The Post PC Era
- Are Tablets Ready for Mainstream Business?
- A Cultural Model for Security in the Post-Desk Era
- Personal Computing Without the PC
- Scaling to the Sky
- The JavaScript Revolution
- Tablet Application Infrastructure Considerations

CA Technologies Perspective
- On Demand Demands
- Passing the Torch: Lessons from Past and Future Revolutionaries of Our Generation
- Big Iron Results in Big Energy Savings
- Commoditization in the Cloud Calls for a Sticky Solution

Technical Notes
- Collaborating Through Shared Displays and Interacting Devices
- Profile of Wilson MacDonald: Mentor Inventor
Introduction to the Post PC Era Issue
by Marvin Waschke, Senior Principal Software Architect, Editor in Chief of the CA Technology Exchange, CA Technologies

The Post-PC Era.
We all know we are in it. Every trade magazine has something about it, but I, for one, still am somewhat dazed by the entire idea. I know what the phrase means: the replacement of general-purpose desktops and laptops with highly portable personal computing devices. But what has the Post-PC Era changed? That is not so clear to me.

Therefore, I was pleased that the CATX editorial committee chose the new era as the theme for the May 2012 issue of CATX. We were pleased at the response when we put out our call for submissions and I have learned a lot about what the technology experts at CA Technologies see in the Post-PC future.

The new era has serious implications for all software vendors. Developers have new choices of venues for their products. Apps have taken center stage and there is no reason to expect that they will decline in importance. Developers do not write apps like traditional applications. There are new tools and they require planning that the industry has not had to cope with before. Several articles in this issue shed light on those issues.

Post-PC security gives IT the whim-whams. The whole phenomenon of BYOD (Bring Your Own Device) makes security officers shake their heads in bewilderment. Connecting a “dirty” unauthorized device to a corporate network used to be career limiting action, but now it is practically an employee right. Corporate security officers have been forced to rethink the whole idea of the internal network perimeter. When combined with the rise of cloud deployments, BYOD has inspired an entire new view of security.

In this issue, we offer to you the depth of CA Technologies insight into the new era. You won’t be reading much about how cool iPads look, or the latest app for Androids, but you might learn something about how to write an industrial strength app for an Android, how to secure an iPad or how Post-PC changes IT management.

In addition to articles on the new era, we have a selection of insight from CA thought leaders. We have also added a section of that we call “Technical Notes.” Lately we have received a number of short monographs that go too deeply into a single subject to fit with our high-level insights, but are too focused to be an article. Therefore, we have added our “Technical Notes” section. I find the notes compelling for their depth and clarity.

Please enjoy our new issue.

Let us know what you think about this issue, or what topics you’d like us to cover in a future issue. Send an email to catj@ca.com.
CA Technology Exchange

Table of Contents

CA Technologies Perspective

1  On Demand Demands
   Dr. Gabriel Silberman, Senior Vice President and Director, CA Labs

3  Passing the Torch: Lessons from Past and Future Revolutionaries of Our Generation
   David Messineo, Senior Software Architect, CA Technologies

14 Big Iron Results in Big Energy Savings
   Cynthia Curtis, Vice President and Chief Sustainability Officer, CA Technologies

16 Commoditization in the Cloud Calls for a Sticky Solution
   Jacob Lamm, Executive Vice President, Strategy and Corporate Development, CA Technologies

The Post PC Era

19 A Personal Viewpoint: Are Tablets Ready for Mainstream Business?
   Marcel Hartog, Principal Product Marketing, CA Technologies

22 A Cultural Model for Security in the Post-Desk Era
   Malcolm Ryder, Senior Software Architect, CA Technologies

33 Personal Computing Without the PC
   Sridhar Chakravarthi Mulakaluri, Director of Operations, CA Technologies

36 Scaling to the Sky
   Dr. Steve Versteeg, Research Staff Member, CA Labs, and Cam Hine, Postdoctoral Fellow, Swinburne University of Technology

38 The JavaScript Revolution
   Nathan Giardina, Senior Software Engineer, and David Tyree, Senior Architect, CA Technologies

46 Tablet Application Infrastructure Considerations
   Kenneth Kranz, Enterprise Architect, CA Technologies

Technical Notes

61 Collaborating Through Shared Displays and Interacting Devices
   Dr. Steven Greenspan, Research Staff Member, CA Labs

65 Patently Innovative: Profile of Wilson MacDonald: Mentor Inventor
   Karen Sleeth, Senior Principal, CA Technologies
On-Demand Demands

by Dr. Gabriel Silberman, Senior Vice President and Director, CA Labs, CA Technologies

What do pilots, drivers and control room operators have in common?

Everyone agrees that flying a plane requires extensive training, practice and experience. After all, planes are very complex pieces of machinery that continuously interact with the environment, other aircraft, and air traffic control. Pilots are continuously fed large amounts of information and it is essential for a safe flight that they have the ability to understand the potentially complex meaning of it all, synthesizing a response and acting upon the data correctly.

For most of us who do not fly planes, we are faced with a similar challenge when we get behind the wheel of a car. Drivers also go through training, but its length and scope is much more limited than for pilots. Yet, there are similar learned behaviors both pilots and drivers - or operators - use to analyze their surroundings, plot their course and react to unexpected events.

These behaviors also apply in a third scenario, this one a control room setting where people monitor a complex system using IT tools. Similar to the two previous scenarios, a control room operator or administrator is faced with a flood of information demanding attention and possible action.

Experienced operators scan a scenario, whether directly by observation (for example, through the windshield) or through a set of instrument readings. Through this scan, an operator inherently assigns a priority to the various items being observed, and, because it is done fairly unconsciously, it is very hard to teach to less experienced individuals. The question then arises whether it is possible to provide assistance aimed at directing attention to events or instrument readings in a sequence conducive to safer or more efficient operation. And to do so in ways that are not too intrusive (e.g., many flashing items competing for attention) nor require many actions on the part of the operator (e.g., read lots of text, click on items, close pop-ups).

In a recent patent application filed by CA Technologies, co-authored by researcher Dr. Steve Greenspan and myself, we propose the use of signaling on a display’s periphery to guide the operator’s attention in a sequence determined by the relative importance of information in the various areas of a display, or windshield, in case of a vehicle. More importantly, we suggest a non-intrusive approach to ascertaining whether the operator has seen the important information, by tracking their eyes instead of requiring some other physical action or spoken command. To close the loop, a learning component can be added in this scenario to record the outcome of the attention focus and use it

About the author

Dr. Gabriel Silberman is Senior Vice President and Director of CA Labs. He is responsible for building CA Technologies research and innovation capacity across the business.

Leveraging the skills of the company’s researchers and technical experts worldwide, CA Labs engages with leading-edge researchers in academia, professional associations, industry standards bodies, customers and partners to explore novel products and emerging technologies in support of CA Technologies key growth areas.

Gabby joined CA Technologies and established CA Labs in 2005. His experience in academic and industrial research has contributed to the success of the collaborative model of exploration into emerging and disruptive technologies.

Prior to joining CA Technologies, Gabby was program director for the IBM Centers for Advanced Studies (CAS), where he was responsible for developing and adapting the collaborative research model for IBM worldwide. Previously, Gabby was a manager and researcher at IBM’s T.J. Watson Research Center where he led exploratory and development efforts, including work in the Deep Blue chess project.
for training purposes, both for the individual operator as well as to pass along best practices of more experienced individuals. For example, knowledge on the sequence used by an experienced operator to scan a composite display, or groups of displays, in response to a complex event, may be captured and used for later analysis and training.

As Steve Greenspan points out in his article, Collaborating through Shared Displays and Interacting Devices, in this publication, despite the best efforts to have technology automate the role of operators, humans are still essential to handle unexpected or unusual circumstances. When this happens, operators must usually contend with a sudden demand to process a large volume of information efficiently and reliably, and under time pressure. To do so in an “on demand” world requires a new approach to the development of tools to augment operator capabilities, training and just-in-time information delivery. CA Labs researchers, together with academic partners at Carleton University in Ottawa, are continuing to explore these topics, together with knowledge management and novel visualization techniques to push the boundaries of what is possible today.

Reference:

Despite the best efforts to have technology automate the role of operators, humans are still essential to handle unexpected or unusual circumstances.
Passing the Torch: Lessons from Past and Future Revolutionaries of Our Generation

by David A. Messineo, Senior Software Architect, CA Technologies

The recent deaths of Dennis Ritchie, John McCarthy, and Steve Jobs, all within a single year, were poignant and collectively marked the passing of a great era in technology. Each man, in his own way, was the face of a common movement: the proliferation of freedom through the transparency of information. Like most movements that persist, ultimately it is the ideas that provide the scaffolding from generation to generation. Who will leverage this scaffolding to mine the contributions of the great minds before them in order to change the world we live in?

This article both reflects on the historical achievements and speculates about future possibilities as we head into the next generation. It concludes with some observations about what powers this movement and how any individual can influence it.

The First Invention of Revolutionaries

In 1439, Johannes Gutenberg created what many consider the most important invention of the second millennium -- the printing press with movable type. The ability to reliably publish and distribute knowledge, cheaply and broadly, radically changed the structure of power across the world. It fed the Renaissance movement, recording stories of heroism and extending the hand of scientific research to the masses. In 1455, Gutenberg would become an unlikely army for Christians by printing the Bible, arguably the best-selling book of all time, altering the politics of our world forever. No less important, the printing press would serve as the mighty sword for one man, Martin Luther, to publish his 95 Theses, widely regarded today as the primary catalyst for the Protestant Reformation. With one invention, Gutenberg unleashed the power of complex systems, where hair-like fractures within a community lead to massive faults in the fabric of society.

While this issue of CA Technology Exchange focuses primarily on the technical and business aspects of the post-PC environment, there is another aspect just as important: society. Much like Adam Smith’s “Invisible Hand” influences our economy, the means of communication and the methods by which people are connected significantly impact the post-PC society. How does Facebook’s ability to link people, ideas, communities, and cultures together change our societal values? How does having instantaneous responses through Siri influence our state of mind? How does having information, available anytime and anywhere impact our decision-making? How does unlimited capacity and bandwidth allow us the freedom to pursue and sell our ideas around the world? And how does an almost unlimited form of universal communication change the way we interact with one another, whether family member, friend, or colleague, personally or professionally?

2011 and the Passing of Three Revolutionaries

No one innovation will ever have the societal footprint or geographic impact of Gutenberg’s printing press. The story of history is long, however. As 2011 passes...
into the archives of history, it’s worth reflecting on three men who radically changed the world in ways similar to Gutenberg: Dennis Ritchie, John McCarthy, and Steve Jobs. Their contributions inspired a truly empowering element of mankind, illuminating the remarkable ability to scaffold upon the successes (and failures) of one another, standing upon giants to change our society and to realize the betterment of all who seek the freedom to explore.

**Dennis Ritchie: Building a Revolution through Software Architecture**

Dennis Ritchie (1941 - 2011) contributed to the digital era with three important revolutionary ideas. His first was a common language. One of the leading challenges in commerce today, as it has been for centuries, can be attributed to differences in language between the parties involved. When writing contracts in multiple languages, it is far from easy to ensure they are legally equivalent. Yet, the power of a contract between provider and consumer is reflected in its details. The primary reason many contracts are written in English is to assure a parallel understanding by all involved parties, since English is globally accepted as the language of business. It is the equivalency of interpretation (a common language) that is critical for allowing swift and efficient communication. Similarly, before the PC made its debut, there were several computer-based languages that proliferated, primarily through academia and large companies. While these languages were tailored to take advantage of their respective environments, they all had significant disadvantages too, chief among them the ability to integrate with and exploit one another’s features.

While at AT&T Bell Labs, Ritchie invented and implemented the C programming language. C was novel and powerful in its native ability to provide low-level access to hardware and overall coding structure, and it was simple and inexpensive. For years, Ritchie’s classic text "The C Programming Language"\(^1\) was considered a robust environment for would-be programmers of all types. What made C truly revolutionary, however, was its portability and openness, thereby supporting deployment across heterogeneous environments. Guttenberg gets credited as much for his successful printing process, as for it being widely copied and improved upon. His approach to printing became a "standard" then, just as C has now. Even today, C continues to make significant contributions to software architecture through C-inspired languages like JavaScript, C++, and C#.

Ritchie went further, however, when he helped create the UNIX operating system\(^2\). While C was the mechanism by which code was written across platforms, UNIX became the preferred backbone for integrating devices. While today we enjoy making phone calls instantly from anywhere in the world, 50 years ago such connectivity was manually established through a sequence of telephone operators individually connecting lines together. The UNIX operating system changed that, consolidating several manual tasks to one seamlessly automated process, and ultimately solving a complex scaling and accessibility challenge. Additionally, like C, UNIX was an open environment, becoming the precursor to Linux, an operating system that powers smart phones worldwide. Through Linux, UNIX essentially became the operating system for the Internet and effectively kicked off the open source movement, which is the third of Dennis Ritchie’s accomplishments, though shared with many people.

**John McCarthy: Building a Revolution through Knowledge Architecture**

John McCarthy’s (1927 - 2011) journey into the digital era wasn’t through technology, per se, but through the mind. Whereas Ritchie focused on software architecture, McCarthy focused on computing from the perspective of a
cognitive scientist. In creating a language called LISP (List Processing), he coined the term Artificial Intelligence (AI). Instead of creating interfaces to hardware, he created interfaces to the mind. With respect to LISP, it's particularly interesting to note that it's the second oldest mainstream language still in active use (only Fortran is older). This longevity reflects the optimism that AI will ultimately benefit mankind by creating a means of distributing knowledge, much like the availability of books did in Gutenberg's time. The fundamental difference, from a revolutionary point of view, is that the concept of a book has evolved to a level of being almost three-dimensional, allowing for a form of bi-directional conversation.

McCarthy's implementation of computerized chess (one of the first) based on published strategies was an example of this evolution. To provide such capabilities, McCarthy introduced tree-like data structures that excelled at deduction, a signature design that is still pervasive in AI systems today.

Like C, LISP continues to be a significant contributor to the digital community by influencing languages such as Scheme, the leading language used to teach students programming, F#, a modern era Internet-optimized language that has strengths in parallel processing and data management, and Prolog, the leading language used today to support AI work. LISP and its variants are often used in modern robotics, fraud detection, and airline scheduling, among others. LISP was also the backbone of such AI simulators as ELIZA and iOS Siri. Noted computer scientist Alan Turing, who created the “Turing Test”3, would no doubt be proud of how far AI has come under the stewardship of John McCarthy.

McCarthy's accomplishments hardly end with LISP and AI alone. In fact, much as Linux became the operating system for the Internet, McCarthy's notion of time-sharing (i.e. time-slicing) has evolved to become the basis for networking, electronic commerce, and now cloud computing. As if inspired by Nostradamus himself, McCarthy's ideas are what fuelled and continue to accelerate the adoption of the Internet, through the notion of “infinite bandwidth” and transparent availability to information and knowledge across all domains. The economics of such access is directly attributable to the technical ability to time-share.

Steve Jobs: Building a Revolution through Social Architecture

While the accomplishments of Dennis Ritchie and John McCarthy are generally unrecognized by the public, those of Steve Jobs (1955 - 2011) are well known. Steve Jobs is seen by many as the pioneer of the computer revolution. As the brainchild of the iMac, iTunes, iPhone, iPod, and iPad, Jobs revolutionized the advantages of being portable, tethered to a community rather than a machine. Jobs defined a cultural vocabulary for coolness. Whereas technologists saw Bill Gates as being the ultra cool mascot of everything digital, it was the general public that ultimately endorsed Jobs, and Apple, as "cool".

Today, having an "iCool" product is a discerning symbol of class. While Jobs was clearly a visionary and a technology pioneer, he was ultimately a genius in business, innovation, and product design too. His real impact is only now being fully appreciated in this post-PC world. There is no doubt Bill Gates will be seen as the leader of the personal computing world, but Jobs will be remembered as the one who outsmarted Gates by redefining what "personal" meant. The stock price of Apple and Microsoft reflect just how differently the two companies are viewed when it came to as being innovative.
Steve Jobs, more than anyone else in history, both humanized and commercialized personal computing while maintaining the advantages of global connectivity. The "i" in Apple's products clearly indicates this is not a purchase so much as a lifestyle. To Apple, you're not just a customer; you're an individual with his or her own unique needs. The device works the way you want. Yet simultaneously, you're part of a bigger "i", as in society, whether it is an organization, a country, or a generation. Jobs connected the universe of individual uniqueness with the universe of collective power in a way no one before him had been successful in commercializing. Instead of a platform powered by technology, Jobs created a platform powered by people. Jobs defined the post-PC world by creating the ecosystem of universal mobility juxtaposed with personal enablement.

Whether you are an iPhone user or a Droid user, the platform of applications built on these devices has changed society, by not only untethering you from the personal computer but by creating a mobile lifestyle. No more keyboards or mice – it is all touch and voice now. No more suitcases filled with papers and laptops with spreadsheets and presentations to carry with you everywhere – it's just a phone and a repository of documents sitting in a cloud somewhere. No more leaving a message or waiting back for a response from email – responses are instantaneous with texting. No more lengthy memoranda or urgent

Jobs connected the universe of individual uniqueness with the universe of collective power in a way no one before him had been successful in commercializing.
messages — we simply "tweet". Finally, and most profoundly, no more corporate and personal address books – everything you need to know about a person is in Facebook, LinkedIn, or some other social network. This IS your address book. These are the individuals that make up your life. This is the platform of your life. While Steve Jobs didn't invent many of these mobile applications, he constructed and deployed a platform for them by making things simple. Like Guttenberg, Jobs changed the way knowledge was distributed; however instead of imprinting movable type on paper, Jobs imprinted ideas on the brain.

**Revolution as a Catalyst for Innovation**

Dennis Ritchie, John McCarthy, and Steve Jobs were not simply innovative; they each built a platform that established a base from which their ideas could evolve. Each was exceptionally smart and received many achievement awards honoring their respective contributions. Yet what separates their accomplishments is how they captured power through the eyes of accessibility. Whether it was through hardware, software, or mind-ware, each of these men aggressively identified the need to network clearly, quickly, accurately, succinctly, and oportunely, seeking to bridge the gap between societal challenges of the day. They became the giants that enabled others to achieve remarkable feats.

While Guttenberg may have been recognized as the father of the most important invention in the second millennium, it remains unsettled who will be awarded such distinction in the third. Will it be Bill Gates, arch-nemesis of Steve Jobs, who created the PC revolution by lowering the cost of access to technology, standardizing software, commercializing the operating system, and reinventing capitalism by creating more millionaires than likely anyone else in history? Will it be Tim Berners Lee, the inventor of the World Wide Web (on a NeXT computer of all things)? Will it be Google co-founders Larry Page and Sergey Brin that aggregate content and provide search engines that pinpoint specifically what you're looking for? Will it be Steven Wolfram's “Wolfram-Alpha”, a computational knowledge engine that leads the next generation of accessibility?

These individuals, many others, and the companies they represent have and will continue to encourage changes (purposeful or not) to our society in the post-PC world. However, two gentlemen stand out as effecting the same kinds of radical revolutionary changes that Dennis Ritchie, John McCarthy, and Steve Jobs enabled: Jeff Bezos and Mark Zuckerberg.

**Jeff Bezos – Version 1: Building the Revolution through Individual Attention**

It's been rumored that a certain toddler named Jeff Bezos (1964 -) tried to dismantle his crib with a screwdriver. Years later, he graduated summa cum laude from Princeton University, and wrote his business plan for Amazon on a cross-country drive from New York to Seattle. With an introductory text on selling books in hand and a couple of engineers as partners, he built (as he called it) “the biggest bookstore on earth”. One has to marvel at such an assertion, given his operation ran out of a garage and carried little real inventory. Just a few years later, however, his biography was to include being named Time magazine's 1999 Person of the Year and living life as a multi-billionaire. So what is the magic of Jeff Bezos and his beloved creation Amazon?

