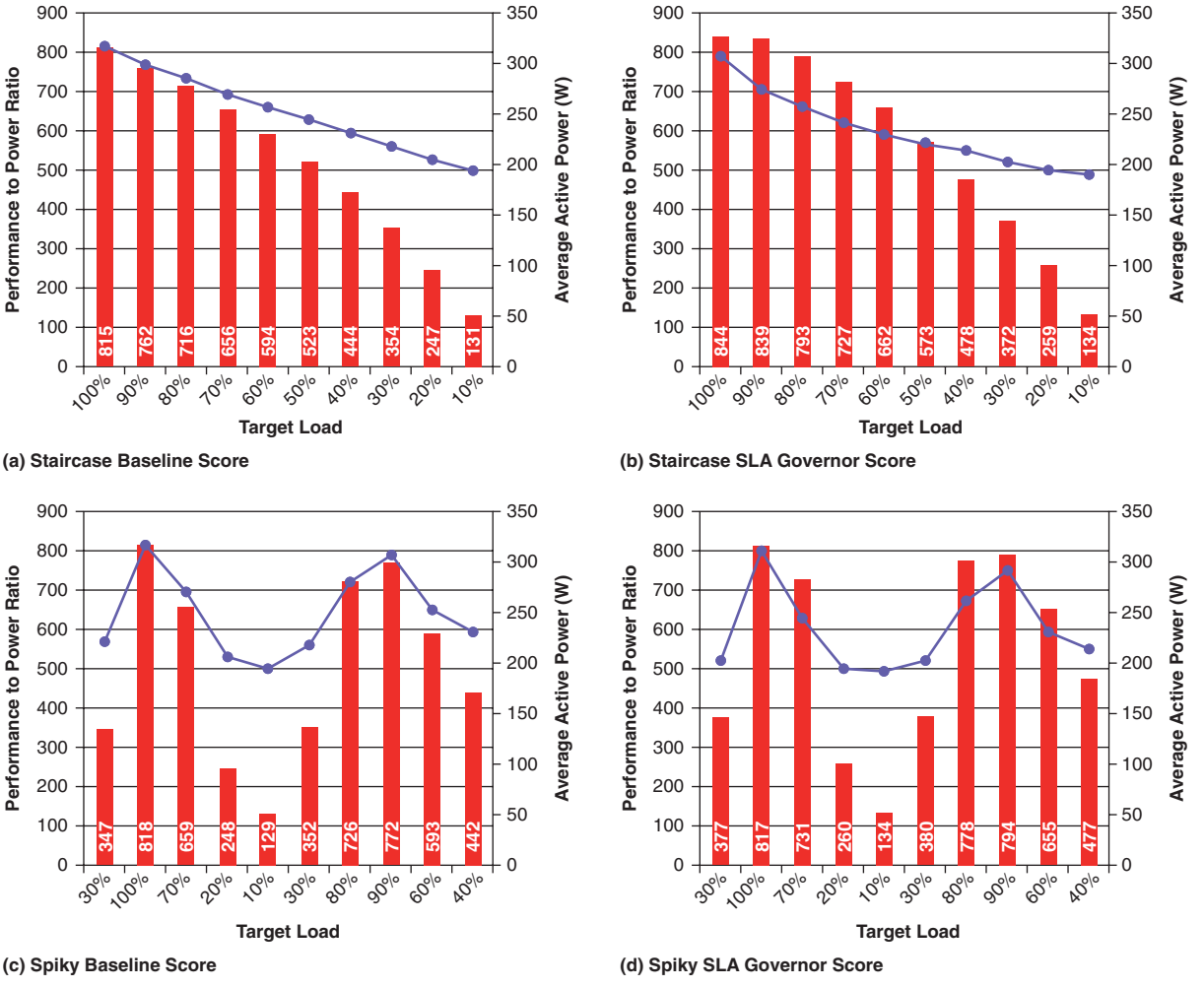


Figure 7: SPECpower for the staircase and the spiky workloads
(Source: Intel Corporation, 2012)

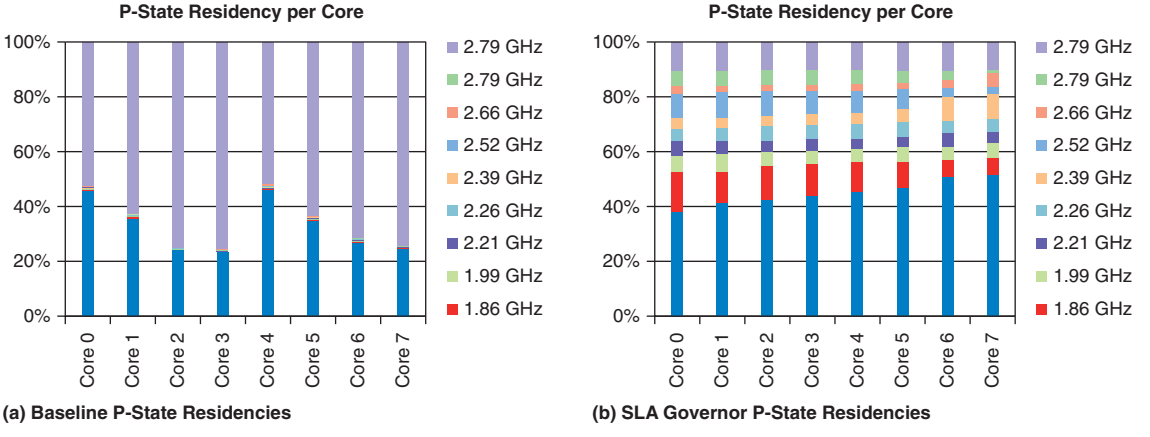


Figure 8: P-state residencies for the staircase workload
(Source: Intel Corporation, 2012)

"The performance impact of core consolidation on application performance is much more difficult to predict, due to latencies related to exiting sleep states of idle cores, caching effects, and so on."

Extending SLA Governor with Core Consolidation

In addition to scaling processor frequency, we also extended the SLA Governor to consolidate work on fewer cores, thus allowing the idle cores to transition to deep sleep (C3/6 states). Unfortunately, the performance impact of core consolidation on application performance is much more difficult to predict, due to latencies related to exiting sleep states of idle cores, caching effects, and so on. For instance, we observed temporary performance spikes in TPoX both while reducing and increasing the number of available cores to the application. Further analysis using machine status registers (MSRs) revealed increased application CPI, cache miss rates, and branch miss-prediction rates compared to non-consolidation. Our results indicated that some of the processor cores are indeed spending more time in sleep states C3/6 due to consolidation. However, when these cores need to be woken up, we incur the penalties of sleep exit latency and cold caches. In order to mitigate the performance spikes with core consolidation, we modified the SLA Governor to provide temporary frequency boosts to the remaining cores. Using the temporary frequency boosts, we were able to alleviate the performance and SLA penalties, but we lost the additional power savings of core consolidation. Another reason why our core consolidation approach did not lead to additional energy savings is that the OS and processor were already quite aggressive in transitioning cores to sleep states at a much finer granularity than the control of the SLA Governor. As mentioned earlier, the cores were already spending on average 59 percent of their time in C1/C3/C6 sleep states. Although our initial results with core consolidation did not yield additional energy improvement, we plan to revisit this technique in the context of multiple concurrently executing workloads and virtualization.

Future Work

We designed the SLA Governor to be flexible and extensible. The first implementation used in evaluations described here implemented just P- and C- state course adjustments. New capabilities are being added in a staged manner. The next steps will improve and extend the SLA Governor in several directions. First, we will improve the ability of the governor to react to rapid changes in workload demand without violating the SLA. We plan to achieve this by including additional application information, such as transaction arrival rates. Second, with the incorporation of transaction arrival rates, we will add proactive P- and C-state corrections instead of waiting until their effects are manifest in response times. Third, we will extend the SLA Governor to manage all system resources in a coordinated manner, and not just CPU. In particular, we plan to start managing DRAM power states by directing memory controller policies to alter clock speeds and rank-idle thresholds. This will require online machine learning and modeling application performance in terms of memory and CPU power states. Fourth, we will use feedback from hardware in selecting the lowest power configuration among several possible alternatives. Fifth, we plan to extend the SLA Governor to operate in the context of multiple concurrent applications and virtual machines. Over a longer time span, we will

allow an application to receive callbacks through which power and thermal saturation conditions can be signaled, so that a capable application can act to defer or deprioritize nonessential activities.

Conclusion

In this article, we motivated engaging applications actively to the management of their energy footprint, by implementing an SLA-aware power management governor. Compared to a well-tuned, state-of-the-art Linux power management governor, we showed that SLA-aware power management, outside the OS and with interaction with the application, can achieve energy savings up to 8.3 percent for TPoX without any degradation in performance, and 7.3 percent better power normalized performance for SPECpower. In addition, we showed that our governor improves the efficiency of the system for both benchmarks. All our experiments were implemented on enterprise server systems, and using an extensive measurement infrastructure throughout the system stack, we were able to provide not only with important insights and tradeoffs regarding our governor mechanism, but also with general insights for power management in server systems.

References

[1] ACPI website. http://www.acpi.info.

[2] SPECpower workload. http://www.spec.org/power-ssj2008, 2011.

[3] XML database benchmark: Transaction processing over xml (tpox). http://tpox.sourceforge.net, 2011.

[4] M. Andreolini and S. Casolari, "Load prediction models in web-based systems," in *Proceedings of the 1st international conference on Performance evaluation methodologies and tools*, 2006.

[5] V. S. Arun et al., "Power-aware qos management in web servers," in *Proceedings of the 24th IEEE Real-Time systems Symposium, RTSS'03, Cancun*, 2003.

[6] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, 40:33–37, December 2007.

[7] D. Brodowski, "Cpufreq governors – information for users and developers." http://www.mjmwired.net/kernel/Documentation/cpufreq/governors.txt.

[8] D. Brodowski, "Linux kernel cpufreq subsystem." http://www.kernel.org/pub/linux/utils/kernel/cpufreq/cpufreq-set.html.

[9] L. Brown, "Saving energy with intel-idle cpuidle driver." http://video.linux.com/video/1788, 2010.

[10] J. Corbet, "The cpuidle subsystem," online article at LWN.net. http://lwn.net/Articles/384146/, 2010.

[11] X. Fan et al., "The synergy between power-aware memory systems and processor voltage scaling," in *Workshop on Power-Aware Computing Systems*, pages 164–179, 2002.

[12] K. Flautner and T. Mudge, "Vertigo: automatic performance-setting for linux," *SIGOPS Oper. Syst. Rev.*, 36(SI): 105–116, 2002.

[13] J. Gantz et al., "The expanding digital universe", white paper sponsored by EMC, 2007.

[14] Y. Hayamizu et al., "Application-aware power saving for on-line transaction processing using dynamic voltage and frequency scaling in a multicore environment," in *Proceedings of the 24th international conference on Architecture of computing systems*, ARCS'11, Berlin, Heidelberg, 2011. Springer-Verlag.

[15] Intel, "Intel® Intelligent Power Node Manager," White paper at: http://www.intel.com/Assets/PDF/Node-Manager-A-Dynamic-Approach-To-Managing-Power-In-The-Data-Center.PDF.

[16] C. Isci et al., "Live, runtime phase monitoring and prediction on real systems with application to dynamic power management," in *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 39, pages 359–370, Washington, DC, USA, 2006.

[17] W. Kim et al., "Performance comparison of dynamic voltage scaling algorithms for hard real-time systems," in *Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium*, RTAS '02, Washington, DC, USA, 2002.

[18] K. Kumar et al., "Memory energy management for an enterprise decision support system," in *Proceedings of the 17th IEEE/ACM international symposium on Low-power electronics and design*, ISLPED '11, Piscataway, NJ, USA, 2011.

[19] J. Li and J. Martinez, "Dynamic power-performance adaptation of parallel computation on chip multiprocessors," in *Proceedings of the International Symposium on High-Performance Computer Architecture*, 2006.

[20] X. Liu et al., "Chameleon: application level power management with performance isolation," in *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTI-MEDIA '05, New York, NY, USA, 2005.

[21] A. Miyoshi et al., "Critical power slope: Understanding the runtime effects of frequency scaling," in *Proceedings of the 16th Annual ACM International Conference on Supercomputing*, 2002.

[22] R. Nathuji and K. Schwan, "Virtualpower: coordinated power management in virtualized enterprise systems," in *Proceedings of*

twenty-first ACM SIGOPS symposium on Operating systems principles, SOSP '07, New York, NY, USA, 2007. ACM.

[23] M. Otagiri and M. Kutami, "Trends of energy saving in data centers and fujitsu group's approach," *Fujitsu Scientific and Technical Journal*, 2010.

[24] V. Pallipadi and A. Starikovskiy, "The ondemand governor, past, present, and future," in *Ottawa Linux Symposium*, 2006.

[25] C. Poellabauer et al., "Feedback-based dynamic voltage and frequency scaling for memory-bound real-time applications," in *Proceedings of the 11th IEEE Real Time on Embedded Technology and Applications Symposium*, pages 234–243, Washington, DC, USA, 2005.

[26] M. Poess and R. O. Nambiar, "Energy cost, the key challenge of today's data centers: a power consumption analysis of tpc-c results," *Proc. VLDB Endow*. 1:1229–1240, August 2008.

[27] S. Siddha, "Multi-core and linux kernel," white paper sponsored by Intel Corporation, 2005.

[28] V. Anagnostopoulou et al., "SLA-Guided Energy Savings for Enterprise Servers," Short paper to appear in *Proceedings of 2012 IEEE International Symposium on Performance Analysis of Systems and Software*, ISPASS'12, New Brunswick, NJ, 2012.

Author Biographies

Vlasia Anagnostopoulou is a graduate student at the University of California, Santa Barbara. Her research interests include the energy efficiency of large-scale enterprise systems. She interned with Intel's Software and Services Group in 2011. Contact: vlasia@cs.ucsb.edu

Dr. Martin Dimitrov obtained his B.S. degree in computer science from Bethune–Cookman College in 2004 and his PhD in computer science from the University of Central Florida in 2010. Martin joined Intel in 2010 as a systems engineer. Currently, Martin works in enabling and optimizing enterprise software for Intel server platforms. In addition, Martin is an active researcher in the Greenpoint initiative, which aims at optimizing system energy through collaborative software-hardware approaches. Contact: martin.p.dimitrov@intel.com

Dr. Kshitij A. Doshi is a principal engineer in the Software and Services Group at Intel Corporation. He has a bachelor of technology degree in electrical engineering from the Indian Institute of Technology (Mumbai) and a master's degree and PhD in computer engineering from Rice University. His research interests span operating systems, optimization of performance, power, and energy in enterprise solutions, database architectures, and virtual machines. Contact: kshitij.a.doshi@intel.com

ENERGY ADAPTATION FOR MULTITIERED DATA CENTER APPLICATIONS

Contributors

Muthukumar Murugan
University of Minnesota

Krishna Kant
Intel Research

David H.C. Du
University of Minnesota

"The key premise of EAC is to right-size the design of data center energy infrastructure and to provide smart control."

"Energy Adaptive Computing and its efficient implementation in data centers hosting multitiered applications."

Data center equipment, including the power infrastructure, is typically overdesigned in order to provide guaranteed service under stress conditions. In this article, we investigate the efficient implementation of a new paradigm called Energy Adaptive Computing (EAC) in data centers that host multitiered web applications. The key premise of EAC is to right-size the design of data center energy infrastructure (this includes power supply, power delivery and distribution infrastructure, and cooling infrastructure) and to provide smart control in order to cope with short-term deficiencies in available or consumable power. The possibility of deficiency in available power admits the use of variable power sources such as renewable sources, and deficiency in consumable power admits free cooling and other undersized cooling alternatives. We discuss coordinated control operations across tiers and at different time granularities in order to provide a dynamic allocation of power where it is most needed in order to satisfy the given QoS constraints. We present detailed algorithms and experimental results to show that the proposed technique can significantly reduce the probability of violating the QoS constraints and yet results in substantial savings in power. When the QoS constraints cannot be met, the proposed technique can gracefully degrade the QoS in order to provide the best effort service within the prevailing energy bounds. We demonstrate that the proposed technique can save power in the networking and storage subsystems as well. We also show how the rescheduling of background operations can further help in coping with the supply variations when a significant portion of the energy is drawn from local renewable sources.

Introduction

Large scale data centers that provide Internet services to geographically dispersed users are increasing in both number and popularity. The quality of service in these data centers is typically measured in terms of response times; that is, the time interval between the time of issue of request by the user and the time when the user gets back the response. Oversizing of data centers to meet peak demands is the norm of the day in order to avoid adverse impacts on user perceived *quality of service* (QoS) during high utilization periods. However this practice is extremely expensive since the overprovisioned resources consume significant amounts of energy even when idle. In our recent works we proposed and investigated a new paradigm called *Energy Adaptive Computing* (EAC)[1][2]. In this work we investigate a specific incarnation of EAC, which is the manifestation of Energy Adaptive Computing and its efficient implementation in data centers hosting multitiered applications ranging from e-commerce applications to large scale web search applications. We present an energy adaptive framework for these multitiered data centers.

The use of clean energy for data center operations in order to achieve sustainable IT has attracted a lot of interest in the recent past[3]. A compelling reason behind these efforts besides sustainability concerns is that power costs constitute around 31 percent of the total cost of ownership of these data centers[14]. The use of renewable energy could reduce the operation costs significantly over the long term, perhaps at the cost of increase in the initial investment in the power infrastructure. The use of renewable energy resources however has associated challenges. Energy supply from these energy sources is not constant and has both short and long term variations. Short term variations can be dealt with the help of energy storage like batteries, UPS, and so on. Long term variations in energy supply (such as, for example, diurnal variations in availability of solar energy) are difficult to deal with, since large-scale energy storage is extremely expensive. In either case, adapting to changes in workload patterns in addition to supply variations is essential for feasible and yet sustainable operation of these data centers.

"Adapting to changes in workload patterns in addition to supply variations is essential."

The energy profiles of renewable energy sources provide opportunities for predicting the available energy during different time periods. If energy availability is predicted to be low during a certain period of time, some work can be done during the previous energy plenty periods. For example, in a data center supporting a web search application, web crawling and indexing operations can be done during energy plenty periods. Workload can also be migrated to data centers located in places where there is surplus/cheap energy available.

Large-scale web service applications have data center operations supported by multiple clusters that are located in different geographical locations. Distributing the load between these data centers depends on a number of factors like the proximity to users, cost of routing, geographical load balancing, and so on. Another factor that needs to be considered in the geographical load distribution decisions is the electricity cost in different locations. Figure 1 shows the different power control actions that are taken at different time granularities in such a data center with multiple clusters. The largest time window T1 in Figure 1 represents the time granularity in which the power supply variations take place. The control actions at this time granularity may include predicting the available power for the next control period and migrating load away from energy deficient clusters. In the multitier web server scenario that we consider in this article, assuming that the services are stateless, workload migration involves redirecting more traffic to data centers with surplus energy. The number of servers in each cluster needs to be adjusted depending on the available power. Workload needs to be redistributed based on the number of servers that are kept powered on after the execution of the control actions.

"The number of servers in each cluster needs to be adjusted depending on the available power."

Some of the nodes might experience thermal constraints due to inefficient cooling or high thermal output. The demand variations and thermal constraints of the nodes need to be continuously monitored and reported to the managing entity (for example, a tier-level load dispatcher) and the load and power budgets of the nodes need to be adjusted accordingly. This happens at a smaller time granularity ($T2 < T1$) than the power variations. At an even smaller time granularity

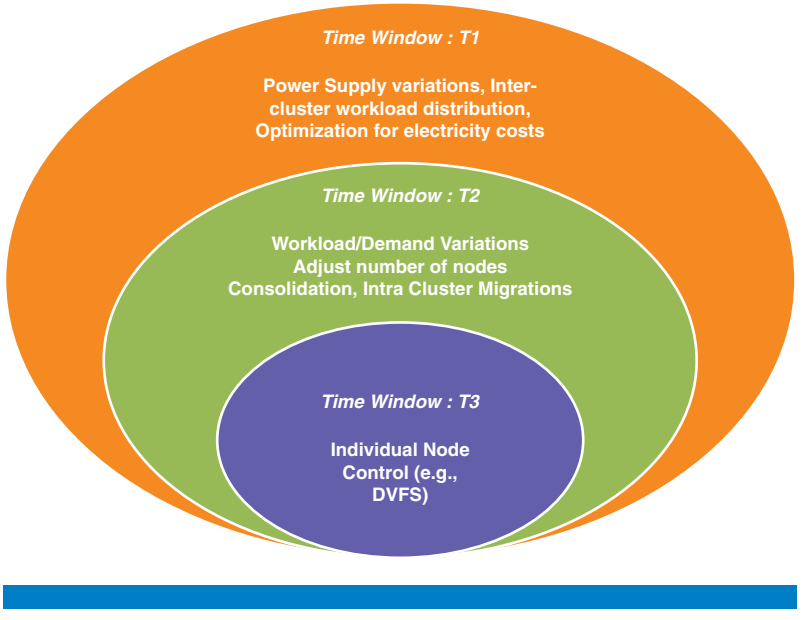


Figure 1: Power control actions at different time granularities (Source: Intel Corporation, 2012)

(T3 < T2), the control actions may include adjusting the operational frequency of individual nodes or putting the nodes in shallow sleep modes if available.

