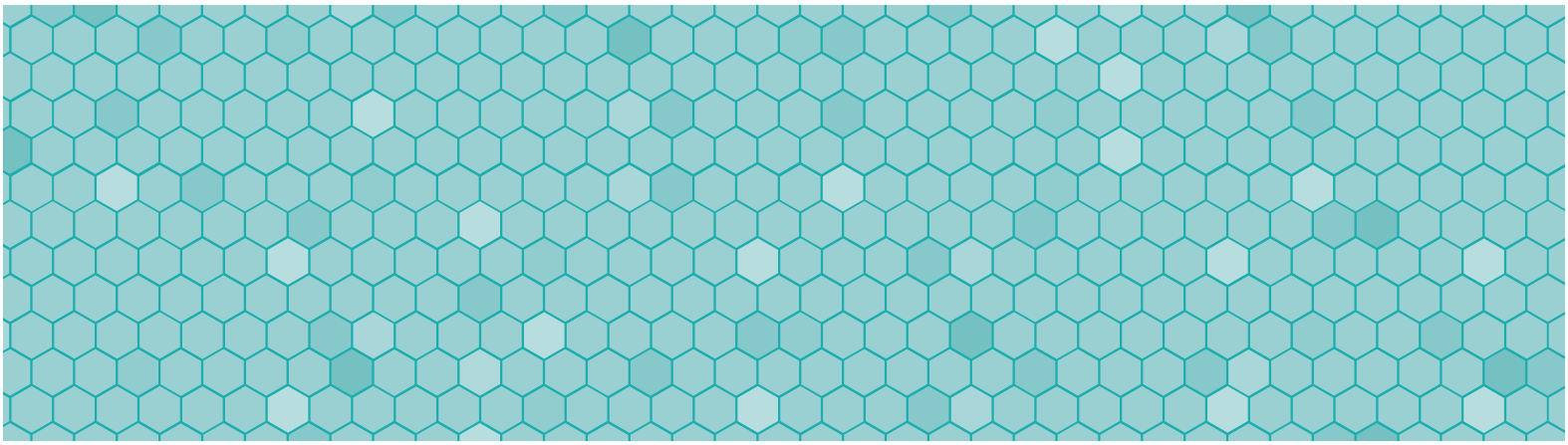


The Cloud as a Tectonic Shift in IT

The move to the cloud represents one of the largest paradigm shifts to ever affect the IT landscape. More than just a simple technology evolution, the cloud fundamentally changes many of the cornerstones on which IT was built. From redefining the concepts of operating systems and middleware, to revolutionizing the way IT services are built and consumed, the cloud is ushering in an era of change unlike any we have ever seen.

In this white paper, CloudBees provides an overall perspective on the huge disruption happening across our industry today, examines the effect it is having on traditional IT concepts and reviews the new IT service and consumption models that have emerged as a result. Following that, some predictions are made about where the tectonic shift brought upon by the emergence of the cloud will lead us in the future.

CloudBees

The Java™ Paas Company

Table of Contents

Learn More – Video Resource

Sacha Labourey, CEO, presents:
The Cloud as a Tectonic Shift in IT
http://www.cloudbees.com/why-platform-service.cb#SachaCloud

The Industrialization of IT

History shows that human being tend to be pretty smart at predicting the predictable. Any time linear evolution happens, it tends to be pretty easy to look at where things are going, at what pace, and guess where it is going to land at some point. Take the cost of hard drives as an example (see graph below). Looking at the last 30 years, it seems like it has never been very hard to guess what price a hard drive would be 3, 5 or even 10 years from now. The same could be said about Moore's law on CPUs. All of that is good and very predictable.

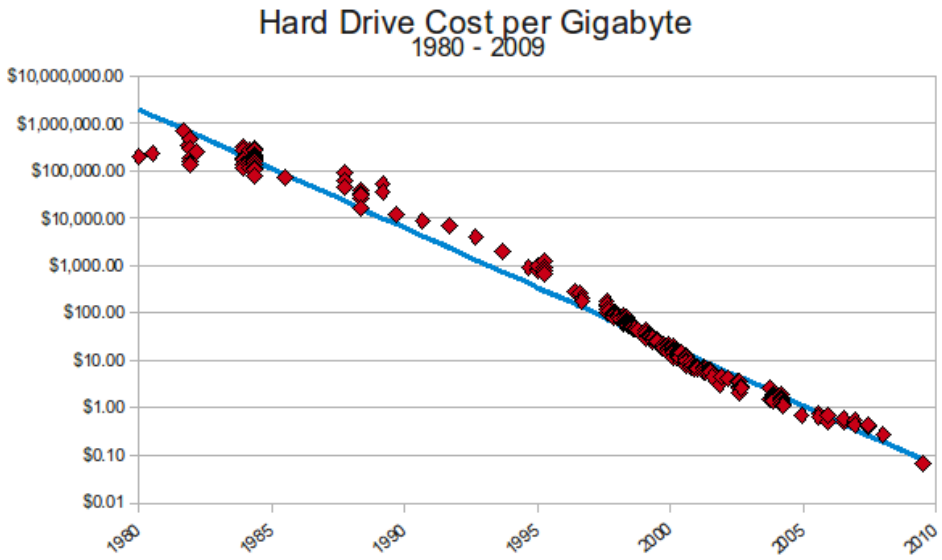


Figure 1 – Cost of hard drives per gigabyte
(Source: http://www.mkomo.com/cost-per-gigabyte)

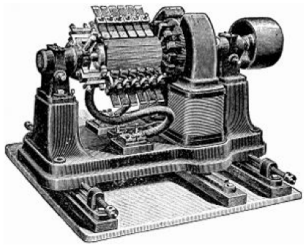
On the other hand, when paradigm shifts happen – a real shift, not just mere evolution within a given paradigm – we do not have the same ability to predict the future. This is because paradigm shifts can follow multiple option paths, some of which are incompatible with what we are doing today. And because these options can lead to entirely new behaviors, it is difficult to predict where things will eventually land. The other consequence is that once you've experienced the shift, it becomes hard to remember or figure how things were being done in the past.