Wal-mart often gets credited with fundamentally changing the retail industry. From a small five and dime store in 1962 to the 18th largest public corporation
and the biggest private employer in the world (with 2 million employees), there is no doubt about Wal-mart’s global dominance. Amazon, however, radically changed the retail industry in a manner that Wal-mart hasn’t and likely can’t (short of an acquisition of Amazon). The story behind Bezos and Amazon starts with his first redefining and then exploiting the digital universe. While Walmart mastered the physical storefront, Amazon cornered the virtual one. However, such a simplistic view ignores what truly distinguishes Amazon from ordinary retail organizations, and what makes it a fitting contribution to the second revolution in the post-PC era.

Amazon started as a commerce platform to sell books. Today, it sells almost everything a typical individual requires and is arguably considered the largest store in the world for selection and availability of merchandise. Its service is top-notch, and its use of novel features like one-click shopping, the associates program (selling items through other entity’s web sites), the advantage program (selling your items through Amazon), and Prime membership (free expedited shipping and on-line access to films) are all designed to steer people from traditional mortar shops like Wal-mart into Amazon’s virtual warehouse of products.

Two characteristics, when paired, help explain why Amazon is so fundamentally successful – individual access and attention. Today most retail shops get your attention by promoting loss leaders and other incentives to encourage purchases. Many retailers provide club cards, membership mailing lists, and online access to coupons in an effort to customize the experience to a customer’s unique tastes. Through one simple observation, Bezos radically changed the manner in which retail engaged in a digital universe. Shortly after receiving his first sets of orders, he noticed by careful examination of individual book orders that he could discern interests and start recommending books to others with similar purchases. While buying associations through technologies like CART was not new, the ability to leverage such analysis and make recommendations in real-time reflected a golden opportunity. The genius behind Bezos was that he took such analysis to a new level, providing Amazon the means to accomplish the following:

- Maintain historical purchase information by individual and groups
- Maintain wish lists of books by providing “stored” purchase carts
- Notify consumers of books they placed on a watch list
- Allow consumers to identify books they already purchased
- Allow individuals to create lists of books about a particular topic
- Provide for customer reviews, track reviewers influence, and maintain a scorecard of top reviewers based on quality
- Look inside the book (as if you were in the bookstore)
- Profile the book based on category and reading level
- Track recommendations based on localities, companies, or groups

By creating a digital ecosystem from which to sell, Bezos created a wealth of information about an individual’s buying habits assessable at the point of interest. Imagine influencing an individual’s voting choices right at the point of decision. Amazon’s digital presence is not just about access – but attention. Politicians everywhere should be envious.

Amazon further reached out and realized the platform that drives their business could drive others. They have released a slew of business services to support businesses, including support for large databases and computing bandwidth. Through the use of the cloud, they have significantly reduced the costs of providing sounds, images, videos, and software services. The power of the cloud is all about sharing, and Amazon has created the cloud that powers the world of the future.
into business by taking a lot of the up-front risk out and providing a worldwide
distribution for products of all types. They have focused on accessibility, having
country and language specific sites, all tailored to sell products developed by
both individuals and small and large companies alike. In many respects they
have provided equality to the retail process by levelling the advantages one
provider has over another. Amazon's platform serves as a mighty arbiter of
fairness, and has the customer base to prove it. By itself, however, these areas
are not really specific to a post-PC environment. While having virtual access to a
workstation is convenient, by itself it really doesn't change the equation until
there are specific applications that can exploit it as a platform. Such
applications are only now starting to surface. Where Amazon truly fits into the
post-PC environment is with its Kindle platform.

Jeff Bezos – Version 2: Educating the Revolutionaries by Nurturing the
Meme Machine

Libraries have been around for centuries. From the first substantial library in 334
B.C., the Great Library of Alexandria, to the largest of today with over 33 million
volumes4, the U.S. Library of Congress, libraries have been at the forefront of
creating an educated population. However, such education was often restricted
by birth, by class, by family, or by luck itself. Keeping the population uneducated
was a popular way of maintaining power.

It was around 1870 that Andrew Carnegie, having amassed the second largest
fortune of all time by an individual, initiated his efforts in philanthropy. Carnegie
understood the power of knowledge, and of access. He essentially established
the public library system in the United States, while also giving generously to
Canada, the United Kingdom, and other nations. Gutenberg had revolutionized
the printing of books, significantly increasing their availability and distribution,
but the cost of acquiring books was considerably higher than most could afford.
Therefore, having a collection of books was rare, often limited to universities
and to the rich. Carnegie foresaw the creation of the local libraries as
fundamental to the education and welfare of society. While Steve Jobs created a
technology platform based on people, it was Carnegie and other such
philanthropists that gave us the informed individual. The town library was the
platform for knowledge. As the public education system evolved, the concept of
school libraries was successfully adopted.

Much as reframing the retail market in the digital world has changed the buying
experience, so has the digitalization of knowledge. The Kindle and similar
devices have arguably changed the concept of a library. While the notion of an
e-book is not new to this generation of graduates, access to e-books formerly
required a PC tethered to the network. Files would often be stored on a hard
drive, or on a thumb drive for more portability. Either way, it still required access
to an interface. The traditional PC or Mac, while fine for doing day-to-day work,
was not very portable, and the actual experience was quite different from
reading a physical book.

Portability was about more than just having access to a book – it was
maintaining the same advantages of having a physical copy, including the
ability to quickly scan through pages and mark them up. Customers also needed
device that worked in differently lit environments, while avoiding muscle
strain to the eyes. Through the implementation of E Ink technology, Kindle met
these challenges. Furthermore, because the cost of creating a digital copy of a
book is almost negligible, the cost of buying a book could be cut in half,
extending the access of literature of all types to a wider audience. Moreover, the
ability of the Kindle to control access revolutionized the notion of a lending library. You no longer had to worry that the library did not have the book you needed; you could purchase access for a limited time for a small fee or subscribe to a service and have as much access as you needed.

The digitization of material created the possibility to democratize knowledge. The Internet created the platform to exchange material. The Kindle, exploiting the display technologies of Apple’s products, provided an interface, allowing almost unrestricted and transparent access to content. This access provided for the proliferation of ideas and the organic assembly of memes across a society conveniently assembled through social networks like Facebook. The doors to the library are open!

**Jeff Bezos – Version 3: Building the Revolution through the Commercialization of Fairness**

Much like Steve Jobs envisioning a world of un-tethered access and smart devices, Jeff Bezos envisioned a world of personal libraries. By leveraging the unique shopping habits of millions of people, Bezos built Amazon to connect readers with one another and creating an atmosphere and presence rare among on-line retailers and next to impossible to mimic by mortar-based establishments. The rather dramatic announcement by Bezos that digital books outsold physical ones during the 2011 holiday season demonstrates how revolutions often exploit and ultimately raze the structures creates by the ones before it.

Today, as the historical center of reading material, many public and educational libraries are replacing outdated books with Kindles, allowing patrons and students access to virtual libraries around the world. In a post-PC world, the Kindle has radically changed the availability of content by making most of it available for a small fee or free. Such access provides the Kindle with one additional game-changing innovation: it allows the notion of self-publishing to be economically viable. As an author, you can publish your book on the Amazon platform for almost nothing. The quality of that book can be judged by your real audience as opposed to the editors of a large publisher, who often have different motives or incentives. Once again, Amazon’s platform proves to be an arbiter of fairness, democratizing the benefits once available only to large publishers.

**Mark Zuckerberg – Version 1: Harnessing the Power of the Crowd to Empower a Revolution**

There is no doubt that genius and necessity of invention are key to establishing the next step in societal evolution. But often genius and innovation take another form, one not just of invention but of observation. Such was the case in 1961 when Edward Lorenz was using a computer to help predict weather. In taking what he perceived to be a meaningless shortcut, entering the number .506 instead of .506127, he observed widely differently weather patterns. From this small difference was born what Lorenz later called “The Butterfly Effect”, noting that such an insignificant amount could be accounted for by changes caused in the atmosphere due to the normal wing motion of a simple butterfly. Such dynamics are also known as the “Snowball Effect” and reflect that systems, whether biological, societal, or financial, can be very sensitive on initial conditions, with widely varying behaviors resulting from negligibly different beginnings.

In many ways, Mark Zuckerberg (1984 –) observed a similar structure with “social ties”, and created a platform to accelerate its impact in the world. Like
many innovations, the beginning was not obvious; the actual accomplishment was realized as much by luck as by planning. While Zuckerberg was attending Harvard, it was the custom of many universities to publish the names and pictures of new students as a mechanism to encourage socializing amongst one another and the existing student body. One such college referred to this as a facebook, and hence was “born” an industry. As Facebook the company (and its software) started taking shape, more people began subscribing to its membership, allowing others to connect with them across a spectrum of common characteristics. Whether it was a shared academic institution, a common interest in music, movies or sports, a trade skill, or a team, Facebook allowed people to find one another and stay connected in a manner that was previously unlikely because of the time and effort involved. Facebook, unlike any platform before it, focused on leveraging the past as a venue to the future.

Facebook’s success was no more guaranteed at the beginning than Amazon’s. Like Jeff Bezos, Mark Zuckerberg foresaw a future where the flow of information was critical to the success of his product. Driving people to Facebook would be an absolute requirement. Like Lorenz, Zuckerberg’s power of observation, and his response, ultimately defined his genius. If the flap of a butterfly could change the course of a storm, what revolutionary potential would be ignited by the power of a thousand, 10 thousand, or 10 million contributors? Now imagine the force that is Facebook today, which boasts 800 million users as of January 2012.

At the time Zuckerberg first got started, there were many tools available to create “social networks”. Many sites offered group forums, distribution lists, and chat rooms. There were technologies like texting and RSS that provided for broadcasting too. While these tools provided for community interactions, most individuals did not have the time, knowledge, or discipline to “integrate” these technologies into spaces convenient to their unique lifestyles. This severely limited the potential base of possible contributors, and ultimately restricted the size of the overall social network. With Facebook, the interface suddenly made it possible for anyone to “microblog” with almost no previous experience and an absolutely minimal setup burden.

Facebook’s ease-of-use naturally made the conventional communities of practitioners more accessible and easier to join as a regular member. The platform made it easier to spontaneously publish, subscribe, and respond, allowing a degree of freedom and expression at an unprecedented level of practicality and richness. Much like the liberation that Bezos brought to publishing, Zuckerberg provided a means for people to focus and exploit their ideas at their own discretion and convenience. Good ideas would spread faster, cheaper, and more broadly than ever before. Not surprisingly, a Darwinian-like feedback system evolved, helping to improve ideas for those authors inclined to invest the time to update them. Effectively, Facebook provided an easy-to-use interface for integrated (“rich”) 2-way communications, without the restraints of conventional third-party intermediaries. The power of change is now influenced as much by “social capital” as it has historically been by economic capital” or political capital.

Mark Zuckerberg – Versions 2, 3, 4, 5…: Manufacturing a Revolution through Opportunity

With Facebook, Zuckerberg added to the scaffold created by Ritchie, McCarthy, Jobs, and Bezos. What each of these inventions provided was a means of tying society closer together, leveraging technology to bridge the common human challenge of distances, languages, interfaces, and education. Zuckerberg
arguably added another, that of opportunity.

Opportunity has always been a key ingredient to creating and sustaining a revolution. Opportunities are born from ideas that, once surfaced, find the means to attract and hold the attention of their audience to the point where action can be taken. It was in 1962 that Everett Rogers described this “socialization” phenomenon in his book “Diffusion of Innovations”. Rogers studied hundreds of cases of innovations with varying degrees of success and proposed four main elements that influence the spread of ideas: (1) the innovation potential, (2) communication channels, (3) elapsed time, and (4) a functioning social system. Zuckerberg, whether directly or indirectly, has provided the momentum for a new functional social system, crafted from the aforementioned innovations, and driven by the post-PC society.

In fact, unlike many of the other innovators discussed here, one could argue the Zuckerberg is deliberately driving a mission to create not just a new technology platform, but a society based on ideas and information. While certainly Facebook is a money-making enterprise, it’s not at all clear that revenue alone is its driving force. Zuckerberg seems driven by the need to connect people together. Facebook has already expanded the use of the Internet across all generations. In reality, Facebook has changed the overall manner of communication. In a nod to Steve Jobs, Facebook’s recent platform move to develop iFrame Applications will enable the company to establish a method to further evolve social applications, including new kinds of Customer Relationship Management (CRM) applications. Today, for example, Facebook is used to provide a platform for business and government organizations of all types. By melding the concept of customer and provider into a powerful platform fostered by individual and collective recommendations, Facebook is changing the nature in which influence is peddled throughout our society. Its influence has recently even found its way into politics, becoming a key mechanism to raise money for candidates that would likely have been left out in the past. Facebook is manufacturing a revolution through the realization of opportunity.

**Innovation and Invention – the Cornerstones of a Revolution**

Both individually and collectively, Dennis Ritchie, John McCarthy, Steve Jobs, Jeff Bezos, and Mark Zuckerberg have changed the world we live in. In saying our farewells to Ritchie, McCarthy, and Jobs, we need to reflect not only on their specific accomplishments, but also on what they have taught us about changing the world. I doubt that any of these individuals saw the world simply as pre-, during, and post-PC. The PC was simply an instrument through which to realize a vision -- a vision that we as a society can bring perspective to now and in the future, as evident by the timeline of innovation below.

---

**TIMELINE FOR REVOLUTIONARY INNOVATIONS**

*Revolutionary Enterprises*

*Revolutionary Technologies*

Illustration created by the author.
Creating a Revolution – Lessons from the Experts

If there is something to take away from all of these innovators, it may be that changing the world is about accommodating a few common principles:

- First, *accessibility* is the key to stickiness. Whether you're accessing a device through C or someone's knowledge through AI, being able to have access is paramount. Often this comes in the form of common languages, common structures, common interfaces, and common tools that have evolved through “open” source movements and standards committees. Interestingly, the economics of Amazon and Facebook have been driven through access to their ecosystems as well.

- Second, *connections* create value exponentially. Whether you're connecting resources through an operating system like UNIX, connecting customers through a platform like Amazon, or connecting individuals through an interface like Facebook, having the ability to find and maintain a connection is critical. A fascinating read behind the mechanics and philosophy of connections, particularly around social networks, is “The Strength of Weak Ties” by Mark Granovetter⁶.

- Third, *platforms* create an architecture for innovation. The innovations we've identified here, whether post-PC or not, all focused on various kinds of platforms. From way back in Gutenberg's time, to Carnegie and the more recent past, a platform became the means of evolving new solutions while taking advantage of the past. Backward compatibility, a feature best demonstrated by Bill Gates, is important to keeping momentum behind a movement. Taking a lesson from Ronald Coase's “Theory of the Firm”⁷, platforms reduce the cost of transactions, whether it be financial as he defined it or societal as Bernd Heinrich did in his illuminating text “Bumblebee Economics”⁸.

- Fourth, *influence* acts as a noise capacitor. In 1948, Claude Shannon published his seminal work on information theory⁹. Broadly speaking, Shannon discussed the need to reliably communicate information over noisy lines. With so much information, over so many different interfaces, in so many contexts, innovation must continually find ways to reduce the noise and focus on information. Apple's Siri takes an inductive approach, while Wolfram Alpha focuses on deduction. Relationships based on trust within the Facebook and LinkedIn social networks establish and influence the validity of ideas and information.

- Lastly, a *vision* beyond pure economics is essential. All of these innovators (excluding Gutenberg) either benefited directly or indirectly by their ideas in financial ways. However, their energy was not focused on revenue alone. That is why Larry Ellison of Oracle, for example, will not likely enjoy the celebrity that Steve Jobs has in the general public. The individuals highlighted here set their goals on bridging the penultimate gap between now and the future. They saw a better way, and they pushed their ideas forward. Ultimately, they provided a path for us to follow and improve upon.

From Gutenberg to Zuckerberg, these men of technological history have carried the torch of innovation from one revolution to the next. They serve as examples of an individual's ability to empower society, which reflects the true nature of the post-PC era.

References:

2. Ritchie and his colleague, Ken Thompson, wrote the UNIX kernel in C in 1973.
4. http://www.loc.gov/about/generalinfo.html#2010_at_a_glance
Big Iron Results in Big Energy Savings. Why Mainframes are sustainable.

by Cynthia Curtis, Vice President and Chief Sustainability Officer, CA Technologies

When people think about Green IT, you don’t find a lot of people who immediately say, “Oh yeah, that’s all about mainframes.” Affectionately known as Big Iron, this tried-and-true technology that has been around for years is not on top-of-mind when a discussion about energy efficiency in the data center comes up. But it should be for medium and large companies with significant computing requirements, according to Dayton Semerjian, General Manager, Mainframe at CA Technologies.

“Mainframes are quite simply the most energy efficient computing systems available today. To start with, an entry-level mainframe provides the computing capacity of about 1500 servers and it does so in a device about the size of a refrigerator,” says Semerjian. “And because mainframes work as a single integrated system, they are designed to make extremely effective and efficient use of their vast computing resources.”

As I discussed in a blog post earlier this year, energy savings is one of the key benefits of server virtualization in a distributed server environment in a data center. By having multiple virtual machines run on a single physical server, companies have used virtualization as an effective means to lower energy costs. But here again, don’t count mainframes out.

Mainframes have been running at a 95 to 100% virtualization rate for years – well before it became all the rage in distributed environments. The closed environment of a mainframe allows the system to make extremely efficient use of its computing resources in much the same way a more complex, distributed virtualized environment does.

More Reasons Why Mainframes Are Energy Efficient

- Lower total power costs: According to Sine Nomine, a research and analysis firm, mainframes are responsible for 1/7 the total power cost, when compared to distributed server environments.

- Lower HVAC costs: Sine Nomine also estimates that mainframes are responsible for only 1/20th the energy costs of dedicated HVAC systems.

- Less space: According to IBM, mainframe environments only require 40%
of the physical data center space required by distributed server environments.

- Less area to cool and maintain.
- Less support: Fewer boxes and more efficient, integrated machines means smaller support staff per unit of computing capability. At up to a 25% reduction in support reduction according to Clabby Analytics, this means less energy consumed supporting the machines.

Mainframes may not right for every business. However, for the many organizations where they do make sense, there are clear sustainability benefits, including lower energy costs, lower operating costs and a rapid return-on-investment. And when you add in a mature, stable, secure environment, what’s not to like?
Commoditization in the Cloud Calls for a Sticky Solution

by Jacob Lamm, Executive Vice President, Strategy and Corporate Development, CA Technologies

To ensure its success in the cloud, it has never been more important for a product to be “sticky”. Paradoxically, given the nature of the cloud, “stickiness” has never been harder to achieve. With cloud computing, the commoditization trend long observed with hard goods, such as mother boards or salt, can now be observed with software products. As markets mature, competition increases. To stay in the game, manufacturers try to match each other feature for feature. Goods become homogenized and eventually commoditized. This leaves software manufacturers to focus on how to retain customers by making products sticky in an environment where software choices abound and are increasingly swappable.

In a previous article on commoditization¹, I referenced the maker of Wonder Bread, which capitalized on the commoditization of the toaster by marketing pre-sliced loaves as “the best thing since sliced bread.” Two months after the article was published, USA Today reported that the company filed for bankruptcy². Was it something I said?

The article stated that Hostess Brands, which also manufactures Twinkies, “has enough cash to keep stores stocked with its Ding Dongs, Ho Hos and other snacks for now. But longer term, the 87-year-old company has a bigger problem: health-conscious Americans favor yogurt and energy bars over the dessert cakes and white bread they devoured 30 years ago.”³ If the iconic Twinkie can’t maintain market share, then what hope is there for the rest of us?

I thought about our own IT industry. The cloud and its inherent flexibility allow for frequent software component changes as a company’s needs change or a competitive product is perceived as more feature-rich or economical. Within the cloud, a company presumably is less committed to any particular software component. Changing products may cause some disruption, but nothing like the months or years of planning and testing that may accompany the replacement of a deeply entrenched legacy system. And with pay-as-you-go pricing, the upfront investment needed in the cloud is smaller than would be needed for other pricing models. Again there is less commitment to a particular application.