"We investigate elaborate control actions that need to be coordinated in a single data center hosting a multitiered web service application."

"An efficient planning of power control operations is important to minimize impact on QoS."

In this article we investigate elaborate control actions that need to be coordinated in a single data center hosting a multitiered web service application. The power control knob that we use is the number of servers that are kept powered on in each tier. The expected delays with different number of servers in each tier is determined with the help of queuing theoretic models and the best configuration that minimizes delay violations is chosen. After the best configuration is chosen, some servers need to be turned on and some others need to be turned off. These operations are not instantaneous and involve significant overhead. An efficient planning of these operations is important in order to minimize the effect on QoS. Much of the research works in the past have focused on formulating the power/performance tradeoffs as optimization problems[12][13] and propose solutions that minimize power costs and/or performance impacts. In this work, we explore strategies for implementation of such optimization solutions. The proposed technique also accounts for heterogeneity that may arise as a result of differences in available or consumable power in different servers due to limitations in their thermal capacities or insufficient power budgets.

Power Management in Multitiered Data Centers

Many data center applications like web search, e-commerce, and other transactional services require processing of user queries by servers that are organized into multiple tiers. The number and type of servers in each tier determine the response times for the user queries. Traditionally, power-hungry,

high-end servers were used in the backend tiers. Recently, newer data center architectures have been proposed where a large number of cheap commodity servers replace the powerful and expensive servers[4]. We consider such a scenario where there are multiple commodity servers in each tier that are typically homogeneous. A load dispatcher in each tier assigns the queries to the appropriate servers based on their capacity and current load. In this section we describe the architecture and modeling of multitiered data centers.

Architecture of Multitiered Data Centers

Typically multitiered data centers have a frontend tier that accepts user queries and serves as a gateway to the data center. The next tier is the application tier that processes the queries and contacts a third database tier when necessary and returns the response back to the frontend nodes, which process these responses (for example, convert to HTML) and send them to the users. The response time perceived by the users depends on the processing and queuing delays in each tier and the total number of sessions that are active in the data center. The typical architecture of a multitiered data center is shown in Figure 2. Figure 2(a) shows a shared storage architecture in which the data is stored in a common storage

"The response time perceived by the users depends on the processing and queuing delays in each tier."

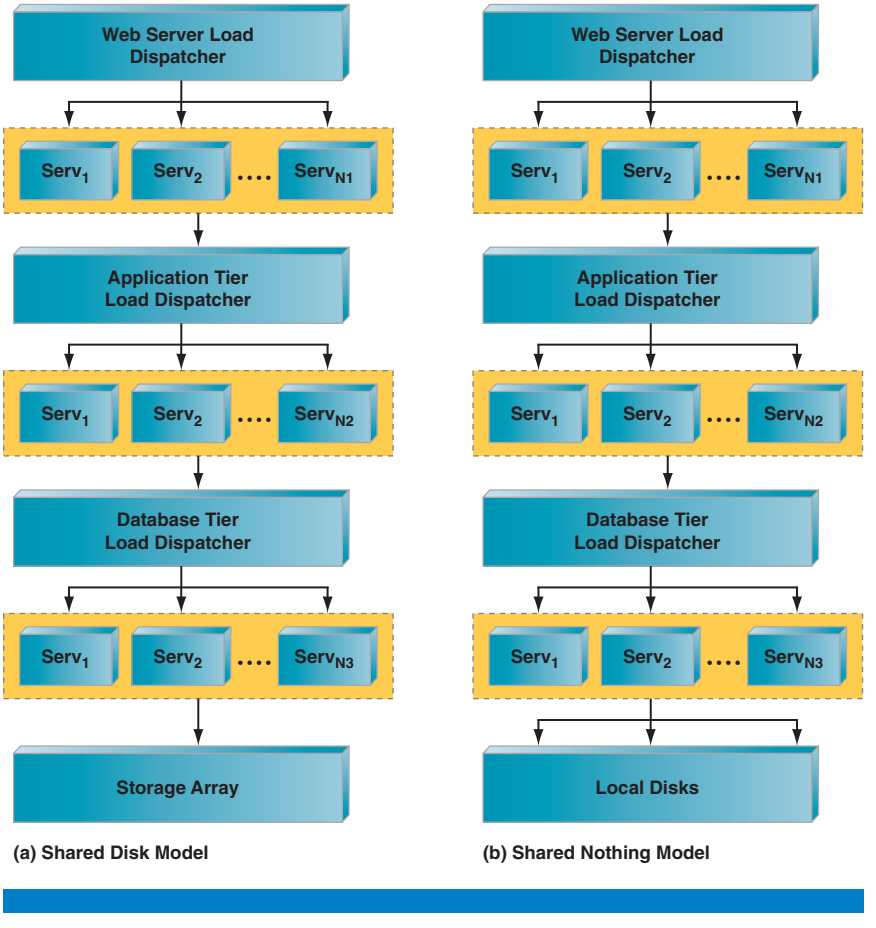


Figure 2: An example data center architecture with three tiers (Source: Intel Corporation, 2012)

"Queuing theoretic modeling of data centers to estimate the delays involved."

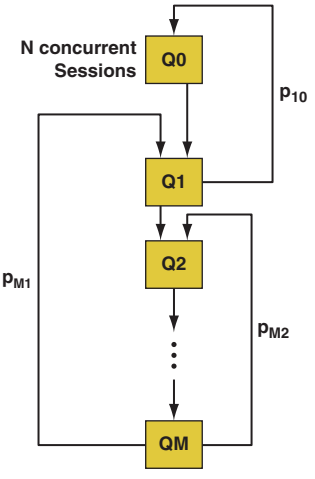


Figure 3: Closed Queue Model of a multitiered data center
(Source: Intel Corporation, 2012)

array of disks and is accessed by all servers. Figure 2(b) shows the case when each individual server stores a portion of the data in its own local disk. We discuss the data storage model later in the section "Data Storage Model."

Modeling of Multitiered Data Centers

An important step in determining the number of servers in each tier is to determine the overall delay across all tiers given the current workload demand of the data center. We leverage on queuing theoretic modeling[16] of the data center to estimate the delays involved. Let us assume in each level/tier i, the arrival rate to each server j in level i is λ_j^i. The requests pass through the same tier multiple times. The average number of sessions in progress handled by the data center is N. With the above assumptions, the multiple tiers can be analyzed as a closed network of queues as shown in Figure 3.

The user requests originate from queue Q_0 and pass through each tier multiple times. The delay at Q_0 corresponds to the user think time, which is the time spent by the user after receiving the response for a request and issuing a successive request. Q_0 is an infinite server queue that generates N concurrent sessions on the average.

Mean value analysis (MVA) is a popular technique for analyzing delays in networks of queues where each queue satisfies the $M \Rightarrow M$ property. This property states that if a queue is fed with Poisson input, the output process will also be Poisson. For a closed network, the MVA algorithm starts with the population of 1 and recursively computes the queue length and response time at higher population levels. The implementation of MVA is shown in Algorithm 1.

Algorithm 1: Mean Value Analysis

$n_i(0) = 0, i = 1, 2, 3 \ldots M$
$i(N) = 1/\mu_i + (1/\mu i) \times ni\ (N-1), i = 1, 2, 3 \ldots M$
$\tau_i(N) = N/(\tau_i(0) + \sum_i \tau_i(N)), i = 1 \text{ to } M$
$n_i(N) = \eta(N) \times \tau_i(N)$ (Little's Law)
where,

$\eta(N)$ is the throughput of the system with N customers
$i(N)$ is the average delay of the ith server when there are N customers in the system and
$n_i(N)$ is the average number of customers in queue i when there are N customers in the system

In the case of FCFS scheduling discipline such as the one assumed in this article, $M \Rightarrow M$ property holds only for exponential arrival and service time distributions (M/M queues). Since exponential service time is not practical, we use a simple modification to extend the MVA for queuing networks where the $M \Rightarrow M$ property does not hold[8]. The Nth arriving customer will find $n_i(N-1)$ customers in service in tier i, of which $U_i(N-1)$ will be busy already receiving service from the server in tier i. The average residual service time γ_i of customers is given by,

$$\gamma_i = s_i\ (1 + CV_i^2)/2 \quad (1)$$

where CV_i is the coefficient of variance of service times. The response time of the customer is therefore given by Equation (2).

$$\tau_i(N) = (1/\mu_i)[1 + n_i(N-1)] + U_i(N-1)(\gamma_i - s_i) \quad (2)$$

Hence this value of $\tau_i(N)$ can be substituted in Algorithm 1 to get an estimate of the delay values for the FCFS service discipline with a general service time distribution. The mean service time of each tier depends on the thermal and power constraints of the individual servers and the arrival rates. The thermal constraints are ignored temporarily and it is assumed that each server in the tier can run at full capacity. Hence for different configurations, that is, different number of servers in each tier, MVA can be used to calculate the mean overall delay across all tiers for the requests. The configuration with the minimum power consumption that can satisfy the QoS guarantees for the requests is then chosen.

Data Storage Model

As mentioned before, in traditional architectures, a few powerful servers are used in the database tier and they manage data in the backend databases in large arrays of hard disks that are interconnected by high speed storage area networks (SANs). However, with the advent of clustered and distributed storage systems (for example, NoSQL, key-value stores), these high-end servers are being replaced by a cluster of a large number of commodity servers, each of which handles a portion of the database. The cluster of servers may store the data in two ways.

"The cluster of servers may store the data in two ways – shared disk and shared nothing."

- *Shared Disk:* The cluster of nodes share a common storage array of disks connected via high-speed storage area network. This is very similar to the traditional database architecture described above.
- *Shared Nothing*: In this case, the servers have the data stored in their local disks as shown in Figure 2(a). Data blocks are typically replicated and stored in multiple locations to increase both reliability and performance.

In this article we consider both the above-mentioned architectures. In the shared disk model, the storage tier is modeled as a separate tier. We assume that the data access probabilities of files follow a Zipf distribution[5]. The access probability of a file is proportional to $1/r^{\alpha}$ where r is the rank of the file and α is the parameter of the Zipf distribution close to 1. Thus a few files are very popular and accessed more frequently. The rest of the files are accessed very rarely. The number of disks that are powered on to avoid delay violations is calculated as before using MVA. The disks storing the most popular files are kept powered on. The other disks that store less popular files are kept in the standby state and the requests to these disks incur longer delays. Let the disks be labeled 1 to N and the number of disks that are powered on be disk 1 to disk N_1. The mean service time of the storage tier is thus given by Equation 3.

$$\mu_{mean} = \sum_i \mu_1 P_i + \sum_j \mu_2 P_j, \quad i = 1 \text{ to } N_1 \text{ and } j = N_1 + 1 \text{ to } N \quad (3)$$

where μ_1 is the average service time of a disk that is powered on and μ_2 is the sum of the average service time of the disk and the disk spin-up time from standby to ready state.

In the shared nothing storage architecture model, when a node in the database tier is shut down, the data that it was handling is migrated to other nodes. However there is a minimum capacity requirement for the stored data. Naturally, there are a minimum number of nodes that need to be powered on and hence there is a minimum power budget for the storage tier.

"With renewable energy resources infrastructure needs to adapt to the available power profiles."

"Control the number of servers that are powered on and adjust the distribution of requests to different servers."

Server Selection in Each Tier

With renewable energy resources and in some cases even with conventional power grids, the available power budget to the data center is not constant. The output of solar power plants may be affected by clouds in the sky and/or the relative position of sun. The available power from wind power plants varies continuously depending on wind strength and directions. In these cases, it is essential for the infrastructure to adapt to the available power profiles. In the multitier scenario that we discuss in our current work, the primary adaptation mechanism that we use is controlling the number of servers that are powered on for serving the requests and adjusting the distribution of requests to different servers. The former step is executed at time intervals of ΔT_s and the latter step is more frequent and is executed at time intervals of ΔT_l ($<\Delta T_s$). Given the available (predicted) power for the next ΔT_s period, the first step is to determine the number of servers that need to be powered on in each tier. Let the number of tiers be M. As mentioned before, different servers have different thermal capacities and hence have different service rates. Assuming that there are N_1, N_2, N_3 . . . N_M servers in each tier, if we were to account for capacities (constrained by thermal limits) of each server individually and then try to determine which servers should be kept powered on in each tier to meet the overall delay bound, the corresponding algorithm would have a complexity of $O(2^{N_1+N_2+N_3+\cdots N_M})$. In order to avoid this, we devise an approximation scheme as follows.

- *Step 1:* At the beginning of each time interval T_S, assume that all servers can be run at full capacity ignoring their thermal limits. The delay values with different number of servers in each tier can be calculated using MVA as explained earlier in the section "Modeling of Multitiered Data Centers." Determine the minimum number of servers that need to be kept up and running in each tier to meet the overall delay bound – $O(N_1N_2N_3 \ldots N_M)$ algorithm. It is to be noted that typically $M \leq 3$ in most data centers.

- *Step 2:* Let the total number of servers in each tier i that need to be powered on given by the above step be N_i. Due to thermal constraints N_i servers that can operate at full capacity may not be available in a tier i. Hence we choose the best servers so that the impact on delay is minimized.

We formulate the situation in Step 2 as a knapsack problem. The knapsack problem[15] can be defined as follows.

Definition 1 (Knapsack problem): Given a knapsack of fixed capacity (here, $\sum_i P(S_i)$, $i = 1, 2, 3 \ldots N$, where $P(S_i)$ is the power consumed by server S_i that

is chosen to fill the knapsack) the objective is to fill the knapsack with a set of items ($\{I_k\}$)so that the total value of items ($\sum$ value (I_k), $\forall k$) packed in the knapsack is maximized and the sum of weights of the items ($\sum$ weight (I_k), $\forall k$) does not exceed the capacity of the knapsack.

Here the weight of each item (weight (I_k)) is the power consumed by the individual server and the total capacity of the knapsack is the total power budget of the tier. We explain how we assign the value of each item (server) (value (I_k)) later. The knapsack problem is an NP–hard problem. We use a greedy approximate solution—the items are arranged in sorted order of value (I_k)/weight (I_k). Then the items are picked one by one and packed into the knapsack as long as the sum of weights of the knapsack is less than the capacity of the knapsack. An instance of the knapsack problem is solved at each tier and the capacity of the knapsack in each instance is the power budget of the tier. The total available power budget of the data center is allocated to each tier proportional to the number of servers that need to be kept powered on in that tier as determined in Step 1 above.

"An instance of the knapsack problem is solved at each tier."

We now explain how the value (I_k) is assigned to each server to include other factors besides service time. The objective of the knapsack algorithm is to maximize the value of items that are packed in the knapsack. We know that it might take several seconds for a server to be powered up from a deep sleep/inactive state. We call this the activation period (T_a) of the server. Hence when choosing between multiple servers, the servers that are already up and running need to be preferred to the ones that are shut down or in deep sleep states. On the other hand when a server needs to be powered down, its workload needs to be redistributed to other servers, which will in turn increase the load in the other servers. The pending sessions that the server was handling need to be completed. Above all, the service rate of the server is the most important factor in determining its value. Hence the value function of each server needs to incorporate all these factors. Hence the value of server I_k is given by the following equation.

$$\text{value}(I_k) = w_1\mu_{I_k} + w_2\lambda_{I_k} + w_3 R_{I_k} \quad (4)$$

where μ_{I_k} is the service rate of server I_k and λ_{I_k} is the arrival rate of server I_k.

As mentioned before, in the shared disk storage tier model, the number of disks that need to be powered on is calculated by MVA. When filling the knapsack, the disks are ordered according to the popularity of the files they store and the disks storing the most popular files are chosen first to fill the knapsack.

Load Dispatcher

The service rate μ_j^i (of server j in tier i), is different for different servers depending on power budgets and thermal constraints. For instance even with sufficient power budgets, certain servers can only be operated at half their capacity due to inefficient cooling (for example, servers in the top of racks) and the service time of a server running at 50 percent of its maximum frequency is almost double that of a server running at its maximum frequency of operation

"Even with sufficient power budgets, certain servers can only be operated at half their capacity due to inefficient cooling."

(ignoring memory access and other delays). The relationship between the power consumption and thermal limitations of a server were derived in our previous work[1] and is given by the following equation.

$$T(t) = [T_a + [T(0) - T_a]e^{-c_2 t}] + c_1 e^{-c_2 t} \int_0^t P(\tau) e^{c_2 \tau} \, d\tau \quad (5)$$

where T_a is the ambient temperature, $T(t)$ is the server temperature at time t, and c_1, c_2 are thermal constants. Power budgets are allocated to servers so that the thermal constraints of individual servers are not violated and the capacity of servers are proportional to the power budgets.

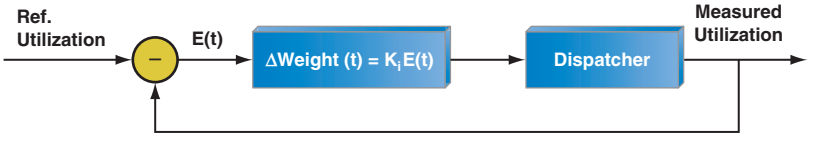


Figure 4: Integral Controller for load dispatching (Source: Intel Corporation, 2012)

Figure 4 shows the design of the load dispatcher. The load dispatcher in each tier periodically (ΔT_l) infers the average service times of the individual servers and the current arrival rates for each server. The average utilization for tier i is then given by $\rho_i = \lambda_i / \mu_i$ where λ and μ are the average arrival rate and service times of the tier. The dispatcher in each tier i then adjusts the arrival rate of queries λ_i for each server j, so that the utilization $\rho_i^j = \lambda_i^j / \mu_i^j$ is the same ($= \rho_i$) for all servers j. The service rate reduction can be due to a number of reasons. The servers might be running at lower operational frequencies by means of techniques like dynamic voltage and frequency scaling (DVFS)[6]. An alternative technique is to periodically force the servers to go to deep sleep states with low power consumption in order to reduce their power consumption[14] or prevent their temperatures from increasing beyond certain limits. Irrespective of the control mechanism used, the load dispatcher has to make sure that the load is shared proportional to the capacities (service rates) of the servers. The integral controller periodically adjusts the weights for each server depending on the difference between the target and measured utilizations. The input to the controller is the measured utilization of each server. The error term $E(t)$ is the difference between the target and measured utilizations at time t. The new weights for the time period $(t, t+1)$ are given by the following equation.

"Load dispatcher makes sure that the load is shared proportional to the service rates of servers."

$$\text{Weight}\ (t+1) = \text{Weight}\ (t) + K_i E(t) \quad (6)$$

where K_i is the integral constant. The arrival rates of each server for that period are then adjusted proportional to the weights assigned to each server.

Scheduling Background Jobs

In many typical data centers, besides the processes that cater to services from the web service clients, some batch jobs, backup services, and maintenance

"Batch jobs, backup services and maintenance jobs are scheduled periodically that run in the background."

jobs are scheduled periodically that run in the background. Most of these jobs have relaxed deadlines and can be flexibly scheduled. The frequency of these jobs is generally low. If the availability of power is low during a certain time period, the scheduling of these background jobs can be postponed. In our proposed scheme the scheduling of processes in the individual nodes is FCFS. Since all calculations of delays until now do not account for the background jobs, we maintain a separate queue for these jobs to avoid interference with regular queries. Whenever the query queue is empty, the node level scheduler schedules the background jobs. The scheduling scheme can also be altered so that the starvation of background jobs is avoided by setting a threshold after which these jobs are forced into the query queue. Since these jobs are infrequent, this forced scheduling happens only rarely and hence does not affect the response times of the queries much.

Planning of Power Control Operations

As explained in the previous section, at each time step ΔT_s the algorithm to optimize for power and performance based on the predicted power supply is executed and the number of servers that need to be powered on is adjusted. New servers need to be powered on and some servers need to be turned off. In either case, workload needs to be reassigned and the sessions that are already in progress need to be completed before the servers handling those sessions are turned off. These operations cannot all be done instantaneously and each of them has associated costs.

"Power control operations cannot be done instantaneously and each of them has associated costs."

Let the initial state of servers at time t be $X(t)$ and the goal state given by the knapsack algorithm is $Y(t+T)$. Here the term state refers to the set of servers that are powered on and turned off and their workloads. The interval between time t and $t+T$ of length T is the planning horizon. A transition from state $X(t)$ to state $Y(t+T)$ involves multiple steps. An exploration of each possible state transition has a time complexity factor that is exponential in the number of servers involved. Hence we employ search heuristics that are well known and widely used in artificial intelligence for planning. One such technique is the A* search algorithm[7] that employs a greedy best first search technique to find the least cost path from the initial state to the goal state. The cost function $f(x)$ from one state to another consists of an accumulated direct path cost value to the next state $g(x)$ and a heuristic function $h(x)$. The next state that is chosen is the one with the minimum cost function $f(x)$. The sufficient and necessary condition for optimality is that the heuristic function $h(x)$ is admissible. An admissible heuristic is one that does not overestimate the cost to the goal state.