A perfect example of a paradigm shift leading to unforeseen behaviors is the phone. The emergence of phones was so significant because people no longer needed to travel in order to communicate. But what's even more interesting is the emergence of mobile phones. When mobile phones appeared, traditional wired phones were already a commodity available in most houses – so it was a paradigm shift for a device that was already quite widespread. It took less than a decade for most people in the U.S. to have a mobile phone and use it to dictate some of their daily behavior. Today,

mobile phones are such an inherent part of our lives that we can barely remember how life was without them. Think about it for a moment: Has a kid ever asked you why phones used to have a cord in the past?

Since understanding the impact of a paradigm shift is not an easy endeavor, let's look at a very powerful analogy to help us understand just how significant one can be.

A Story Backwards



The first electrical generators appeared at the end of the 19th century. They were big, complex, expensive and fragile. As such, they were only accessible to a few highly profitable companies.

Yet, very quickly, cities and countries started understanding the economic potential electricity could bring in terms of competitiveness. In 1878, the Exposition Universelle in Paris catalyzed the acceleration of this process. We should remember this fair as one where some of the inventions that most deeply impacted the 20th century were demonstrated: Alexander Graham Bell's telephone, Thomas Edison's phonograph and megaphone, as well as the completed head of the Statue of Liberty, which was ready to be shipped to New York City. Yet, some of the most impressive demonstrations were related to the use of something new: **electricity**.

In the years that followed, many cities started investing in power plants: St. Moritz in 1879, London and New York in 1882 and Grenoble in 1883. These cities understood that for their inhabitants and factories to ever afford electricity, it had to be a "community" investment. So the earliest power plants were the pride of the communities in which they were built. They were the *Googleplex* of the 19th century!

Exhibit 2 - Power plant built in 1906 in La Chaux-de-Fonds, now protected by UNESCO
(image source: http://en.wikipedia.org/wiki/File:UsineElectriqueLaChauxDeFondsCH1.jpg)

Yet, those efforts were by far not enough to create an environment where electricity could be consumed as a "utility." To start with, there were no standards. As an example, the picture below of Paris from around 1913 shows a single city with different networks and very different capabilities and connectivity: mono-phased, bi-phased, five wires, three wires, etc.

Exhibit 3 – Paris in 1913, segregated by type of electrical network
(Source: http://seaus.free.fr/spip.php?article601)

So as a consumer, what kind of freedom did you really have? Your equipment had to be customized for a specific zone and if the plant in your zone required maintenance you would essentially be without power. This showed the limits of community-scale and proprietary networks.

During the 20th century, massive standardization and network consolidation occurred – from the plug (size, shape, voltage, current, frequency, etc.) to the distribution grids and the producers. A lot of work was done. Fast-forward to the end of the 20th century, and consumers could buy any equipment or device anywhere, plug it in and use it.

Also, given the importance of electricity in both our lives and our economy, governments often played a strong role in the marketplace. Even though most power plants and distribution networks are owned by private companies, governments ensure they followed strict rules because no developed economy can advance without a hyper-reliable electricity production and distribution chain.

Where does that leave us today? Electricity providers are organized into grids in order to provide a highly available stream of energy. Distributors negotiate and dispatch this electricity to consumers, who then have access to a fully standardized commodity. Done.

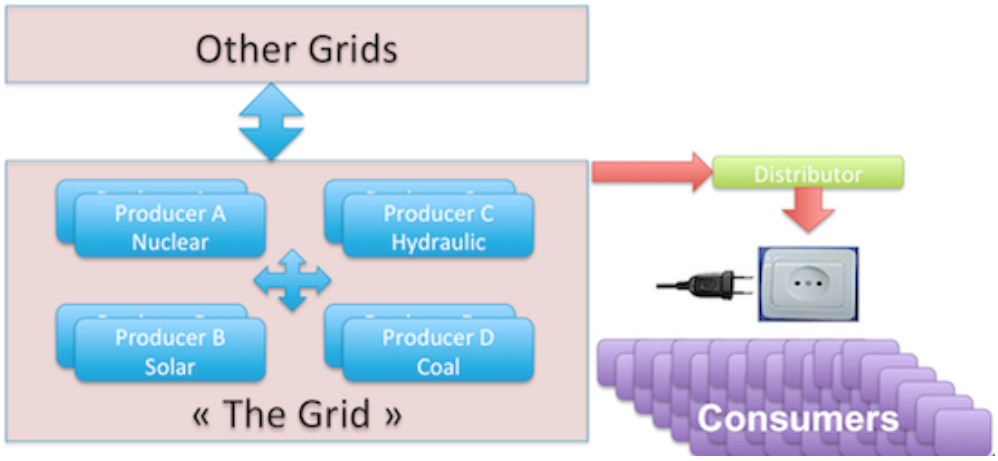


Exhibit 4 - The electrical grid, today – standards everywhere

The State of IT Today

The evolution of electricity over the last century has distinct, direct parallels with the current state of IT. While these are different beasts in many aspects, numerous similarities remain. They are both significantly important in our lives and as a fabric of our economy. They are sophisticated technologies that are hard for non-specialists to manage. And they are "virtual goods," in the sense that what they produce is not like traditional consumer goods.