How do you not only keep but increase market share when your software product exists in an environment that touts among its most attractive features the fact that software products can be easily replaced with little effort or cost? In other words, how do you make a software product sticky in the cloud? Not in the sense of being too difficult to disengage from, but rather in the sense of being too attractive to leave? The answer is to leverage the cloud’s unique characteristics while taking a “back-to-basics” approach to satisfying, even delighting, customers.

Product Appeal

A “must-have” for product stickiness is of course the dependability of the...
product itself. Does it do what it claims to do and does it do it well, even elegantly? Is the deployment experience smooth? Here is where initial trust is built between the customer and vendor.

Once a product is found to work reliably, the feature race is on. What do competitive products offer? And at what price? To maintain and build market share, products may not have to meet each competitor’s new feature or discount one-for-one, but some effort is needed on this front to stay in the game. Conveniently, the cloud accelerates the rate at which new features can be introduced and facilitates easy deployment.

Preferable to merely matching competitors’ features is inventing new ones that increase product breadth and appeal to a broader customer base. A clear advantage goes to those products that contain useful and yes, cool, innovations. This requires an investment in development and research and goes a long way toward building a brand that is perceived as relevant.

For example, Big Data has been getting a lot of exposure lately, though many enterprises are still challenged to harness its power in the data center. To the extent software products can make sense of Big Data and uncover valuable nuggets of information, they will find themselves with a committed customer base.

Similarly, products that take advantage of the mobile device trend will enjoy customers unwilling to revert to more restrictive platforms.

**Customization Appeal**

At first glance, customization may seem more likely in on-premise installations. However, the cloud offers many opportunities for content customization. In fact, because software in the cloud is meant to accommodate a wide range of users, customization opportunities are of huge importance in product design. The degree to which content has been customized either for or by the customer has a direct correlation to product stickiness. For example, investments in customized interfaces and workflows, and knowledge bases built over time, all maximize a product’s usefulness. With tailor-made content, there may be little reason to look at competitive products, conveniently obviating the need for inconvenient retraining efforts.

The prospect of walking away from and having to replicate a significant customization effort is an added deterrent.

**Community Appeal**

A software manufacturer must not only make the product itself hard to leave, but make the community surrounding the product hard to leave as well. By “community” I am referring to shared experiences that a consumer feels comfortable - even happy - being a part of. When referring to communities in the cloud, there is strength in numbers. A strong community denotes strong brand allegiance. Factors that strengthen the community also strengthen the brand. A dependable, glitch-free product promotes a community that values quality. Product innovations promote a community that values progressiveness. Along those lines, Apple has done an especially great job of building teams of product loyalists who wouldn’t think of leaving their community of self-proclaimed techno geeks.

A software manufacturer must not only make the product itself hard to leave, but make the community surrounding the product hard to leave as well.
The experiences that make up a community include all interactions that surround the product. Good service is paramount to promoting a strong community and should characterize technical and customer support and sales. Robust and intuitive websites will help with this, but even cloud-based businesses may involve human contact, which should be caring and knowledgeable.

A community is somewhere customers are comfortable maneuvering. One-stop-shopping, transactions and configurations that leverage previously entered or learned information, and consistent interfaces contribute to ease of use.

Social media offers additional ways for users to connect with each other and the vendor to share information. The more connections made and the more valuable the information shared, the more value is associated with community membership. Technical user communities have an especially strong hold over customers who both contribute and surf for product tips and extensions.

**Summary**

To retain and capture market share in the labile cloud environment, attention must be paid to improve and optimize every experience a consumer can have related to purchasing, deploying and using products. The cloud’s unique characteristics should be leveraged towards this end. These efforts will pay off by making software manufacturers successful and their products sticky-- not too difficult to disengage from, but instead too attractive to leave.

**References**

1 Lamm, Jacob. The Best Thing Since Sliced Bread: Opportunity Born of Commoditization, CA Technology Exchange, November 2011

2 Maker of Twinkies, Wonder Bread files for bankruptcy, USA Today, January 11, 2012

3 USA Today
Are Tablets Ready for Mainstream Business?
by Marcel den Hartog, Principal Product Marketing, CA Technologies

Business people are just like ordinary people. And so they have tablets.
I frequently travel and see them in use at airports, in planes and in hotels. They are a convenient travel companion, light, connected and full of useful and entertaining applications. Since we are carrying them around anyway, why not use them to do email, read next day’s PowerPoint presentation, or study the one that was sent out after the yesterday’s important meeting? Why not use a tablet instead of a laptop? It can do almost everything the laptop can do anyway?

A tablet is a personal device, really personal. It offers no privacy. Once configured, it’s configured for you and not for anybody else. All the Apps you install assume there is one user on the system. This includes email accounts, social networking (with some exceptions), you name it. Once you have entered your details, everybody else using the tablet can see everything you have done unless you just use your browser, tell it never to remember passwords and close all tabs when you are done.

Here is something I found quite shocking. The first time I entered a password, I noticed that just before changing it into an asterisk (“*”), the screen showed the letter I just typed! I was shocked and in the weeks that followed, I found I could simply collect most of the passwords of people using an iPad. Call me paranoid, but a password should be protected and never, ever show up on a screen. It is easier to fix your eyes on a single place on the screen and remember characters than it is to figure out what people are typing. Android users can disable this feature (though it is on by default). The more business oriented Blackberry does not have this problem because the manufacturers have understood that it is sometimes better to trade in some user friendliness for better security.

Imagine people sitting close to you entering their password (which you could easily remember) and then reading their downloaded email. Opening attached presentations and spreadsheets and browsing through them. So you could read not only what arguments they were using to convince a potential buyer, but also why their CEO was unhappy about Q2 and the actions he took to improve on that. Why the market potential in Eastern Europe looks so good for a specific offering and how much they were spending on Marketing. Information that was not meant to be seen by you, but that could only be avoided seeing by closing or averting your eyes. You could have been that competitor, You could have been the friend of the manager under fire, or You could have been the customer...

For tablets to be adopted as business devices, this problem should be solved in a “tablet” way. For example, an app could remember passwords under control of a master password entered once, or go further and use the camera or microphones that most tablets have to implement biometrics security. With many apps already available that use the camera and built in microphone for many different things (apps like SoundHound or Shazaam recognize hummed

About the author
Marcel den Hartog is Principal Product Marketing EMEA for CA Technologies Mainframe solutions. In this role, he is a frequent speaker on both internal (customer) and external events like GSE, IBM’s zUniversity and many others where he talks about CA’s mainframe strategy, vision and Mainframe trends. Marcel joined CA in 1986 as a Pre-sales consultant. Before this, he worked as a programmer/systems analyst on VSE and MVS systems, starting with CICS DL1/IMS and later with DB2. He is still an expert in CA Easytrieve and Cobol and has hands-on experience with many CA products. He was responsible for managing CA’s pre-sales teams in The Netherlands, Belgium and South Africa for a number of years. Prior to his current role Marcel worked as a Linux Development Architect for CA’s Linux and Open Source team. In that role, he served almost two years as a board member of the Plone Open Source Community.
songs or other apps that use the built-in cameras in very creative ways), this could be a killer app for business people.

**Are Business Users Ready for Tablets?**

One of the problems we face is the fact that the tablet is used for both personal and business purposes. At home, we don’t have to hide our tablet from others, and we seem to forget that in public places, we have to be more careful. A tablet gives you feeling of being almost in a private space that can make you forget the unfriendly world around you where you have to be alert. Using important and sensitive information comes with responsibilities, and this is an often forgotten topic when we discuss new technologies that will be used in public spaces.

A tablet is no laptop and it also is no smart phone. The screens of tablets have brilliant, rich colors and are very easy to read, even from a “safe” distance. When handing out tablets to business users or allowing them to use their own for business purposes, we must warn them about the social responsibilities of carrying a tablet.

**Is Mainstream Business Ready for Tablets?**

Once the enterprise accepts that its employees and clients are using tablets, the next step is determining what to do to take advantage of this. The temptation is to say “We have many systems that already have a web interface; people can surely use them on a tablet.” Yes they can. But tablets are popular because they do tablet things, not because they do the same things any laptop can. And using a browser in a tablet can be a frustrating experience because most websites are not designed for a tablet! A simple redesign to avoid some of the generic issues (URL’s too close to each other is the most annoying one) can already do miracles. If we want to use tablets as management tools, we have to build the right applications for them. And this means going back to the drawing board, doing proper analysis and architecting and assigning a group of developers to it. After all, if we can turn our client facing applications into easier to use and more attractive apps, will we do more and better business?

Yes, absolutely. But there are other things for you to consider.

The hours in which people access your systems and websites will change, and so will the behavior of your systems. Performance management is key; apps need to be snappy and the back-office needs to be well maintained and monitored. Users remove slow responding apps in seconds, and they do leave comments behind in the various app stores/app markets. Once the damage is done it can take months for an app to become popular again.

Second, you can make your staff more productive. The tablet is with them all day, and they will be more interactive and socially connected with their colleagues. The management portals should be easily accessible and easy to use by upper management. Nobody wants to admit that they are not able to use a tablet app.... Do not underestimate the cost to build an app for a tablet device. The
technology is new, the way people interact is new and we have all experienced in IT that “new” means “longer projects and unexpected cost”...

**Are Tablets Ready for Mainstream Business?**

Sure they are. A device that can show me where certain stars are (http://vitotechnology.com/star-walk.html) by just holding it up to the sky can do anything. I personally experience the effectiveness of my tablet daily and see many other business people who seem to feel the same. All we have to do is educate our business users how to keep others from looking over our shoulder when we type our passwords. And maybe some direction when it comes to email usage, storage of important documents, use of proper encryption and security controls, and so on.

I still remember the first weeks of using my tablet. I felt something I had not felt for a long time. This was a device that had a potential I could not even grasp. After downloading more and more apps, using my own tablet for business and travel more and more, I came to realize that it’s almost like a mobile phone. I was a happy person without one, but once I got it, I could not even imagine what life was like before. I feel the same with my tablet.

Tablets are here for business, ready or not. Welcome!

*I was a happy person without one, but once I got it, I could not even imagine what life was like before.*
Introduction

While the technology for information security is rightly serious about defending against intrusion and breach, the cultural dimension of security strategy may be the most important key to getting a return on the investment in protecting information. The most secure environment is one in which every participant is routinely motivated to protect their personal benefits by minimizing or preventing the improper access and uses that result in losses.

Twelve years into the new millennium, with relentless innovation of information tools, we have crossed a line and are now accustomed to the idea that we get most things done in a “Post-PC Era” of computing. Information streams flow everywhere with few boundaries and, as information users, we expect to be able to get information from anywhere at any time, unless circumstantially prohibited. Yet neither the “personal” nor the “computing” aspect of what we do has gone away. In fact, both have intensified, and diversified, by orders of magnitude. So what does it mean to say “Post-PC”?

What we're really talking about is not the PC, but the desk. Portable laptops broke away first, but carried the desktop paradigm with them. The desktop itself evolved from being physical to digital, then from digital to virtual. Now, the desk is an idea that we realize on-demand; instead of going to it, or carrying it around, we compose it and bring it just-in-time to wherever we are. This in turn leads to improvisation, and the desk itself, as the organizing instrument, disappears.

The consequence of this evolution is twofold: a different set of structures to contain and arrange the information must be used, and the information protection once based on securing the desk must be built into securing the new structures. The paradigm has changed. In the Post-PC era, the information user is not only a fundamental part of the structure, but is its most influential component. Securing the structure must be done, just-in-time, by qualifying the user. Security architecture must incorporate the user as a component, which makes behavior, or culture, the key to security success.

The Lesson of Conversations and Personas

The pressure to provide on-demand access to information is more intense than ever, but it has not changed the basic strategy for protecting the information. A user needs permission to do the following tasks:

- Enter the information container
- Search for the information
- Use what is found

These three fundamentals are supported, respectively, by a user ID or credential, a content subscription, and an approval. It is, basically, that simple.

What is now interesting about the strategy is how to manage and execute it in the future. On the management side, there is a parallel view of the
circumstances:
- Container <=> Source
- Search <=> Topics
- Permitted use <=> Requirement

The parallel seems to be very abstract until we give the group of terms on the right side a name: a conversation. In a conversation, with the parties taking turns, a party goes to someone else, selects what they are interested in, and pursues a certain kind of value from what is heard.

To do that, each party (source) addresses the other according to protocols (how to interact), grammars (how to describe), and ideas (what to express). The interaction is regulated. The information that is used in the interaction is constrained for the purpose of being appropriate to the pursuit. It is this propriety that is the goal of the interaction. Security has the same primary goal, not to prohibit but rather to govern the propriety of what is delivered or exchanged. In the wide-open world of spontaneous improvisational personal computing, security needs an implementation that is as effective as how we manage conversation.

On the execution side of the strategy, the “protocols, grammars, and sharable ideas” of the conversation are deliverables that must be supplied as the pipelines of the desired information. In effect, the pipelines are made of rules of assembly.

Rules-based security is, of course, not at all a new idea; however, things have changed. Historically, the rules have been used to declare the one way that information could be used and who was allowed to use it that way. Today, the objective of the rules is to support providing different versions of the same information, according to the profile of the individual user.

The diversity of users is already familiar to us, for example, through user interfaces that dynamically adapt and differ according to user profiles or roles. However, for security purposes, key “interfaces” must be implemented and manifested at a deeper level than the surface touched by the user. This must occur at the rules level and at the information level.

Yet first, and equally important for security effectiveness, the user profile now needs to be discovered on-the-fly, because users can change them so easily from one moment to the next. Personalization is the feature that lets users distinguish their profile from other users. With personalization, a set of options is selected by the user to describe what the user is like, and interactions between the user and the tools proceed based on the selected options. We normally think of this as configuring the tool but, from the point of view of the tool, we are actually configuring the user. As we know already, the result is that the same tool can support different kinds of users.

In the post-PC era, a large portion of a company’s workforce is post-desk, with many individual workers composing their mental workspace on-the-fly with a broad collection of information-handling tools. Any work session can use multiple different tools to manipulate any given type of information; and many types of information may be manipulated by a given type of tool. The diversity of tools addressing the diversity of information can create interaction possibilities too numerous to predict so, as any session of information usage begins and proceeds, the challenge is to have the experience be successfully self-organizing to conclusion.

Security has the same primary goal, not to prohibit but rather to govern the propriety of what is delivered or exchanged. In the wide-open world of spontaneous improvisational personal computing, security needs an implementation that is as effective as how we manage conversation.

The diversity of tools addressing the diversity of information can create interaction possibilities too numerous to predict.
Dynamic self-assembly of the information pipeline requires that the elements being called on to cooperate know how they are allowed to interact with each other. The resulting just-in-time pipeline is responsive to the improvisations of the user. In principle, we understand this already, having had a decade or more of experience inventing and refining heterogeneous, multi-component connectivity. But to achieve the modern goal of managing appropriate information transfer, we want the information itself to be able to declare how it can and should be used. This ability of the elements to explicitly self-organize is represented in an architectural principle called Service Orientation.

**The Impact of Services**

Services are the advertised appropriate uses offered by something available for interaction. The advertising is done by any number of mechanisms that allow the appropriate uses of that thing to be identified on-demand when the item is found. As long as finding the item can immediately trigger its advertising, we can say the item is self-declaring, and from there rules can be invoked to decide if the item will interact with something else.

Information that is self-declarative predisposes its assembly into the stream of delivery that reaches a user. Familiar examples of making information service-oriented include classification and tagging. By recognizing that these proactive, content management actions indicate what uses are appropriate, we understand that they are elements of a security strategy. In more exotic implementations, heuristic determination of the information's conceptual and functional types may be called on to serve the same purpose, performing the declaration just-in-time.

The strategic goal of appropriate information transfer requires that the user is also an entity that self-declares. The more improvisational the user is, the more likely the user’s profile can change from one time to the next. Therefore, each session of information use must compare the user’s immediate profile to the target information, a comparison during which rules determine how the user and the information interact.

**Just-in-Time Security**

Just-in-time security, like conversation, has the ability to run flexibly and continually without a predicted endpoint. It is a processing technique, not just an event. The circumstances constraining what the information user obtains are managed, but the management is formalized in a logical model instead of a physical environment. Consider the following model:
An entity involved in the interaction brings an explicit profile to the interaction event. The profile includes protocols (how to interact), grammars (how to describe), and ideas (what to express) brought into play as services supported by the entities in the information interaction.

At an event level, each entity can make a request for service from other entities.

An agent can manage the visibility of an entity’s services.

A broker can perform discovery and negotiation of entities and their services.

Negotiations can be based on rules.

Entities can subscribe to agents and to brokers.

Any number of entities can “chain” themselves together, creating the pipeline or channel needed between the beginning point and the final endpoint (for example, user and target information).

In this model, the strongest point of enforcement is in the negotiations. The broker, in effect, may be the gatekeeper of information by determining whether any appropriate fulfillment is available to the requester.

To emphasize that the model is just a management pattern, consider that these entities, agents and brokers could all be people – that would be one way of implementing it, but the same pattern has the same purposes if the entities, mechanics and functions are provided mostly as technologies.

Meanwhile, what we have realized about people (information users) equipped beyond the classic desktop or laptop PC is that the same individual may show up in a given information environment with a profile that varies from one occasion to another. This is particularly noteworthy because, with the service-oriented model, including agents and brokers, the primary risk of “breached security” is not anticipated or described as a user breaking the rules that assemble the pipeline unless the user is also the broker.

Instead, the most likely point of breach is in the presentation of the user’s profile. In the model, an agent represents the user. Therefore, misrepresentation is an issue. With the unprecedented variety of information tools in hand, the profile of the user (Entity X) has a much greater opportunity to change from one interaction to the next, while the profile of the information (Entity Y) remains unchanged. Two important scenarios associated with this volatility are: unanticipated profiles that pass lax qualification criteria, and intentional disguise or impersonation.

The primary means of managing the possible volatility of the user profile are:

- Exclusion (no accommodation provided)
- Supervision (choice of profiles is prescribed and monitored)
- Training (choice of profiles is coached toward a habitual state)
- Incentives (choice of profiles is encouraged by reward)

Here, management does not mean “to control” as much as it means “to direct”. The intent is not to reduce access to information, but to assure appropriate information transfers. Management here assumes that, if it is successful, the person will voluntarily present a profile most likely to result in an appropriate use of information. As a result, allowing multiple profiles is less of a risk, and much less of a threat.

With risk being managed, the flip side of the coin is the business benefit of flexibility for the user. Increasingly, business opportunities to advance or
preserve advantages are identified by the user before anyone else. Both innovation and agility are predicated on being able to capitalize on those opportunities in the heat of the moment.

When we factor in benefits, it is all the more notable that, of the above four user management options, two are primarily a cultural influence on the entire population of users. The management message is that what is good for the business is rewarding for the individual worker. Training and incentives, when clearly aimed at moving the user towards community-desired outcomes, recruit the community’s peer influence as a type of feedback, reinforcing the interest each user has in reducing security risks. The intended recurring user experience of security managed this way is that the individual’s best personal outcomes tend to result from the user’s effort to strongly comply. We look for this co-operative persona.

**Beyond the Desk**

There have always been famous and familiar non-desks in business, including:

- Telephones
- Golf courses
- Restaurants

Not coincidentally, the common thread of productivity in those famous non-desks has always been one thing: the appointment. Knowing and agreeing, ahead of time, on what kind of information would be made available and when, is the classic paradigm of the non-desk.

We left the desk behind because business competition pressured us into doing that. Business itself actually made desks inconvenient. Now looking back over the long history of classic non-desk business, what’s the worst thing that could happen? Easy: spending the appointment with the wrong person.

So, from the point of view that you are a user pursuing information, “security” means that you want assurances that when you need to find something, you will find it, you can do something about what you found, and that it is actually the right thing to do something about.