"A search algorithm employs a greedy best first search technique to find the least cost path from initial to goal state."*

Figure 5 shows the different state transitions during the execution of the required changes for adaptation. Let us now consider the steps involved during state transition. Let $\{S_i\}$ be the set of servers that are currently serving the requests. Let $\{A_i\}$ be the set of servers that need to be shut down and $\{B_i\}$ be the set of servers that need to be powered on. When a server has to be turned off, no new requests are forwarded to the server and the sessions that it is currently serving need to be completed. The workload of the server to be shut

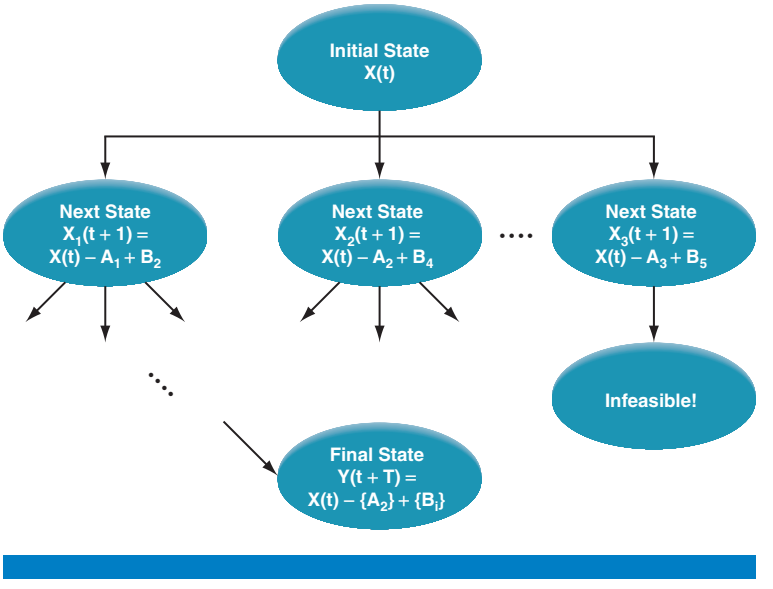


Figure 5: State transitions from initial to goal states (Source: Intel Corporation, 2012)

"The path to reach the goal state needs to minimize the time taken for the control actions to be completed."

down will have to be redistributed to other servers that are already up, thereby increasing the average delay in those servers. The path to reach the goal state from the start state needs to minimize the time taken for the control actions to be completed.

The cost function $g(x)$, which is the cost of getting to state x, is therefore the estimated time to reach the state x from the start state. The heuristic function $h(x)$ is given by the following equation.

$$\begin{aligned} h(x) &= \max(Q_j(x)/\mu_j(x),\ T_a), && \text{if } N_A(x) \neq 0 \\ &= Q_j(x)/\mu_j(x), && \text{if } N_A(x) = 0 \end{aligned} \tag{7}$$

where j is the server with the maximum queue length in state x and $Q_j(x)$ is the queue length of j. $N_A(x)$ is the cardinality of the set of servers that are activated in state x. T_a is the time of activation of a server. $Q_j(x)/\mu_j(x)$ is the virtual work that needs to be completed before the server can be completely shut down. It can be easily seen that the heuristic function is never an overestimate of the actual delay involved since it is a conservative estimate of the most time-consuming control action (deactivating/activating a server). Hence this heuristic function is admissible. Any state is considered infeasible if the power consumption during or at the end of the state is more than the power budget of the tier. A server with a long queue length will most likely have a large workload to be redistributed and hence shutting it down will increase the utilization of the other servers. Hence it is better to mark it for deactivation towards the end of the planning horizon since most of the servers that are marked for activation would be fully functional by then and will be ready to accept the workload of the servers with long queues. In case of the shared nothing storage model, data also has to be migrated from nodes that are

being shut down in the database tier. Once the shortest path of state transitions is determined, the execution of planning operations are carried out after a constant time T_D, which is the time taken for data migrations to complete. Also any state is considered infeasible if the total number of nodes that are powered on in that state is less than the required minimum number of nodes.

The planning process can also be made more dynamic by means of model predictive control (MPC) techniques[9] where at time t, the actions are planned for every time instant $t + \tau$, $\forall \tau \in \{0, 1, \ldots T\}$ and only the control actions for time $t + 1$ are implemented. Then at time $t + 1$ the same process is repeated for the receding time horizon T. However one drawback of this technique might be that this would turn out to be an expensive process considering the large state space. Besides, these techniques might take a long time to converge and are not essentially optimal.

"The planning process can be made more dynamic by means of model predictive control techniques."

Simulation Experiments

In this section we evaluate the techniques presented in previous sections via detailed simulations. We built a discrete event simulator in Java for our experiments. The event queue is a priority queue ordered by event times. Each scheduled event has an associated handler class that executes the corresponding operations and schedules the next event when applicable. As soon as a query arrives, it is sent to the wait queue of the tier. Based on the weights determined by the integral controller, the queries are assigned to appropriate processors. Then an End Processing event is scheduled after the service time of the processor. When the query has completed processing, it is either sent to the previous tier or to the next tier. In our simulations the data center has three tiers (besides the infinite server queue Q_0) and each tier processes a query twice except for the last tier. In the shared disk model the disk array is modeled as a separate tier. There is one integral controller object for each tier. The integral controller is invoked periodically at intervals of ΔT_I and whenever there is a change in the number of nodes in the tier. Planning events occur at time intervals of ΔT_S. The total power supply varies every ΔT_S and the number of servers in each tier is determined by MVA. Then the knapsack problem instance is solved for each tier to determine which servers need to be powered on and which servers need to be shut down. The A* search algorithm is run to determine the best search path and then the steps in the path are executed one by one.

Simulation Parameters

We conducted some real-time experiments to set the parameters in the simulator. We used a Dell* 2650 server with Intel® Xeon® processor and running Ubuntu* Server. Linux kernel supports five different governors for scaling CPU frequency. The performance governor runs the CPU at the highest possible frequency and the powersave governor runs the CPU at the lowest possible frequency.

"Linux kernel supports five different governors for scaling CPU frequency."

Utilization	Performance Governor	Powersave Governor
20%	266 W	266 W
40%	327 W	310 W
60%	355 W	340 W
80%	385 W	360 W
100%	405 W	380 W

Table 1: Power Consumption at Different CPU Utilizations
(Source: Intel Corporation, 2012)

Table 1 shows the power consumed by the server at different CPU utilization levels under the two governor settings. It can be seen that at lower utilization levels the power consumed is not significantly different for the two governor settings but at high utilization levels the difference in power consumed is high. In newer generation servers, with a wider range of CPU frequencies, the difference in power consumption of the highest and the lowest frequency values might be even larger. The values of parameters c_1 and c_2 in Equation 5 were found to be 0.2 and −0.008 respectively in our previous work. We use the same values for our simulations. As mentioned before, we assume that multiple commodity servers are used in each tier and the maximum number of servers in each tier is 40. There are three tiers of servers and each query passes through the first two tiers twice and the last tier once. The average service times of tier 1, 2, and 3 were assumed to be 20 ms, 40 ms, and 120 ms, respectively. In the shared disk model, the average tier 3 service time was assumed to be 100 ms and the service delay of the disk array was assumed to be from a uniform distribution with mean 10 ms. If the disk to which the request is issued is in the standby state, an additional delay of three seconds was added to the service delay to account for the spin-up time. The overall processing time of a query that arrives into the system without incurring any waiting is around $2 \times 20 + 2 \times 40 + 120 = 240$ ms. The utilization of the data center is changed by changing the number of concurrent sessions. We assume that the SLA requirement is that the average delay of the queries ≤ 1 second. The power supply is assumed to consist of a combination of a renewable energy source whose output varies periodically and a constant backup power from a conventional grid supply.

"The power supply consists of a combination of a varying renewable energy source and a constant backup power."

Together, these two resources can support the data center operations at full load (≈ 50 kW). The reason behind this model is that in the current scenario, renewable energy resources alone cannot support data center operations entirely[10][11]. Hence we assume that when all the servers in the data center operate at full load, a portion of the necessary power to support the operation of the data center comes from conventional (backup) power sources. The available backup power is denoted as BP kW. The remaining power comes from a renewable energy source and is denoted as GP kW. Hence when the renewable energy source is operating at full capacity, BP + GP ≈ 50 kW. In our simulations we assume that all of the available green energy is utilized first before using any power from the backup power resources.

We now investigate the energy profiles of renewable resources that our proposed scheme can successfully manage to adapt to. Figure 6 shows the proportion of green and brown energy required to support the data center operations when the available backup power is 20 percent of the total power required to support the entire data center ($0.2 \times 50 = 10$ kW in our case) and the renewable energy varies uniformly between 50 and 100 percent of the remaining 40 kW. We can see that when the number of sessions is small, the entire data center operations can be handled by renewable energy alone. When the number of sessions increases, the variability in renewable energy requires that some backup power needs to be used occasionally. We can see that even when the number of sessions is as high as 1000, only 38 percent of backup power is needed to guarantee the delay bounds. On the average, only 15 percent of the total power consumption comes from the brown energy sources. This is because in our scheme only the required number of servers are kept powered on in each tier to meet the SLA requirements. It is to be noted that we do not explicitly optimize the consumption of brown energy. Our scheme naturally reduces the energy consumption by powering on only the necessary number of servers in each tier to maintain the delay bounds.

"On the average, only 15% of the total power consumption comes from the brown energy sources."

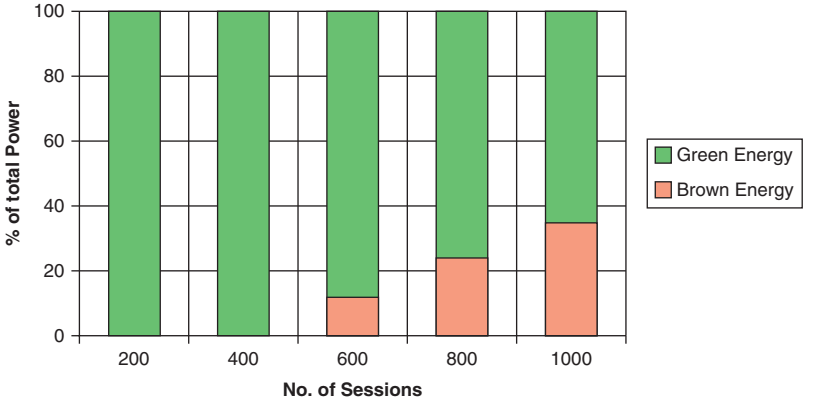


Figure 6: Proportion of green and brown energy consumed when the renewable power varies between 50 and 100 percent (Source: Intel Corporation, 2012)

Next, we evaluate the performance of our scheme under an energy constrained situation where the available energy is (at times) not sufficient to support the required number of servers to meet the SLA requirements. In order to simulate an energy-constrained scenario, we assume the following power supply model. Figure 7 shows the available renewable power profile that we assume for our simulation in a period of one hour where the available green energy fluctuates between 0.45 GP and 0.55 GP kW which corresponds to a variation level of 55 percent. It can be seen that the available renewable energy fluctuates every 10 minutes. With the presence of adequate energy storage infrastructure, these fluctuations may be at the granularity of hours.

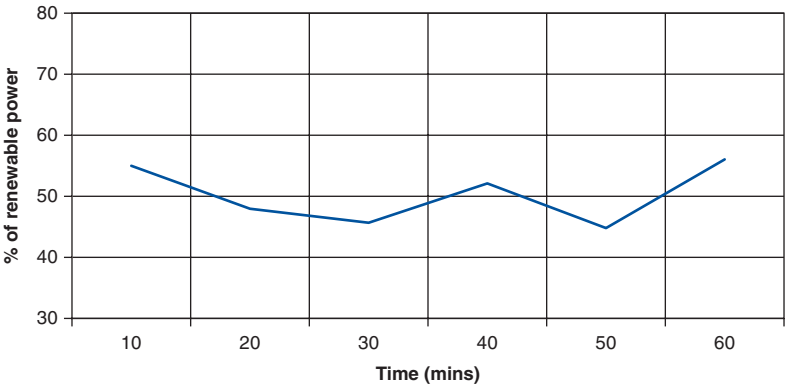


Figure 7: Renewable power profile assumed in simulation (Source: Intel Corporation, 2012)

Figure 8 shows the delay incurred when the number of sessions is 1000 and the backup power is 20 percent. The power profile assumed is the same as shown in Figure 7. The baseline scheme that we use here is one where the available power budget is equally shared among all tiers. It can be seen that when the power budget is allocated equally, the delay values are 35 percent more than our scheme on the average. A blind allocation of power to the tiers without accounting for the tier delays will result in high waiting times even when a large number of servers are powered on.

"A blind allocation of power to the tiers without accounting for the tier delays will result in high waiting times."

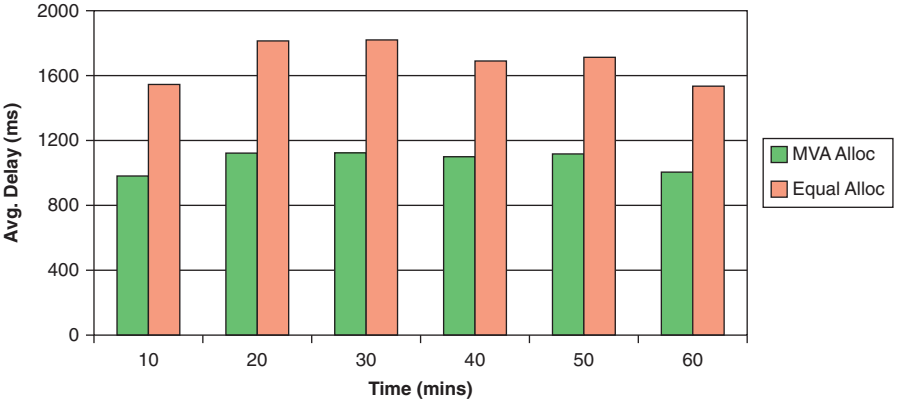


Figure 8: Average delay values with 20 percent backup power and 1000 sessions (Source: Intel Corporation, 2012)

Figure 9 shows the number of background jobs scheduled when the renewable energy power profile is as shown in Figure 7. We assume that the jobs take 100 ms to complete and are recurrent every 1 minute. We see that the number of background jobs scheduled decreases when the available energy decreases. Even though the scheduling of jobs is not optimized explicitly, since there is a separate queue for the background jobs and the integral controller dispatches the queries according to the capacity of servers, the scheduling of background jobs is simply postponed until the power budget of the tier increases.

"Scheduling of background jobs is postponed until the power budget of the tier increases."

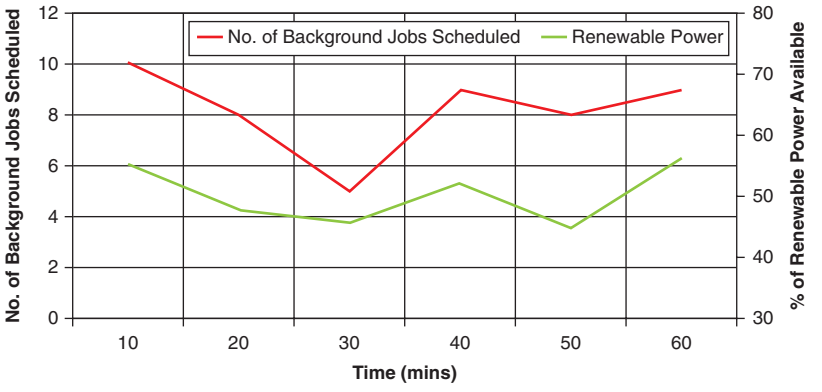


Figure 9: Number of background jobs scheduled
(Source: Intel Corporation, 2012)

Another important aspect of our scheme is its ability to account for the thermal capacities of the nodes. We realize this by means of the load dispatcher. The load dispatcher takes the service rate of each node in the tier as input and allocates the appropriate arrival rates to the nodes so that the utilization is the same for all nodes.

"Account for thermal capacities of nodes with the help of load dispatcher."

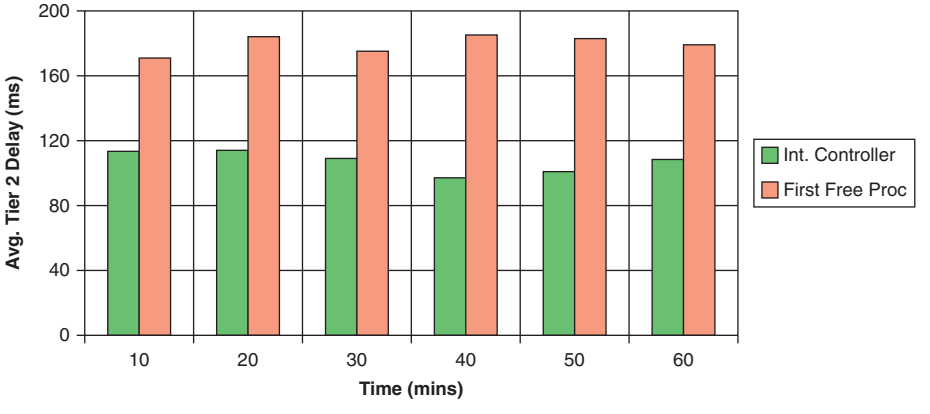


Figure 10: Average delay in nodes at high temperature zones
(Source: Intel Corporation, 2012)

To demonstrate the effectiveness of the load dispatcher, we set the ambient temperatures of 50 percent of the nodes in each tier at 35°C and the rest of the nodes were at an ambient temperature of 25°C. The nodes in the high temperature zones cannot operate at full capacity and their power consumption is limited by running them at a lower frequency as given by Table 1. The maximum temperature limit was assumed to be 70°C and the power limit is determined by Equation 1. We assume that these servers are running at a lower frequency such that the service times are increased by 1.5 times the original service times. With these settings, the average delay values in the high temperature servers in Tier 2 are shown in Figure 10 for 1000

"Integral control based load dispatching reduces the delay values by almost 40%."

concurrent sessions. The baseline scheme that we use here for comparison is one that dispatches the queries to the first available free processor without considering the thermal limits/service rates of the nodes. It can be seen that our integral control based load dispatching scheme reduces the delay values by almost 40 percent compared to the baseline scheme. We observed that at low utilization levels, the delay values in these nodes are zero since they are not turned on. This is because when the knapsack problem instance is solved, these nodes are the least preferred to be packed into the knapsack because of low service rates. But at high utilizations, choosing these nodes becomes inevitable. However, the integral controller continuously adjusts the arrival rate into these nodes so that their average utilization is not high.

Figure 11 shows the average delay of the storage tier when all disks are on and the MVA allocation is based on the Zipf access distribution. We see that the delay incurred by turning off the disks storing least popular files is only 10 percent more on the average than the delay when all disks are powered on. Thus our proposed technique has minimum impact on the delay values while adapting to the available power profile.

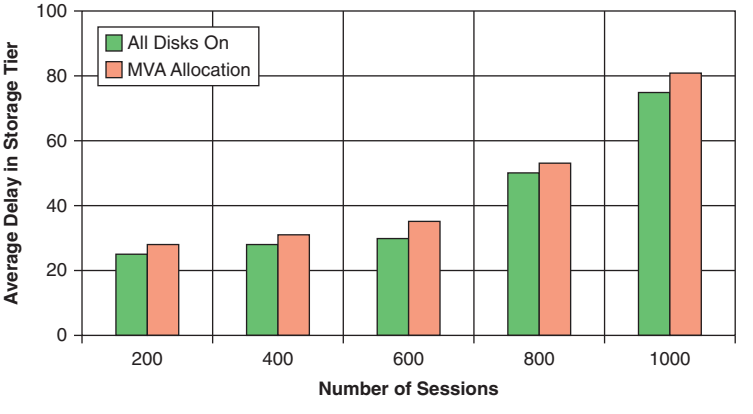


Figure 11: Average delay of the storage tier in shared disk model (Source: Intel Corporation, 2012)

Conclusion

We have investigated the infrastructure adaptations that need to be done in a data center hosting multitiered web services in order to cope with energy and thermal variations. We presented the control actions that need to be executed at different time granularities and their implementation in a multitiered data center. We also showed that an efficient implementation of the control actions is necessary to reduce overhead associated with the control actions. We have also demonstrated that our scheme can adapt to significant variations in the available power supply with minimum dependence on conventional grid. A coordinated approach such as the one described in this article is necessary for simultaneously coping with energy variability and variations in demand in data centers.

"Our scheme can adapt to significant variations in the available power supply with minimum dependence on conventional grid."

References

[1] K. Kant, M. Murugan, and D. H. C. Du, "Willow: A control system for energy and thermal adaptive computing," in IPDPS, 2011, pp. 36–47.

[2] K. Kant, "Challenges in Distributed Energy Adaptive Computing," in Proceedings of ACM HotMetrics, 2009.

[3] Google Green Energy Initiative, http://www.google.com/about/datacenters/energy.html

[4] L. A. Barroso, J. Dean, and U. Holzle, "Web search for a planet: The Google cluster architecture," IEEE Micro, Vol. 23, March 2003.

[5] E. Pinheiro and R. Bianchini, "Energy conservation techniques for disk array-based servers," Proceedings of the 18th annual international conference on Supercomputing, 2004.

[6] G. Dhiman and T. S. Rosing, "Dynamic voltage frequency scaling for multi-tasking systems using online learning," in ISLPED '07: Proceedings of the 2007 international symposium on Low power electronics and design, 2007.

[7] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2009

[8] K. Kant, "Introduction to computer system performance evaluation," McGraw-Hill, 1992.

[9] D. Kusic, N. Kandasamy, and G. Jiang, "Combined power and performance management of virtualized computing environments serving session-based workloads," Network and Service Management, IEEE Transactions on, vol. 8, no. 3, pp. 245–258, September 2011.