Yet, when we look at the state of IT today, it's clear that we are at a comparable level that electricity production and distribution was a century ago.

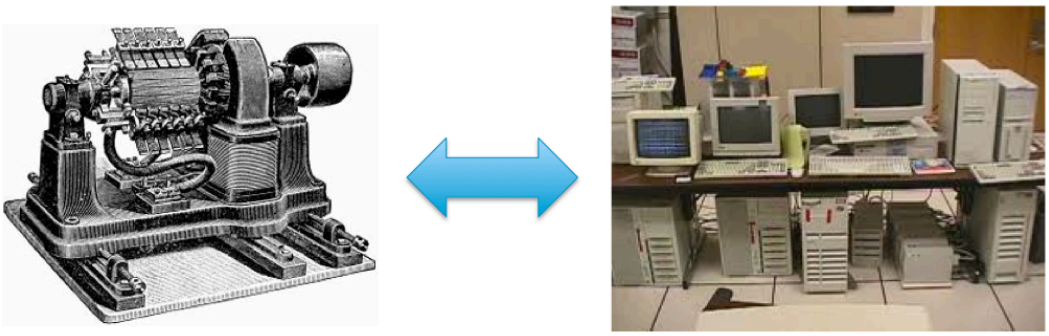


Exhibit 5 – On the left, the first electrical generator. On the right, Google's first data center

Even when a company has clearly identified business objectives and knows what service it would like to implement, the path between this idea and its implementation seems like the map of Paris in 1913. There are so many options to consider

– everything from hardware, operating systems (OS) and backup, to network and firewall configurations – that the business objectives may get lost in the shuffle.

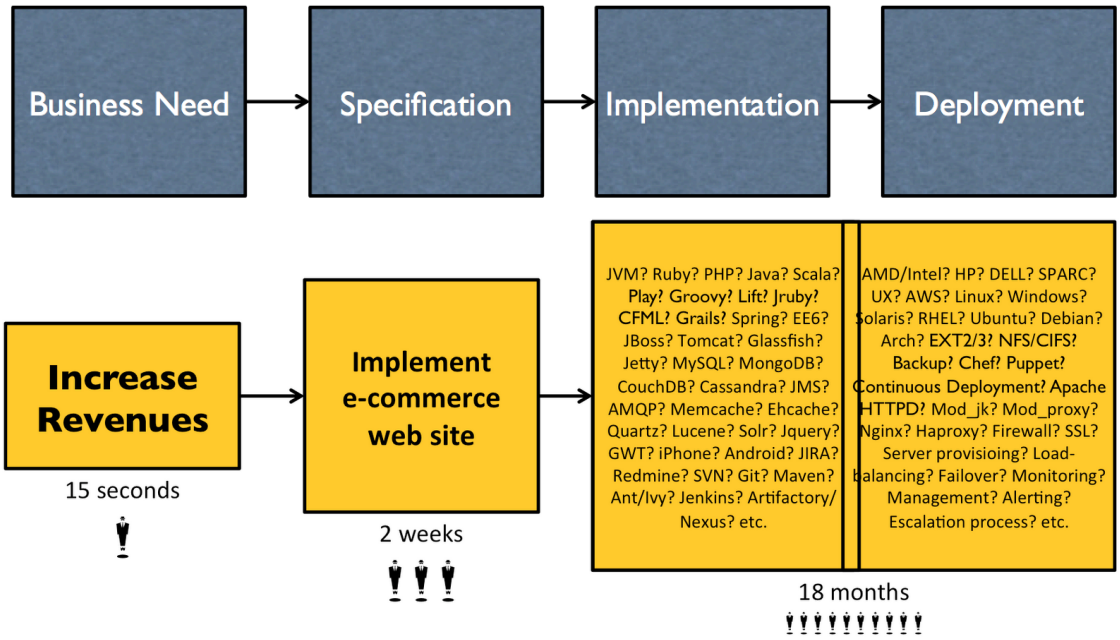


Exhibit 6 – Typical timeline of a business-critical IT system

Today, many computing resources are custom built. Companies set up their infrastructures in unique ways; every application requires a different setup and environment and the overhead is high. It's almost as if each company built its own power plant, based on the best available architecture and a well-defined maximum amount of computing and storage resources.

Thanks to the Internet, only the bandwidth component is a relatively mature IT layer: Companies understand that building their own WAN made no sense, so they have begun outsourcing their communication needs to a set of interconnected bandwidth vendors and do not really know – nor care – through which tube their bits are travelling, so long as service-level agreements (SLAs) are met. This relative maturity doesn't apply to either the compute or storage/data layers.

Is there anything we can learn from the evolution of the production, distribution and consumption of electricity in the 20th century and what's taking place with cloud computing?

From Selling Books to Initiating the IT Revolution

While many had dreamed about it, Amazon did it. In 2006, the company announced Amazon Web Services (AWS), and initiated the phenomenon that we now group within the "cloud computing" basket.

As any online vendor has learned – often the hard way – every tenth-of-a-second increase in response time equals a fair percentage of lost customers (source: http://www.useit.com/papers/responsetime.html). To mitigate that risk, companies typically own enough resources to accommodate peak levels of demand. But since a number of online businesses have a strong seasonality, a considerable portion of the infrastructure ends up sitting idle the rest of the time. Amazon's genius was to make those extra resources available through an API, as an on-demand service available via a pay-per-use model. And guess what? It worked great.

In just a few months, AWS was a great success. The graph below shows that about a year after launching, the bandwidth consumed by AWS customers had already eclipsed Amazon's own consumption.

