Above the Clouds: A View of Cloud Computing

Rakesh Gupta
Rakeshg25@gmail.com

Abstract

Cloud Computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way IT hardware is designed and purchased. Developers with innovative ideas for new Internet services no longer require the large capital outlays in hardware to deploy their service or the human expense to operate it. They need not be concerned about over provisioning for a service whose popularity does not meet their predictions, thus wasting costly resources, or under provisioning for one that becomes wildly popular, thus missing potential customers and revenue. Moreover, companies with large batch-oriented tasks can get results as quickly as their programs can scale, since using 1000 servers for one hour costs no more than using one server for 1000 hours. This elasticity of resources, without paying a premium for large scale, is unprecedented in the history of IT. Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (SaaS). The datacenter hardware and software is what we will call a Cloud. When a Cloud is made available in a pay-as-you-go manner to the general public, we call it a Public Cloud; the service being sold is Utility Computing. We use the term Private Cloud to refer to internal datacenters of a business or other organization, not made available to the general public. Thus, Cloud Computing is the sum of SaaS and Utility Computing, but does not include Private Clouds. People can be users or providers of SaaS, or users or providers of Utility Computing. We focus on SaaS Providers (Cloud Users) and Cloud Providers, which have received less attention than SaaS Users. From a hardware point of view, three aspects are new in Cloud Computing.
1. The illusion of infinite computing resources available on demand, thereby eliminating the need for Cloud Computing users to plan far ahead for provisioning. 2. The elimination of an up-front commitment by Cloud users, thereby allowing companies to start small and increase hardware resources only when there is an increase in their needs.3. The ability to pay for use of computing resources on a short-term basis as needed (e.g., processors by the hour and storage by the day) and release them as needed, thereby rewarding conservation by letting machines and storage go when they are no longer useful.

Key words: *computing- designed- potential- Cloud- organization- datacenters- capital- transform- provisioning*

Introduction to Cloud Computing

Cloud Computing is a new term for a long-held dream of computing as a utility, which has recently emerged as a commercial reality. Cloud Computing is likely to have the same impact on software that foundries have had on the hardware industry. At one time, leading hardware companies required a captive semiconductor fabrication facility, and companies had to be large enough to afford to build and operate it economically.

Table 1: Quick Preview of Top 05 Obstacles to and Opportunities for Growth of Cloud Computing.

	Obstacle	Opportunity
1	Availability of Service	Use Multiple Cloud Providers; Use Elasticity to Prevent DDOS
2	Availability of Service	Standardize APIs; Compatible SW to enable Surge Computing
3	Data Confidentiality and Audit ability	Deploy Encryption, VLANs, Firewalls; Geographical Data Storage
4	Scalable Storage	Invent Scalable Store
5	Bugs in Large Distributed Systems	Invent Debugger that relies on Distributed VMs

However, processing equipment doubled in price every technology generation. A semiconductor fabrication line costs over $3B today, so only a handful of major "merchant" companies with very high chip volumes, such as Intel and Samsung, can still justify owning and operating their own fabrication lines. This motivated the rise of semiconductor foundries that build chips for others, such as Taiwan Semiconductor Manufacturing Company (TSMC). Foundries enable "fab-less" semiconductor chip companies whose value is in innovative chip design: A company such as nVidia can now be successful in the chip business without the capital, operational expenses, and risks associated with owning a state-of-the-art fabrication line. Conversely, companies with fabrication lines can time-multiplex their use among the products of many fab-less companies, to lower the risk of not having enough successful products to amortize operational costs. Similarly, the advantages of the economy of scale and statistical multiplexing may ultimately lead to a handful of Cloud Computing providers who can amortize the cost of their large datacenters over the products of many "datacenter-less" companies. Cloud Computing has been talked about, blogged about written about and been featured in the title of workshops, conferences, and even magazines. Nevertheless, confusion remains about exactly what it is and when it's useful, causing Oracle's CEO to vent his frustration:

Our goal in this paper to clarify terms, provide simple formulas to quantify comparisons between of cloud and conventional Computing, and identify the top technical and non-technical obstacles and opportunities of Cloud Computing. Our view is shaped in part by working since 2005 in the UC Berkeley RAD Lab and in part as users of Amazon Web Services since January 2008 in conducting our research and our teaching. The RAD Lab's research agenda is to invent technology that leverages machine learning to help automate the operation of datacenters for scalable Internet services. We spent six months brainstorming about Cloud Computing, leading to this paper that tries to answer the following questions:

What is Cloud Computing, and how is it different from previous paradigm shifts such as Software as a Service (SaaS)? • Why is Cloud Computing poised to take off now, whereas previous attempts have foundered?
• What does it take to become a Cloud Computing provider, and why would a company consider becoming one?
• What new opportunities are either enabled by or potential drivers of Cloud Computing?
• How might we classify current Cloud Computing offerings across a spectrum, and how do the technical and business challenges

differ depending on where in the spectrum a particular offering lies?
• What, if any, are the new economic models enabled by Cloud Computing, and how can a service operator decide whether to move to the cloud or stay in a private datacenter?
• What are the top 5 obstacles to the success of Cloud Computing—and the corresponding top 5 opportunities available for overcoming the obstacles?
• What changes should be made to the design of future applications software, infrastructure software, and hardware to match the needs and opportunities of Cloud Computing?

2 What is Cloud Computing?

Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (SaaS), so we use that term. The datacenter hardware and software is what we will call a Cloud.