[10] C. Stewart and K. Shen, "Some joules are more precious than others: Managing renewable energy in the data center," in Workshop on Power Aware Computing and Systems HotPower), 2009.

[11] I. Goiri, K Le, T. D. Nguyen, J. Guitart, J. Torres and R. Bianchini, "GreenHadoop: Leveraging Green Energy in Data-Processing Frameworks," In Proceedings of the seventh conference on Computer systems, EuroSys 2012.

[12] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems, 2011.

[13] N. Buchbinder, N. Jain, and I. Menache, "Online Job Migration for Reducing the Electricity Bill in the Cloud,", Proceedings of the 10th IEEE International Conference on Networking (IFIP Networking '11).

[14] N. Sharma, S. Barker, D. Irwin, and P. Shenoy, "Blink: managing server clusters on intermittent power," in Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems (ASPLOS), 2011.

[15] S. Sahini, "Approximate algorithms for the 0/1 knapsack problem," J. ACM 22, 1, pp. 115–124, Jan 1975.

[16] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi, "An analytical model for multi-tier internet services and its applications," in Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems, 2005.

Author Biographies

Muthukumar Murugan is a fourth-year PhD candidate at the University of Minnesota. His research interests include energy and thermal adaptive computing and data center power management, design and applications of solid state storage, and performance and power management of large scale storage systems in the cloud.

Dr. Krishna Kant is currently a program director in the Computer and Networks Systems (CNS) division of the National Science Foundation. At Intel Research, he has worked on future server architectures and technologies. His recent research interests include sustainability and energy issues in data center design, robustness of Internet infrastructure, and cloud computing and configuration management security.

Dr. David Hung Chang Du is a Qwest Chair Professor of Computer Science and Engineering at the University of Minnesota, Minneapolis. He is the Center Director of a newly established NSF multi-university I/UCRC Center on Intelligent Storage. Dr. Du's current research focuses on intelligent storage systems, multimedia computing, sensor networks, and cyber-physical systems. He also has research experience in high-speed networking, database design, and CAD for VLSI circuits.

STORAGE POWER OPTIMIZATIONS FOR CLIENT DEVICES AND DATA CENTERS

Contributors

Hung-Ching Chang
Department of Computer Science, Virginia Tech

Erik Kruus
NEC Research Laboratories America, Inc.

Thomas J Barnes
Technology and Manufacturing Group, Intel Corporation

Abhishek R. Agrawal
Software and Services Group, Intel Corporation

Kirk W. Cameron
Department of Computer Science, Virginia Tech

Storage devices are essential to all computing systems that store user data from desktops, to notebooks and Ultrabooks™ to data centers. Hard disk drives (HDDs) or solid state drives (SSDs) are today's most popular storage solutions. Active power for storage devices has significant impact on the battery life of client devices. In the data center, the amount of energy required for thousands of HDDs/SSDs competes with that of small cities. In this article, we propose a range of storage power optimization techniques that apply equally well to client devices and data centers. A number of power conscious optimization techniques along with multiple case studies are discussed. On notebook/Ultrabook platforms our techniques achieve power savings of ~420 mW for HDD/SSD power during a media playback scenario. Furthermore, for an "always connected" scenario, our techniques achieve 7.5X higher retrieval rates with a 5 mW storage power budget. For data center storage, we demonstrate 61.9 percent power savings from using a swap policy optimization strategy.

"By 2015 there will be more than 60 exabytes of data stored in global data centers with ~800 terabytes per second of traffic generated through global networking infrastructure."

Introduction to Storage Issues

Global technology growth has fueled the need for vast storage systems. Social networking currently drives the growth of huge data centers and compute farms. Facebook alone has a user base of 845 million members who have shared more than 100 petabytes of photos and videos and produced an average of 2.7 billion "likes" and comments a day in just the final three months of 2011[1]. It is projected that by 2015 there will be more than 60 exabytes of data stored in global data centers with ~800 terabytes per second of traffic generated through global networking infrastructure[2]. Energy-efficient storage of these vast volumes of data is a looming challenge for the community.

On a common hardware platform, we have seen how data efficiency can be improved at a software level. Here we will consider what can be done to make software lay out data more intelligently. Awareness by users, system designers, and software writers is useful. We will first consider what can be done within the scope of the single-user scenario and also touch on what can be done in larger-scale storage scenarios. The current issues center around usage of solid-state devices (SSDs) as compared with hard disk drives (HDDs). We will be brief and present order of magnitude take-away messages, indicative of a median user or device.

Storage Device Power States

The Advanced Configuration Power and Interface (ACPI) standard defines D-states (D0, D1, D2, and D3) for device power states. The D0 state refers to "on" while the D3 state indicates "off." There is no definition for D1/D2 states.

Generally, the D1 state saves less power than the D2 state but retains more contexts and, thus, has less exit latency going back to the D0 state. In storage devices, Serial ATA (SATA) bus interfaces and SATA devices have power states that map to ACPI D-states.

The ATA8-ACS (ATA Attachment 8 Command Set) standard defines four available power states for SATA devices. They are, in order of decreasing power consumption:

- D0 – Device is active or immediately ready to process commands.
- D1 – Entered after idle period with no commands to be processed, or on receipt of an IDLE IMMEDIATE command.
- D2 – Entered on receipt of STANDBY (IMMEDIATE) command.
- D3 – Entered on receipt of a SLEEP command.

Hard disk drive devices spin when the drive is active or in idle states, and the drive is spun down when the device is in standby or sleep states.

"Hard disk drive devices spin when the drive is active or in idle states, and the drive is spun down when the device is in standby or sleep states."

SATA also defines three PHY layer (or interface) power states. They are, in order of decreasing power consumption:

- PHY Ready – PHY logic and PLL are both on and active.
- Partial – PHY logic is powered, but in a reduced state. Exit latency is no longer than 0.1 ms.
- Slumber – PHY logic is powered, but in a reduced state. Exit latency can be up to 10 ms.

Figure 1 provides a hierarchical view of the SATA device and link power states[3].

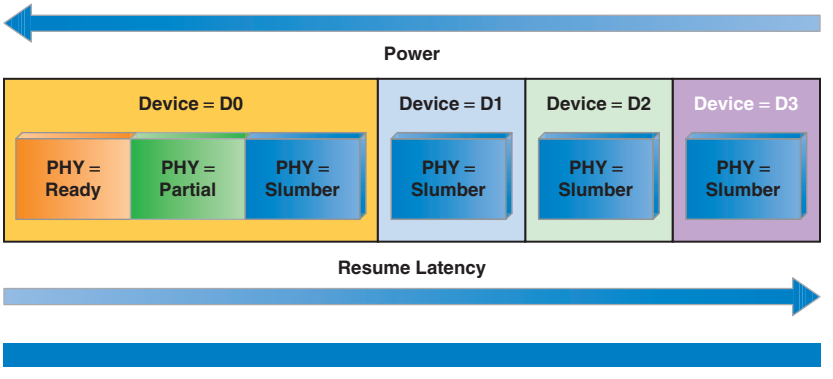


Figure 1: SATA drive power state hierarchy
(Source: Intel Corporation, 2012)

It is important to note that the PHY is not required to be in a Slumber state when the device is in a D1, D2, or D3 state. While this may be the likely condition of the interface when the devices connected are in a low power state, it is not a requirement.

New standards are emerging at the drive and platform level that enable more fine-grain control of storage power consumption; these include SATA DEVSLP

"New standards are emerging at the drive and platform level that enable more fine-grain control of storage power consumption."

"DEVSLP state provides the option for the storage device to completely disable the SATA interface, yet still maintains the exit latency close to slumber state."

and ACPI Run-Time D3 (RTD3)[4]. A new DEVSLP link state has been added to the original three power states (ready, partial, and slumber, and DEVSLP) while Run-Time D3 provides a system-level power option in addition to the existing four SATA device power states (active, idle, standby, and sleep). DEVSLP state provides the option for the storage device to completely disable the SATA interface, yet still maintains the exit latency close to slumber state. RTD3 allows the SATA device to be place into power-off while the rest of the system remains powered, providing an aggressive power-saving opportunity for the system on the long period of storage device idleness usage scenarios.

Device State (Link State)	In-state Power (mW)	Time To Transition(s)	Latency to Active (s)
Active (Ready)	>1000	0	0
Idle (Partial)	100	10^{-6}	10^{-4}
Idle (Slumber)	50	10^{-3}	0.01
Idle (DEVSLP)	5	0.02	0.02
Off- RTD3 (n/a)	0	1.5	0.6

Table 1: Storage Device Power States[4][5][6][7]
(Source: Intel Corporation, 2012)

Table 1 lists in-state power, time to transition, and latency to active from the storage power state. Note that the values are representative for current drives, or in the case of DEVSLP and RTD3, expected values for future drives. Actual values will vary from drive to drive and are implementation-specific.

Figure 2 illustrates a storage device that features the latest DEVSLP and RTD3 power states. As the system begins I/O activity, the storage device is first woken

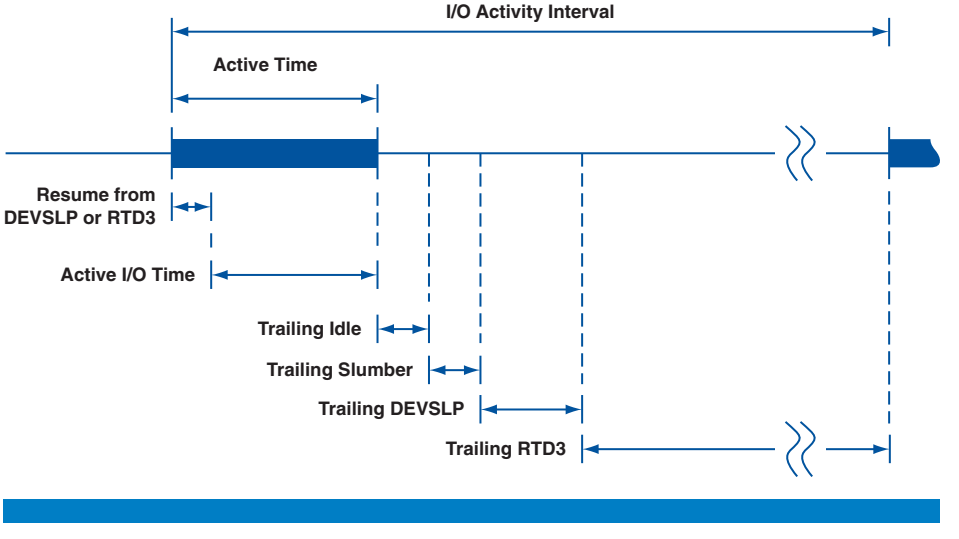


Figure 2: Storage power states with repeating I/O activity
(Source: Intel Corporation, 2012)

up from its low power state (resume from DEVSLP or RTD3). The system then issues one or more I/O commands to the storage device and remains active while commands are being executed. When all I/O are completed the active time comes to an end; the drive becomes idle, and then after a delay it is commanded to drop into slumber state. After another delay, the drive is commanded to go into DEVSLP state. After a final delay, the drive is issued a standby immediate command to ensure that all data is flushed to nonvolatile media, and then placed in RTD3 D3 (turned off). The next I/O issued by the system will again wake the device from the DEVSLP or RTD3 state and repeat this cycle.

SSDs versus HDDs

Solid state drives (SSDs) use flash memory to store data, meaning there are no moving read/write heads as in traditional hard disk drives (HDDs). SSDs read and write faster than HDDs; even the cheapest SSD models are much faster than mechanical spinning HDDs. For common specification-style benchmarks typical SSD drives are roughly 60 percent faster than hard drives for sequential access. SSDs often use less power when active and outperform HDDs by 500x for random access I/O benchmarks.

"SSDs read and write faster than HDDs; even the cheapest SSD models are much faster than mechanical spinning HDDs."

The primary drawback of SSDs is cost. Generally SSDs cost 10-20x more per GB than their HDD counterparts. For example, a 64 GB SSD can cost as much as a 3 TB hard drive. Thus, in system designs, SSDs account for less storage size than HDDs. SSDs have also been criticized for high performance variance and poor longevity[8].

Despite the many advantages of SSDs over HDDs, the associated cost for high capacity has limited SSD proliferation. While SSDs are appropriate for mobile devices to potentially enhance battery life, they are too expensive to provide the capacity needed in typical enterprise environments. While SSD costs remain high and with the growing popularity of cloud services, the most popular enterprise solution currently is the HDD with a RAID configuration for redundancy. In some cases, hybrid systems using SSDs as caches have been proposed to balance cost, capacity, and energy consumption[9].

"Despite the many advantages of SSDs over HDDs, the associated cost for high capacity has limited SSD proliferation."

Related Work

Previous efforts in storage energy efficiency focus on studies of the designs of new storage architectures combining HDDs and SSDs[10][11][12], the tradeoffs between energy and reliability[13][14], the designs of low power algorithms[15][16][17][18][19][20][21][22][23][24], and the tradeoffs between power and quality of service (QoS)[25][26][27].

A number of projects combine HDDs with SSDs[10], or other mixtures of components with different capabilities to build up an energy-proportional storage framework[11][12]; that is, obtaining energy consumption proportional to "load," an idea that has been well studied in the context of CPU utilization.

Some address system design tradeoffs such as energy versus reliability[13], reliability requirements for intermediate data[14], and energy efficient MapReduce[15][17].

Research on storage caching algorithms is relatively mature addressing important areas such as distributed caching[18][19], prefetching[20][21], and write-caching[22][23]. Partly because of its simplicity, and partly because of mobile and cloud computing, replicated storage has seen a lot of developmental research recently[24][25]. Replica placement work increasingly addresses QoS tradeoffs and guarantees[25][27]. Note that work addressing replica placement in tree structures is relevant to designing caching strategies for large-scale "tiered" storage systems of commercial interest. There is a close relationship between caching strategies and replication. One useful approach is to view replication as caching on a persistent storage device.

"The primary goal of this work is to minimize overall power use within storage device categories for common usage scenarios."

The primary goal of this work is to minimize overall power use within storage device categories for common usage scenarios. In contrast to previous approaches, this work is primarily focused on reducing storage power by proposing novel software techniques that minimize end-to-end energy consumption. In particular, we use an application-driven approach to optimize several scenarios on both client devices and the back-end enterprise system providing services. This article examines an arsenal of techniques to achieve higher power efficiencies in the context of media playback, always-connected scenarios, and block swap policy management.

Notebooks/Ultrabooks™ Storage Energy

A Notebook computer provides the convenience for portability with enough computing capability to support web browsing, photo editing, media playback, and many other usage scenarios. Battery life is a key criterion for users evaluating notebook products. Intel-based notebooks, called Ultrabooks, were designed to extended battery life without compromising performance. Ultrabooks have a minimum of five hours of battery life and up to eight hours of all-day usage. Therefore, it is very important for Ultrabooks to use low-power software optimization techniques to aggressively utilize the component-level power management on each device to extend the battery lifetime. This section discusses the software optimization on the side of the storage device.

"It is very important for Ultrabooks to use low power software optimization techniques to … extend the battery lifetime."

From Figure 2, it is observable that the key to save energy for storage devices is to consider maintaining the device in the low power state (DEVSLP or RTD3) for as long as possible between two I/O intervals. On the hardware, several points can be improved including the wakeup interval (DEVSLP and RTD3 to active time), the idle time, slumber timeout (time to transition, see Table 1), and even the power consumption in the DEVSLP state. Software should aim to prevent the device from frequently being woken up by consolidating I/O read/write commands, which will extend the I/O interval between I/O activities and provide the device with more time to spend in DEVSLP/RTD3 low power states.

Several optimization techniques will help reduce the storage power. Depending on the workload patterns, each type of usage scenario requires different software optimizations for the best use of the storage power states to reduce power. In the next two sections, we continue to discuss which optimizations can be applied for media playback and always-connected usage scenarios and, later on in the section "Storage Power Optimization Results," we will introduce a storage power simulation to demonstrate how much energy can be saved on the storage device from applying these optimization techniques.

Case Study: Storage Power Optimization for Media Playback

Power consumption during media playback can be understood fully once a complete platform activity picture is painted. Figure 3 shows events across the platform that impact total platform power. There are three processing functions that are handled on the core today in most media playback execution: audio processing, video processing, and associated interrupt processing. The storage interrupts are the hardware I/O interrupts generated by Wi-Fi* and storage devices.

"Power consumption during media playback can be understood fully once a complete platform activity picture is painted."

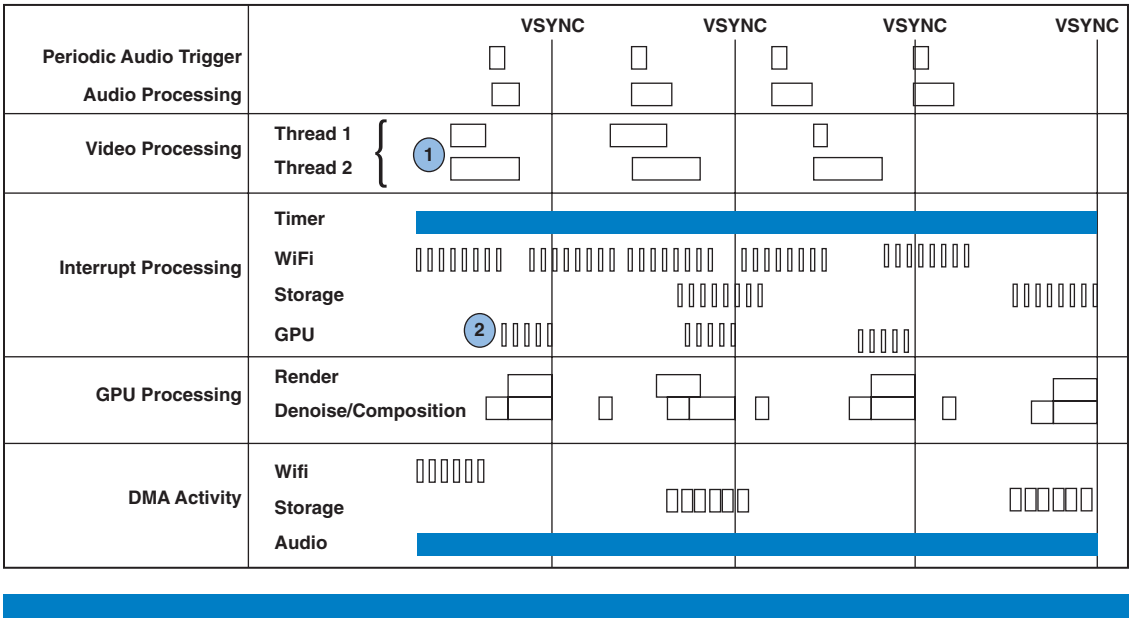


Figure 3: Platform activity profile during media playback (Source: Intel Corporation, 2012)

Media players today access storage every 2–3 frames (which translates into intervals of 60–100 ms). This time interval is inadequate for storage devices to utilize their low-power states during media playback. Ideal applications should buffer content for 10 seconds or more and then put storage devices to sleep. While 10 seconds worth of pre-buffering is sufficient for solid state devices, hard drives and hard drives with a solid state cache require much longer idleness windows to transition to low power state. For hard-drive–based media playback, multiple minutes of pre-buffering is required to reduce storage power substantially.

"Ideal applications should buffer content for 10 seconds or more and then put storage devices to sleep."

"Users expect their devices to have the latest, freshest data whenever they access them, and they expect the device to go multiple days without requiring a recharge."

Case Study: Storage Power Optimization to Extend "Always Connected" Battery Life

The proliferation of mobile devices with "always connected" characteristics combined with long battery life has raised the bar for power consumption on Ultrabook platforms. Users expect their devices to have the latest, freshest data whenever they access them, and they expect the device to go multiple days without requiring a recharge. From a storage power perspective, frequency of access is again a key parameter determining average power consumption, followed closely by managing the duration of time spent in higher power states during transitions in and out of active operation.

For the "always connected" scenario, the workload is modeled as repeating I/O interval consisting of a period of I/O activity followed by a period of I/O inactivity as shown in Figure 2. The duration of the period of activity and period of inactivity are set by the application. As a result, the Active Time and Inactive Time are determined by the applications' I/O pattern. During the Inactive Time, the storage device enters progressively lower power states. The deeper power state the storage device is in, the more electronic parts in the device are turned off, and, thus, more power savings is expected.

An additional source of power consumption is the energy expended when the drive transitions between power states; this is referred to as transitional energy. For instance, when an SSD turns off, it must first save its internal state; this can involve writing hundreds of megabytes of data of a period of hundreds of milliseconds, which consumes significant energy. Transitional energy has to be taken into account when determining when to make state transitions; transitioning to a lower power state may not result in a net power savings if the energy expended to enter (and exit) the state was greater than the energy expended by staying in a higher power state for the same amount of time. The time at which the energy consumed by power state *x* equals the energy consumed by transitioning into, and out of power state *x*-1 over the same period of time is referred to as the "recoup" time.

"There are two major strategies for reducing storage power in the always connected scenario…"

There are two major strategies for reducing storage power in the always connected scenario: 1) reduce in-state power and 2) reduce time in higher power states. SSD drives can reduce in-state power by, for instance using LP-DDR memory rather than conventional DRAM for storing drive state. More aggressive entry into lower power states has a performance impact, but it is more tolerable in "always connected" state.