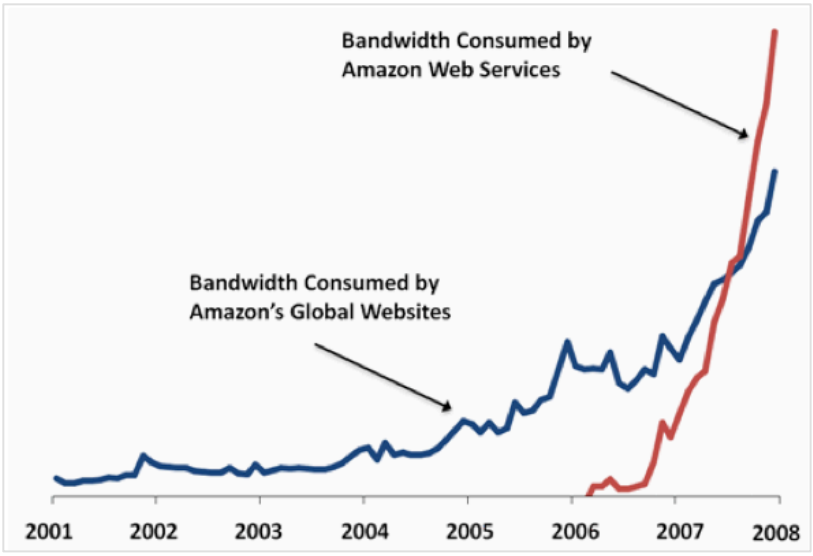


Exhibit 7 – Bandwidth consumed by Amazon's own websites, compared with bandwidth consumed by Amazon Web Services
(Source: Amazon Web Services blog, http://aws.typepad.com/aws/2008/05/lots-of-bits.html)

Very quickly, engineers realized that the arduous process of provisioning new servers they had been used to could now happen in just a few clicks with AWS! This opened new doors. Now, they were able to use compute resources like a disposable razor blade: take one, use it, toss it. Start an application, add resources to handle a peak in traffic and decommission it immediately after the fact.

A similar process in the enterprise can take weeks and more frequently, months. And once you obtain a resource, you typically get it for five years.

But, engineers also realized that not everything was customizable – they had to accept some level of standardization. Is this tradeoff acceptable? Is a more streamlined IT environment acceptable when resources are almost instantaneously available? The more customized you make it, the less automated it can be, the harder it is to reach a critical mass and the more you move away from a utility model. Think about it, it is a bit like asking for an 80-volt electrical plug just for a specific toaster. Objectively, providing an 80-volt source might end up being more efficient for a specific case, but that type of custom requirement is exactly what we've tried to eradicate. Today is all about standardization and simplicity, not constant customization.

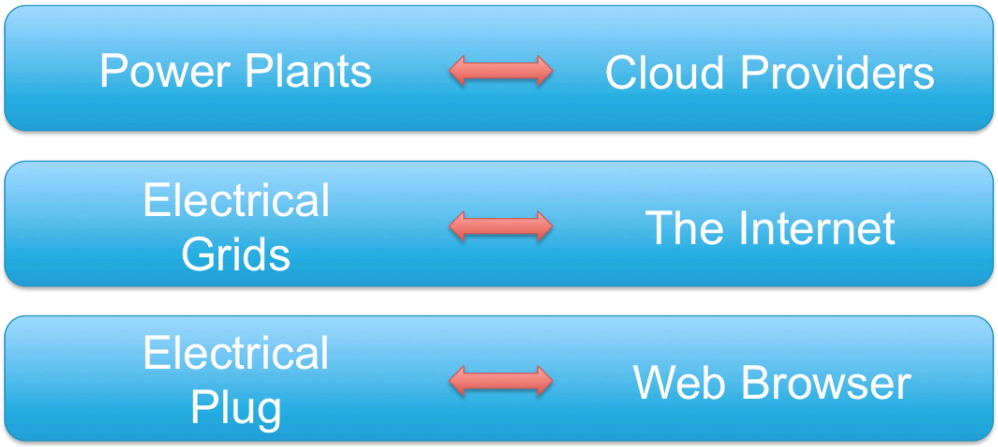


Exhibit 8 – Mapping electrical delivery to IT/cloud services delivery

It's a safe bet that cloud computing and IT in general will evolve in a very similar fashion to the production, distribution and consumption of electricity: Cloud providers are the new power plants, the Internet unifies clouds into a highly available grid and distribution and consumption is standardized around specific browsers (HTML5, CSS3, JavaScript, etc.). IT will move away from à la carte systems, on-premise data centers and customized client-side technologies. The critical mass on each of these layers will make any competing/proprietary technologies decidedly non-competitive.

Much like the transition of wired phones to mobile phones, cloud computing is not a mere evolution, but a true paradigm shift. Its consequences are still hard to precisely foresee, but it will without a doubt impact the current IT landscape – especially when it comes to how we develop software and identify leading IT vendors.

The Irrelevance of IaaS

Because they are not a straightforward evolution, true paradigm shifts make it hard to foresee what's to come. Yet, when facing such drastic change, the natural response is to try to replicate what we know and have done to date in the new environment.

One of IT's occupations is to build software stacks, which often include virtualization technologies, OS, middleware, databases and more. The stack is capable of evolving over time and integrating new software versions and patches with as little disruption as possible. While there is a lot of science to it, there is also quite a bit of art in there.