Today, we like to think of this non-desk paradigm as having been superseded as well – by something we call “on-demand”. The on-demand paradigm does not require an appointment. It works mainly because we act with reliance on two very advanced things: how to interact with an information system, and how to find information and the systems that host it. We do this as a pursuit with an agenda.

A long version of the story about “agenda-based pursuit” is not as interesting as a short version. The short version is that information pursuit today occurs within a vast “open” universe of material, and the most proven way to efficiently get something from pursuit is not to simply be demanding, but to use a broker and an agent.

When brokers and agents work together for you, a “channel” is formed, and it becomes possible to make that channel increasingly discriminating by virtue of specifying your demand. The channel then does a lot of the work for you, which creates a degree of practical freedom to explore, with less chance of coming up empty.

Strategically, the collection of these channels will amount to the degree of information security being maintained. As security practitioners, we will call on
this to help deal with all users, notably including those whose agenda is not so friendly.

Meanwhile, the most likely threat of rendering information unprotected is a motivated insider’s knowledge of how to do it. Technical experts exist both within and outside of the community and can always pose a threat to information security from either direction. But insiders, who need not be experts, have the kind of opportunity that makes it more dangerous if they lose the motivation to protect information and instead allow or perpetrate its misuse. The primary advantage that an insider has is the opportunity to move information from a normally protected place to an unprotected place.

To complement the attractiveness and ease of being compliant, being non-compliant must be either equally unattractive or as discouragingly difficult. As part of a security strategy, the intent of instituting this resistance is likewise under more and more pressure from the vastly increasing opportunities for leaving information unprotected.

Today, once information is “out of the gate, off and running”, it is still effectively impossible to take it out of circulation. The value of this circulating information might degrade over time, but the immediate impact of its initial inappropriate exposure could be harmful enough to make that later degradation a moot point. The originator of the information may be directly victimized by its subsequent misuse, or be held liable for the after-effects that its improper use has on others.

The motivation of an individual who places information in an unprotected state might be attributed to any number of things that may be anticipated by types but are not foreseeable by timing. What’s worse, operating freely beyond the desk-based paradigm, the majority of information end-users now make “ordinary” demand harder to confront than before. It is not the volume of users that is the problem; instead, when information pursuit is a machine-driven event, it takes other machines to stand up to the demand generated. Free-wheeling users armed with machines are what demand looks like today. The information may hop carriers incredibly quickly, entering multiple different channels and formats that leave little trace or trail, so that we no longer fully know where else the information appears.

In the face of such risks (and associated costs) of information becoming unprotected, we should take a tactical stance, noting several factors where tools can either cause the problem or help eliminate it:

- The likelihood of information being recognized when the antagonistic insider first finds it
- The length of time that the information remains in the location that is accessible to the insider
- The inability of the information to leave any evidence that it has been manipulated in a certain way

But hypothetically, short of directly reducing the number of potential users, the only real tactical defense against this risk is to make the information itself highly perishable whenever it exists outside of its planned and authorized environment. The information could automatically disassemble unless its lifespan was extended by the storage mechanism intended to be its host. The embedded “glue” that holds the data together as information could stop working and the data could dissolve or revert to a disorganized state. Such a capability seems futuristic but we’ve thought of it before: the Mission
Impossible tape that self-destructs in fifteen seconds, digital rights management and/or the software license that auto-expires, and of course the poison pill of malware... In fact, the disappearing glue feels less like fiction and more like the end of an R&D climb.

Today, the universal “glue” that we work with is different from that; it is connectivity. The pipeline established between the user and the information source differs in the following ways:

- By making the connection perishable, we begin to minimize the opportunity that the information can be improperly displaced.
- Randomizing the connectivity further minimizes the improper opportunity by reducing the predictability of unchallenged access.
- The accounting of each access of the information stages a history of interactions that may be positioned as highly consequential, subject to investigation and interpretation based on circumstantial evidence.

It is beyond the intent of this article to call out the many intervention technologies already in use on information transport. However, techniques that make inappropriate opportunism expensive for the user strongly address a strategic objective of culturally implementing security. The greater that expense becomes, the lower is the probable perceived value of the opportunistic behavior, and therefore fewer people will do it.

Naturally, a point of balance must be found between the importance and the expense of obtaining the information. Management’s responsibility is to establish the “market value” of the information within the community of intended users, and this matter requires that the information be dealt with not just as material but as an asset. Addressing a cultural level of influence, a strategic technology for information security is therefore technology that manages information as an asset. In association with that strategy for information security, content management and knowledge management are both essentials.

The Culture Shift

So far, we have characterized an aspect of security called Assurance that we can think of as playing “offense,” while we can characterize the defensive side of security as Prevention. We have also looked at various circumstances in a way that differentiates the supply of and demand for information.

The Information Culture: Strategic Framework for Protection of Information

| DEMAND: Request information of appropriate value | PREVENTION: Prevent omission of protection |
| Asset management | User identification |
| Provisioning rules | Knowledge management |
| Subscriptions | ???
| Content management | Surveillance |

On the Prevention side, what is most noteworthy in the new era of multi-user, on-demand information access are two intertwined dominant issues: anonymity and persistence.

On-demand behavior in the emergent new era is spontaneous, high-speed, and
relentless. Information access devices have proliferated in consumer information usage. Targeted for consumers, the devices bring with them a set of expectations that make sense for fostering selling and community-building. However, the expectations do not align with the typical requirements of production management within the business organization, where cost, quality and legal factors must be carefully orchestrated under executive accountabilities for performance.

The dramatic increase in the number and variety of access devices makes for diversity that challenges security’s ability to be adequately supportive. What is most challenging is that the devices are machines with the extraordinary ability to continue information searches and requests indefinitely. This amplification of demand on information sources means that an inappropriate attempt to access information may be far riskier, the worst case amounting to a sustained attack under cover of a much larger volume of demand.

At the same time, outside of production management work, broadcasting is the main type of initial information exposure conducted by suppliers. However, this is true not so much by intention as by default.

With little control over the actual subsequent distribution of information on the internet, information suppliers tend to think about information in terms of two locations: areas where information goes unprotected, and areas where it does not. This segregation of information intuitively makes sense; we know that, in ordinary life, security is generally about creating and maintaining protective locations and about setting location boundaries according to what range we can afford to control.

Just as we would outfit a traveler to be able to protect himself in an unknown environment, we now need to understand how to get information to remain protected as it passes through the endless expanse of locations within the internetworking that can expose it. We need this understanding because, due to available technologies, we must assume that the information is more likely to move than it is to stay still. The information is the traveler.

Logically, the two primary options for preventing inappropriate exposure are:

- Create secure channels of passage for the information
- Enable the information to otherwise protect itself

How are these modes utilized already? Again the answer is familiar, at least by analogy.

For a secure channel, the traveler can resort to disguises, escorts and bodyguards, or can call a taxi if he suddenly feels he is standing on the sidewalk of a not-so-great neighborhood. Getting timely directions to the routes that are already safe is important. This range of self-protection, from the full-time to the just-in-time, requires that the traveler has the ability to maintain a direct link to the form of protection for as long as necessary.

Taking this idea a little further, we can say that the traveler subscribes to the protection, and that each form of protection is a service. To make this work, the traveler must be noticed leaving the original Safe Location X, which should invoke the active protection in one or more ways. Thus, the first challenge we have in managing this approach is to determine that the traveler is in motion and consequently to apply the service.

When information is the traveler, surveillance is the base-level requirement for activating the protective service. The main reason why suppliers do not
implement surveillance of their own information is that surveillance is thought to be costly, but making surveillance inexpensive is not about settling for the low-price method. Instead, surveillance is about using the method most likely to protect information that generates high levels of benefit when it reaches its supplier's intended destination. This understanding about surveillance reveals the cultural side of security: the value that the information supplier attributes to the information should provide the justification for how it is treated.

In today's hyper-connected information environment, the most prevalent practices for information discovery are not based on business-level concerns. Seen as private property, information accessible through current consumer-class technology presents the essential problem of being sufficiently well-managed in a space dominated by, and largely engineered for, the expectations for public access. Creating adequate spaces for private information makes sense mostly if there are equally adequate channels to those spaces. The outlook for information protection is that the greater the percentage of information users subscribing to authorized channels, the safer a supplier's information can be. Strategically, this means that users must come to prefer channels over ad-hoc discovery; to promote this evolution, the supplier should help the user construct the channel and provide benefits to the user in that channel. In doing this, a critical adjustment occurs when, as a subscriber, the user stops being anonymous and becomes identifiable.

From the user's casual point of view, anonymity is too easily confused with privacy. The correct distinction is that, with anonymity, we don't know who the user is even if we know what they are doing; whereas, with privacy, we don't know what the user is doing even if we know who they are.

Beyond the boundaries of organizational membership, the default expectation that an information user has today is anonymity. This expectation is reinforced by technology that maximizes the convenience of making an anonymous information acquisition for a private use. Properly speaking, anonymity does not cause misuse but, when trying to protect information, anonymity of users makes it harder to prevent the inappropriate exposure of the information.

Information suppliers want to exercise forms of deterrence, but a protection strategy should be a long-term stance that breeds inherent, not just circumstantial, protection. If protection is successfully based at a cultural level, most information circulated by users will continually be at lower risk. Instead of information suppliers designing security levels to reduce the number and type of attempted breaches, these suppliers will design security levels to manage information in ways that make a breach increasingly less useful to the would-be intruder given obviously-available alternatives.

The information supplier should try to replace the most popular idea – that anonymity creates the most convenient access – with a different idea: privately benefiting from a public domain is not hampered by being identified. To promote this, the supplier should make it easy for users to recognize when information the supplier puts in a public domain is suitably valuable for private uses. Establishing this pattern of promotion encourages users to do predictable things and additionally to ask suppliers if they can have things that are not already there. The requests give suppliers the opportunity to arrange for appropriate security before new information is released.

A Cultural Framework for Security

Compiling the additional observations above completes a draft framework for

Surveillance is about using the method most likely to protect information that generates high levels of benefit when it reaches its supplier's intended destination.

Suppliers will design security levels to manage information in ways that make a breach increasingly less useful to the would-be intruder given obviously-available alternatives.
tackling the cultural side of security. The draft is not expected to be exhaustive but instead representative. In this framework, a variety of practices are recommended. Independently, many of these practices are enablers of things outside of security concerns. But to address the security strategy culturally, the positioning of the practices within the framework is as important as their inclusion. And the security goals of these practices can be approached by defining how the respective practices offer their protective influences with the consistent nature and visibility of known services. These definitions or services will vary according to the specific organization considering them; however the framework provides the same guidance to all organizations.

**The Information Culture: Strategic Framework for Protection of Information through Security Services**

<table>
<thead>
<tr>
<th><strong>DEMAND:</strong> Request information of appropriate value</th>
<th><strong>PREVENTION:</strong> Prevent omission of protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSURANCE: Ensure inclusion of protection</td>
<td>User profiling Knowledge management</td>
</tr>
<tr>
<td>Asset management Provisioning rules</td>
<td></td>
</tr>
<tr>
<td>Supply: Engineer value into the information format</td>
<td>Channels Information marketing</td>
</tr>
<tr>
<td>Subscriptions Content management Surveillance</td>
<td></td>
</tr>
</tbody>
</table>

Implementing security at a cultural level means shaping and adopting the value system that is to be practiced by all members of the information user community. The adopted value system influences behaviors; behaviors drive information usage; and outcomes of the usage reinforce the value system.

That path to a desired conclusion is attractive but, despite the best logic and planning, cultural change is usually more about directing influences than it is about achieving predictable results.

Today, because of consumer adoption of sophisticated IT, the default expectations that information users have are based on the idea that security is an environmental condition and that the environment is present before the arrival of any producers of the information in it. In that view, any information “found” in an unprotected environment is “up for grabs” unless some other party actively chases it down to enforce its exclusivity.

That perspective sees security merely as an “ecology” separated from management. To instead provide security as a managed effect, the expectation needs to be set by a policy comprised of the following points:

- Security is a service
- Most of the valuable information in the environment is available through the service
- The service is operated by a combination of suppliers and provider
- Different types of security are offered as different classes of service
- All classes of service must conform to some independent standard that can be audited
- Outside of the service, there is not a state that will be called or treated as “secure”

This is a very tall order, more of a target state than a reality. However, we have proof that organizations take this level of ambition seriously and learn to accommodate it. The most obvious precedent for this level of ambition about
adoption is a working reality: Finance. In recent memory, the chaos of non-standard business accounting led to the Sarbanes-Oxley Act, a cultural force that had the primary effect of very painfully changing expectations everywhere about tolerance of risk.

In a different but similar example, the worldwide ISO organization translated much of the IT Infrastructure Library (a de facto standard called ITIL) into layers within an actual industry-ratified standard (ISO20000), another cultural force that changed expectations about what was tolerable – this time regarding IT management effectiveness.

We call these examples “cultural” because, as game changers, they had to be adopted (as a fundamental stance) even more so than implemented (as a specific new current state). Those who did not try to adopt could not play at all, because their purpose for being in operation was to do things co-operatively with other organizations, and other organizations did not cooperate with parties that did not adopt.

Many may think of these “adoptions” as being draconian impositions, greatly restricting the options of users, but the idea of security is not one that stands in opposition to creativity, differentiation, or advantage. As with Finance, the purpose of practicing strategic security is to advance the cause of the organization at a behavioral level. Also, we already have great experience with the required modes of transitioning and implementing the practice. In making the related efforts to reach the goal, we see that “getting” security relies on administration, whereas “keeping” security relies on management.

**Conclusion**

This discussion is not intended to provide a comprehensive prescription for how to establish security in the new era of mobile wireless smart devices, machine-to-machine communications, and the ongoing survival or eventual supplanting of Ethernet and TCP/IP.

Clearly, the advanced systems broadly available as a computing environment for personal use allow people to more easily consider alternative modes of producing things. This in turn will fuel more experimentation, more demand, and more permutations of things that may warrant protection or distribution.

Against that backdrop, a socialization of security must be a top priority, and technology needs to be just as responsible for that socialization as it is for the properties and products that need to be secured. Services should become a gateway that also practically implements security, and role enablement tools should be the installed components of the security architecture.

As technology continues to morph and improve in the direction of individual user autonomy, security must evolve in a way that positions it as the best connector of users to suppliers, and not as a barrier.

This will not be a controversial stance to take, as anyone who places and values their financial holdings in an institution already knows. Instead, the evolution of security will highlight the need to more deliberately assign value to the things being demanded, and to treat them appropriately in all respects, not just as a matter of protection. Security will not be an independently successful operation, but instead will be one strategic dimension of managing the value of desired business properties.
Personal Computing Without the PC
by Sridhar Chakravarthi Mulakaluri, Director of Operations, CA Technologies

Abstract
With the adoption of information technology growing quickly in our daily lives, it is imperative to understand the way IT is being consumed by people. This article recognizes the universal need for personal computing and focuses on publicly accessible personal computing. An analysis of the economics of kiosk computing and the impact on society, including the potential disruption to the current trend (iPad, Blackberry, and so on) is presented. Some common experiences and use cases in India are described along with the potential outcomes of this trend. This article helps us rethink and redesign the tools required to manage the changing way in which IT is provisioned and consumed.

Introduction
Technology has been invading the present day urban lifestyle. It has also started making its way into the rural regions where access and use of relevant information is redefining the economy and lives of many marginalized citizens. Regardless of their socio economic background, people are finding ways to use information technology to improve their daily lives.

Changes in the internet over the past 15 years are changing many business models and redefining how people access goods and services. Embracing this opportunity, technology companies have started developing the latest devices to access and consume these new age services such as i-Pod, i-Pad, tablet PC, and so on. The design of these latest devices is very intuitive, making them very easy for almost everyone to use. This has again changed the game by redefining the accessibility of technology.

Under these circumstances, where more and more people are becoming aware of the power of information and communications technology (ICT), they are also aware of the high cost of the latest devices. Many of them have become a style statement and are fashion accessories, although at a steep price. Consequently, this marginalizes much of the user population again.

Both users and providers have started recognizing the potential and the opportunity associated with ICT, which redefines the concept of personal computer to personal computing and making it device independent. Many of the latest devices are being sold on the premise that people who do not own them or use them are old school. However, the device provides a mere convenience of access and does not in any way effect the core service. This, in fact, provides an opportunity where a user can still enjoy the convenience of ICT without owning the device to access the information or service. Welcome to the new post-PC era where accessibility and point of access are becoming independent.

The Opportunity
Everyone wants to access and use most of the modern amenities and services like transportation. Similarly, the need for information has become all pervasive.

About the author
Sridhar Chakravarthi Mulakaluri is the Director of Operations at CA Technologies in Hyderabad. He has 20+ years of rich and versatile experience across various verticals, which includes working with the Indian Air Force, manufacturing, and into IT and business consulting, as well as an entrepreneurial stint for a few years.

Sridhar has worked as a Management Systems Auditor and consultant in the areas of Information Security, Disaster Recovery and Business Continuity Management, IT Service Management, Quality Management and Project Management. He has helped more than 200 organizations to improve their systems and processes. He has experience as a developer, tester, process consultant and project manager in the IT industry, and has trained in Japan on various manufacturing processes and techniques.

Sridhar drives the various CSR and Sustainability projects at CA Technologies. Sridhar also volunteers as a coordinator for IT Industry CSR Forum in Hyderabad and with Dr. Reddy’s Foundation.
and the demand is growing exponentially with trillions of gigabytes of data being added to the already cluttered internet every day. The ability to use a PC is still limited and is a major deterrent to accessing and using the technology driven services. However, the improved user interface has made this redundant. The best example that comes to mind is the ATM. Regardless of the level of education and technology literacy, the ATM was designed to enable access to the selected banking services by almost everyone. Education is no longer a barrier due to multi-lingual ATMs and visually impaired people can access some of the modern ATMs. By allowing access to one’s bank account through any ATM, the banking sector has come up with another interesting model. Although this service was chargeable for some time, today it is almost free.

While the banking industry has made good inroads into the market by leveraging technology, the other industries have yet to catch up. The Indian Railways has installed information terminals at major ticket reservation centers where people can browse for seat availability, booking confirmation status, and train related information. While all these facilities, including internet ticket booking, are also available through a website and secure payment gateway, the kiosks remove the dependency on the device for general information.

A similar phenomenon is noticeable in movie theaters where public kiosks are installed to allow people to buy tickets without going through the trouble of queues. While on-line ticket booking exists, these kiosks offer the flexibility of accessing the service without owning the device. In airports, free Wi-Fi services and kiosks where people can browse the internet free of cost are being set up. All these kiosks have machines where access is intuitive and very user-friendly.

The Economics

Today these kiosks are viewed as an additional facility to the customers; however, they can be turned into profit centers, not only through cost reduction, but also as a marketing platform.

E-commerce has shown that the consumer who accesses web services need not be the paying customer. Companies can identify opportunities for advertising, product display, and exclusive access to services, and so on. This will make the whole proposition much more attractive and economically viable. The kiosk should be attractive enough to draw attention as well as users who will use the facility.

Take the example of airport displays. A few companies have tried them already by installing exclusive kiosks that provide access points to their products or services. Today we depend on human intervention to do the same thing, which limits the potential. It is very similar to a vending machine. Today physical and tangible products are sold through vending machines. Why not services? What kind of services can be sold with minimum customization? The challenge is to identify and offer these services through these kiosks.

This means that we need a lot of product/service innovation where they become off-the-shelf products. Many financial services are very intricate and complicated and we need an experienced financial advisor to explain them. However, if the same products are made simpler and clutter free, the ease with
which these services can be bought and sold can improve drastically, making an ideal candidate for kiosk sales.

All the companies who want to offer their products and services through the internet or these kiosks must focus on product innovation. The simpler the products are to use, the easier it is to make a buying decision, and the faster the sale will be. Another big concern is the secure payment gateways. When it is addressed through comprehensive messaging and market education, this concern will go away.

Another challenge is the organization and the availability of latest information. This is where a good number of jobs can be created by proactively creating the information and knowledge banks. The information should also be made available in multiple languages to address the user challenges related to education. This is another interesting aspect of economy driven by these kiosks.