Enterprise Storage Energy

Consolidation provides the most common single source of energy savings. Rather than have each desktop with an underutilized HD, virtualized storage pools all users so that vast excess storage capacity can be reduced. One buys fewer hard drives in total, expanding storage capacity as needed, and hopefully keeps each spinning hard drive doing some substantial amount of work. Estimates of how much datacenter energy usage is related to storage range from

10 to 40 percent, and depends on how the data center workload is weighted toward CPU/storage. For example, Google can use every CPU cycle they have, whereas other enterprises may need more storage capacity than CPU capacity.

Energy savings is one driver for the trend for enterprises to consolidate and virtualize storage and workloads. Surveys in the last years have rated energy efficiency as the top concern of between 20 and 40 percent of IT managers. In fact, the US federal government planned to shut down approximately 137 data centers in 2011, and 800 of its 2094 large data centers will be shut down by 2015. These workloads will migrate to larger, more energy-efficient data centers or cloud computing.

"Surveys in the last years have rated energy efficiency as the top concern of between 20 and 40 percent of IT managers."

Now once a workload has moved to a large data center, mechanisms besides consolidation can be used to make storage more energy efficient. *Tiering* is often set up as a pyramid of hardware: expensive SSD at the top, mid-layer SAN, and bottom layer of an array of commodity hard drives (RAID) with high resiliency (resistance to hardware failure). Using upper layers as a cache for all "active" data can allow idling and transition to lower-power modes. Once again, different environments will have different answers to how much storage should be fast SSD storage (high tier) and how much should be lower-speed HD storage (low tier).

In most cases data center storage has been kept agnostic about compute requirements; however, a major energy win can be targeted by turning off entire machines[16] when both storage and compute layer can agree. In this section we describe instances to explore storage system design. We introduce our job-based block swap policy with replica support and garbage collection. In the section "Storage Power Optimization Results," the EEffSim simulator will be presented, and the simulator will be used to evaluate the energy-optimized swap policy for the enterprise storage systems.

Case Study: Block Placement Policy for Distributed Storage Systems

To achieve energy saving among disks, block swap policies organize disks in an ordered list and attempt to relocate accessed blocks so that disk position eventually reflects access frequency and concurrency. This allows collocating frequently accessed blocks on a smaller subset of disks to enable turning off idle or less frequently used disks.

"To achieve energy saving among disks, block swap policies organize disks in an ordered list."

Job-based Swap Policy

We used the current disk I/O rates (averaged over, say, 600 seconds) and disk ON/OFF state to determine if we should swap the data block or not, using a set of heuristic rules: (i) if all disks are operating at a comfortably level of activity, block swapping can be avoided altogether; (ii) block swapping destinations could be remembered, as swapping jobs, to help keep client blocks on a reduced number of target disks; (iii) hysteresis within conditions triggering job creation can avoid noisily switching jobs. Corrective actions were simplistic jobs that swapped all I/O directed to a source disk to a given destination disk. Job deactivation could occur when the triggering condition was returned to well

within the normal regime, if the swap destination I/O rate became too high, or if either source disk or destination disk turned off. The key concept here is to activate block-swap jobs when it is really obvious something needed fixing.

High-Latency Outliers

There is a challenge of applying job-based block-swap policy. When two consecutive writes of the same block to an OFF disk occurred, the second write could be delayed considerably by intervening block-swap operations. To address this, we implement a message queuing model that could handle foreground and background messages within separate queues at each node. This leads, in turn, to implementing more complex lock management, since locks for background operations were observed to interfere with the fast path of client (foreground) I/O. In particular, it is useful for the initial read phase of a background block-swap to take a revocable read lock. When a foreground client write operation revokes this lock, the background operation can abort the block-swap transaction, possibly adopting an alternate swap destination and restarting the block swap request.

"… we implement a message queuing model that could handle foreground and background messages within separate queues at each node."

Replica Support and Garbage Collection

An initial implementation uses a data structure highly optimized for the case where a single replica is the only (or majority) case, where default mappings and other tricks are used to save a large amount of memory. In such systems, garbage collection (GC) is a relatively crude "brute force" operation, not intended to reflect a realistic GC. It enables us, nevertheless, to identify a highly desirable mode of operation. In the next section we present some early simulation results of moving from a system based on zero free space and full block swap to one with free space and GC. Note that the policies are still of a most fundamental type, where dynamic state is system load information and static state is limited only to maintaining job descriptors for the Job-based swap policy.

Storage Power Optimization Results

This section describes the power metrics that can be used to evaluate the effectiveness of the low power optimization techniques for minimizing storage power followed by discussion of some of the results achieved by applying these optimizations on different usage scenarios.

Notebooks/Ultrabooks™ Storage devices

We have discussed software power techniques for the media playback scenario and the "always connected" scenario on notebooks and Ultrabooks back in section 5. Here we evaluate the power techniques on a real instrumented system and on a spreadsheet simulator.

Media Playback Optimization for Storage Power

System on measuring storage device power: We use an Ivy Bridge Software Development Platform (SDP), which is configured with a two-core Ivy Bridge processor with 4 GB DDR3 memory running the Windows* 7 64-bit operating system. The Ivy Bridge SDP is pre-instrumented for storage device

power measurement for the NetDAQ system. The average data rate is held constant at 1.2 MB/s. The experiment is carried out using a test application to read data from a file with adjustable sleep time to emulate buffer time. From test to test, we reboot the machine and wait for at least 5 minutes to eliminate the background noise.

Results: Figure 4 shows the HDD average power consumption for different storage access intervals. With a 100-millisecond interval the HDD is at 1.4 watt average power consumption. With a 1-second access interval, which is a 10x longer access interval than the 100 millisecond baseline, the HDD power savings is increased by 21 percent. As the access interval scales to 20 seconds, the power savings of HDD is almost 30 percent compared with the 100-millisecond access interval.

"With a 1-second access interval … HDD power savings is increased by 21 percent."

Figure 4 and Table 2 describe the average power of a SSD storage device with buffer time from 0.1 seconds to 10 minutes. With a 1-second access interval, the power savings of the SSD is about 60 percent compared with the 100-millisecond access interval. As we adjust the access interval to 5 seconds and 10 seconds, the SSD power drops to 154 milliwatts and 89 milliwatts separately, which is 70 percent and is 83 percent compared to the 0.1-second baseline. The SSD power savings benefits more from the software optimization of buffer time approach than the same approach on the HDD storage device.

Access Interval	100 ms	1 s	5 s	10 s	20 s
SSD Average Power	0.5 W	208 mW	154 mW	89 mW	
HDD Average Power	1.4 W	1.1 W	1.02 W	1.14 W	0.98 W

Table 2: Media Playback Buffering
(Source: Intel Corporation, 2012)

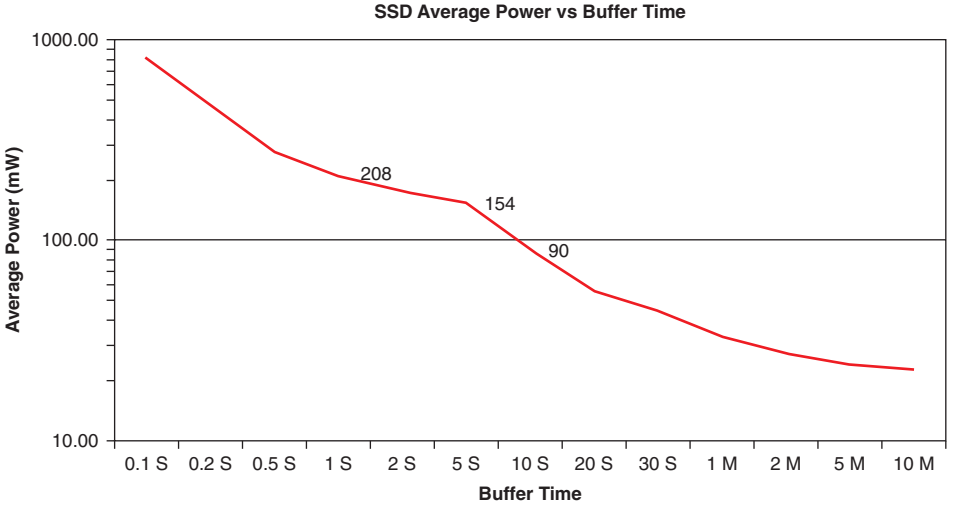


Figure 4: HDD/SSD average power versus access interval
(Source: Intel Corporation, 2012)

"Storage can dissipate a significant amount of power if accessed frequently."

The storage can dissipate a significant amount of power if accessed frequently. Similar to other components on the platform, storage devices require a period of idleness to transition to low power state and decrease power consumption. The idleness required is recommended to be greater than one second before a storage device transition into a low power state.

"Always Connected" Optimization for Storage Power

Storage power simulator: We develop a spreadsheet storage power simulator for the "always connected" scenario. In this simulator, the "always connected" I/O access pattern is modeled as a recurring pattern of brief system activity in which application data is updated from the network, followed by a longer interval in which the platform is in its lowest power state; the combined active and low power time is referred to as the wake interval, as shown in Figure 2. The parameters as shown in Table 3 provide the in-state power and time, and transition energy and time, for the state transitions that take place in a single wake interval.

SSD Storage Characteristics	
Active power	1.3 W
Idle power	800 mW
Trailing idle time	10 ms
Trailing slumber time	5 s
Slumber Power	125 mW
DEVSLP Power	5 mW
DEVSLP Resume energy	100 ms @ 1.3 W
RTD3 Context Save Data	256 MB
RTD3 D3 (Context Save) Energy	1.2 s @ 2.6 W
RTD3 D0 (Context Restore) Energy	200 ms @ 1.3 W

Table 3: Storage Simulator Parameters
(Source: Intel Corporation, 2012)

Here we discuss the implementation of the storage power simulator in detail. For the simulator to calculate the associated energy, several time intervals are needed. The time intervals required for the simulator are (1) wake interval time, (2) total I/O time, (3) time in low power states while I/O is in progress, (4) what power state is transitioned into in the remaining time, and (5) time spent transitioning into the selected low power state.

In the simulator, the (1) wake interval time is fixed. The (2) total I/O time is subtracted from the wake interval time. The simulator assumes the I/O time increases with the interval time, because the same amount of data is transacted per unit time, regardless of the activity time. And, on determining the (3) time spent in low power states while I/O is in progress, the simulator can model the I/O as distributed evenly across the active time or model the I/O as arriving in

bursts that are distributed evenly across the active time. Moreover, determining (4) what power state is transitioned into in the remaining time is based on the whether the power state can be entered and exited in the available remaining time, and whether the transitional energy will be recouped in the available time, compared to the next highest power state. In practice, the drive always transitions into slumber; it may then transition into DEVSLP or RTD3 D3 state based on the above criteria. Last, the simulator determines the (5) time spent transitioning into the selected low power state as well as the time spent transitioning out of the lowest power state.

Results: During normal operation the timeout from slumber to DEVSLP is maintained at 5 seconds in order to minimize the impact to resume time from DEVSLP on I/O performance. In the "always connected" case, performance is less critical, and the 5-second waits in slumber account for the majority of power consumed by storage at short wake intervals. Figure 5 shows the distribution of power consumption for an SSD based on the storage power simulator and the power and transition time assumptions shown in Table 3.

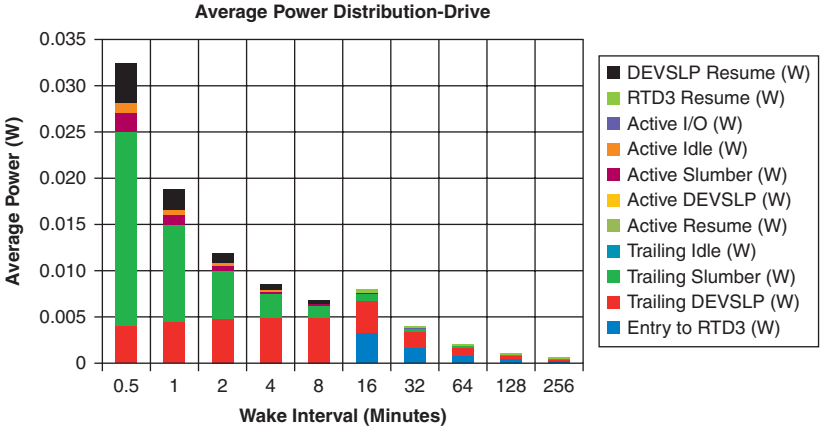


Figure 5: Average power versus wake interval, with 30 minutes wake interval for storage power in the 5 mW range (Source: Intel Corporation, 2012)

For a system to achieve multiple days of idle battery life with a reasonably sized (and priced) battery requires total platform power consumption while idle on the order of 100 mW. With the budget for storage power in the 5 mW range, the system cannot wake more frequently than every 30 minutes; from a "latest, freshest data" perspective, 30 minutes is clearly an unacceptable cadence for updating users data.

"… to achieve multiple days of idle battery life with a reasonably sized (and priced) battery requires total platform power consumption while idle on the order of 100 mW."

Figure 6 shows the distribution of power consumption for an SSD with slumber power reduced by 4x, context save energy reduced by 5x, and trailing time in slumber reduced to 1 second. Compared to the baseline shown in Figure 5, power consumption for 30-second wake interval is cut by one third, and the 5-mW wake interval has moved from 30 minutes to 8 minutes.

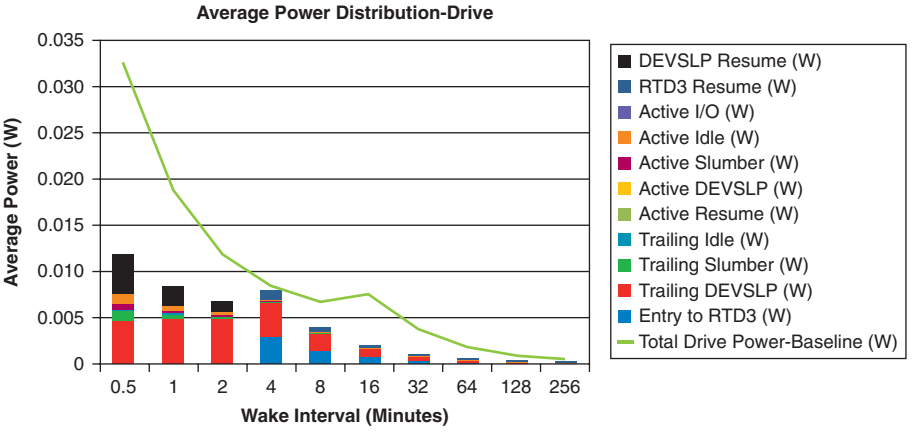


Figure 6: Average power versus wake interval, with 8 minutes wake interval for storage power in the 5 mW range
(Source: Intel Corporation, 2012)

If software can indicate to the storage driver the average time until the next active interval, then the drive can immediately transition to the power state with the best recoup time, as shown in Figure 7. With this optimization in place, power consumption is now below 5 mW at the 4-minute wake interval.

"In the "always connected" case for a system to achieve multiple days with a reasonable battery capacity without charging, the optimization involves both hardware and software approaches."

In the "always connected" case for a system to achieve multiple days with a reasonable battery capacity without charging, the optimization involves both hardware and software approaches. The SSD storage device should focus on power reduction at the slumber, DEVSLP power states, and the energy spends in context save/restore operations. A software approach is used to ensure longer wake interval and shorter DEVSLP wait.

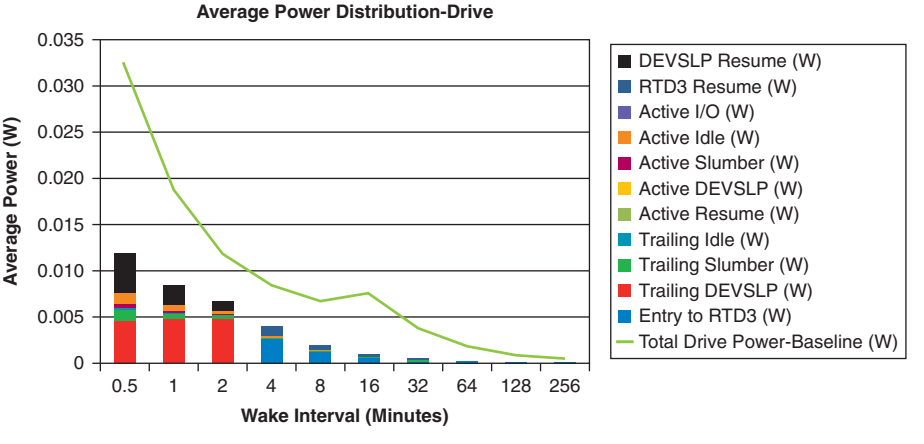


Figure 7: Average power versus wake interval, with 4 minutes wake interval for storage power in the 5 mW range
(Source: Intel Corporation, 2012)

Enterprise Storage Devices

In this work we use both the network model[28] and the DiskSim[29], but with much-simplified abstractions that gloss over fine details of both network and device to obtain energy usage estimates of heterogeneous storage devices including SSDs.

EEffSim Simulator

We have built the EEffSim[30] simulator as shown in Figure 8 that allows:

- A framework to compare block placement policies for energy savings and frequency of high-latency events (> 1 s)
- Investigation of novel SSD-friendly data structures that may be useful in real implementations
- Speedy simulation (> 1_106 messages per second) of very large distributed storage systems using approximations in modeling disk access latencies
- Accuracy in low-IOPS (I/O per second) limit, with good determination of energy savings due to disk state changes

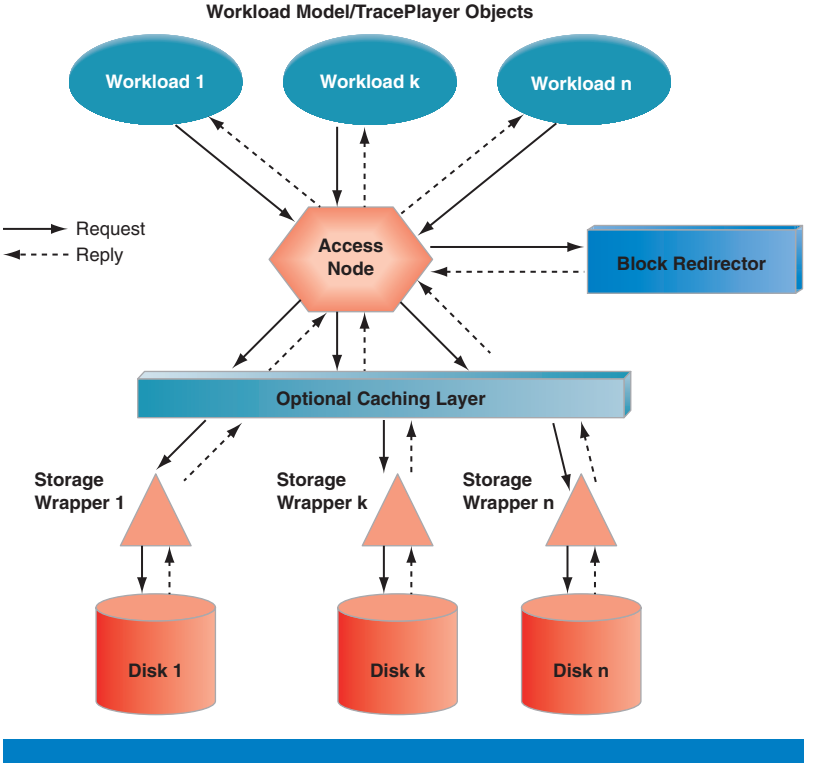


Figure 8: Overview of major components in EEffSim Simulator (Source: NEC Labs America, 2012)

To this standard discrete event simulator core we have added support for block swap operations. The swap operation protocol is built to present minimal delay to the fast path of a client request—in fact, in a tiered solution, readjustment of lower layer data layout can be done with no effect on client latency. Figure 8

"Storage wrappers have two main functions: to select an LRU block and to track free space."

shows various objects that can populate simulator test cases. Storage wrappers have two main functions: to select an LRU block and to track free space. The block redirector handles background operations for full block swap, or for the faster replica-creation operations that are possible if the system is aware of free space. Because disk spin-up is on a time scale 3–4 orders of magnitude larger than typical disk access times, errors on the level of milliseconds for individual I/O operations contribute negligibly to block swapping policies governing switching of the disk energy state. This approximation holds well in the low-IOPS limit, where bursts of client I/O do not exceed the I/O capacity of the disk, and accumulated approximation errors in disk access times remain much smaller.

"… even very simple online heuristics, in which swap decisions are made as incoming I/O requests are received, are capable of substantial energy savings."

Results

Here we show that even very simple online heuristics, in which swap decisions are made as incoming I/O requests are received, are capable of substantial energy savings. The online approach can minimize the I/O with respect to epoch-based data reorganizations, where the data to be moved may need to be reread, as well as quick response to non-optimal data arrangements. We will also show that large benefits can be gained by moving from generic full block swap to using replication.