Consequently, when discovering the cloud, IT initially tried mimicking what they had been doing for decades: Pick a server, install an OS and some additional software and configure, manage, patch and monitor it. But when it comes to the cloud, the server was not a recognizable, brand name server running in a private data center. Instead, it is a "no-name," standardized virtual server running in Amazon's worldwide data center infrastructure. So if the brand doesn't matter at all, why not just cut and paste our last 30 years of blueprints and apply them to the cloud? With no CAPEX investment required and very little time needed to provision new servers, it's a change for the better. However, this scenario fails to capitalize on all the cloud has to offer. Here, it is merely an evolution of the traditional on-premise model, not a real paradigm shift.

Initially applied to servers, under the name Infrastructure as a Service (IaaS), the founding attributes of the cloud – elastic, on-demand, pay-as-you-go capabilities – can actually be applied to other layers of the stack. Two of those layers are Platform as a Service (PaaS) and Software as a Service (SaaS). Let's examine how IaaS, PaaS and SaaS all relate to each other.

IaaS vs. PaaS vs. SaaS

IaaS sits at one extreme of the cloud continuum. It provides a way for users to consume the basic building blocks of IT – the compute, network and storage layers – as a service. This is probably the most flexible layer of all, but also the most complex. It is flexible because access to a bare-bones server allows you to install whatever you want on that machine, including a specific OS, a specific version of that OS, low-level drivers and more. With IaaS, you are really working with systems – not just applications or data, but a full-fledged stack of software.

But in doing so, you end-up performing a lot of low-level IT tasks. Worse yet, executing these in the cloud is typically harder than doing it on-premise. Cloud environments are implemented based on the assumption that they might come and go. To take advantage of elasticity, new servers must be spawned automatically, requiring that all of the typical IT tasks be highly automated – not just the setup of new instances, but also the synchronization of those resources with each other.

For example, using the underlying IaaS-API, the provisioning logic must be able to dynamically update load-balancers whenever clusters get modified, discover and collect under-used resources and so on. And while you are not in charge of the hardware maintenance, you are still very much in charge of patching, upgrades and other software maintenance. The typical audience for IaaS consists of system operation and DevOps engineers who are replicating the on-premise art of IT in the cloud. If you are interested in migrating an existing system – and not just a specific application – from an on-premise server to the cloud, you will want to use IaaS.

At the other extreme of the continuum sits SaaS. Whenever a business need is generic enough, chances are high that you'll find a company providing the solution as a ready-to-use service. Typical examples include CRM, ERP, support portals and collaboration tools. If you can find a SaaS deployment that fits your requirements, you will realize that they typically offer a huge productivity boost. They can often be customized to fit specific requirements, but if you have other requirements that are not covered by the solution, you may be stuck.

It is important to note that a growing number of SaaS solutions have started offering a set of APIs that will automate specific tasks through an external script instead of, for example, having to rely purely on a human clicking on a GUI. This capability becomes very important as the number of SaaS solutions being consumed grows, as companies will need these APIs to synchronize and/or integrate some areas of their SaaS solutions.

Yet, those APIs will typically not be able to change the behavior of the solution per se, but only impact it at its periphery. Consequently, SaaS solutions typically deliver great productivity gains as long as they offer what you need. The typical audience for SaaS can be any business end user but you'll also find a number of solutions, such as e-mail gateways and source code analysis, aimed at more technically minded end users, like developers.

In between IaaS and SaaS sits PaaS. If you are interested in developing and deploying custom applications, then PaaS is for you. You can see PaaS as "the middleware of the cloud." It provides an abstraction layer over low-level IT elements, including servers, storage and networking, and enables software developers to work with such concepts as "applications," "databases," "source-code building" and "application testing." With PaaS, you don't have to worry about setting up servers, firewalls, build farms, load-balancers or databases. You'll only have to focus on your business needs and what your application needs to do; the PaaS vendor provides the rest.

This is a very important distinction that might not seem obvious at first: IaaS focuses on hosting **full system stacks**, while PaaS focuses on **applications**. The PaaS provider delivers a set of pre-defined, state-of-the-art application runtimes following the best practices in that space, which developers use to deploy their applications. PaaS users should never have to care about what OS, load-balancer and configuration is being used, or whether or not it should be upgraded. Those are the concerns of the PaaS provider. And while it might be more efficient to request an 80-volt plug for a toaster – or a customized runtime environment for just one specific application – this is what kills the ability for a provider to offer a high-quality service at a highly competitive price.

The typical audience for PaaS consists of software developers and architects.

So, Which One Should I Use?

The big question that many businesses face as they move to the cloud revolves around which layer they should be using.

Most companies are already using some sort of SaaS solution, such as Google Mail, Salesforce.com or Zendesk. But, whenever the need for "custom" work arises, they'll typically adopt an IaaS solution. The reason is simple: building

software stacks is what they are doing today on-premise. This is really the evolutionary approach that helps a company change, while remaining in a comfort zone. But they quickly realize that while the first steps are trivial, building a fully automated, scalable and resilient infrastructure is more complex than doing it on-premise – and requires different, quite specialized technical talents.

If a company is interested in running existing legacy systems, then this might be the right solution. Whenever you want to customize a system at a relatively low level, you need to have full access to the server – as well as specific drivers and a specific version of the file system or database. But remember that in this case, the IaaS provider will only help you by maintaining the hardware equipment and environment – you won't receive any help detecting issues with your stack, or patching and upgrading it. All of these activities remain in your hands.

Should you really move existing systems "as-is" to the cloud? This is not always the best choice. A better course of action is to leave the existing legacy systems and stacks that require a specific configuration on-premise. So, the next time you have a new business problem to solve, ask yourself a very simple question: Can I find a SaaS solution that fits my needs? If the answer is "yes," then take it – you'll get the biggest productivity gain by using an already implemented, already tested, fully maintained offering.