The biggest advantage the kiosk model will have over the handheld devices is the ease of maintenance and upgrading. All kiosks can be dumb terminals/thin clients that access servers with the latest information, which results in low maintenance costs and better antivirus management, access restriction, and content management. Unlike kiosks, iPods and iPads will become obsolete very quickly as the cost and duration of ownership is a challenge with handheld devices. In comparison, kiosks can access resources from the public cloud and pay per use, further offsetting their costs.

**Impact on the Present Trend**

A future where a handheld device for accessing information becomes redundant may not be far off. While the “fashion accessories” continue their market strategy, that market may not really grow. It will be more of a replacement market. A loyalty program or buyback discounts would make more sense there.

However, the number of users who would like to use ICT will grow exponentially and their needs must be met to realize the market potential. This will result in many public kiosks and will become a challenge to create and attract users. One possible opportunity could be creating a set of products and services that are available exclusively through these public kiosks, which drives the traffic there.

These can also be collocated in restaurants, coffee shops, bus stops, super markets, and other public places. This can develop into a complex business model where the property owner, the kiosk owner, and the information owner can have multi-level agreements. Based on the location, there can be some exclusivity of services as well. For example, a grocery shop may promote certain recipes and cuisine based on the ingredients being promoted.

Another very important area that is opening up rapidly is e-governance. Many government services are offered online and the kiosk model will help citizens access these services. We can have customized kiosks that will make life easy for the users in terms of accessing the right information at the right time. In the near future, more and more citizens will want to access and use web-based services without necessarily owning a computer.

Considering all these factors, we can conclude that kiosk driven personal computing is destined to grow significantly and has a global potential, particularly in developing countries.
At CA Labs we’ve been working on a tool for internal use - dubbed “Reacto” - which provides a new approach to measuring the scalability of a software component in an enterprise-scale environment. (From now on we will refer to the component we are testing as the “component-under-test” or CUT.) The general approach of the Reacto framework is to create lightweight, realistic, models of the interaction between the CUT and the other components or systems in the environment. These models are then executed in an emulation engine and the CUT communicates with the emulated components as though it were talking to real systems. The key aspect of the research is to balance the modeling such that it is rich enough to “fool” an unmodified CUT into thinking that it is talking to real systems, but lightweight enough such that tens of thousands of instances of model systems can be executed simultaneously in the emulation engine.

The upcoming release of CA CloudMinder® Identity Management, part of the CA CloudMinder suite, presented a perfect opportunity to use Reacto to validate CA CloudMinder’s scalability at managing many thousands of endpoints. The component of CA CloudMinder that communicates directly with the managed endpoint systems is the Java Connector Server (JCS). Reacto was used to emulate 10,000 mock endpoints for the JCS to manage, enabling us to measure the scalability and performance of the JCS itself. Each emulated endpoint had its own IP address and an independent state.

Using Apache JMeter, we invoked the JCS to conduct a set of typical identity management operations on each endpoint. We executed this test using up to 100 concurrent threads, and used CA Introscope® to monitor the JCS memory usage, CPU usage and response time. The test confirmed that CA CloudMinder and the JCS satisfactorily scale to manage 10,000 endpoints. We also provided the resource consumption information to the software architects. One unexpected result of the test was the discovery that the endpoint caching mechanism used by the JCS used more memory than expected. This information was thus used by the software architects to improve the design of the JCS. This demonstrates the real utility and practical benefit of using emulation for scalability testing.

Comparing Reacto to some other approaches, there is a gap that is filled by emulation. One common approach used by QA teams is to use virtual machines as the test endpoints. We compared the resource consumption of our emulation to that of using VMware virtual machines. The response times of the

About the authors

Dr. Steve Versteeg is a Research Staff Member with CA Labs, based in Melbourne, Australia. His role is to coordinate collaborative research between universities and CA Technologies. His current projects are in the areas of software engineering, role engineering and anti-malware.

Steve’s PhD research was in the area of neural simulation. A well studied neural circuit was used as a case study for re-creating robust behaviour in computer systems.

From 2004 until early 2008, Steve worked at WMind LLC as a senior developer and researcher on an experimental automated futures trading system. The system searches the history to identify instances in the past most similar to the current scenario and then predicts the most likely future movements. Steve holds a PhD in Computer Science from the University of Melbourne.
Reacto emulated endpoints were similar to those of virtual machine endpoints, but Reacto’s resource consumption was far less. For a similar number of endpoints, Reacto uses about 100 times less CPU time and 1000 times less memory, plus Reacto endpoints require very little hard disk storage. The low computing resource consumption enables us to emulate 10,000 endpoints on a single physical host. VMware ESXi 5.0 has a limit of a maximum of 25 virtual machines per physical core. Another kind of tool commonly used in performance testing is performance tools like HP Load Runner. The important distinction here is that performance tools are generally designed to emulate a large number of clients to test a server. But what we required for the IdentityMinder-as-a-Service testing was a large number of mock servers.

Reacto’s approach does correlate very nicely with a product that CA recently acquired - ITKO LISA. The CA LISA® Virtual Services Environment for Performance Testing application automatically creates models from the observed message traces between a CUT and the other services in the environment. This enables customers to accurately test the impact of new versions of a system without affecting the production environment and without needing to access the real systems in the production environment. CA Labs is now collaborating with the ITKO team.

Cam Hine is a Postdoctoral Fellow at the Swinburne University of Technology in Melbourne, Australia. Cam has been collaborating with CA Labs for five years, working on the Reacto/Kaluta Large Scale Testing project during his PhD candidature (also at Swinburne) and as a post-doc. Cam’s PhD thesis is titled “Emulating Enterprise Software Environments” and investigated large scale emulation and modelling.
The JavaScript Revolution: How JavaScript will shape the future of cloud computing

by Nathan Giardina, Senior Software Engineer and David Tyree, Senior Architect, CA Technologies

The JavaScript Revolution is underway, a technology that has been around since 1995\(^1\) is poised to dramatically change the shape of enterprise and cloud computing. Traditionally thought of as a client and browser side technology, NodeJS\(^2\) has managed to bring JavaScript onto the server with surprising results; allowing for complex framework development, traditional app server like tools, and making major in-roads in enterprise and cloud computing. Today’s JavaScript not only has the performance and scalability required to develop and service the cloud, but also offers a relatively low time-to-market for new products and a true “write-once-run-everywhere” environment.

Given these benefits, is JavaScript poised to become the technology that weaves the cloud computing fabric of the future?

Before that question can be answered, a definition of cloud computing is required. Cloud computing is a new way to host and manage computing resources; instead of enterprises managing their own computing resources they are instead hosted and managed on the Internet.

This new paradigm brings many new challenges with it and technologies that may have been well suited to enterprise solutions are not feasible in the world of cloud computing. In the Cloud applications are accessible from anywhere, are heavily dependent upon web browsers and other open standards, and may be accessed simultaneously by thousands of customers at once. Traditional enterprise technologies frequently lack the scalability, ubiquity, and fluidity required to operate in this new environment. JavaScript on the other-hand has found itself well suited to meet these challenges.

JavaScript, which is typically thought of as web browser only programming language, is now being utilized far beyond web browsers. Look at any highly dynamic web application today and JavaScript is likely the underlying technology. But with the introduction of NodeJS, server-side adoption of JavaScript is growing. Many large companies have started to make commitments to NodeJS and JavaScript in public and visible ways. Microsoft has announced on MSDN\(^3\) that their cloud computing service, Azure, will now support NodeJS as part of their offering, while LinkedIn has found that using NodeJS in support of their mobile applications has allowed for a significant performance improvement over other popular technologies\(^4\), EBay also uses NodeJS to meet its scalability needs and Yahoo has announced Manhattan and Mojito as their platform offerings, which are basically a NodeJS cloud and an MVC architecture for NodeJS\(^5\). Many other companies\(^6\) have also started adopting NodeJS and JavaScript to meet their scalability and performance needs.

The adoption of NodeJS is partially due to how widely supported it is in the open-source software development world. Examining GitHub, a popular code hosting site, used by more then 1 million open source developers\(^7\), it becomes

---

JavaScript was initially designed in 1995 for Netscape Navigator 2.0, and is a blend of both Scheme and Java\(^2.5\).

---

About the authors

Nathan Giardina is a Senior Software Engineer with the Advanced Technologies Group in the Office of the CTO at CA Technologies. He has worked for CA Technologies for over 11 years and has had experience with several major products in that time as well as every aspect of the software lifecycle from research and development to sustaining and support. Nathan is well versed in Java, J2EE, Perl, and Ruby, and C/C++; he is currently focusing on JavaScript and NodeJS.

Nathan has written several mobile applications for Android and iPhone and is interested in cross mobile platform development.

David Tyree is a Senior Architect with the Advanced Technologies Group in

---
apparent that JavaScript is having a growing influence. GitHub's most chosen language for their over 2 million repositories is JavaScript, now comprising 20% of all software projects on GitHub (see Figure 1)\(^8\).

The JavaScript revolution is occurring and will continue to grow as more organizations struggle to meet the need for highly dynamic and scalable cloud and web applications with current technology. But why is this occurring? Because JavaScript has significantly matured as a technology and its unique features are particularly well suited to building highly scalable applications. Though originally conceived as a programming language for use in web browsers, and a victim of the web browser wars in the late 90s, JavaScript hasn't had a particularly great reputation.

Previously it has been difficult to achieve consistency across browsers leading to distinct code bases of JavaScript for each browser (software that worked in Internet Explorer didn't work in Netscape, and vice-versa). The drive towards highly dynamic web applications and the introduction of new web browsers such as Google’s Chrome and Apple’s Safari have shifted the browser wars in favor of JavaScript, resolving most of the incompatibilities and focusing browser manufacturers on performance instead.

Microsoft, Google, Apple, and Mozilla are trying to achieve the fastest JavaScript environment possible for their respective web browsers, bringing advanced technologies such as Just-in-time compilation\(^9,9.1,9.2\) and ‘type inference’\(^10\) to the various JavaScript execution environments. The results of these technologies are that JavaScript is beginning to become competitive performance wise with entrenched enterprise platforms and languages (see Figure 2)\(^11\).

The drive towards highly dynamic web applications such as GMail, Google Docs, and other thick-client browser based web applications will only continue to drive investment in JavaScript performance.

The proliferation of smart phones will continue to leverage performance requirements on JavaScript, along with making it much more ubiquitous. With the introduction of the iPhone in 2007 the possibility of having a full browser on a smart phone was realized, and by doing so made JavaScript one of the most

---

David's experience with web technology extends far beyond JavaScript and NodeJS, he has extensive experience building web applications using a variety of technologies including PHP, Perl, Ruby, C++ and Java and J2EE.

Prior to joining CA Technologies, David worked at Verizon Business as a Senior Architect in Verizon's Managed Security offerings, focused on both compliance and security event management offerings. David holds a number of patents in the field of computer security, his background is strongly focused on computer security.

Nathan and David have been collaborating with other team members to research and prototype applications using NodeJS and Javascript, both for cloud applications and mobile applications.
widely distributed technologies available today. Considering that almost every
desktop and smart phone now has a fully functional browser with a JavaScript
execution environment on it, and the adoption of NodeJS for server-side
JavaScript, the promise Java once offered of “write-once-run-everywhere” is fully
realized in JavaScript.

The server side portion of JavaScript, NodeJS, was developed in response to the
demands for highly scalable network I/O. NodeJS is based upon Google’s V8
JavaScript engine (V8 is written in C++), but offers a unique value proposition;
instead of gaining scalability with multi-threading similar to other enterprise
based languages, NodeJS is built leveraging the asynchronous nature of
JavaScript to provide a tightly coupled and robust event based I/O capability.
This event based I/O is very different from the thread-based model found in
most development environments today.

But how does JavaScript’s asynchronous nature help scalability for cloud-based
applications? Essentially asynchronous programming is a different
programming model which in many cases is much simpler then the comparable
multi-threaded model, and avoids the overhead of context-switching between
threads, allowing NodeJS to provide a significantly higher level of scalability
when dealing with I/O. Traditional desktop applications do not require efficient
and high performance I/O as there is often little delay between a desktop
application’s standard I/O (reads and writes to the hard drive are near
instantaneous), and the computer itself is often more than well equipped to
handle activity from just one user.

Moving past the need for traditional desktop applications and into applications
that are housed in the cloud, there is an increased need for more efficient I/O
performance. A program is no longer just accessing information on a hard disk,
but rather communicating with another server on the network for data. These
servers are no longer running just one application for one user, they are running
at peak performance, handling as many requests as they can and as many users
as they can. These individual servers all comprise the fabric that make up the
cloud, and each of them is highly dependent on I/O performance, and in this
case JavaScript’s event driven paradigm is often more efficient, performs better
at high load, and is easier to develop than traditional thread based models.

JavaScript allows functions (closures) to be parameterized and executed when
I/O becomes available, meaning that processing cycles are not wasted until the

Going mobile!
PhoneGap is a framework for
developing mobile applications
that work across multiple different
smart-phone platforms.
It leverages the smart-phones
native browser and JavaScript to
access native device features such
as cameras and GPS.
data or resources related to the task are available. NodeJS makes hooking functions to I/O related capabilities possible; for example a function can be hooked to a file read, the function will be executed once a file has been loaded from disk, so until the data for the file is available the function won’t consume CPU resources. NodeJS works similarly with network I/O. This is important because most operating systems are highly optimized around handling I/O and offer specialized function calls to trigger tasks when I/O becomes available (i.e. epoll[10]).

What is a closure?

A closure is when a function and its scope are parameterized, for example:

```javascript
function makeACounter()
{
    var i=0;
    return function(){ return i++; }
}

//counter is a closure
counter = makeACounter();
counter(); // returns 1;
counter(); // returns 2
```

How closures work with asynchronous I/O

Closure can be used to handle I/O. In the example below, a file is read and then imported into the database. The important thing to remember is that the functions won’t be called until the associated I/O is available.

```javascript
//onComplete is a closure provided by the caller
// in NodeJS almost all closures expect an 'err'
// argument as the first argument to the closure.
function importFile(filename,onComplete){
    fs.read(filename,function(err,data){
        if(!err){
            db.import(data,onComplete);
        } else {
            onComplete(err);
        }
    });
}
```

Couple the ability to handle large amounts of I/O on the server-side with the ubiquity of the JavaScript execution environment, and you now have a technology being heavily invested in by technology bellwethers, that has penetrated almost every type of computing device available today.

The advantages for JavaScript go beyond its ubiquity and potential scalability: demands for more dynamic web applications have brought other valuable additions to the JavaScript landscape, specifically JavaScript Object Notation (JSON)[13]. JSON is a data interchange format similar to XML, and is widely supported and used. It was initially proposed by Douglas Crockford and specified in RFC 4627[14] and is supported natively on every JavaScript platform. JSON makes interchanging data between JavaScript environments efficient and easy. Unlike XML, JSON is not strictly defined and is easily validated, but being native

JSON: JavaScript Object Notation

Here is what a person object might look like when specified in JSON:

```json
{
    lastName: 'Tyree',
    firstName: 'David',
    email: 'david.tyree@ca.com'
}
```
to JavaScript makes it particularly easy to both produce and consume; the trade-off being simplicity instead of strong-typing. JSON makes it very easy for servers and clients to communicate and to communicate between JavaScript environments.

Ubiquity, scalability and ease of communication make for a pretty compelling cloud computing underpinning, but is JavaScript really poised to succeed? Examining trends in technology can help illuminate the future for JavaScript and its potential as a platform. The drive towards and demand for highly dynamic web applications will continue to shift how we think about web browsers. Once commonly considered only thin clients, many organizations are making large investments in dynamic web applications, all based upon JavaScript; in fact with the introduction of HTML5, many of the features of Flash are now accessible to JavaScript. Even Adobe recognizes the future of JavaScript and has started making investments in JavaScript by purchasing PhoneGap, a JavaScript based mobile device development platform, and phasing out Flash based technologies in favor of JavaScript technologies (This is partially driven by Apples unwillingness to support Flash on its mobile devices).

Continued technology advancements will drive web browsers to become the thick clients of the future - this is apparent when considering the recent additions to web browsers. Most popular web browsers now contain advanced 3D APIs, storage APIs (including SQL), image generation APIs, sound APIs, file APIs, geo-location APIs, network APIs, drag and drop APIs etc. These additional APIs are accessible via JavaScript and have made the web browser a very compelling application platform.

It seems likely given the investments of Microsoft, Apple, Google, Mozilla and Adobe that advances in JavaScript on the client side will continue.

Given the advances on the client side (web browser), what is happening on the server side to advance JavaScript towards becoming the cloud computing technology of choice?

At the moment JavaScript via NodeJS on the server side is relatively immature, and is growing organically. NodeJS has over 5600 modules available via NPM (Node Package Manager), providing a vast amount of functionality and capabilities. But many of these modules are not in a common package format making their reuse difficult at best. The open source community is working to address these needs, by working towards a common module format that will make re-use of JavaScript modules on both the client and server side much easier. Furthermore the open source community is also quickly building out frameworks and libraries, providing an ever growing level of functionality for NodeJS.

Examples of some of the modules for NodeJS:

<table>
<thead>
<tr>
<th>SQL Databases</th>
<th>Oracle, MySQL, PostgreSQL, ODBC, SQLite, Drizzle, etc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoSQL Databases</td>
<td>Redis, MongoDB, CouchDB, Memcached, etc</td>
</tr>
<tr>
<td>Other</td>
<td>LDAP, XML, SOAP, HTML5 Canvas, REST</td>
</tr>
<tr>
<td>Web</td>
<td>ExpressJS, Jade, HAML, etc</td>
</tr>
</tbody>
</table>

The growth of NodeJS in software components is yet another indicator of what is to come, but not the only one, many companies are beginning to offer hosted NodeJS environments and PaaS services, for example Microsoft, Nodester.
Nodejitsu, Heroku, and AppFog. These organizations are making bets that JavaScript and NodeJS will have a bright future and that adoption of NodeJS and JavaScript will continue to grow.

**Short list of NodeJS providers:**

<table>
<thead>
<tr>
<th>Provider</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joyent</td>
<td><a href="https://no.de/">https://no.de/</a></td>
</tr>
<tr>
<td>NodeJitsu</td>
<td><a href="http://nodejitsu.com/">http://nodejitsu.com/</a></td>
</tr>
<tr>
<td>Heroku</td>
<td><a href="http://devcenter.heroku.com/articles/node-js">http://devcenter.heroku.com/articles/node-js</a></td>
</tr>
</tbody>
</table>

Given not only the continued investment in JavaScript and web-browsers, and the suitability of NodeJS for cloud-based services, the argument for JavaScript continues to grow; however there are detractors. JavaScript does have a number of challenges facing it, specifically the high learning curve, a series of poor language design decisions, and the vastly different development environment compared to existing popular enterprise environments.

JavaScript and related technologies do have a steep learning curve compared to many other technologies and languages. This is due to both JavaScript's loosely-typed prototype-based design and its asynchronous nature, setting it fairly far afield from Java, C#, PHP and other more traditional synchronous OO (Object Oriented) based languages. Design patterns and approaches that work well for these languages aren’t immediately obvious how to apply in JavaScript, resulting in many software developers fighting the language instead of embracing it. Callback function calls are a prime example of this shift in design patterns. Most other languages make it difficult to pass function calls as parameters; in JavaScript however, it’s as easy as passing any other variable because of JavaScript’s loosely-typed nature. This can be frustrating to a typical Java programmer who is used to a function performing a task and then returning back a result.

The steep learning curve is further perpetuated by a number of poor design decisions (variable hoisting, global variables, the ‘with’ keyword, type conversions, etc.) that were implemented in the language. These decisions often create even more stumbling blocks for developers and implementers who are new to JavaScript. Because of the steep learning curve and design flaws a number of efforts are under way to both provide easier to use languages on top of JavaScript, such as CoffeeScript, and to further improve JavaScript’s existing design in the form of ECMAScript Harmony. Although it is important to mention that there are some efforts by Google to replace JavaScript with alternative languages such as Dart, they have yet to gain significant traction.