The energy usage results for no-swap and job-based block swap policies are presented in Table 4, where disks turn off after 15-second idle. The first column describes how many times each of the daylong traces ran. The second column represents the energy usage (in joules) for the no-swap policy, which disables block-swaps. This state is a non-optimal configuration of one disk per client. Over a one day trace period, no-swap clients maintained their dedicated disks on for 14–100 percent of the time, averaging 63 percent on. The third column shows energy usage with respect to the no-swap policy, using a simple job-based policy. As the system learns the client access patterns, energy usage is about a third of what it originally was. Writes are offloaded as desired, without later rewriting them to some original location. The job-based policy is based on detecting overloaded and underloaded disks (or total ON disks not proportional to total load), and creating swapping hints such as "swap from disk A to disk B." These swap hints are active until conditions on disks A and B allow their removal (hopefully having removed the original underload/overload condition).

# of repeats	no-swap (joule)	Job-based
1	578275	61.9%
2	1156475	42.8%
4	2312867	33.3%
8	4625621	28.4%

Table 4: Cumulative energy usage for no-swap and job-based policies (Source: NEC Labs America, 2012)

The traces we used here are taken from Verma et al.[12] where an epoch-based subvolume replication method is used to reorganize data. They found that data reorganization led to 0.003 percent of requests suffering high latency due to unavoidable disk spin-ups; we found (with no replication and including the full warm-up period) 0.005 percent with high latency. Client I/O patterns are displayed graphically in Figure 9 for the SRCMap traces. Note that these traces are on average quite inactive, which means very simple and quite aggressive swap policies can perform quite well.

"… that these traces are on average quite inactive, which means very simple and quite aggressive swap policies can perform quite well."

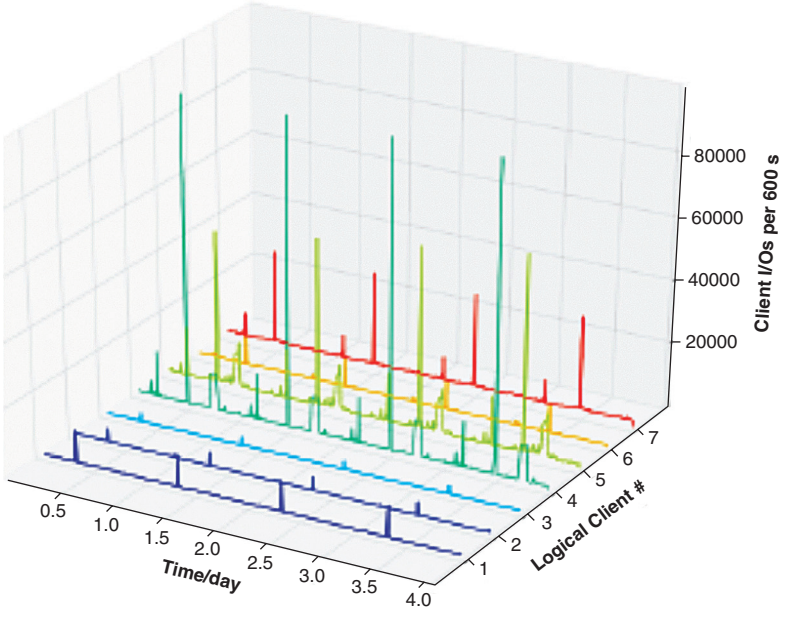


Figure 9: Client I/O characteristics with no swapping and each client assigned to a unique disk
(Source: NEC Labs America, 2012)

We now illustrate the benefits arising from moving from mere data reorganization to replication. In Figure 10 and Figure 11, we plot system behavior for the SRCMap traces of Figure 8 using the same aggressive job-based swap policy repeating the traces 4 times and using = 2 minutes. Figure 10 shows no swap, and Figure 11 shows "swapping" (replication) with 5 percent free space and garbage collection. The no-swap case (Figure 10) has every client assigned to a unique disk and results in disks being on about 80 percent of the time (left Figure 10). Disk I/O is very low except during spikes of client activity. In the legend of the middle Figure 10, the <dl>=10 s value refers to a total client latency above what the disk could provide, not including the long waits for disks to spin up. It is a crude QoS indicator reflecting the sum of response time above the nominal random read time.

Other QoS indicators were monitored too, such as full I/O latency distributions, disk spin-up statistics and I/O rates. On the right of Figure 10 we see that even with a 2 minutes idle time before turning off, dedicated

"Other QoS indicators were monitored too, such as full I/O latency distributions, disk spin-up statistics and I/O rates."

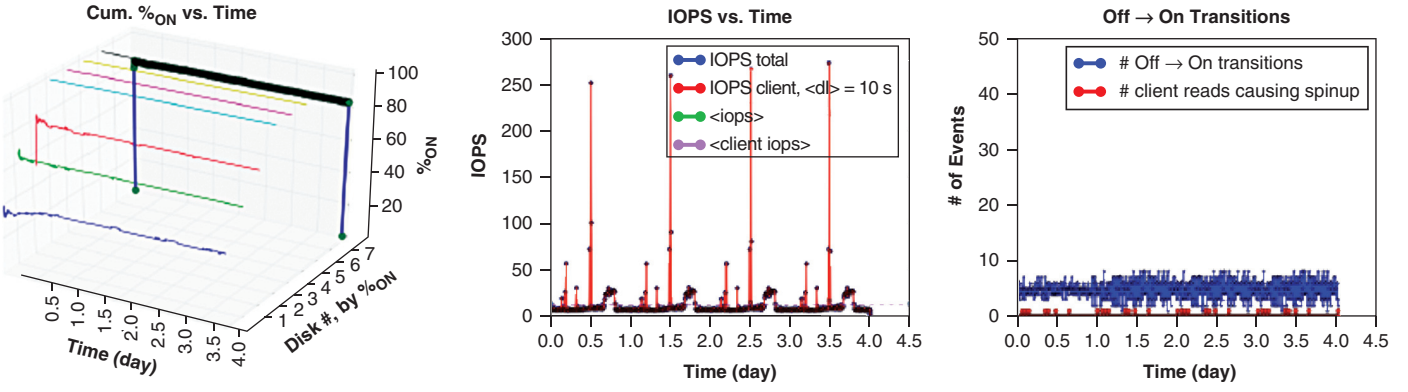


Figure 10: "Trace" clients, no swap. Rows: cumulative I/O rate at each disk, client, and total IOPS and transition counts (Source: NEC Labs America, 2012)

disks turn on and off very many times a day (this can be unacceptable in data centers due to concerns about disk lifetime). Here, though, we focus on energy efficiency more, since additional refinements (such as investigating more than just two queuing models, and definitively addressing how load factors are measured, for example) may result in larger QoS improvements than minor swap policy refinements.

In Figure 11, swapping is optimized to a replication operation and avoids these lru-disk spin-ups. The net effect of reducing the impact of background data migration yields: (i) appreciable energy savings, (ii) dramatically reduction in the number of disk spin-ups, (iii) only improvement possible for QoS. As evidenced in Figure 11, we have well under 20 percent energy usage, <dl>=21 s, even fewer disk spin-ups and almost no data reorganization apart from the first day. Note that if incorporated within a file system, free space is known and we would not need a separate block-level tracking.

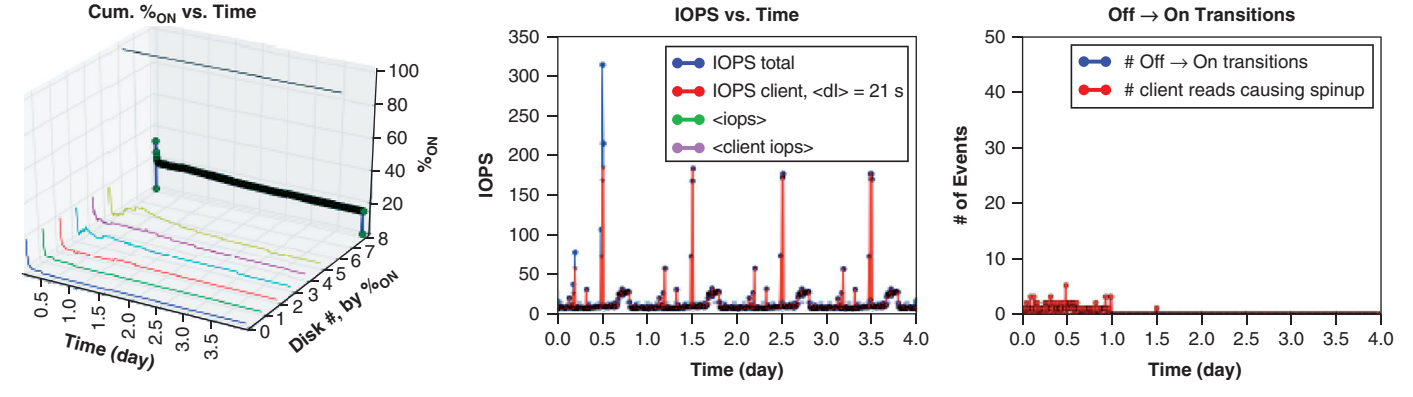


Figure 11: "Trace" clients, aggressive job-based swap policy with 5 percent free space, secondary replicas, and garbage collection. Rows: cumulative I/O rate at each disk, client, and total IOPS and transition counts (Source: NEC Labs America, 2012)

We use more sophisticated and conservative swap policies to handle larger sets of clients of high-rate, extremely bursty I/O patterns. As clients become more bursty, and demand greater disk bandwidth, the swap policy must become more conservative in order to maintain QoS. When the system is pushed to the limit, with demanding synthetic clients, swapping activity must be reduced to very conservative levels to avoid impacting client QoS, and convergence to an efficient operating regime can take a long time. For such cases, we have developed more conservative and robust swap policies, able to take into account not just current system load, but also client I/O and correlation statistics. Naive swap hints must be replaced by a more global estimate of the total expected background swapping load on the storage devices. For simulations with 100 or more clients it is useful to scale down disk and client rates, as well as working set sizes, by a factor of 10 or so: for many test cases, energy efficiency predictions are affected less than 5 percent by such scalings.

"When the system is pushed to the limit, with demanding synthetic clients, swapping activity must be reduced to very conservative levels to avoid impacting client QoS."

Summary and Conclusions

This article described the storage energy consumption in notebook and Ultrabook portable systems as well as in an enterprise environment, and identified the potential power savings in storage devices for the client/server systems. Specifically, media playback and "always connected" usage scenarios are discussed in Ultrabook devices, and the block swap policy was talked about in server systems. Power savings can be achieved by keeping the storage device in low power state as much as possible; low power optimization techniques can be implemented in the application and the OS/driver to minimize the overall platform power. The article described at length several low power techniques such as buffering content in media playback, wake interval power optimization in "always connoted" scenario, and swap policies with replica support and garbage collection in enterprise systems. It showed the power benefit that can be realized by use of these techniques in each of the target usage scenarios for client/server environment. With the shift towards the mobile/cloud computing paradigm, such low power techniques are the key to providing a longer battery life experience to the end users and to supporting a lower cost infrastructure for the cloud service hosting enterprises.

References

[1] B. Bosker, "Facebook IPO Filing Reveals Its Stunning Size: A Private Jet, $1 Billion In Profits, And More," The Huffington Post. Retrieved: 3/1 2012. Available: http://www.huffingtonpost.com/2012/02/01/facebook-ipo-filing-revea_n_1248434.html

[2] "The Efficient Data Center," Intel Corporation, White Paper, November 2010. Available: http://www.intel.com/content/dam/doc/white-paper/cloud-computing-efficient-data-center-paper.pdf

[3] "Serial ATA Advanced Host Controller Interface (AHCI) 1.3," Intel Corporation, Reference Guide, April 2011. Available: http://www.intel.com/content/dam/www/public/us/en/documents/technical-specifications/serial-ata-ahci-spec-rev1_3.pdf

[4] "Serial ATA Device Sleep (DevSleep) and Runtime D3 (RTD3)," Intel Corporation and SanDisk Corporation, White Paper, Dec 2011. Available: http://www.sata-io.org/documents/SATA-DevSleep-and-RTD3-WP.pdf

[5] "Designing Energy Efficient SATA Devices," Intel Corporation, Reference Guide, April 2011. Available: http://www.intel.com/content/dam/doc/reference-guide/sata-devices-implementation-recommendations.pdf

[6] "PCI Express Architecture Power Management," Intel Corporation, White Paper, Nov 2002. Available: http://www.intel.com/content/dam/doc/white-paper/pci-express-architecture-power-management-rev-1-1-paper.pdf

[7] "SATA Power Management: It's Good to Be Green," www.serialsata.org, White Paper, April 2009. Available: http://www.sata-io.org/documents/SATA_Power_Management_article_final_04_08_2009.pdf

[8] P. Schmid and A. Roos, "Should You Upgrade? From A Hard Drive To An SSD," Retrieved: 3/1 2012. Available: http://www.tomshardware.com/reviews/ssd-upgrade-hdd-performance,3023.html

[9] A. Ku, "OCZ RevoDrive Hybrid: Solid-State Speed With Hard Drive Capacity." Retrieved: 3/1 2012. Available: http://www.tomshardware.com/reviews/revodrive-hybrid-ssd-hard-drive,3044.html

[10] H. Jo, Y. Kwon, H. Kim, E. Seo, J. Lee, and S. Maeng, "Ssd-hddhybrid virtual disk in consolidated environments," in Proceedings of the 2009 international conference on Parallel processing (Euro-Par'09), pp. 375–384, 2010.

[11] J. Guerra, W. Belluomini, J. Glider, K. Gupta, and H. Pucha, "Energy proportionality for storage: impact and feasibility," ACM SIGOPS Operating Systems Review, vol. 44, no. 1, pp. 35–39, 2010.

[12] A. Verma, R. Koller, L. Useche, and R. Rangaswami, "SRCMap: energy proportional storage using dynamic consolidation," in Proceedings of the 8th USENIX conference on File and storage technologies, pp. 20–20, 2010.

[13] A. Gharaibeh, S. Al-Kiswany, and M. Ripeanu, "ThriftStore: Finessing Reliability Tradeoffs in Replicated Storage Systems," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 6, pp. 910–923, 2011.

[14] S. Ko, I. Hoque, B. Cho, and I. Gupta, "Making cloud intermediate data fault-tolerant," in Proceedings of the 1st ACM symposium on Cloud computing, pp. 181–192, 2010.

[15] Y. Chen, L. Keys, and R. Katz, "Towards energy efficient MapReduce," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-109, Aug 2009.

[16] R. T. Kaushik and M. Bhandarkar, "GreenHDFS: towards an energy-conserving, storage-efficient, hybrid Hadoop compute cluster," in Proceedings of the 2010 international conference on Power aware computing and systems (HotPower'10), pp. 1–9, 2010.

[17] Y. Chen, A. Ganapathi, A. Fox, R. Katz, and D. Patterson, "Statistical Workloads for Energy Efficient MapReduce," EECS Department, University of California, Berkeley, Technical Report UCB/EECS-2010-6, 2010.

[18] B. Gill and L. Bathen, "Optimal multistream sequential prefetching in a shared cache," ACM Transactions on Storage (TOS), vol. 3, no. 3, p. 10, 2007.

[19] M. Tawarmalani, K. Kannan, and P. De, "Allocating objects in a network of caches: Centralized and decentralized analyses," Manage. Sci, vol. 55, no. 1, pp. 132–147, 2009.

[20] Z. Zhang, K. Lee, X. Ma, and Y. Zhou, "Pfc: Transparent optimization of existing prefetching strategies for multi-level storage systems," in The 28th International Conference on Distributed Computing Systems (ICDCS '08), pp. 740–751, June, 2008.

[21] S. Bhatia, E. Varki, and A. Merchant, "Sequential prefetch cache sizing for maximal hit rate," in IEEE International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 89–98, August, 2010.

[22] B. S. Gill, M. Ko, B. Debnath, and W. Belluomini, "Stow: a spatially and temporally optimized write caching algorithm," in Proceedings of the USENIX Annual technical conference (USENIX'09), pp. 26–26, 2009.

[23] A. Batsakis, R. Burns, A. Kanevsky, J. Lentini, and T. Talpey, "Awol: an adaptive write optimizations layer," in Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST'08), pp. 1–14, 2008.

[24] X. Sun, J. Zheng, Q. Liu, and Y. Liu, "Dynamic Data Replication Based on Access Cost in Distributed Systems," in 4th International Conference on Computer Sciences and Convergence Information Technology, pp. 829–834, 2009.

[25] C. Cheng, J.Wu, and P. Liu, "QoS-aware, access-efficient, and storage efficient replica placement in grid environments," The Journal of Supercomputing, vol. 49, no. 1, pp. 42–63, 2009.

[26] N. Xiao, W. Fu, and X. Lu, "Qos-aware replica placement techniques in data grid applications," SCIENCE CHINA Information Sciences, vol. 53, pp. 1487–1496, 2010. 10.1007/s11432-010-4036-3.

[27] M. Bsoul, A. Al-Khasawneh, Y. Kilani, and I. Obeidat, "A threshold-based dynamic data replication strategy," The Journal of Supercomputing, pp. 1–10, 2010. 10.1007/s11227-010-0466-3.

[28] A. Varga, "OMNeT++," in Modeling and Tools for Network Simulation, pp. 35–59, 2010.

[29] J. S. Bucy, S. Jiri, S. W. Schlosser, G. R. Ganger, and Contributors, "The disksim simulation environment version 4.0 reference manual," Carnegie Mellon University Parallel Data Lab, Technical Report CMU-PDL-08-101, May 2008.

[30] R. Prabhakar, E. Kruus, G. Lu, and C. Ungureanu, "EEffSim: A discrete event simulator for energy efficiency in large-scale storage systems" in The 2011 International Conference on Energy Aware Computing (ICEAC'11), pp. 1–6, 2011.

Author Biographies

Hung-Ching Chang received a BS degree in electronic engineering from Huafan University, Taiwan. He has been working toward the PhD degree in computer science at Virginia Polytechnic Institute and State University since 2007. He has worked as an intern in the Software Services Group at Intel. His major research interests include high performance parallel computing, power-aware computing, and computer architecture.

Erik Kruus began in chemical physics and was a tenured chemistry professor for 8 years at UQAM. After two years as a dot-com programmer he began work at NEC Research Labs, where he has been a research staff member for ten years, working in areas such as artificial intelligence, biophysics and, for the last five years, in the data storage department.

Thomas Barnes is a software architect in the Storage Technology Group at Intel Corporation where he is responsible for path finding solid state storage solutions for client platforms. Thomas has more than 20 years of experience in system and embedded firmware design, development, and performance optimization. Thomas studied electrical engineering at the University of Pittsburg and holds five patents.

Abhishek Agrawal has over 10 years of industry and research experience. He is currently a senior technical lead in the Software Services Group at Intel and leads software power enabling for Intel's client and Intel® Atom™ based platforms. He chairs the industry-wide power management working group for Climate Savers Computing Initiative and has authored several industry whitepapers and technical papers on energy-efficiency in refereed international

conferences and journals as well as co-authored a book entitled *Energy Aware Computing*. Abhishek is a member of IEEE and ACM and is on many industry panels and IEEE/ACM conference committees on green computing.

Kirk W. Cameron received a BS degree in mathematics from UF in 1994 and a PhD in computer science from LSU in 2000. He is currently an associate professor of computer science at Virginia Polytechnic Institute and State University. He directs the SCAPE Laboratory at Virginia Tech, where he pioneered the area of high-performance, power-aware computing to improve the efficiency of high-end systems. He has received numerous awards and accolades for his research and publications including the National Science Foundation Career Award in 2004, the Department of Energy Career Award in 2004, the USC COE Young Investigator Research Award in 2005, the Best Paper Nominee SC06, the VT COE Fellow in 2007, the IBM Faculty Award in 2007, the Uptime Institute Fellow in 2008, and was invited to the 2008 National Academy of Engineering Symposium. He is on the editorial board and the editor for the IEEE Computer "Green IT" column. He is a member of the IEEE and the IEEE Computer Society.

A TRANSFERABLE BLUEPRINT FOR ENERGY-EFFICIENT OPERATIONS: A MATURE DATA CENTER CASE STUDY

Contributors

Keith Ellis
Intel Energy and Sustainability Lab, Intel Labs Europe

Charles Sheridan
Intel Energy and Sustainability Lab, Intel Labs Europe

"Data center facilities, … constitute the fastest growing contributor to the ICT global carbon footprint."

"Mature data centers are indicative of the situation facing most of the existing building stock."

The ever-increasing digitization of modern life has resulted in the growth of the information and communication technology (ICT) sector. ICT accounts for approximately 2 percent of global carbon (CO_2) emissions and is comparable to the airline industry. ICT growth has seen the increased deployment of data center facilities, which constitute the fastest growing contributor to the ICT global carbon footprint.[1]

While there is an abundance of literature relating to newly designed or new resource-efficient data center developments, there is less focus on "mature" facilities commissioned at a time when sustainability was less a consideration. Mature data centers are indicative of the situation facing most of the existing building stock, where often the challenge is not a lack of technological options but rather a problem with understanding which actions will have the greatest impact on resource efficiency.

The project described in this article details how an adapted quality-based improvement methodology was utilized in pinpointing opportunities for significant, measurable energy reductions in the area of server load and HVAC, while offering a common language that challenged the often siloed approach to data center operations.

The methodology is posited as a generic transferable blueprint for energy- and resource-efficient operations that is complementary to energy management standards such as ISO 50001 (EN16001).

The article begins with a brief background introduction. Methodology selection is then discussed and this is followed by a methodology overview description. The project use-case detail is then outlined and the article concludes with a summary/discussion section.

Introduction

The project described in this article was born out of a number of separate but synergistic interests relating to the data center domain and sustainability trends.