But if the answer is "no," it probably means you must implement some custom service. In that case, you will want to use PaaS, not IaaS. As a company, your added value lies in your ability to create new services that help you differentiate from the competition – not managing servers, firewalls and load-balancers.

The bottom line is that 5 to 10 years from now, existing systems will be de-provisioned and replaced by SaaS solutions, and new custom applications will be developed and deployed on PaaS. IaaS will be comparatively too complex to use and require sophisticated – and hard to recruit – talents.

Will IaaS Disappear?

Much like power plants are not likely to disappear, IaaS is here to stay. However, PaaS and SaaS vendors will emerge as the dominant visible species – and a number of SaaS solutions will actually be delivered using PaaS.
PaaS and SaaS vendors will be the ones caring about the infrastructure layers, as well as servers, load-balancing, backup and maintenance. You won't.

Does this mean that IT consumers should not care about what IaaS framework their PaaS/SaaS solution is based on? Not at all. But they should care for different reasons.

First of all, customers will want to make sure some applications are co-located for latency reasons and they will want to make sure that others are hosted in a reputable country, typically for legal reasons.

Customers will also want to know what uptime SLA the IaaS solution is held to, as they will not directly manage this relationship nor use these resources. Instead, they will use a PaaS vendor as a proxy to address these underlying concerns.

Last but not least, all IaaS solutions are not created equal. Some offer services that go well beyond the mere renting of computing and storage resources – such as offering a Content Delivery Network (CDN) or providing access to sophisticated database engines. When running on such an IaaS architecture, the PaaS/SaaS provider could resell those additional services, provide the end customer with an abstract interface to access them in a vendor-neutral fashion, or, more simply, let the customer use them. In the latter case, the customer will want to make sure it is running on that particular IaaS, despite delegating systems and software stacks to a PaaS/SaaS provider.

As we've seen, "the cloud" is actually composed of several distinct layers, each providing value in a very different fashion to certain audiences. In particular, IaaS, through vendors like Amazon, catalyzes the adoption of the cloud and is getting an enormous amount of attention. But as cloud layers mature and enterprises become more educated about them, focus will shift away from IaaS and move towards PaaS and SaaS, the more directly actionable layers. Therefore, unless you are a PaaS vendor yourself, chances are you won't directly care about IaaS; you'll just move right to PaaS and SaaS solutions.

The Death of Operating Systems (as We Know Them)

When adding new services to their IT portfolio, most companies will go through the same thought process: First, they'll look for a SaaS solution that fits their needs. If they find one, they'll use it. Otherwise, they'll build a custom application using PaaS.

As you can see from that simple process, both decisions are made without any regard for the low-level software stack. When you are using SaaS, you only care about the services offered. When you are using PaaS, you care about the application runtime and services offered to you as a developer. These services include persistence, security, transactions, elasticity, failover and messaging.

By design, the underlying layers – hardware, OS, virtualization, networking, etc. – are handled and abstracted away by the PaaS or SaaS provider, making them totally invisible to the end user. So in the future, IT end users won't have to care about the server-side OS. The PaaS and SaaS providers will.

Now, let's dig one level deeper and look at how those PaaS and SaaS providers will behave.

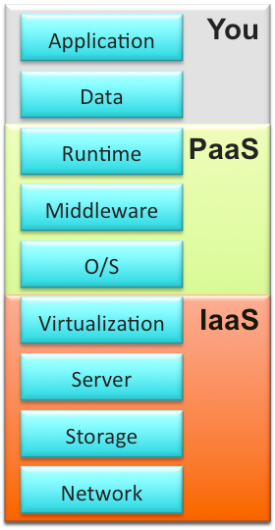


Exhibit 9 – The software & infrastructure stack

What OS will PaaS Use?

When it comes to what makes the ideal OS, the majority of PaaS providers want something that is highly flexible, customizable, easy to upgrade and maintain and supported by an active community/ecosystem. At a high level, the feature set should be laser-focused, providing a hyper-robust OS that can deliver lightweight OS-level virtualization, fine-grained security and highly controllable resource allocation – with the lowest possible overhead. Ideally, this should result in a limited number of very small, well-defined OS images. These are frequently referred to as "Just Enough Operating System," or "JeOS," and are typically as small as 60 MB in size.

Several specialized Linux distributions match those constraints, which makes Linux the prime choice for many providers.

What OS will PaaS NOT Use?

Many may be wondering why PaaS providers don't routinely use Windows. While windows has a very active development community and vibrant ecosystem, and is pretty good at finely allocating resources among different tenants, it loses out as the OS for PaaS for two reasons.

First, Windows is not known to be the best OS when it comes to creating JeOS images and customizing, scripting and spreading them around. The second reason is simply cost! Why would a PaaS provider, using only a very small subset of an OS, spend a considerable amount of money per core for a full-fledged OS, when that value is not even going to be offered to the developer or end user? Windows is simply a no-go for the cloud and this will obviously have a long-term impact on Microsoft, since Windows is the main hub of their current ecosystem.

Now, does this mean Microsoft will be the only company impacted by this change? Not at all, even Linux vendors such as Red Hat will be impacted by the shift. Initially, as companies try using the cloud via the IaaS layer, Red Hat will do very well, as many organizations will choose Red Hat Enterprise Linux over Windows.