There have been other contenders for this language revolution: Ruby, Java, .NET, and PHP just to name a few. Each of them, while not failing, will not be the language of the cloud for various reasons. .NET failed due to it’s single platform design; it only runs on Windows, and while Microsoft might have some success with .NET on Azure, it will never be as prolific or as widely adopted as other languages. Java is struggling due to many factors, not the least of which being speculation about Oracle’s long term plans for Java, but also the complexity of building enterprise applications in J2EE. While Android definitely gave Java a boost, currently Java doesn’t quite “run everywhere”, it requires a large
infrastructure and configuration for it to function, and has a very large server footprint. Ruby and PHP can be held up as other excellent contenders however their lack of ubiquity, and investments from technology bellwethers will keep them at a disadvantage to JavaScript.

Designing cloud based applications using NodeJS and JavaScript will seem foreign to many implementers who are familiar with J2EE and other similar application frameworks. NodeJS and JavaScript require a significantly different architecture approach and different components. Instead of a single application server implementers are likely to use multiple small discrete processes, as NodeJS is inherently single threaded, with message buses and storage components coordinating processes. This style of architecture, if designed correctly is particularly apt for construction of cloud services, as it can scale fairly easily.

The JavaScript revolution is underway; a technology once despised has begun to become more widely used with greater and greater investment being placed in it. Its ubiquity, scalability and unique feature set have the potential to make it the fabric used to weave the future of the cloud.

References
1  http://oreilly.com/pub/a/javascript/2001/04/06/js_history.html
2  http://www.nodejs.org/
2.5  http://www.2ality.com/2011/03/javascript-how-it-all-began.html
4  http://venturebeat.com/2011/08/16/linkedin-node/
6  https://github.com/joyent/node/wiki/Projects,-Applications,-and-Companies-Using-Node
7  https://github.com/blog/936-one-million
8  http://github.com/languages
9  https://developer.mozilla.org/En/SpiderMonkey/Internals
9.1  http://code.google.com/apis/v8/design.html
11  http://shootout.alioth.debian.org/u32/which-programming-languages-are-fastest.php

Figure 2 reprinted pursuant to the following permission:
Copyright © 2004-2012 Brent Fulgham
All rights reserved.

- Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
Neither the name of "The Computer Language Benchmarks Game" nor the name of "The Computer Language Shootout Benchmarks" nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

12 http://kovyrin.net/2006/04/13/epoll-asynchronous-network-programming/
13 http://www.json.org/
14 http://www.ietf.org/rfc/rfc4627
15 http://html5demos.com/
18 http://www.apple.com/hotnews/thoughts-on-flash/
19 http://search.npmjs.org/
20 http://www.commonjs.org/specs/
21 http://jashkenas.github.com/coffee-script/
22 http://ejohn.org/blog/ecmascript-harmony/
23 http://www.infoworld.com/d/application-development/first-look-google-dart-vs-javascript-179852

45
Tablet Application Infrastructure Considerations
by Kenneth Kranz, Enterprise Architect, CA Technologies

Introduction

When smart phones were finally accepted within the IT infrastructure, the thought was that a flood of IT-centric applications would soon follow, yet that did not happen. As time has demonstrated, smartphone-based IT application monitoring and control apps are challenged by the cramped screens and limited input of the form factor.

Tablets, on the other hand, are almost as portable yet offer significantly more options. While sales of tablets, specifically iPads, have been mostly limited to the consumer market, their use in the workplace has been growing; many times despite many corporate IT restrictions. It is reasonable to expect that, in the very near future, tablets will become first class citizens in corporate America – with some employees receiving tables in place of desktop or laptop computers. Consequently, demand for tablet-based IT applications will soon rise. If this prediction plays out, IT directors will have to ponder the implications of developing and supporting tablet based applications.

This article discusses:

- Hardware and software architectures that are well suited for tablet deployments.
- Skills that are required to develop this new breed of application.
- Strengths and weaknesses of developing applications for tablets rather than traditional platforms.
- Differentiation between tablet-based applications and web-based applications tailored to tablet browsers.
- The fundamental bandwidth limitations that shape and lead to these solution approaches.

This article is for both IT managers contemplating engaging their staff in the tablet development space, and members of the technical software development staff who want to know how this new technology will affect them. I will delineate the boundaries and obstacles faced by IT managers and executives considering tablet application development, and hope to provide insight into this emerging field and relate tablet development to the traditional issues of client/server projects.

Tablet History

Before discussing the impact of tablet development on IT staff or infrastructure, a brief history of tablet marketplace is in order. Tablet history has been uneven and difficult to follow. This is especially true when you consider the blurred distinction between extremely “smart” smartphones and what the tablet can do today.¹

While IBM displayed the Simon concept phone in 1992, the first true smartphone appeared in the marketplace nearly ten years later. Even then, IBM was not the first to envision a tablet-like device. In the late 1980’s Apple

About the author

Kenneth Kranz is an Enterprise Architect with the Catalyst-USM Project. He is currently helping to formulate architectural solutions incorporating multi-tenancy within the Catalyst product line.

Ken joined CA Technologies in May of 2011. Prior to joining CA Technologies, Mr. Kranz was a principal architect for Cablevision Systems Corporation (CSC). At Cablevision he was responsible for the design and development of Click-to-Call, a Skype like SIP based telephony product. In addition he had significant input on the design of the backend content delivery architecture for their iPad based TV viewing app.

Prior to joining Cablevision, Ken was a senior architect for Sun Microsystems, Inc. (SMI) for 12 years and a member of their distinguished engineering mentoring program. While there, he was involved in a wide range of emerging technologies and products; not the least of which being the SOA based Java Composite Application Platform Suite (Java CAPS) Open Enterprise Service Bus.
introduced the Newton line of devices that were commercially available until 1998. Before that, in 1968, Alan Kay of Xerox PARC envisioned the Dynabook. In his abstract, he described the Dynabook as a “personal portable information manipulator”\(^2\). It is hard to say if he actually envisioned the first laptop (it had a keyboard) or tablet (it has a slate form factor) but it was not far off the mark of where we are today.

In 2000, early entries in the smartphone market included the Symbian based Ericsson R380 and Nokia 9210. They were soon followed by Palm/Handspring, Microsoft, Research in Motion (RIM), and many other handset vendors. The first company to introduce a modern tablet was Microsoft circa 2001 with they released specifications for their Tablet PC (and later recast as the Slate Computer) running Windows XP Tablet Edition, which superseded Windows for Pen Computing – are you following all this? Be it ergonomics, hardware limitations, or simply timing, and like the Newton, the Tablet PC eventually fell out of favor.

Around the same time, Palm Inc. was working on their Treo and then Pre series of smartphones. The Palm phones were highly convergent, boasting touch-screens, email, web browsers, contacts, memos, to-do lists, SD slots, camera, graffiti, Wi-Fi, Bluetooth and 3rd party applications. Around the same time, RIM, having already locked in the corporate market, was focusing on their “berry” line of phones and applications. Because of the BlackBerry’s secure network policies, they became an early favorite of the corporate world.

More recently, Palm launched the Foleo – yet another precursor to the tablet - (see Figure 1 – Tablet Device Time Line) in 2007. The Foleo was actually a Sun notebook computer that ran Linux.\(^3\)

![Figure 1 – Tablet Device Time Line](image)

While never intended as a tablet, the Foleo was designed to access the Internet via Wi-Fi tethering with the Treo (and only the treo). The aim of the Foleo was to offer a device that was more functional than a small smartphone, yet more portable than a standard laptop. It also seamlessly mapped files from the Treo to the Foleo. While the Treo was successful, the Foleo was not. Despite its lack of market success, the Foleo was the first concrete attempt in the marketplace to close the usability gap between the limited smartphone and the real-estate rich laptop.

By the beginning of 2008, the major players in the “smart” smartphone market
effectively consisted of Palm, RIM and Apple. While we might look back and marvel at how obviously successful the iPhone was, it was not readily apparent at the time. Back then, RIM was the most popular device, with double the sales of the iPhone. In third place, many people considered Palm’s WebOS powered Treo/Pre line as the odds on long-term favorite from an innovation standpoint. While the iPhone was a strong contender, many dismissed its popularity as being limited to “Apple devotees”. Case in point, while Apple’s sales were stronger than Palm’s, they were weaker than RIM’s – again, RIM being driven by the corporate market.

In Q4 2008, iPhone sales accelerated and have not significantly slowed (see Figure 2 - Worldwide Apple iPhone Sales⁴). This chart depicts a significant up-tick in iPhone sales in Q4’08 with accelerated sales moving forward. In fact, the only dips in iPhone sales post Q4’08 were tied to either anticipated new device releases or supply issues⁵. This raises the question, what happened in Q4, 2008?

In the illustration below (Figure 3 - Application Store Unveilings) we have taken the Tablet Device Time Line (Figure 1) and augmented it with the Grand Openings for the various application stores – beginning with Amazon’s Kindle Direct Publishing in November of 2007 and ending with Amazon’s Appstore in 2011⁶.
In this new timeline, you will note the unveiling of the Apple iOS App Store in June of 2008. This opening coincides with the uptick noted in worldwide iPhone sales. While the iPhone was the killer device, the App Store acted as the catalyst to rapid growth.

The fact is the term “App Store” has become so synonymous with Apple (at least in their mind) that they issued a trademark infringement suit against Amazon over the use of the term “AppStore” – note the lack of a space between “App” and “Store”. While the judge denied Apple’s request that Amazon stop using the name in the short term until the issue can be settled in court, tune in this fall for the final act.

Without the iOS App Store, it is conceivable that the iPhone would still be a niche player in an already crowded smart phone market. Instead, The App Store changed everything. No longer are users required to visit external web sites to purchase applications, never without knowing if the site is legitimate or the software beneficial. Instead, applications obtained via the App Store are vetted by Apple to be safe and, in theory, functional.

Because the iOS App Store is tightly integrated into the everyday operations of the phone, the process of obtaining an application is significantly streamlined. A built in notification process helps to keep the installed applications updated with the latest versions available. With the App Store, users feel confident that downloading a new version of an application will not negatively affect the use of the phone or the applications and their historic data. While the iPhone is a brilliant device, the App Store was the significant enabling technology that helped spur the phone’s ultimate success. In late 2010, Apple extended the App Store to include Mac OS X applications. In other words, Apple has extended the application store concept to include their MacBook and MacBook Pro lines.

Needless to say, the concept of a device, be it a tablet, smartphone or laptop, with an associated application store is here to stay and will become a growing consideration for application developers and IT managers moving forward.

### Tablet Development Infrastructure Considerations

We have already demonstrated that tablet development entails some form of interactions with an application store. Therefore, it is reasonable to ask if there other considerations that will affect an organization’s staff or infrastructure? Unfortunately, not only is the answer to this question “yes”, but we have not even addressed all the issues surrounding the application store.

In this section, I will address other factors one should contemplate when entering the app development arena. These factors will have wide-ranging effects. Some will influence staffing decisions while others the architectural approach. In the latter case, the impact might manifest itself in the layout of software design or changes to backend infrastructure and interfaces.

### Platform

### Tablet Software Platforms

The area that has the potential for the largest impact on an organization’s development staff is the choice of platforms to support. Unless you have been living under a rock (or slate), it is safe to assume we are all aware of the two
platforms that have gained the most news headlines: Apple’s iPad with its underlying iOS operating system and the various Google-based Android tablets.

The choice of platform will significantly affect the required skillset of the development staff and will significantly influence the choice of platform. Development of Android applications requires expertise mainly with Java in a Linux environment (although Android also supports C/C++). It is fair to say that a great deal of Android development is accomplished using Eclipse and the Android SDK plugin.9

Objective-C, and the Cocoa Framework in the standard Xcode environment, is at the center of Apple iOS development. The iOS development toolkit is freely available from Apple10 though you have to register for the privilege11. While application development is free, there is a yearly subscription fee to place an app in the App Store. The paid subscription also provides access to the community forums. Do not confuse Objective-C with C++. Objective-C is C augmented with object oriented extensions and SmallTalk-style messaging.

Overall, Eclipse-based Android and Xcode-based iOS development skillsets do not overlap. Consequently, there is no magic button to convert an app developed for one platform to the other12. If you are writing an app for the consumer market to run on both the iPad and Android, anticipate more than a simple redeployment.

Developing web-based applications are more general. Both Android and iPad browsers support JavaScript, HTML5, and the WebKit framework. Although web based development is an option, unfortunately there are large classes of applications for which an Internet connection and remote (server-based) execution is not acceptable. For those use cases, only application development will suffice.

**Tablet Hardware Platforms**

Given the volatility in the tablet market, it is reasonable to be concerned about platform choice. In early 2011, HP’s WebOS powered TouchPad looked like a strong contender to take on the iPad. Palm Inc. originally developed WebOS to power the Pre and Pixi line of smart phones. For a time, some (including me) considered WebOS to be the technology leader in the device OS race. In early 2011, no one could fault you for choosing the WebOS powered TouchPad. By September of that year, the magnitude of your mistake would be evident as HP called it quits after 90 days of lackluster sales. Considering that Palm was once the darling of the smart phone industry, this was the second major setback for WebOS. With former Palm CEO Jon Rubinstein’s recent departure from HP, it is unlikely that WebOS will ever see the light of day again; at best, it may re-emerge as an embedded HP printer OS. His departure coupled with HP’s lack of participation even puts the open source WebOS project in jeopardy.

In light of the list of devices that litter the tablet highway, it would be foolish to assume that the iPad will remain the only significant player in the market. New, less expensive devices have recently appeared. Others will be joining the fray in early 2012. Many will embrace the “consumption only” mode and shatter the low-end price threshold. These include the new Kindle Fire, Nook, and the Kobo Vox. These devices, while not feature rich, have the potential to tap the consumers who like the iPad, but cannot afford it13. If you are planning to develop an application for the consumer market, these devices belong on your radar. Many of these new devices are Android powered.

*In light of the list of devices that litter the tablet highway, it would be foolish to assume that the iPad will remain the only significant player in the market.*
On the other end of the spectrum are the “feature rich” tablets/slates. Both the Lenovo ThinkPad Tablet and the Toshiba THRiVE challenge the iPad on features. They view the iPad as extremely portable, but too limited to outright replace a laptop for most IT professionals – a fair assessment. These new tablets are extremely portable laptops in a tablet form factor. Both tablets will benefit from the upcoming release of the newest Android OS, Ice Cream Sandwich.

There are other, currently lesser, players in the market. RIM has recently introduced their BlackBerry PlayBook. While too early to call the game, the PlayBook does not show signs that it will challenge the Apple iPad’s lock on the market any time soon, but the PlayBook does have several potential advantages. The first is RIM’s underlying highly secure network. The BlackBerry’s email encryption and server infrastructure is viewed as extremely secure. RIM’s encryption is thought so secure that several governments banned its use until anti-terrorism concerns were addressed and back door access was granted. The PlayBook may be able to leverage this existing technology to differentiate itself and gain a foothold.

Second, RIM has long and strong relationships in the corporate IT world. Migrating BlackBerry users to PlayBooks is a logical path. Finally, RIM already offers a large stable of corporate centric business applications. It is logical to expect that these packages are either automatically available on the PlayBook or, as is the case with their email client, will soon be ported.

MeeGo is another option. Like Android, MeeGo is a Linux-based open source OS project sponsored by the Linux Foundation (Android is sponsored by Google). MeeGo is not specifically designed for tablet applications. If you are looking to develop an enterprise wide application on a bare platform, MeeGo may be worth consideration. On the other hand, if you are targeting a wider consumer market, choosing MeeGo might be premature.

Not to be forgotten, Microsoft has been steadily working on perfecting its tablet/slate version of Windows. While the Microsoft tablet OS has been mostly eclipsed by iOS, Android and even RIM, they are far too large to give up without a fight. Though the window of opportunity (pardon the pun) is closing, they have the potential to re-enter the market and regain market share. Several manufacturers (ASUS, Lenovo, Samsung, and HP) have released Windows 7 versions of tablet computers. Early indications are that the Windows 8 tablet OS might have what it takes to not only challenge iOS but leapfrog Microsoft to the front of the pack.

Application Store (continued)

Returning to the App Store, it is more than a convenient mechanism for downloading applications. The App Store made the fat client cool again! In the late 1990’s there was a push to get away from thick clients on the PC. Back then, it was common to develop large applications that would have to be installed throughout an enterprise. Popular application development tools included Visual Basic and Power Builder. Thick clients tended to be feature rich. Because they were closely paired with the underlying hardware platform, they could create tailored, powerful widgets. Back then, long before Ruby or JavaScript, thin clients had limited functionality. Thick client developers (they never described themselves as FAT) derided browser apps as being “green screens with color”. Web developers shot back calling their counterparts “Fat Developers”. On the down side, their deployment images were large and, back then, the internet was not the ubiquitous download medium it is today. Network bottlenecks strained even dedicated intranets. Product releases on multiple floppies, CDs or DVDs were the norm – a product I once worked on spanned 22 DVDs! Despite the size and distribution issue, the killer of the thick client was library conflicts. Unanticipated library conflicts at install time were consuming larger and larger portions of corporate IT support budgets.

The combination of the application store and the underlying Internet support infrastructure has helped mitigate the fat client installation, update, and library issues of the past.

Whether you are developing an application for the Android or iPad markets, users can conveniently purchase and download your application from the application store via the Internet. Over time, it will even track and notify your install base when updates are available. Apple also supports applications geared for corporate IT departments. These applications are not for the public App Store. The Enterprise Application Development Store allows you to offer applications that are freely downloadable to targeted or licensed end users.

There are downsides to the App Store:

- First, the App Store is successful, in a large part, because they own the market. If the iPad were to drop from its current 90% market share down to a more realistic level, then the effort to write and support apps for iPad tablet devices will not seem as attractive. A three-way race between Apple, Android and Microsoft application stores means that IT staff must become acquainted with three separate development infrastructures.
Second, cost. Apple extracts a premium for applications placed in their store. Apple now collects 30% of revenue for applications sold via their App Store. While applications provided on the Enterprise App Store are freely downloadable, access to the Enterprise App Store itself is not. Even apps that are "free" for download via the App Store have a hidden cost to the developer, namely the iOS Developer SDK subscription cost. Over time, as realistic competitors enter the market, this might change. For example, because of Google’s aggressive advertisement pricing model, Apple is rumored to be changing their iAd policy. Just as Google’s ad policy impacts iAd, future competitors might cause Apple to change the pricing models for both the consumer and Enterprise App Stores.

Finally, even if your app is “free” for download, placing it in the App Store has a cost. Placing an application in Apple’s App Store is not a simple process. Every app must go through an approval process to "ensure that applications are reliable, perform as expected, and are free of explicit and offensive material." This review process can be lengthy and require changes and updates to an application. Depending on the visibility and complexity of your application, plane rides and face-to-face meetings with Apple’s technical staff may be required. Do not underestimate the time and cost of the process.

Browser vs. App

Not all applications targeted for the tablet must be made available via an application store. For the most part, the browsers on both the iPad and Android tablets support HTML5. Android browsers tend to be less restrictive, for example, they support flash player. Unlike the case with apps, Apple cannot control web-based content. Consequently, there are no required approval processes.

In addition, you are not limited to the default browser installed with either of the devices. Firefox, Opera, iSwifter, Mercury, Atomic, and Skyfire are available for Android and iOS – though not all are available for the iPad. To get around iPad restrictions, some of these browsers render iPad destined pages remotely and then ship graphical screens shots back to the tablet – similar to the way Kindle’s new Fire browser makes use of the Amazon’s AWS cloud to help render pages faster (a trend that may soon become very common).

In the browser-based solution, the browser must ultimately point to a URL which means establishing and maintaining some sort of web presence. With app development, web-based supporting servers are only required for applications that use web services. In addition, some applications, by their nature, require a close relationship to the underlying hardware and, consequently, a web solution is not practical (e.g. alarm clock).