At Intel, data centers are one of the largest consumers of energy outside of fabrication plants and account for about 70 percent of Intel's IT direct energy use. Similar to the wider ICT sector the majority of these data centers can be described as "mature data centers" and were designed with resilience rather than resource efficiency in mind. Typically, within such data centers, for every 1 watt that is supplied to IT equipment another 1 watt or more is consumed by supporting infrastructure, primarily cooling and power.

Intel Labs Europe (ILE) has an interest in ICT trends, energy efficiency research, and in understanding if process improvement methodologies are complimentary to a sustainability context. Intel IT had an increased appreciation of the impact of its data center operations in terms of energy consumption, while Intel Facilities had an interest in reducing its bottom line energy expenditures.

The project presented the team with the opportunity to align these separate interests with wider corporate sustainability goals.

Additionally having assisted the European Commission in developing the Code of Conduct for Data Centers, and having achieved participant status, ILE and IT were keen to build on the best practices contained within the Code and promoted by industry bodies like the Green Grid.

A cross-functional team, including IT, facilities, factory, and ILE members, conducted a holistic study of one mature data center ecosystem. The application of Lean Six Sigma (LSS) as the primary method focused the team on the issues with the greatest impact while Systemic Innovation for Teams (SIfT) helped in improving solution identification. The specific project goal was to improve the energy efficiency of the use-case data center. The overarching objective was to develop a "transferable blueprint" methodology for measuring, monitoring, and managing towards energy-efficient operations, whereby the most appropriate technologies could be selected as part of an effective solution.

Methodology Selection

Given energy trends and the pervasiveness of mature data centers within Intel and industry, ILE sought to analyze a test case data center to determine how its energy efficiency could be improved, thereby extending its viable lifespan. The overarching research objective was to understand the importance of people, process, and technology in the context of sustainable goals and to identify appropriate process improvement methodologies.

"The … objective was to understand the importance of people, process, and technology in the context of sustainable goals and to identify appropriate … improvement methodologies."

Improvement methodologies, such as Lean Six Sigma (LSS), Lean, Structured Problem Solving, Model-Based Problem-Solving, and Systemic Innovation for Teams (SIfT) all offered different context-dependent advantages, but no one methodology was considered to be adequate in all contexts.

Given the envisaged need for a process-oriented view of energy consumption the decision was made to position LSS at the core of any project methodology. The fundamental LSS equation is $Y = f(Xi)$, where Y corresponds to the output of a process and is a function of its Xs. If viewing a data center as a type of factory with defined inputs, outputs, and a process, then energy efficiency, Y, is a function of the efficiency of the contributing elements, Xs, of the data center ecosystem. The widely accepted data center PUE metric, designed to illustrate the ratio of energy consumption between IT and facilities within a data center ecosystem, is very much aligned to the $Y = f(Xi)$ formula. As such it was felt

LSS could help in identifying the specific elements that constituted the system under study and to what level those variables interacted and affected energy efficiency.

The decision to utilize LSS would also prove pragmatic as most stakeholders were accustomed to "lean thinking" and practices. Energy efficiency was thus presented as a special case of waste reduction. Additionally, the decision to use LSS resulted in strong support by the LSS program group, who wanted to challenge a lingering organizational perception that LSS methods were valid only in Intel's manufacturing environments.

"A common critique of six-sigma is that while it can pinpoint where to focus on improving, the method provides little guidance to determine how to improve."

A common critique of six-sigma is that while it can pinpoint where to focus on improving, the method provides little guidance to determine how to improve. Intel had included Advanced Systematic Inventive Thinking (ASIT) within its LSS training as a response to the need for guided ideation. However, it was felt from the outset of the project that the facilitated nature of SIfT could prove a valuable addition in augmenting the ideation process of the improve phase.

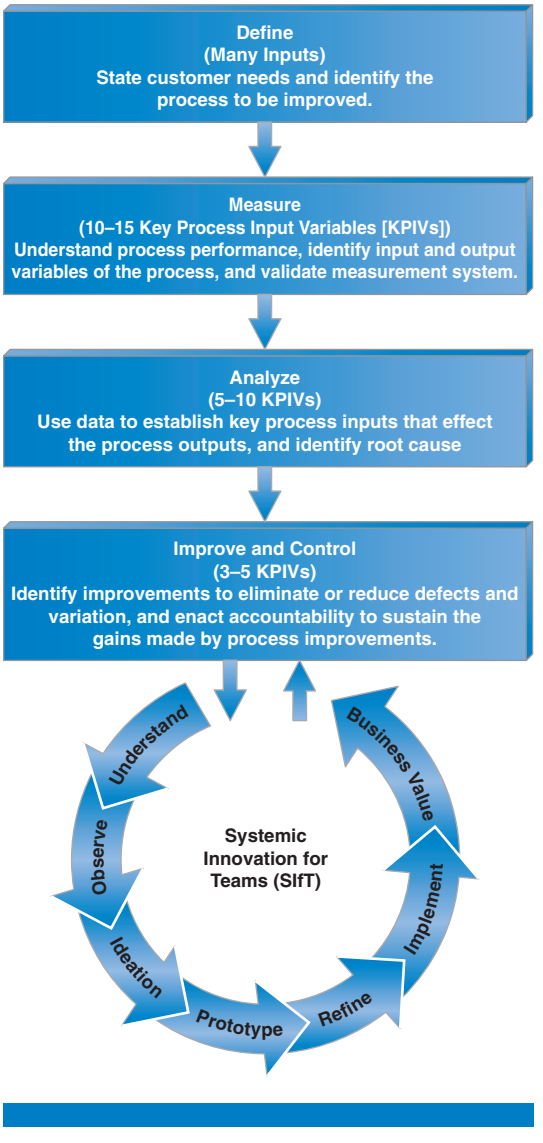


Figure 1: LSS and SIfT combined (Source: Intel Corporation, 2010)

Methodology Overview

The LSS methodology follows the five-step DMAIC process in narrowing the entire field of inputs down to the few key performance input variables (KPIVs) that have the most impact. For our project, we applied these steps to analyzing the energy efficiency of our project data center. A more detailed description of DMAIC follows.

LSS DMAIC

- *Define.* We defined our project charter; the cost of poor quality (COPQ); a list of suppliers, inputs, process, outputs, and customers (SIPOC); and our high-level processes. This step helped us identify what aspects of the data center energy consumption were important.
- *Measure.* We developed a process map and performed a measurement system analysis to establish a benchmark for how the data center was performing in each of the KPIVs identified in the Define step.
- *Analyze.* During this step, we analyzed the underperforming KPIVs for the data center, identified in the Measure step, to determine why they were underperforming. Some of the tools we used included a cause-and-effect matrix, failure mode and effects analysis (FMEA), multivariate analysis, regression, and statistical modeling functions.
- *Improve.* We identified what needed to be done. Important aspects of this step included application of predictive modeling and solution selection. SIfT played a key role in this step.
- *Control.* Having identified and implemented improvements we developed a control plan to ensure changes could be sustained overtime.

These steps are illustrated in the top half of Figure 1.

SIfT

Systemic Innovation for Teams is a step process targeted at enabling an organizations core asset, teams, to consciously innovate, consistently.[2]

"Systemic Innovation for Teams is a step process targeted at enabling an organizations core asset, teams, to consciously innovate, consistently."

The SIfT methodology is designed to help participants understand the what, why, and how of a challenge and to develop an innovative solution. SIfT uses a 6-step process: Understand, Observe, Ideation, Prototype, Refine, Implement, and Business Value as shown in the bottom half of Figure 1.

How it works is this: an organization or team has a problem or opportunity; it wants a better product, service, or process. An eclectic diverse team of experts and nonexperts is assembled (nonexperts are just as important!) to address the given challenge, by working through the following iterative process:

- *Understand.* The team first shares what they know about the opportunity to understand the current situation.
- *Observe.* They then go and observe and document the "user" experience (*user* is an expansive term here; it can be a human or even a wafer) to find the "points of inspiration."
- *Ideation.* A number of techniques can be employed here ranging from brainstorming to lateral thinking to TRIZ, depending on the depth required. As IDEO like to put it: "it is managed chaos: a dozen or so very smart people examining data, throwing out ideas, writing potential solutions on Post-its* and attaching to a wall."[3]
- *Prototyping.* The team then quickly mocks up no-frills prototypes of the best concepts that emerge. Prototyping is key; seeing ideas in a working tangible model is far more powerful than reading about it. Observation, ideation, and prototyping are the core trio in the SIfT process and the workshop focuses on participants learning and applying the techniques.
- *Refine.* After the team has defined possible improvements, the process transitions from expansion to contraction as the team narrows the choices down to a few possibilities focusing on the outcome with the customer.
- *Implementation.* Leverages what Intel excels at—flawless execution.
- *Business Value.* No innovation is complete until value is extracted. The final often-ignored step is business value extraction, where the team analyses and measures the business value impact of the innovation.

Using LSS and SIfT together

Both LSS and SIfT are process-based approaches to business improvement with the business value emphasis within SIfT being completely homogenous with the LSS "voice of the business" concept.

Both methods promote a cross-functional approach to problem resolution and guide users through sensing, understanding, deciding, and acting in ways that create value. Although LSS constituted the primary analysis methodology, the SIfT process significantly enhanced the Improve and Control LSS phases as identified in Figure 1.

"Although LSS constituted the primary analysis methodology, the SIfT process significantly enhanced the Improve and Control LSS phases."

The data gathered during the LSS Measure and Analyze phases fed the front end of the SIfT process, helping to ensure that there was strong factual basis to support the Understand and Observe steps of SIfT; in fact these two SIfT steps were essentially replaced by the LSS phases. In turn, the Ideation and Prototyping steps of SIfT fed back into the LSS Improve and Control phases. Specifically, the SIfT input impacted LSS factors such as Modeling and Solution Selection.

"The LSS concept of "voices" helped us balance … often contradictory views."

Any business improvement we recommended needed to take into account employees, cost efficiency, customers, and sustainability. The LSS concept of "voices" helped us balance these often contradictory views.

The voice of the customer (VoC), the voice of the business (VoB), and the voice of the employee (VoE) form the traditional critical-to-quality (CTQ) triangle—adding the voice of the environment and the voice of the process/product to form a pentagon, as shown in Figure 2.

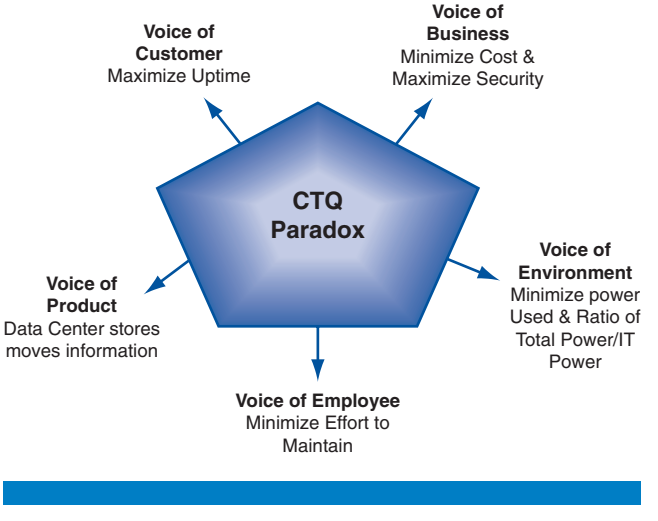


Figure 2: The CTQ paradox (Source: Intel Corporation, 2010)

"Adding the voice of the environment and that of the process/product alters the focus from purely economic … to sustainable …"

Adding the voice of the environment and that of the process/product alters the focus from purely economic (maximum benefit for minimum capital), to sustainable (maximum added value using a minimum of resources). In other words, Figure 2 illustrates the paradigm faced by organization in trying to deliver more while consuming less.

The Data Center Use Case

In accordance with the LSS emphasis on cross-functional improvement, the project team members were selected from diverse groups including Intel Labs, IT, and facilities staff. The team was assisted by an Intel LSS Master Black Belt whose purpose was to provide advanced guidance in applying methods and tools.

Define

The team developed a project charter and secured a project champion. In cannot be overemphasized how important this action was, as having buy-in would prove essential in driving support for the process and identified improvements.

"…having buy-in would prove essential in driving support for the process and identified improvements."

The team completed a SIPOC listing suppliers, inputs, process, outputs, and customers. This is essentially a scoping exercise and proved valuable in identifying all the relevant stakeholders. Engaging with those stakeholders early was key.

The team also began to map the data center ecosystem. This task helped to understand what aspects of the data center energy consumption were likely to be important. Iteration between the Define and Measure phase was to prove vital in ensuring completeness.

At this point the project had agreed two high level objectives:

- Improve subject data center energy efficiency by 15 percent.
- Develop a transferable blueprint for resource-efficient operations/ improvement identification.

Measure

The ideal state of a data center ecosystem: IT equipment should only draw power when performing useful work and cooling infrastructure should dynamically match the heat produced by the IT equipment.

In mature data centers it's often the case that IT load and cooling are not dynamically matched. For example, servers are typically left powered on, consuming energy while idle; data center air handler units often have fixed-speed fans that wastefully overcool. We set about trying to map our data center to determine what was dynamic and what was fixed. That process began with the heat source, the IT systems.

Measuring IT Load

As with any measurement exercise we started with defining how to measure. The use of the metered uninterruptable power supply (UPS) as a measurement system seemed an obvious place to start. A UPS is designed to support critical load (in our case the critical load was the IT systems); however in practice additional non-IT load is often powered off the same system. In such scenarios apportioning load becomes an arduous process. In our case stakeholders could not say for sure what the UPS load represented.

"As with any measurement exercise we started with defining how to measure."

We used heuristics and value stream mapping (VSM) techniques to tackle the issue. The electrical panel-board schedules were used to build up a holistic view of which equipment "should" be on the UPS and what additional variables constituted the wider data center ecosystem powered from a dedicated substation. This process identified that, apart from some fan coils and low power auxiliary systems, the load serviced by the UPS was dedicated to IT equipment and could be taken as reflective of the IT side of our PUE equation.

We now needed to do two things:

1. Prove via Measurement System Analysis (MSA) that the UPS could be used to accurately measure IT load changes and hence associated cooling requirements.
2. That the load as per VSM exercise was indeed IT load as per electrical panel schedules.

The team came up with different means of doing this; however some of the more dynamic solutions would effectively constitute projects in themselves. A simple innovative method was decided upon. Using a handheld nonintrusive multicore clamp-meter we measured every rack in the data center calculating the kilowatt power for each. The kilowatt total we measured and the UPS total adjusted for the fan coils and auxiliary systems were within 1.5 percent. Project stakeholders had been identified in the project SIPOC. Through engagement with these stakeholders we identified servers that were end-of-life. Such servers even when idle can consume in excess of 60 percent of full electrical power. We measured their kilowatt load and predicted the expected drop-off at the UPS should these units be decommissioned, that is, disconnected from main power.

"…servers even when idle can consume in excess of 60 percent of full electrical power."

In total 57 units were powered off. The units were powered off in batches over an eight-week period. The predicted drop in IT load was within 2 percent, as shown in Figure 3. This told us the existing UPS system was reliable as a measurement system for changes to IT load. Decommission resulted in a 5 percent IT load reduction.

"Decommission resulted in a 5 percent IT load reduction."

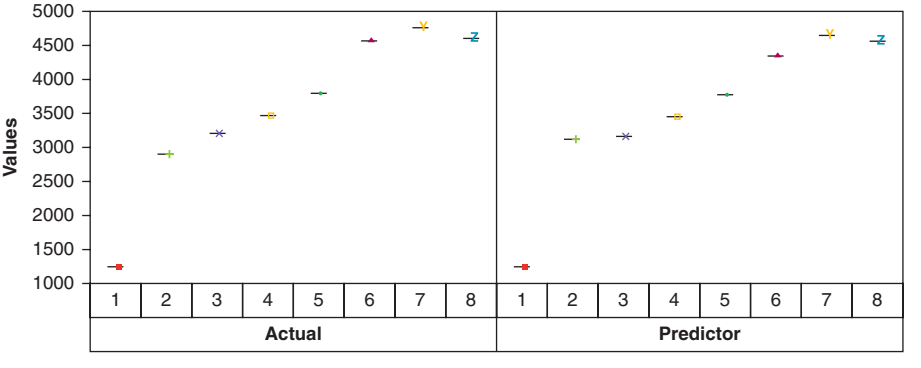


Figure 3: Variability chart actual versus predicted weekly kilowatt-hour drop (Source: Intel Corporation, 2010)

Building a Model View

We used the metering data identified in the VSM exercise to build a dataset to statistically model. The data included IT load, chiller load, and weather data at 15 minutes granularity taken from existing facilities systems. Spot measurements were taken from the data center Computer Room Air Conditioner (CRAC) fixed speed fans, room fan coils, and extractor fans. The dataset was rolled up and modeled.

In parallel we investigated the use of Intel® Intelligent Power Node Manager and Intelligent Platform Management Interface (IPMI) technology as a means of utilizing the servers as sensors. The varied age and capability of our IT equipment made it difficult to use this technology as a complete solution, but it did offer data to augment existing metering and would prove useful in the solution phase of the project.

There was one key variable that would not manifest in our model, one that was not obvious to non-facilities team members. That variable was air, and it is central to data center energy efficiency. Correctly managing how air reaches server inlets and cooling equipment is the key to reducing facilities load while still maintaining equipment integrity. As per best practice, we commissioned a computational fluid dynamics (CFD) analysis of the data center in order to evaluate the air flow effects. In the short term, pending results from the CFD, the team measured the floor tile air flow for the entire data center using locally available equipment, including a pressure hood and thermal camera.

Analysis

The ideal state within an air-cooled data center would be to keep hot aisles hot and cold aisles cold, with no mixing. The thermal camera, while not showing actual air flows, does illustrate the effect of hot or cold air on solid surfaces. As the images in Figure 4 demonstrate, open cable gaps and missing blanking panels can significantly affect the correct distribution of hot and cold air. The thermal camera also pinpointed hot air leakage through server racks that didn't use blanking panels. Without internal blanking panels air can mix, thus reducing cooling efficiency. The images demonstrated to decision makers the effects the invisible variable of air had for energy consumption.

"The thermal … images demonstrated to decision makers the effects the invisible variable of air had for energy consumption."

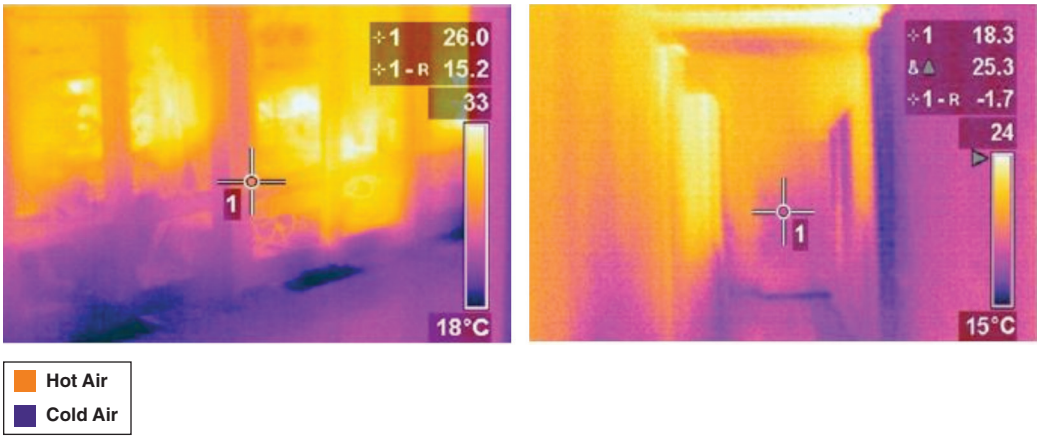


Figure 4: Thermal images illustrating air mixing. Left: gaps in hot aisle causing mixing. Right: ideally cold aisle should be all purple.
(Source: Intel Corporation, 2010)

Using the data obtained, through using the multicore clamp-meter and the flow-hood, we built a spreadsheet-based model that calculated the actual airflow delivered through each perforated floor tile versus the required level

based on the kilowatt rack density. Using this data, we were able to identify areas of overcooling and hotspots. This data would subsequently inform our air optimization strategy and was deemed by the team as a satisficing alternative to CFD.

"The data-centric approach of LSS was central to understanding the complex interactions of the data center ecosystem."

"The … concept of "Practical, Graphical, and Analytical" (PGA) was particularly powerful in identifying issues with the data center's cooling infrastructure…"

Chillers and Cooling Infrastructure

The data-centric approach of LSS was central to understanding the complex interactions of the data center ecosystem. The statistically based modeling application coupled with the six-sigma concept of "Practical, Graphical, and Analytical" (PGA) was particularly powerful in identifying issues with the data center's cooling infrastructure, specifically with the chiller units.