To understand the move away from today's type of Linux distributions, it is important to realize that the unique value of these organizations lies in their ability and expertise in testing their distribution with a huge number of back-end devices and systems, and certifying an ecosystem of thousands of independent software vendors (ISVs). In PaaS and SaaS scenarios, the number of back-end devices being tested declines – AWS, for example, exposes the same hardware on its millions of servers – and the ISVs, having adopted SaaS, aren't directly exposed to the OS anymore. In other words, the OS vendors are the ones orchestrating the ISV ecosystems. As ISVs morph into SaaS vendors, PaaS deployments – not the OS – become the new hub for that ecosystem, not the OS anymore. Consequently, as PaaS and SaaS offerings take the lead, cloud providers will use their own customized version of Linux – one that is either self-supported or managed by IaaS providers at a much lower cost.

What About the Server-Side OS?

When it comes to servers, the OS as we know it today will slowly fade away and disappear. In their place, we will see the emergence of self-supported, IaaS-supported and mini-Linux OS organizations focused on providing JeOS at a much lower

cost. The time of the fully-featured, heavy OS has passed. Also, it is important to realize that this is not just a technically focused discussion: The move away from an OS-centric IT ecosystem will have important consequences for a number of well-established companies in that space.

What About the Client OS?

OS requirements will heavily change on the client side as well. Thanks to the rise of HTML5, CSS3, JavaScript and browsers will become the new unified interface. The underlying OS services required to run a browser-based environment will be very basic. As such, there will no longer be a need for the traditional fat-client OS. Obviously, HTML5 does not yet offer the complete feature set required to fully compete with native applications, so we will have to wait. But by the time, say HTML6 or 7 is released, there will likely be a device-agnostic client "universal plug" strong enough to defeat native/proprietary environments.

Unlike for the server-side OS though, the reason for this evolution won't be so much an attempt to reduce the feature set offered – client devices will grow richer, if anything – but to converge towards the concept of the universal plug.

And as enterprises increase their usage of the cloud and move toward PaaS and SaaS solutions with an HTML5 frontend, the concept of the OS will evolve toward a much more focused alternative offering a fraction of the OS services available today. This change won't just be a technical one, but will come with a complete reshuffling of how ecosystems are organized, and around which vendors they are centered.

The Death of Middleware

As the move to the cloud accelerates, companies will slowly stop caring about the underlying infrastructure and OS. But they will care about the type of services and SLA that will be delivered to them – just not how those services will be implemented. But is the OS the upper-most layer that will be impacted in this death-by-abstraction?

When Salesforce.com started their "no software" campaign, the idea was to make people pause and think. Obviously software was not dead: Salesforce has thousands of engineers producing software on a daily basis. But, does it matter? After all, the company's value lies in the service it delivers, not the software it creates.

Think about it another way: When you are booking a trip through a travel agent, do you care whether their back-end system is running on Windows or Linux? Whether they are using pencils or pens to do their work? No, you care about the value you are getting from the agent. The same concept applies to Salesforce.com. Since moving to a service approach has relegated software to an "implementation detail," what impact will PaaS have on traditional middleware?

What is Middleware Today?

Historically, a successful middleware implementation had to win the heart of two very different audiences, each with very unique conditions to satisfy.

On one hand, middleware has to please the developers that would be interacting with it on a daily basis. Developers want technology that will make their application cooler and faster. They care about ease of use, productivity and being able to prototype things fast. They have a mission – implementing a solution – and middleware has to aid in that quest, hiding all possible lower-level complexity and providing shortcuts for their most frequent tasks.

On the other hand, middleware also has to please IT operations teams. These individuals demand that software is rock solid, scalable, resilient and easy to monitor, manage and patch. IT operations teams know that the value of an application is lost if it goes down at 2 a.m. They are being paid to ensure the company is able to operate at all times, under any circumstances, and have a backup plan in case things go wrong. IT processes are designed to cover all possible cases, with no surprises, not just 80% of what could happen. In this way, IT operations has a lot to do with resilience and inertia.

Yet, in real IT life, a lot of activities require these two teams to work together. Do you need to create a new application? Provision a new application server? Perform load testing? Get a new database instance? Cluster your application? IT life in any company is a constant, high-friction interaction between development and IT operations teams. Nothing is ever quite as easy as it should be and even the most basic request can lead to time- and energy-consuming processes.

Producing new applications and new services results in friction between these two teams because each has totally different DNA, with the only common ground being their love for technology and their desire to see their company be successful. And middleware, as its name implies, naively sits in the middle.

Middleware in the Cloud Era

How does this change when it comes to the cloud? Let's first start by looking at the IT-operations side of the house. As we've seen previously, PaaS is all about providing a service that will allow developers to create and run applications without having to directly care about the infrastructural and operational aspects.

IT has a completely different role in this setup. IT will not be involved in building the infrastructure required to operate the application and they are not going to be monitoring, maintaining and patching the stack. Plus, they will not need to worry about delivering top-notch Java clustering, or setting up firewalls, databases and load-balancers, to name a few. The PaaS vendor will. As such, PaaS becomes the new engine developers will use to get their environments setup. It will be responsible for provisioning, managing, monitoring, patching and evolving that environment.

And don't forget the "S" in "PaaS." PaaS is more than just a set of well-orchestrated software processes. PaaS providers will act as the new support organization developers approach when things don't happen as expected. And even though 99% of support requests have to do with a bug in the application, most developers still need guidance from the platform-support organization to understand the problem.

The bottom line is that from an IT operations standpoint, PaaS acts as a major friction-killer. Infrastructural and operational aspects are now owned by the PaaS provider and are delivered to the developer as higher-level concepts – such as "create

an application" or "scale an application horizontally" vs. "give me an application server" – on a set of standardized notions. Obviously, this helps to significantly reduce friction. Consequently, from an IT operations standpoint, middleware becomes pretty-much non-existent – simply because they do not have to worry about it anymore!