The Network Pipe

Our final infrastructure consideration is the “pipe” leading to the tablet. Generally, tablets come in two flavors: Wi-Fi only or Wi-Fi and data plan (3G or 4G cellular support). With the exception of some Tablet PCs and/or Slate Computers, tablets generally do not have a wired network port. Consequently, any data you propose to push to your tablet application must fit through one or the other (or both) of these network pipes.

If your application is designed to run on Wi-Fi only, you must pay attention to this limitation. While the new N and P wireless routers have higher bandwidth, the distance from the hotspot can significantly reduce network bandwidth. And do not forget the number of G (still somewhat fast) and B routers out there. At the bottom of the pile, there are tablets Wi-Fi tethered to phones on 3G networks. The landscape is vast. You must ponder the pros and cons of the connection options and decide what will and what will not be supported.

Real-World Example

Let us consider the infrastructure impact given an example user story. For this discussion, we will assume the iPad as the delivery platform – though it would apply equally well to an Android device.

User Story: Video Push to iPad

As a Direct Broadcast Satellite (DBS) or Multi Systems Operator (MSO) provider, I wish to push live TV video streams to my subscribers for consumption on their iPad devices. In addition to streaming video programming to my subscribers, due to contractual obligations, I would like to ensure that these signals cannot be easily intercepted by non-subscribers.
This is an interesting use case. While many of the issues apply to any delivery platform, a tablet adds wrinkles and twists that must be uniquely addressed for their infrastructure.

**Browser vs. App for Video Push**

The first decision is whether to build an iPad-specific embedded application or a web-based application. Both an app and a web application will have the necessary controls to provide the video feeds to the device, so either approach is feasible. Consequently, the decision will depend on factors beyond technology, capabilities, and limitations.

It is noteworthy that the iPad does not support Adobe’s Flash Player and, by all accounts, never will; the details of which have been eloquently elaborated by the late Steve Jobs. Alternatively the Safari browser, included with all iPads, is fully HTML5 compliant. With the Safari approach, developers would likely work in HTML5, JavaScript and program to the WebKit API Framework. For app development, they would program in Objective-C using the Cocoa Framework. Depending on the capabilities of your development staff, one option may be more attractive than the other. In addition, Safari and Objective-C are not the only choices – just the most readily available.

The Safari approach implies that servers will be created to host the live TV application, including load balancing and failover support. Calculations will determine both the upstream and downstream request rates and message sizing. Further calculations are required to determine if the plant structure would support these values. If not, then a more streamlined app approach might be the better solution.

The application approach eliminates standing up and maintaining servers to support the live TV web application and the associated upstream bandwidth requirements. It also offloads processing to the tablet’s internal CPU and changes the user’s initiation and interaction with the application. By definition, as an application, the user downloads and installs the live TV package via Apple’s App Store. This app would appear on the tablet’s desktop as an icon – not a bookmark in the Safari browser. The app approach requires time in the schedule for Apple’s certification process - which can be lengthy.

Encryption and SIP (Session Initiation Protocol) support also are important. Objective-C supports more security options than JavaScript and HTML5. This alone could decide the app vs. web approach though more research would be prudent.

Because tablets are usually wireless, consideration must be given to transmission of the signal to the devices. Specifically, will the signal be the video stream provided over the Internet (as with Time Warner and Comcast) or will it be limited to an intranet (Cablevision and Dish Network)? While an Internet stream is friendlier to customers, it may place the “cable company” in breach of contract with the content owner. An intranet solution limits viewing to the home network and it is an effective first-line barrier to the hacking community; it also helps avoid content distribution lawsuits.

**Video Push Hardware Requirements**

Next, we must consider the hardware requirements within the DBS and MSO to support the numerous iPad devices owned by the subscriber base. The input side of the Content Delivery Network (CDN) has a series of N servers hosting j live TV feeds (the channels). (See Figure 4 - Channel to CDN to iPad Mappings.)
Since we are dealing with an existing DBS and MSO, the servers will already exist. Assuming High Definition (HD) TV, each station will generate around 3 Mbps per channel. Each channel server uniquely maps to a server within the CDN. The Y CDNs, in turn, service an expected M tablets in the subscriber base. Each tablet represents around a 3Mbps drain on the feed. As you can imagine, the feed numbers add up quickly. Figure 4 - Channel to CDN to iPad Mappings illustrates the situation.

The figure shows a many-to-one mapping between the channel servers and the CDN servers. The anticipated demands for output that will be placed on the CDN by the tablet community drive the mapping. To determine the precise number of CDN servers needed to meet demand, one must attack the problem from the outbound requirements and consider the maximum number of channels supportable by a CDN.

This is a variation on the classic Knapsack Problem. We have P items (the channels), each with a weight of P[w] and an unlimited number of servers to pack. The goal is to pack as few servers as possible with as many channels as possible without exceeding the server’s input service limits. In other words, pack as many channels into each CDN server, each channel having a weight, such that they can be served to the tablet community. The total number of tablets that connect...
to a particular CDN will represent the total weight for a given server. The ultimate goal is to meet the quality of service requirement with as few servers as possible.

Before we can do this calculation, we need to understand the expected bandwidth requirements between the channel and CDN servers, as well as the CDN servers and the tablets. Clearly, while the channel mappings between the Channel Servers (CS) and the CDNs will vary, the overall bandwidth of the output from the CDN Servers will equal the sum of the inputs to the CDN. Alternatively put, if I map Channel 2 to a CDN then I can only pull Channel 2 out. The advantage of the CDN approach is that, even if I have 500 tablets subscribing to Channel 2, the CDN only generates one output – not 1000 streams. Alternatively, if I map channels 2-20 to a CDN, I might only have viewership on a few channels but the CDN will still stream all 18 channels.

We are stating that the mapping of the CDN server array does not matter. The sum total of the output stream can only equal the sum total of the input stream (see equation 1 below).

\[ T_1 = \sum_{i=0}^{N-1} \sum_{j=0}^{Y-1} (CS[i, j]) = \sum_{k=0}^{Y-1} (CDN[k]) \]

Equation 1 created by author.

While not earth shattering, (1) shows that the network traffic \( T_1 \) generated by mapping \( N \) channels (which, in reality are physically hosted on \( X \) content servers) equals the output from all the CDN servers combined.

For the first part of the analysis, we concern ourselves with the bandwidth between the many CS servers and any one CDN (see equation 2).

\[ T_2 = \sum_{j=0}^{N-1} (CS[C, j]) \]

Equation 1a created by author.

In equation (2), the array of CDNs has been factored out and is now represented by the constant \( C \). The variable \( j \) represents the channels feeding the specific CDN server (recall it is the constant \( C \) now). Therefore, our first constraint is that the total input bandwidth \( T_2 \) for any one CDN server cannot exceed a prescribed threshold maximum value. This value will be determined by the combination of the network topology between the CS and CDN servers, as well as the limit set by the manufacturer of the CDN servers.

The goal of the Knapsack Problem is to completely fill a single backpack with objects of varying weights and sizes and get the maximum weight. The solution is recursive by nature.

In our example, the CDNs represent the backpacks. We have many backpacks \( Y \) to be exact, the packages are channels (of which there are \( N \)), and the weight of each individual channel is a function of the number of viewers (e.g. Nielsen Ratings).
There is a similar constraint on the output side (tablet facing) of the CDN array. The Knapsack Problem demonstrates that we cannot indiscriminately throw items into a bag and expect to get the maximum weight. We also cannot assign channels without regard to demand (viewership).

To do this, we must determine the approximate number of CDN servers required in the array. To derive this value, we first assign a weight for each of the channels – the weight for the Knapsack Problem. We derive these weights from the Nielsen ratings or from internal marketing statistics. Next, we must define a function that can transform viewership ratings into an equivalent server load. Finally, we can divide the sum of the transformed viewership by the manufacturer’s server ratings.

\[
Y = \sum_{i=0}^{N-1} \left[ f(Rv) \right] / Rs
\]

Equation 2 created by author.

Where:
- \( Rv \) is the viewership rating
- \( N \) is the number of channels
- \( Rs \) is the maximum load reading of the CDN server
- \( Y \) is the total number of CDN servers in the array

For reasons that will soon become apparent, we will not have a uniform distribution of channels to CDN servers. Consequently, this value of \( Y \) might not be accurate. To help visualize this, assume the following:
- There are 1000 channels (\( Y \))
- There are 50 CDN servers (\( N \))
- The viewership rating for ordinal Channel 0 is 1000
- The viewership rating for ordinal Channel 1 is 999
- The viewership rating for ordinal Channel 1 is 998
- ...
- The viewership rating for ordinal Channel 999 is 1

Next, map these 1000 channels to our CDN server array in ordinal order. That mapping would look as follows:

<table>
<thead>
<tr>
<th>CDN</th>
<th>Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-19</td>
</tr>
<tr>
<td>2</td>
<td>20-39</td>
</tr>
<tr>
<td>3</td>
<td>40-59</td>
</tr>
<tr>
<td>4</td>
<td>...</td>
</tr>
<tr>
<td>50</td>
<td>979-999</td>
</tr>
</tbody>
</table>

What should be apparent is that CDN 1 hosts the most viewed channels. While CDN server 50 is hosting the least viewed. This not only means that we have placed a large load on CDN 1 and almost no load on CDN 50, but we might have
violated CDN server 0’s maximum server rating (Rs). You may think that a
better approach would map the channels in round robin order. Unfortunately,
this approach does not help spread the load across all the servers and it still
fails to take into account the CDN server’s maximum capacity. For example, if
the viewership demand (i.e. sum total of currently mapped channels) placed on
a server consumes all of the server’s rated capacity, then no other channels can
be mapped to the server.

In reality, there are two separate and distinct requirements we need to meet:

1. Each CDN server should be ideally loaded to its maximum capacity
given the viewership weight given to the channels
2. Derive the idealized minimum number of CDN servers (Y) needed to
service the (N) channels

This new requirement is, in essence, a classic applied math minimum/
maximum equation. With these two requirements, we now can now derive the
optimal channel mappings, as well as determine the minimum number of CDN
servers required to support the load. To do this, we use something we can
describe as a holistic Backpack Algorithm with backtracking. This algorithm
entails:

1. Set the CDN array size to N-x, where x is a tuning parameter – initially
   set to 1
2. Sort the channels from highest to lowest viewership
3. Set the “current server” to the first server in the array
4. for all channels in the array:
   a. Is the server array zero length? If so, it is impossible to map the Y
      channels to the N server. Consequently:
      i. increment the size of the CDN array
      ii. remove the old mapping
      iii. restart the process from step 3
   b. Else, would mapping the “current channel” to the “current CDN”
      server cause it to violate either of the constraints?
      i. If not, map the “current channel” to the “current server”
      ii. Else, remove this server from the lineup and recursively
         invoke this algorithm (starting at step 4)

The result of the above algorithm will be an ideal distribution that spreads the
load effectively across a minimum of servers. Assuming no one channel by
itself exceeds a single server’s capacity, the worst-case outcome from this
algorithm is a CDN server array size of Y – meaning each channel has been
mapped to one dedicated server. This would only happen if each channel’s load
were greater than 50% of each server’s capacity.

There are other concerns, beyond the scope of the channel mappings that must
be addressed. Each video stream from a CDN to a tablet is a concrete 3 Mbps
stream. We must ensure that the overall network can support this volume of
data. Luckily, this calculation is straightforward. For every CDN server, we must
sum up the anticipated viewership feeds – then sum the results from every
server:
Where \( v \) is a constant that represents the 3 Mbps drain associated with each tablet, “ch” is the number of channels assigned to CDN[s].

Additionally, a video quality (QoS) feedback loop must be added from the tablet back to the plant. In addition to streaming High-Definition (HD) video, each CDN must also stream a lower definition (reduced bandwidth) stream. The application then transmits back statistics on the time it took for a packet (SIP or HTML5) to reach the tablet and, by inference, the packet drop rate. This interface could be composed of a publish/subscribe Catalyst interface (more on Catalyst interfaces later). If the drop rate were to exceed a given threshold then the application would have to switch to the lower quality stream. 

This additional lower definition stream must be factored into the previously detailed calculations – that assumes the 3 Mbps value was not high enough to begin with.

### Tablet Specific Limitations

A tablet specific limitation must also be put in place. This limitation would only affect the users in home access where bandwidth is limited; it would not apply in areas with public wifi access. Specifically: if you are limited to accessing the video stream from home then you are effectively limited by the network path to the intranet from within the home (the last mile issue). Consequently, the 3G and 4G access built into some tablets would not come in to play. This would dramatically affect the number of simultaneous tablets that can be supported within a residence. The ‘N’ router specification claims bandwidth speeds about 300 Mbps. Given the placement and distances of routers in the field, a more realistic rate might be 50 Mbps or lower. 50 Mbps may sound as if many tablets in use, but recall every other Wi-Fi device in your house (printers, Netflix, laptops, tablet browsers); each needs a chunk of that capacity. Quickly you realize that the number of concurrent tablets that can be actively engaged in video streaming is probably below 10.

Furthermore, the weakest link rule applies here. In many cases, the feed from the DBS/MSO to the residence (the last mile) is in the single or low double-digit Mbps range. This network bottleneck affects all network users, not just Wi-Fi devices. Consequently, due to multiple factors, a limit on the number of users viewing live TV within a single subscriber location is a required in order to provide a high quality of service to a few users rather than poor quality to all the network users in the house.

The chances of multiple simultaneous viewers on a home network are more likely that one might think. Think about a Super Bowl Sunday party where almost everyone shows up with a tablet. The introduction of “TV on the iPad” will create new social norms. While many may perch traditionally on a couch in front of the HD TV, others may mingle or cluster in smaller groups in other rooms, like the kitchen, hot tub, dining room or at the pool table in the basement; all the while keeping up with the action on their iPads. You might even want to provide a dedicated iPad on the wall of the bathroom – if for no other reason than the novelty of it. 1 envision the crowd in the “HD TV room” ebbing and flowing with the quarter, score and the overall action of the game.

---

**Equation 3 created by author.**

\[
Y = \sum_{i=0}^{N-1} \frac{f(Rv)}{Rs}
\]

This brings up both an interesting phenomenon and a fun parlor game too. In reality for quality of service and failover, several mirror CDN servers will likely stream the same channels. Unless there is additional hardware specifically put in place, they may not stream the video feeds in sync. If two tablets, tuned to the same channel, happen to connect to different CDN servers, they will not see the same output.

Instead, there will be a time or synchronization mismatch between the tablets – some users will seem to see and hear the action on their iPad seconds before the person sitting beside them – neither of whom will see any of it before the dedicated/ traditional TV. It causes an interesting group dynamic in a large setting. Try it with your friends the next time they come over to watch the Big Bang Theory.
Extending the example, a wireless TV in every room is conceivable. TVs that automatically turn on and automatically change channels as tagged viewers move throughout the house could soon enter the marketplace. DirecTV® already offers recording on a central DVR and watching on any wired TV in the house. Currently, moving from room to room is a manual switching process, but technologies on the horizon promise to automate even this. Unfortunately, the current infrastructure supported by the various DBS/MSOs places restrictions on the number of wireless TVs that can effectively be used at once. Eventually, this limitation will evaporate as DBS/MSOs switch from dedicated wired set top boxes to central boxes with built in hotspot endpoints.

Conclusion

It is reasonable to expect tablets to gain better traction in corporate America in the coming years. This will place a burden on IT management and development staff to enter into this new technology landscape. Compounding the issue will be the likely entry of new players with varying hardware and software (OS) technologies and limitations.

Delving into the tablet development space is not as simple as it might first appear. Given their limitations, there are infrastructure issues to take into consideration. These issues include the capabilities of the technical staff, infrastructure configuration and delivery limitations, and the subtleties of dealing with various tablet platform vendors and stores. As we have shown, calculating server capacities and configurations required for adequately performing tablet apps is not trivial. When these issues are addressed, a new world of access and convenience will open for IT support and developers alike.

References

1 I will resist the temptation to call a “smart” smartphone a PDA (Personal Digital Assistant). The history of the PDA’s is equally blurry and predates smart phones. The first successful PDA was the Palm Pilot (circa 1997). Before that, there was the Apple Newton. The history of convergence between the cell phone and the PDA into the “smart phone” could be an article itself. Equally intricate is the differentiation between a PDA and a tablet – made difficult by a cell phone-PDA convergence device like the Handspring Treo. Suffice to say, the PDA was the nail struck slightly off-center while the tablet was dead-on.


3 While at the time, this author was employed by Sun Microsystems, Inc. (SMI) he was not connected with the Foleo project.


6 Technically speaking iTunes, which opened in 2001, is also a form of an App Store.


12 Utilities are purported to exist to allow Android apps to run in iOS and vice versa but I have no firsthand experience and cannot attest to their worthiness.

Thick vs thin, fat vs. thin, “feature rich”? Regardless if you call it “fat”, “thick”, “thin”, etc., we mean a measure of how much of the code resides on the device and how much resides is on the server. The more of the application code on the client (e.g. laptop, desktop, tablet, etc.) the thicker the client. Eventually, the code and libraries combine to reach a subjective “user defined” tipping point, and the application becomes “fat” and, well, that ain’t Phat! Yet, tablet applications are reminiscent of “fat clients” and, in this new incarnation, fat is Phat! Go figure.

While I have used both “greens screens with color” and “fat clients” many times over the years, I cannot recall who coined these terms. The early Internet/ARPAnet days are but a blur.

As new versions of devices (iPads, Android, and others) come to market and older devices remain in play, the old “library” issues of the past may come back in strange new ways... and you heard it here first! For now, things are good. Go ahead and press that “update” button – what could go wrong?


Sip HTML? What, why would I do that? Yet another puzzle! SIP is an IETF protocol for controlling communications sessions that has been around for over a decade. It powers most digital/PBX phones today. It also powers Voice-over-IP communications, like Skype. Competing protocols have been “backed in” to HTML5. This fight has hardly begun. It is conceivable that SIP will go the way of WebOS and be relegated to PBX land, but that is difficult to predict with any certainty.


If I have thousands of tablets subscribing to a CDN, there might be downstream network and switching bottlenecks, but that is not a CDN issue.

To conserve space, the entire algorithm has not been presented. There are additional checks and steps that have not been included.

You have probably experienced this in other online streaming applications when, for example, your YouTube video might periodically become a bit pixelated. Eventually, either through rerouting or simply less demand placed on the network segment, the HD stream is eventually restored.

For more information reference the various Bluetooth, Lower Energy and Near Field Communications web sites and is the subject of a separate paper by this author.
Collaborating Through Shared Displays and Interacting Devices

by Dr. Steven Greenspan, Research Staff Member, CA Labs, CA Technologies

Information devices are getting both smaller and larger as we move beyond the traditional desktop. On the one hand, the lighter, smaller and more portable smart phones and tablets are replacing devices such as desktop computers and laptops (see Figure 1). On the other hand, desktop displays have increased in size or have multiplied on a single desktop, and shared displays have grown to cover entire walls or to form rooms that users can enter. This trend has continued for more than a decade but has accelerated with the introduction of the iPhone, iPad, dense flash memory, and inexpensive large screen monitors.

The next phase of computing beyond the desktop monitor will explore how these new devices interact with one another to support collaborative data collection, analytics, sense-making, and performance.

There are five primary display modalities:

- Ambient (or glance-able) displays with physical sensors (such as a thermostat that changes color whenever there is a power shortage)
- Handheld displays that can easily slip into a pocket or purse
- Personal document displays (tablets, ultrathin laptops) that are mobile and good for reading and editing
- Large desktop monitors used as personal workspaces for simultaneously viewing information from several sources or perspectives
- Very large shared displays that support data immersion or shared viewing; these range from the size of large televisions to the size of walls and may be horizontal (tabletop monitors) or vertical

For each of these, there are debates among users over form and function. For example, within the personal document display modality, some people prefer the editing power of a laptop keyboard and others prefer the interactivity and portability of the tablet.
A sixth display modality, the smart lens, has been promised for over a decade, and is now emerging. A recent NY Times article (Bliton, 2012) suggests that Google Goggles will provide augmented reality by superimposing visual data onto real-world environments. (Will Apple release iGlasses?) Some argue that augmented reality displays will eventually mimic and replace all other displays.