- *Practical.* The data center had three chillers located behind the Fab condenser room. Consistent with the lean concept of "go-see" we identified that the hot air from these condensers discharged in the vicinity of the chillers. The condenser did not warm the air surrounding chiller 1, so we assumed this chiller would run most efficiently. Chillers 2 and 3 were affected by the condenser discharge, so they would have to work harder to dissipate heat to the warmer outside air. We assumed these chillers would consume the most electricity.
- *Graphical.* The modeled graphical analysis of chiller energy shown in Figure 5 helped us determine that chiller 1 was in fact not the most efficient chiller. The model showed how IT load varied but the level of variation was minimal. Practically, one would expect the IT load (the heat source) to have an obvious effect on cooling load however this was not visually apparent. OSA conditions proved to be the dominant factor driving cooling load and this cloaked the effect of the IT equipment heat load.
- *Analytical.* As one would expect with air-cooled chillers, as outside air temperature increases, total chiller power increases. What our data showed however, is that chiller 1 exacerbated that effect by overconsuming power

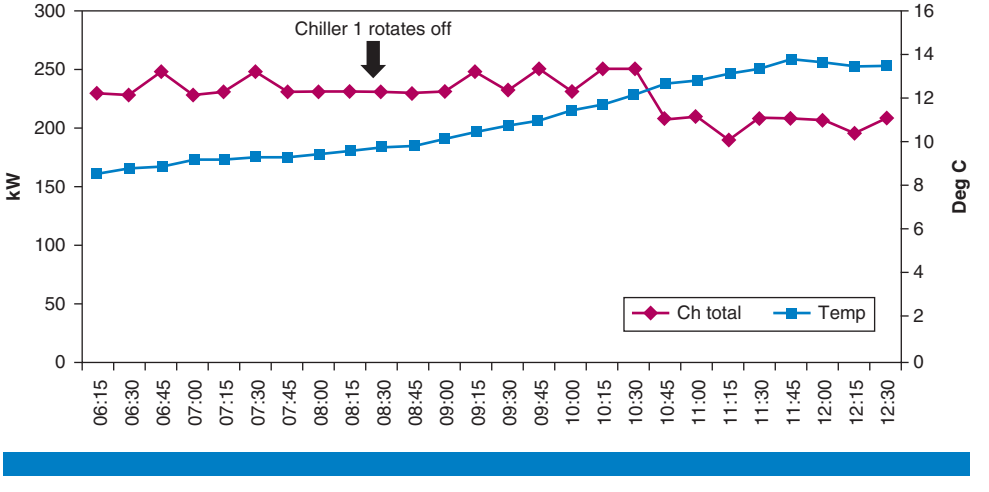


Figure 5: Chiller 1 effect on total chiller consumption (Source: Intel Corporation, 2010)

relevant to the temperature increase. Figure 5 demonstrates that once we turned off chiller 1, chillers 2 and 3 consumed considerably less power in dealing with the same conditions. This was the exact opposite of our practical assumption.

The PGA technique together with the statistical model was very effective in focusing analysis and ensuring that the assumptions were challenged and the right questions were asked. The model allowed us to disregard factors such as wind direction and speed while highlighting the need to include others such as temperature and dew point. However it also highlighted that we were missing something, some additional variable perhaps or some issue with the data because the spot COP measurements taken at the chillers and what the model inferred did not wholly align.

We convened a SIfT session in order to analyze as a group what information we had gathered and to work on possible solutions. What we learned very quickly was that our model did not take into account the additional load of the UPS room itself, which required cooling, or the effects of thermal load on the building. Additionally we discovered that our chilled water system was also supporting a PCCW (process controlled chilled water) loop servicing a Fab toolset. Having the right stakeholder group in the SIFT session was crucial.

The team now understood the dominant effect OSA conditions had on our cooling infrastructure. We appreciated the danger in working off assumptions. We knew that chillers 2 and 3 had a better COP because the compressor air had the benefit of keeping their coils clean, resulting in better dissipation of heat relative to chiller 1. But above all we now realized just how poorly our chillers performed in dissipating the IT, data center building, and UPS room load. Reflecting on the concept "all models are wrong, but some are useful"[4] we proceeded towards the improvement phase confident that we had built a "useful" view of our data center ecosystem.

"Reflecting on the concept "all models are wrong, but some are useful" we proceeded towards the improvement phase."

"During the process nonexpert attendees …proved as important as expert ones …"

Improvement Selection

As a result of our analysis and as part of our SIfT session we developed many innovative improvement ideas. The session used DeBono techniques to flesh out possible improvements. During the process nonexpert attendees (nonexpert in the sense that they were not from facilities or IT) proved as important as expert ones as they brought an unconditioned perspective. It had been a nonexpert team member who had suggested the alternative to CFD, testimony to the fact that sometimes experts cannot see beyond the norms of their domain. Following the SIfT session, project limitations were reintroduced and a solution matrix was used to score feasible paths to improvement.

IT Load Reductions and Dynamic On/Off Proof of Concept

As part of the UPS measurement system analysis we had powered off 50+ units to test the predicted drop. The flip side of this was to illustrate the type of saving that could be achieved. Stakeholders within the data center

identified additional units to be decommissioned now knowing that they could demonstrate the saving. In this sense the MSA proved to be part of the solution in removing nonessential power consuming equipment.

We also demonstrated a workable proof of concept (PoC) that used Intel Intelligent Power Node Manager and Intelligent Platform Management Interface (IPMI) technology to match server power during the work week. When servers were on in the project data center, they consumed a relatively steady rate of power. We used IPMI and a calendar-based GUI to turn identified servers off except during office hours.

The reductions in IT load highlighted the importance of metrics or rather the importance in having the right suite of metrics. PUE indicates how well facilities infrastructure, that is, power and cooling, supports the IT load. However, there was evidence that it had confused decision making with respect to energy efficiency in the test case data center. Elements of the facilities load were fixed, specifically the CRAC fans, and as reducing IT energy consumption did not mean dynamically reducing CRAC fan power it meant PUE increased. Identifying and communicating an appropriate suite of metrics that included kilowatt-hours was essential.

"Identifying and communicating an appropriate suite of metrics … was essential."

Air Management

Consistent with best practice we devised an air management strategy that:

- Blocked all gaps in the floor, fitted racks with blanking plates, and blocked off a sizable point of leakage under the raised metal Floor.
- Commissioned a hot-aisle containment system that uses PVC strip curtains (see Figure 6). It offered advantages in the short term by dramatically improving air management in terms of temperature, static pressure, and bypass air.
- Deployed appropriate floor tile types, based on the kilowatt loading of each individual work cell. A work cell is defined as a 2-foot by 8-foot (610-mm by 2438-mm) area consisting of a server rack, flanked by cold aisle on one side and a hot aisle on the other.

While much of the literature had called out the importance of such actions, none that we encountered had quantified what the result might be. The process adopted meant we could offer data-driven predictions that informed specific solution selection and ROI discussions.

"The process adopted meant we could offer data-driven predictions that informed specific solution selection and ROI discussions."

Chiller Plant

What became apparent during the Analyze phase was that the test-case data center was significantly overcooled and that the air-cooled chillers were inefficient with significant chiller to chiller variance. What was equally apparent was that any solution to improve the chillers could not improve beyond 3.63 COP (coefficiency of performance)—the maximum efficiency for the units new at 25°C. With that in mind, it was decided to work on

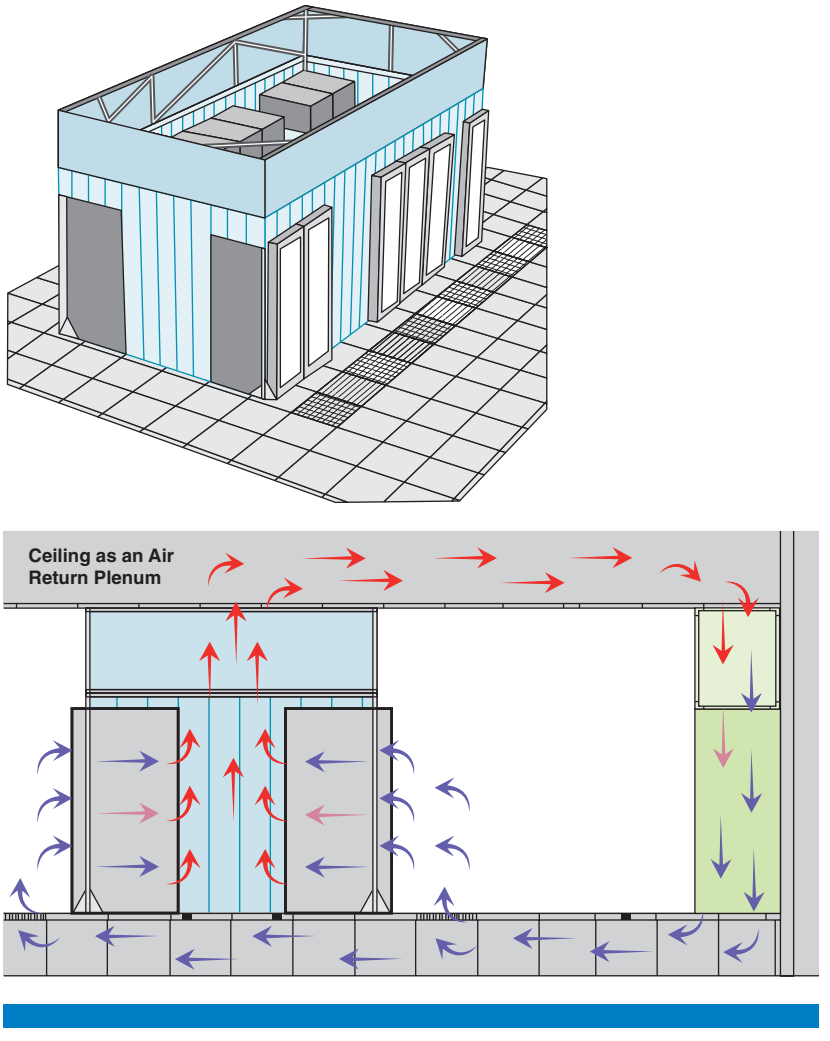


Figure 6: Hot aisle containment
(Source: Intel Corporation, 2011)

improvements to and alternatives for the current system. Initial work focused on existing system improvements and included:

- Cleaning chiller condenser coils as buildup was deemed to be affecting dissipation (this was later stopped as the coil fins were too fragile)
- Metering the chiller system to foster development of real-time (COP) monitoring.
- Turning off some CRAC units after air segregation
- Increasing chilled water temperature after air segregation.

Alternative system improvements focused on understanding the feasibility of high-efficiency heat removal, namely:

- Direct air cooling, airside economization
- Evaporative cooling, wet-side economization

Following a feasibility study it was decided to implement the latter.

"Alternative system improvements focused on understanding the feasibility of high-efficiency heat removal…"

Evaporative Cooling

Following a SEAI (Sustainable Energy Authority of Ireland) supported special investigation, the decision was made to implement evaporative cooling, based on two main factors:

- The availability of an underutilized water tower situated close to the chiller plant.
- The data center's roof could not bear the weight of air handling units required for direct free-air cooling without significant civil engineering costs.

As part of the SIfT session we learned that the evaporative tower option had been examined and rejected based on a prohibitive return-on-investment (ROI). However, the previous study had approached the project from an ideal system perspective and had not considered suboptimal options. Through the SIfT process we considered improvements, not just optimal solutions. An evaporative cooling improvement/solution was subsequently proposed.

The suggestion was to leverage the considerable excess capacity of the existing water tower to precool data center return water. The design incorporated the use of hot taps, a plate-heat-exchanger (HEX), valves, and a primary pump. The HEX sits between the closed and open systems. The return water would pass through the secondary side of the heat exchanger and is cooled by water from the tower pumped into the primary side. While not particularly novel from a mechanical engineering perspective the combination of this sufficing option, coupled with hot-aisle containment and IT improvements, offered considerable improvement in ecosystem efficiency.

Results

"The combination of LSS and SIfT within the project was paramount in pinpointing opportunities for significant, measurable energy reductions …"

The combination of LSS and SIfT within the project was paramount in pinpointing opportunities for significant, measurable energy reductions in the area of server load and cooling and offered a common language that challenged the often siloed approach to data center operations. The improvements are listed in three phases below:

Phase 1: April 2009. Proved the UPS to be a valid measurement system and achieved a once-off 5-percent server load reduction through IT equipment decommissioning, as shown in Figure 7. A further 3-percent server load reduction was identified using Intel Intelligent Power Node Manager and Intelligent Platform Management Interface (IPMI) technology to dynamically match server power to work-week usage for specific service types.

Phase 2: September 2010. Implementation of the air management strategy was completed including the deployment of hot-aisle containment, which enabled:

- 3 of 11 fixed-speed CRAC units to be shut down
- server inlet supply temperature to be raised from 14–21°C
- chilled water supply temperature to be raised from 6–7°C

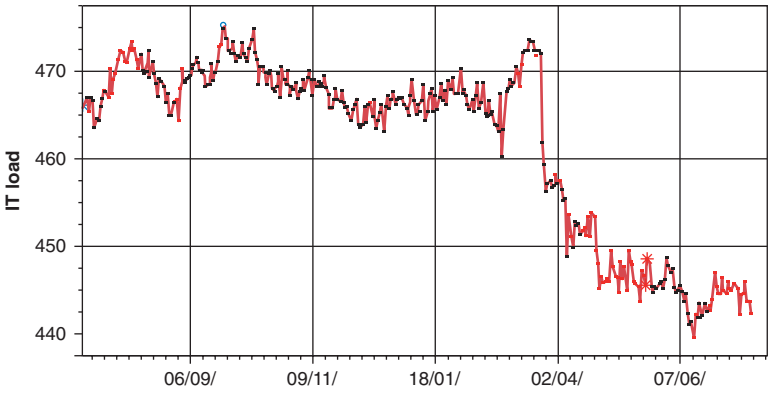


Figure 7: Kilowatt drop in IT load post-decommission (Source: Intel Corporation, 2010)

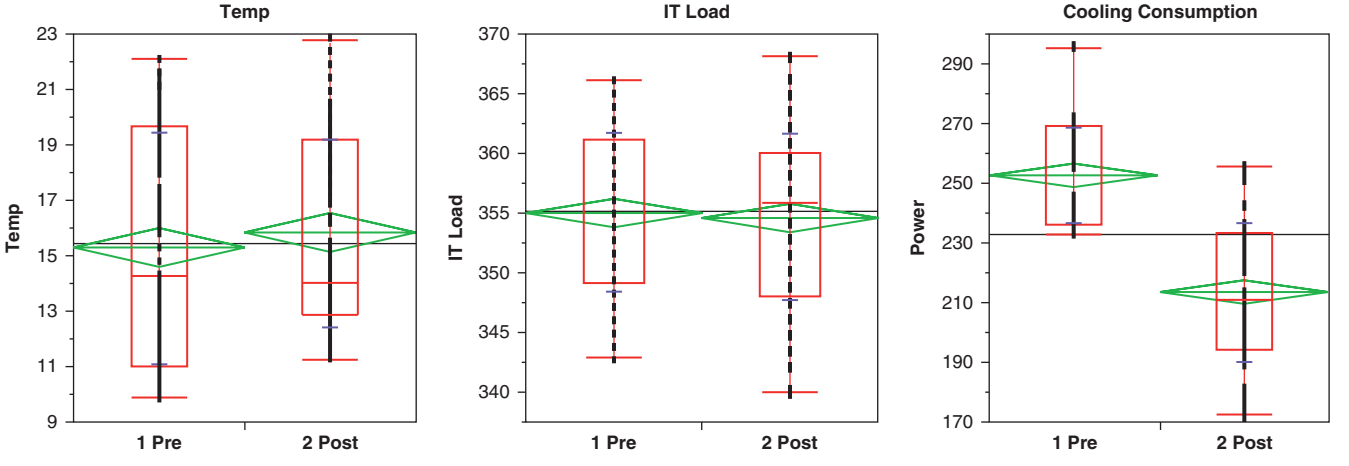


Figure 8: Impact of Phase 2 air management on chiller consumption (Source: Intel Corporation, 2010)

The result was a 27-percent decrease in CRAC fan consumption and a 17-percent decrease in chiller consumption, as shown in Figure 8.

Phase 3: November 2011. Having completed all capital works and having experienced a non–project-related delay:

- chilled water supply temperature was raised to 10°C
- evaporative cooling was made fully operational

This delivered a further 50 percent (100 kW) decrease in chiller consumption resulting in a to-date total reduction of 67 percent. Based on historical weather trends the 2012 annualized improvement is predicted to be 55–60 percent.

"Based on historical weather trends the 2012 annualized improvement is predicted to be 55–60 percent."

Figure 9 highlights the overall trends in IT consumption, chiller consumption, and PUE from January 2010 to January 2012, highlighting the two-step change in baseline.

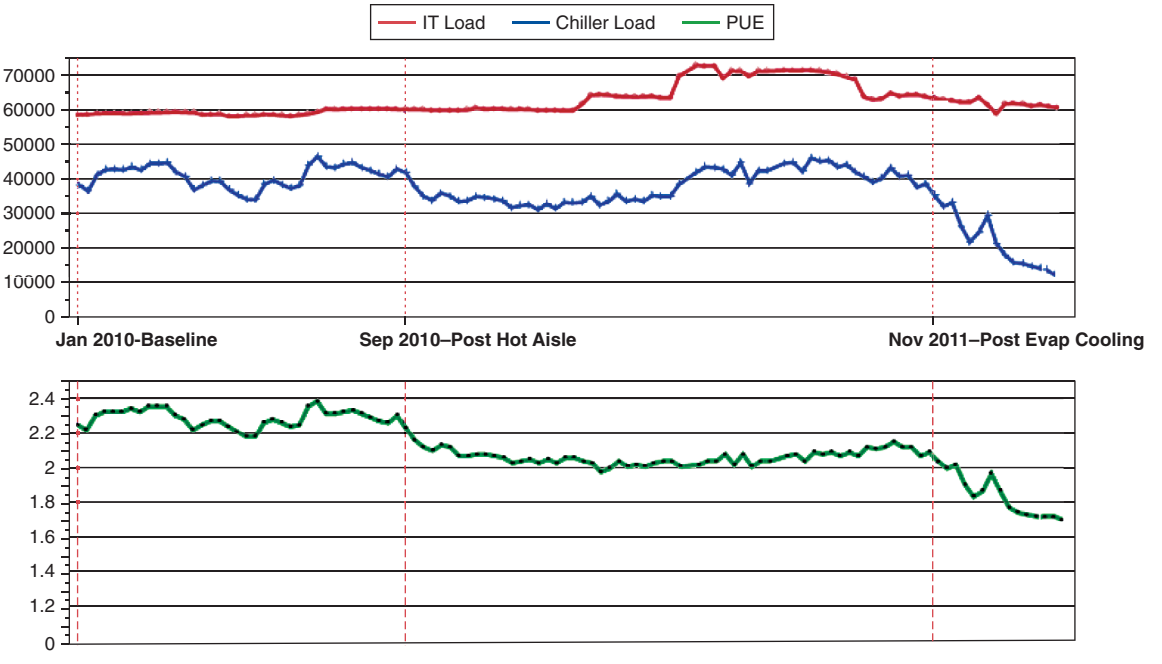


Figure 9: Overall trend for PUE, IT, and Chiller consumption (Source: Intel Corporation, 2011)

In summary, the methodology applied was paramount in delivering the following verified improvements:

- A 26-percent reduction in total energy consumption
- An improvement in PUE from 2.3 to 1.69
- An approximately 993-ton reduction in CO_2 emission

Planned future improvements in the test case will reduce PUE to about 1.58, deemed the highest achievable rating for a mature data center of its class.

Conclusion

Data centers are the fastest growing contributor to the CO_2 footprint of the ICT sector, which represents about 2 percent of total global CO_2 emissions. Mature data centers are typically energy inefficient, consuming 1 watt or more for every 1 watt delivered to the IT load. Such data centers are indicative of the wider building stock designed at a time when energy and resource efficiency was less of a concern.

The authors identified an improvement methodology based on the synergistic combination of Lean Six-Sigma and Systemic Innovation for Teams. The methodology proved highly effective in delivering significant, measurable energy reductions in the test case data center.

The methodology is highly complementary to a sustainable context and is posited as a generic transferable blueprint for delivering resource-efficient operations.

"The methodology is highly complementary to a sustainable context and is posited as a generic transferable blueprint for delivering resource-efficient operations".

Acknowledgements

The authors would like to thank James Eynard, Dermot Honan, Aengus Nolan, Peter Sullivan, John Weir, Rachel Agnew, Luke Fenner, and Kevin Geoghegan.

References

[1] Global e-Sustainability Initiative, "Smart 2020 Enabling the low carbon economy in the information age," available online at www.gesi.org

[2] Kelly, J. et al., "Enabling Deliberate Innovation: systemic Innovation Workshop," IMEC 2006

[3] Nussbaum, Bruce, "The Power Of Design," *BusinessWeek*, May 17 2004, available online at http://www.businessweek.com/magazine/content/04_20/b3883001_mz001.htm

[4] Box, G. E. P., and Draper, N. R., *Empirical Model Building and Response Surfaces*, John Wiley & Sons, Inc., New York, 1987.

Author Biographies

Keith A. Ellis is a research scientist with the Energy and Sustainability Lab, Intel Labs. Keith's prime research interests focuses on ICT enablement in the context of energy/resource efficiency. He has participated on several EU funded and Intel projects in this space. Keith has worked with Intel for 17 years with roles in both manufacturing and IT innovation.

Charles G. Sheridan is the co-director of the Energy and Sustainability Lab, Intel Labs. He co-chairs a leading consortium focused on sustainable computing. Charlie has been involved in several EU, national, and Intel projects in the sustainability domain. Charlie has worked with Intel for 17 years with roles in both automation and IT innovation.