When people hear this, they quickly interpret this as the death of IT. But that's not quite right. The growth of the cloud actually means new challenges are ahead: IT will certainly want to be involved in choosing a PaaS provider, reviewing the SLA and determining on which IaaS solution it can operate. And as they define what solution should be used across their PaaS and SaaS vendors, IT will want to maintain the map of what company data is made available to what application and audience, and what new security guidelines should be added as a result of moving to the cloud. IT will very much remain in charge of protecting the crown jewels of the company – they will now be delegating most infrastructural and operational aspects to specialized cloud providers.

PaaS also radically changes the picture from a development standpoint. Typically, middleware solutions focused on a specific runtime behavior, such as running an application within an application server, running processes or working with rule engines. Some of these providers go as far as delivering a set of tools – or plug-ins – that make the development of solutions for that runtime easier.

This is all well and good from a vendor's perspective, but is this truly satisfying from a developer's standpoint? Developers are using a myriad of runtimes that have to properly integrate with each other. And software has to be developed, tested and validated. Middleware vendors typically never help with those steps. So development teams have to resort to specialized tool vendors for solutions as diverse as continuous integration, static code analysis, code repositories, bug trackers and binary artifact repositories.

To do their jobs properly, development teams typically need to have a relationship with at least a dozen vendors to get the point solutions that are unified to provide the complete workbench they need to achieve their goals. However, throwing together dozens of tools from different vendors still doesn't build a usable workbench. Those tools have to be integrated as a proper whole, versions need to be compatible, custom code has to be written to facilitate integration when it is not provided out of the box and a common identity solution needs to be applied across the board.

To further complicate matters, as each individual tool evolves, patches and new versions for each must be applied without breaking the chain. Doing this properly requires time and diligence: Any service interruption can severely impact the development team's productivity. But who is responsible for building and maintaining that complete development-to-production environment? Some of the most sophisticated IT shops have dedicated teams responsible for that job. But the average – and even above-average – IT team doesn't get anywhere close to that level of sophistication and organization. In most companies, the IT operations team will even refuse to touch that side of the process. Development tools must be handled by developers; IT operations provides only the raw hardware and focuses instead on production-related issues.

The bottom line is that development teams will spend a significant portion of their time and resources building, maintaining and extending their environment, all of which comes at the expense of building software and creating value for the company.

But when using PaaS, the situation is very different. Obviously, middleware is still very much present. Developers still need to handle transactions, security, persistence and asynchronous communication. Thankfully, PaaS solutions offer all of this and so much more.

First, a PaaS vendor will offer a complete, ready-to-use environment to code, build and test your software. Need to create a new project? No problem: In one click you end up with a complete, pre-integrated, fully managed development toolset. All you need to do is add the co-workers of your choice to the project and you are ready to go and collaborate. This way, you can store code and binary artifacts and perform builds and tests in a continuous fashion, without ever having to care about servers, storage or backup – or talking to IT operations. You'll have more time to focus on your job.

Should you need additional tools, just go on the vendor's marketplace, enable that other SaaS and you'll instantaneously get access to a new resource that is properly integrated with the rest of your development toolset. There is no need for your team to spend any time on non-productive activities or ask IT operations for access to the resources you need.

Consequently, while it is very tempting to initially pitch PaaS as the "middleware of the cloud," this thought process is akin to reducing a jet to an engine and a couple of wings. While the features provided by middleware are still part of the features exposed by PaaS, these solutions will provide much broader functional and lifecycle coverage, handle all infrastructural and operational aspects and integrate and deliver these best-of-breed capabilities as a unified, fully managed service. Furthermore, PaaS vendors also act as the support organization for any issues you might encounter during software development, helping you avoid the need to decide where to route your requests. Just ask your PaaS provider – it covers all of that.

The bottom line is that for a developer using a PaaS, the notion of "middleware" is pretty much as visible and interesting as the more technologically intensive aspects of your platform.

While this might seem like a semantic change at first, it is actually a fundamental shift in how developers work and where they are spending their time. And as with any paradigm shift, you must experience it in order to fully understand the implications.

As IT moves to a service-oriented world, much of the friction we have been accustomed to in our daily activities will disappear. We are entering a new world of efficiency – one that eliminates the need to worry about middleware and intensifies the focus on creating value.

Conclusion: The Cloud is IT's Most Important Tectonic Shift

The cloud represents one of the most significant shifts that computing has gone through. As we move towards the cloud, we will discover a new service-based world, where many words that were once common in the average IT shop – like servers, data centers, OS, middleware and clustering – will get erased.

Much like Google and Facebook were not familiar companies a mere decade ago, the IT landscape is due to radically change in the next five years. New entrants with a cloud pure-play will be much more agile in creating value and won't face the internal competition that arises during a paradigm shift, protecting their existing revenues while providing a credible, cost-efficient alternative. Think about it this way: List the companies you think of today as the biggest IT players. Wait five years, perform the same exercise and compare your notes. There's a good chance more than half of the names on that list will be going through hard times and the shift to the cloud and a service-based economy will be the main reasons why.

But IT vendors won't be the only ones impacted by these changes. The cloud has already helped companies increase their competitiveness today and will play an important role in ensuring it tomorrow. Those who continue to reject cloud solutions as not being flexible, secure or good enough will fail under the weight of their own IT costs and lack of agility. As of today, any company creating new IT assets that does not consider the cloud in some form is increasing the legacy burden that will make their move to the cloud more painful and their business less competitive.

This white paper provides CloudBees' vision of the cloud and its impact on development and IT, as articulated by Sacha Labourey, CEO.

CloudBees

CloudBees, Inc.

400 TradeCenter, Suite 4950
Woburn, MA 01801
USA

www.cloudbees.com
info@cloudbees.com