As suggested in the preceding list, different physical formats provide different benefits and encourage different forms of interaction. Over the past decade, we have seen a steady movement away from desktop computer to mobile devices, and from solo tasks to collaborative teamwork. In addition to continuing the integration of social media and crowdsourcing into work practices, the next phase of enterprise computing will extend the use of personal devices to interact with shared displays and shared data. The goal will be to support collaborative sensemaking of big data and rapid responses to dynamic situations. Indeed, the growing availability of big data, and the need for collaboration in real-time analytics, necessitates this evolution.

These trends will impact data centers, network operating centers, and command and control centers. Their future is in delivering highly-optimized, automated services, on a massive scale. Nonetheless, things will inevitably go wrong and the unexpected will happen. Human judgment will be needed, and network and security professionals will be expected to monitor, analyze, and quickly act upon an ever-increasing volume of real-time and historic data. While some of these professionals may be collocated, it is expected that many teams will collaborate across time zones and political boundaries. The technological and psychological challenge is to develop abstraction, visualization, collaboration, policy and measurement techniques that encourage teams to perform optimally when confronted with a myriad of real-time, complex data sources.

Very large and very small display screens each create ergonomic and perceptual challenges. These include ensuring readability and consistently-perceived colors across multiple visual angles that might be experienced in a large room, and reducing the strain of holding tablets at ergonomically, suboptimal distances that allow for both readability and multi-touch interactions.

These perceptual and ergonomic challenges are reasonably well understood. However, the cognitive problems encountered with information overload, situation awareness, and collaboration in an information-rich environment are poorly understood and are topics of intense research.

One of key problems in collaboration is work coordination. When talking to one another, participants must attend to the same aspects of a problem (sharing common ground) and, when working in parallel, they need to minimize redundancy. As an example, team members may need to know that one person is monitoring a sub-network looking for security intrusions, so that others on the team can decide to act upon other types of problems in that sub-network or to look in another sub-network. Likewise, managers need to know how individuals within a team and across teams are distributing and prioritizing the workload.

As they conduct their work, managers and staff will have available to them different information devices ranging in size from very small mobile devices to very large wall-sized monitors. Each type of device suits different collaboration styles. Personal devices (handhelds, tablets and laptops) enable remote fieldwork, mobility within a large workspace, and individual data explorations. Very large wall and tabletop displays allow multiple teams to scan immense networks, crowds of people, multiple graphs of data, etc. In contrast, tabletop displays encourage greater interactivity within the team and with the screen.
itself. Recent work on collaborative tabletop displays explores how gestures, such as tapping the screen and pointing, can be conveyed to remote users (Genest & Gutwin, 2012).

However, these large surfaces are not as good for personal, in-depth explorations of data that include reading text. Increasingly, personal information devices will be used to interact with data on large screens. For example, users could position a tablet vertically on a tabletop map to view more detailed information about events at that section of the map. Ambient displays can also be developed to interact with tabletop monitors. Moving differently shaped objects to different regions of a very large display could create an array of alarms in which the objects change color when certain events happen at their locations; tapping the same colored tab on a tablet could provide more detailed information.

Thus, ambient, handheld and tablet displays will, in concert, offer new techniques for manipulating and navigating shared data. Onsite and remote staff can work together, each carrying a personal tablet, in the following ways:

- Moving information on and off the big screen
- Privately analyzing data on their tablet, drilling down (or up) for more (or less) detail relative to the shared view
- Quickly ascertaining where others are looking and what they are doing, without having to navigate to another screen or send a query to that person
- Arranging physical or virtual ambient displays to signal important events

Combining devices in this way will help address the goals defined by Douglas et al (2007): alerting teams to changes in their environment, displaying information about these changes in the optimal format to support “situation awareness, decision making, and collaboration,” and allowing individuals the personal space to creatively analyze the situation.

As information gets easier to share, questions naturally arise concerning the efficacy of sharing information across personal monitors or on very large displays. Several empirical studies provide a direction for answering this question.

In practice, large shared monitors in control centers tend to be composed of “repeater” displays (Douglas, 2007) that duplicate information from several personal desktops. As might be expected, these tend not to be used and are treated as “eye candy”. There are multiple factors that lead to this underutilization:

- They duplicate information that is readily available on the personal displays
- If the shared display is an unfiltered version of a user’s display, that user’s control icons and cursor are distracting and can obscure data
- The colors and display format that are appropriate for the personal display are often not appropriate for the larger screen

If users customize the look and feel of their displays, the problems are compounded.

Even if colors and formats are standardized, and cursors and widgets are hidden, introducing new technologies into a well-performing center can degrade team and system performance. For instance, chat applications in military settings have been found to degrade overall team performance (Cummings, 2004). Similarly, experimental evidence (Bolstad & Endsley, 1999) suggests that providing a large screen that simply duplicates the displays available to each
user increases mental workload and degrades performance in high workload conditions.

However, in high workload conditions, sharing data through large common displays can improve productivity and coordination if the shared information is carefully crafted. Presenting the relevant subset of data that is required for coordination, as opposed to all the data present on each user’s screen, improves team performance and coordination (Bolstad & Endsley, 2000).

In addition to relevancy, format also matters. A series of studies indicate that multiple-attribute displays are better on small displays, but spatially-grouped, individual displays (for each attribute) work better on very large displays (Yost & North, 2006). Thus, large screens should be used for monitoring and exploring situations that would otherwise clutter personal displays, provide a common ground to support collective situation awareness, and/or permit comparison of different levels of data representation or different sets of data.

We are just beginning to learn how to best coordinate information across different display modalities and how different devices can interoperate to facilitate collaboration, shared situation awareness, and problem-solving. Current research directions and technology trajectories suggest that the command center of the future will increasingly incorporate remote and collocated teams using multiple types of devices that communicate with one another to automatically modify their respective displays.

At CA Labs, with university partners such as the HotSoft Research Group at Carleton University, we are exploring new technologies for visualizing big data, automatically merging and transforming data to appropriate formats as a function of display modality, tasks, and user identities and roles. We are examining how attention can be guided within a team by other team members and by software (as in the case of alerts), and how multiple devices can be used in combination to synergistically enhance situation awareness and responsiveness.

References


“The greatest good you can do for another is not just share your riches, but to reveal to him his own.”
Benjamin Disraeli (1804-1881)

Background

Wilson MacDonald is an inventor and a lead member of the CA Technologies Patent Review Board. He is a named inventor of three issued patents and has several filed patents. His commitment to innovation does not stop there; he is also a mentor for inventors and customers. He believes the five hallmarks of a great inventor are:

- Curiosity
- Knowing you are wrong
- Being able to take an idea and develop it while understanding it’s not perfect
- Working it to the point where it is useful
- Remembering the benefit you are trying to provide

Inventor

Wilson’s quest to be an inventor started young. He was mechanically curious as a child and was driven to disassemble time pieces, recorders, or other items found around the house. Number four of five brothers, Wilson sometimes bore the consequences of not reassembling the objects that often did not belong to him. But the same mechanical abilities provided support with work as a bicycle mechanic through high school, a Bachelor of Science in speech communications from Emerson College, and a Masters of Business Administration from the University of Kansas.

Combining mechanical skill with a keen visual sensibility, Wilson worked for several years in drafting. The visual aspect of his creativity remains strong with him today, and Wilson uses this ability to illustrate solutions to business and technical problems. His skill in using graphics to clarify solutions was integral to the successful filing of Wilson’s first patent, Displaying Resource Performance and Utilization Information. At a time when organizations were spending more on larger pipes and increased bandwidth, it became important for them to understand resource utilization.

Wilson developed a new way to display existing information that ultimately saved his customers considerable expense. Using customer data sets previously displayed in bar charts, he developed a way to convert those bar graphs into pie charts and assigned unique colors to different areas of the infrastructure. Using this display, it was possible to determine large ranges of the infrastructure’s resource usage. Wilson’s customers determined that less than 10% of the infrastructure was used 95% of the time. By viewing and analyzing existing information in a different way, they found that they did not need to purchase more pipes, but just needed to redistribute and balance the load, thus saving them a great deal of money.

About the author

Karen Sleeth is Senior Principal Program Manager for CA Technologies Patent Program as well as CA Press, the retail publishing imprint for CA Technologies.

She launched CA Press and an internal author award program when she joined CA Technologies in 2007.

Karen became the manager of the Patent Program in 2010. Since that time the internal Patent Review Board has doubled in size, which has helped to shorten the time of the review process. The company’s increased focus on innovation related to invention and thought leadership initiatives has brought more exposure to the importance of patents.

Karen enjoys working with inventors and especially helping identify patentable ideas through internal invention harvesting sessions.

Karen is a member of CA Technologies Innovation Team and is PMP certified. Karen is based in North Carolina and graduated from UNC-Chapel Hill.
Mentor

Wilson has always found it personally satisfying to take a complicated issue and break it down to be simple and clear. He is passionate about focusing on the real idea by understanding the problem. He believes that problems are the “food” of the creative process. But that is no surprise to the Johnson O’Connor Research Foundation1, whose aptitude testing group rated him high off the charts in ideation.

With customers who would like to simply implement a solution, he patiently listens, but enjoys the journey they take together to fully understand the problem and ensure the solution agreed upon is the best one possible.

When talking about the benefits of iteration in the Agile development process, Wilson stated, “It goes back to working with customers to give them what the need, which may be quite different than what they think they want. To some degree the Agile process takes that into account, and makes the process easier.”

A modest man, Wilson doesn’t consider himself as technical as the other CA Technologies Patent Review Board members, and he says he “understands enough and can confine [his] role to the mechanics.”

Other members of the Review Board can always depend upon Wilson to act as an encouraging mentor with budding inventors, first pointing out the positive elements in their ideas, then the areas for improvement, and ending with suggestions for the next steps. Mentoring is a natural fit with his character and, taking Disraeli’s quotation to heart, Wilson excels at revealing to inventors their own riches.

1 Johnson O’Connor Research Foundation. http://www.jocrf.org/

Wilson MacDonald
Vice President,
Business Unit Operations,
CA Technologies

<table>
<thead>
<tr>
<th>Issue Date</th>
<th>Title</th>
<th>Inventor(s)</th>
<th>Patent Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>August 2, 2011</td>
<td>Server Side Application Integration Framework</td>
<td>Dushan Fernando, Anthony Candappa, Koustudh Pewar, Nilin Maheshwari</td>
<td>7,992,132</td>
</tr>
<tr>
<td>August 2, 2011</td>
<td>System and Method for Managing Zone Integrity in a Storage Area Network</td>
<td>Subrahmenya Yellapragada</td>
<td>7,991,809</td>
</tr>
<tr>
<td>August 2, 2011</td>
<td>Permanent Pool Memory Management Method and System</td>
<td>David Jordan</td>
<td>7,991,976</td>
</tr>
<tr>
<td>August 23, 2011</td>
<td>Method and System for Computer Security</td>
<td>David Huddleston</td>
<td>8,006,301</td>
</tr>
<tr>
<td>August 23, 2011</td>
<td>Performance Monitoring of Network Applications</td>
<td>Brian Zuzga, John Bley, Mark Addelman, Kretes Ng</td>
<td>8,005,943</td>
</tr>
<tr>
<td>September 6, 2011</td>
<td>Network Fault Manager for Maintaining Alarm Conditions</td>
<td>Lawerence Stable, Mark Sylor, Thomas LaRosa, Bradley Carey, Dave Russo, Kevin Jackson, Albert Briner, Jeremiah Small, Matthew Bocceley</td>
<td>8,015,456</td>
</tr>
<tr>
<td>September 6, 2011</td>
<td>Two-Factor Anti-Phishing Authentication Systems and Methods</td>
<td>Geoffrey Hird</td>
<td>8,015,598</td>
</tr>
<tr>
<td>September 13, 2011</td>
<td>System and Method for Optimizing Database Queues</td>
<td>Edward Kosciuszko, Sreekumar Mencn, Hung-Vuong Vo., John Vincent, Thomas Waring, Joyce Lau</td>
<td>8,019,750</td>
</tr>
<tr>
<td>September 13, 2011</td>
<td>Systems and Methods for Tracking Screen Updates</td>
<td>Mark Godwin</td>
<td>8,019,003</td>
</tr>
<tr>
<td>Issue Date</td>
<td>Title</td>
<td>Inventor(s)</td>
<td>Patent Number</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------</td>
<td>-------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>September 13, 2011</td>
<td>Logical Entity Fault Isolation in Network Systems Management</td>
<td>Timothy Pirozzi, Chris Burke</td>
<td>8,020,033</td>
</tr>
<tr>
<td>September 13, 2011</td>
<td>System and Method for Intelligent Service Assurance in Network Management</td>
<td>Karl Forster</td>
<td>8,019,730</td>
</tr>
<tr>
<td>September 20, 2011</td>
<td>System and Method for Installing Software</td>
<td>David Luft, Sarad Thapa, Vnematsh Granham</td>
<td>8,024,430</td>
</tr>
<tr>
<td>September 27, 2011</td>
<td>Heterogeneous Job Dashboard</td>
<td>An Ly, Arun Padmanabhan, Edward Chen</td>
<td>8,028,265</td>
</tr>
<tr>
<td>September 27, 2011</td>
<td>System and Method for Reactive and Deliberative Service Level Management (SLM)</td>
<td>Lundy Lewis</td>
<td>8,028,066</td>
</tr>
<tr>
<td>October 4, 2011</td>
<td>Programmatic Root Cause Analysis for Application Performance Management</td>
<td>Jyoti Bansal</td>
<td>8,032,867</td>
</tr>
<tr>
<td>October 4, 2011</td>
<td>Method and Apparatus for a Comprehensive Network Management System</td>
<td>Lundy Lewis</td>
<td>8,032,630</td>
</tr>
<tr>
<td>October 11, 2011</td>
<td>Method and System for Managing Information Technology Data</td>
<td>Mark Barrancocha</td>
<td>8,037,106</td>
</tr>
<tr>
<td>October 16, 2011</td>
<td>Intrusion Detection Based on Amount of Network Traffic</td>
<td>Paul Gassoway</td>
<td>8,042,180</td>
</tr>
<tr>
<td>November 1, 2011</td>
<td>Synthetic Transactions Based on System History and Load</td>
<td>Jon Ruiz, Ling Thio, Brian Zuzic</td>
<td>8,051,163</td>
</tr>
<tr>
<td>November 15, 2011</td>
<td>Systems and Methods for Excluding User Specified Applications</td>
<td>Paul Gassoway</td>
<td>8,060,867</td>
</tr>
<tr>
<td>November 15, 2011</td>
<td>System and Method for indicating/Confirming Special Symbols to be interpreted Literally</td>
<td>Wayne Barbarek</td>
<td>8,059,126</td>
</tr>
<tr>
<td>November 15, 2011</td>
<td>System and Method for Live Software Object Interaction</td>
<td>John Michaelsen</td>
<td>8,060,864</td>
</tr>
<tr>
<td>November 22, 2011</td>
<td>Directory Searching Methods and Systems</td>
<td>Rick Harvey</td>
<td>8,065,338</td>
</tr>
<tr>
<td>November 22, 2011</td>
<td>System and Method for Defining and Presenting a Composite Web Page</td>
<td>Kaihu Chen, George Hong Zhi Hsu</td>
<td>8,065,520</td>
</tr>
<tr>
<td>November 29, 2011</td>
<td>Topology Static Zones</td>
<td>Subrahmanya Sarma V, Yellapragada Narasimha C, Reddy Sankar Natarajan</td>
<td>8,069,229</td>
</tr>
<tr>
<td>November 29, 2011</td>
<td>System and Method for Delivering Results of a Search Query in an Information Management System</td>
<td>Edward Blaziejewski, Thomas Angle</td>
<td>8,069,147</td>
</tr>
<tr>
<td>December 6, 2011</td>
<td>System and Method for Interfacing an Application to a Distributed Transaction Coordinator</td>
<td>David Thole</td>
<td>8,074,220</td>
</tr>
<tr>
<td>December 6, 2011</td>
<td>System and Method for Optimizing Storage Infrastructure Performance</td>
<td>Gregory LaMonica</td>
<td>8,073,880</td>
</tr>
<tr>
<td>December 6, 2011</td>
<td>Method and Apparatus for Service Level Management (SLM)</td>
<td>Lundy Lewis</td>
<td>8,073,721</td>
</tr>
<tr>
<td>December 13, 2011</td>
<td>Evaluation of Current Capacity Levels of Resources in a Distributed Computing System</td>
<td>Sanjay Radia, Robert Gingell, Jarry Jackson</td>
<td>8,078,684</td>
</tr>
<tr>
<td>Issue Date</td>
<td>Title</td>
<td>Inventor(s)</td>
<td>Patent Number</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>December 13, 2011</td>
<td>Automated Grouping of Messages Provided to an Application Using String Similarity Analysis</td>
<td>Jyoti Bansal, David Seidman, Mark Addleman</td>
<td>6,178,619</td>
</tr>
<tr>
<td>January 3, 2012</td>
<td>System and Method for Variable Block Logging with Log-Ahead Buffers</td>
<td>Joseph Lynn</td>
<td>6,090,691</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>System and Method for Self-Supporting Applications</td>
<td>Jacob Lamm, Marvin Waschke, Yaakov Shapiro</td>
<td>6,098,809</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>Relational Model for Management Information in Network Devices</td>
<td>Jochen van de Groenendaal, Amitava Chakraborty</td>
<td>6,099,425</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>Using Mobile Agents to Perform Operations of an Information Life-Cycle Management System</td>
<td>Pavan Korlepara</td>
<td>6,099,464</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>Methods and System for Finding Scrolled Regions Within a Tile Cache</td>
<td>Mark Godwin, Vivian Lloyd</td>
<td>6,099,578</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>System and Method for Generating a Unified Service Model</td>
<td>Greg Goodman, Donald LeClair, Maurice Donegan</td>
<td>6,099,439</td>
</tr>
<tr>
<td>January 17, 2012</td>
<td>Data Storage System</td>
<td>Peter Malcolm</td>
<td>6,099,394</td>
</tr>
<tr>
<td>January 24, 2012</td>
<td>Managing Virtual Machines Based on Business Priority</td>
<td>Kouroso Esfahany, Michael Charamonte, Karthik Narayanan</td>
<td>6,104,033</td>
</tr>
<tr>
<td>January 24, 2012</td>
<td>Method and System for Obtaining Digital Signatures</td>
<td>Charles Spitz</td>
<td>6,103,667</td>
</tr>
<tr>
<td>January 24, 2012</td>
<td>System and Method of Specifying Customized User Interface for Rules in Rulebase</td>
<td>Ji-Beng Tan</td>
<td>6,103,610</td>
</tr>
<tr>
<td>January 24, 2012</td>
<td>Adding Functionality to Existing Code at Exit</td>
<td>Jeffrey Cobb</td>
<td>6,104,023</td>
</tr>
<tr>
<td>January 31, 2012</td>
<td>System for Compression and Storage of Data</td>
<td>Carl Johnson, John Casey, Joe Cravy, Timothy Bruce, William Abernathy</td>
<td>6,108,442</td>
</tr>
<tr>
<td>January 31, 2012</td>
<td>System for Generating Optimized Computer Data Field Conversion Routines</td>
<td>Donald Boiten, Kevin Pintar</td>
<td>6,108,528</td>
</tr>
<tr>
<td>January 31, 2012</td>
<td>System and Method to Restore Computer Files</td>
<td>Grant Wang</td>
<td>6,108,357</td>
</tr>
<tr>
<td>February 7, 2012</td>
<td>Method and Apparatus for Supporting Multiple Versions of a Web Services Protocol</td>
<td>Chris Betts, Tony Rogers</td>
<td>6,112,472</td>
</tr>
</tbody>
</table>