When a Cloud is made available in a pay-as-you-go manner to the public, we call it a Public Cloud; the service being sold is Utility Computing. Current examples of public Utility Computing include Amazon Web Services, Google AppEngine, and Microsoft Azure. We use the term Private Cloud to refer to internal datacenters of a business or other organization that are not made available to the public. Thus, Cloud Computing is the sum of SaaS and Utility Computing, but does not normally include Private Clouds. We'll generally use Cloud Computing, replacing it with one of the other terms only when clarity demands it. Figure 1 shows the roles of the people as users or providers of these layers of Cloud Computing, and we'll use those terms to help make our arguments clear.

The advantages of SaaS to both end users and service providers are well understood. Service providers enjoy greatly simplified software installation and maintenance and centralized control over versioning; end users can access the service "anytime, anywhere", share data and collaborate more easily, and keep their data stored safely in the infrastructure. Cloud Computing does not change these arguments, but it does give more application providers the choice of deploying their product as SaaS without provisioning a datacenter: just as the emergence of semiconductor foundries gave chip companies the opportunity to design and sell chips without owning a fab, Cloud Computing allows deploying SaaS—and scaling on demand—without building or provisioning a datacenter. Analogously to how SaaS allows the user to offload some problems to the SaaS provider, the SaaS provider can now offload some of his problems to the Cloud Computing provider. From now on, we will focus on issues related to the potential SaaS Provider (Cloud User) and to the Cloud Providers, which have received less attention.

We will eschew terminology such as "X as a service (XaaS)"; values of X we have seen in print include Infrastructure, Hardware, and Platform, but we were unable to agree even among ourselves what the precise differences among them might be.1 (We are using Endnotes instead of footnotes. Go to page 20 at the end of paper to read the notes, which have more details.) Instead, we present a simple classification of Utility Computing services in Section 5 that focuses on the tradeoffs among programmer convenience, flexibility, and portability, from both the cloud provider's and the cloud user's point of view

From a hardware point of view, three aspects are new in Cloud Computing:
1. The illusion of infinite computing resources available on demand, thereby eliminating the need for Cloud Computing users to plan far ahead for provisioning;
2. The elimination of an up-front commitment by Cloud users, thereby

allowing companies to start small and increase hardware resources only when there is an increase in their needs; and

3. The ability to pay for use of computing resources on a short-term basis as needed (e.g., processors by the hour and storage by the day) and release them as needed, thereby rewarding conservation by letting machines and storage go when they are no longer useful.

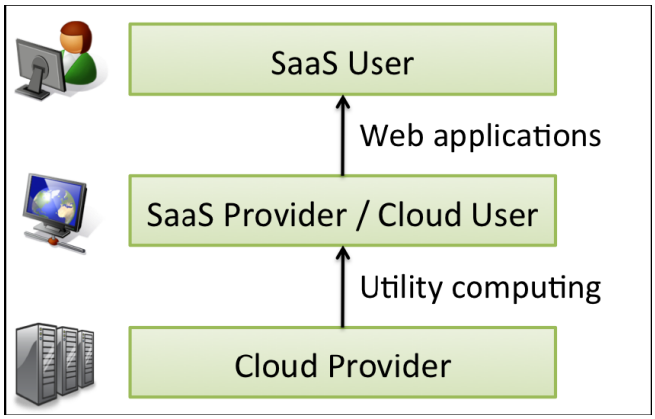


Figure 1: Users and Providers of Cloud Computing.

The benefits of SaaS to both SaaS users and SaaS providers are well documented, so we focus on Cloud Computing effects on Cloud Providers and SaaS Providers/Cloud users. The top level can be recursive, in that SaaS providers can also be a SaaS users. For example, a mash up provider of rental maps might be a user of the Craigslist and Google maps services.

We will argue that all three are important to the technical and economic changes made possible by Cloud Computing. Indeed, past efforts at utility computing failed, and we note that in each case one or two of these three critical characteristics were missing. For example, Intel Computing Services in 2000-2001 required negotiating a contract and longer-term use than per hour.

As a successful example, Elastic Compute Cloud (EC2) from Amazon Web Services (AWS) sells 1.0-GHz x86 ISA "slices" for 10 cents per hour, and a new "slice", or instance, can be added in 2 to 5 minutes. Amazon's Scalable Storage Service (S3) charges $0.12 to $0.15 per gigabyte-month, with additional bandwidth charges of $0.10 to $0.15 per gigabyte to move data in to and out of AWS over the Internet. Amazon's bet is that by statistically multiplexing multiple instances onto a single physical box, that box can be simultaneously rented to many customers who will not in general interfere with each others' usage.

While the attraction to Cloud Computing users (SaaS providers) is clear, who would become a Cloud computing provider, and why? To begin with, realizing the economies of scale afforded by statistical multiplexing and bulk purchasing requires the construction of extremely large datacenters.

Building, provisioning, and launching such a facility is a hundred-million-dollar undertaking. However, because of the phenomenal growth of Web services through the early 2000's, many large Internet companies, including Amazon, eBay, Google, Microsoft and others, were already doing so. Equally important, these companies also had to develop scalable software infrastructure (such as Map Reduce, the Google File System, Big Table, and Dynamo and the operational expertise to armor their datacenters against potential physical and electronic attacks.

Therefore, a necessary but not sufficient condition for a company to become a Cloud Computing provider is that it must have existing investments not only in very large datacenters, but also in large-scale software infrastructure and operational expertise required to run them. Given these conditions, a variety of factors might influence these companies to become Cloud Computing providers:

1. Make a lot of money. Although 10 cents per server-hour seems low, Table 2 summarizes James Hamilton's estimates that very large datacenters (tens of thousands of computers) can purchase hardware, network bandwidth, and power for 1=5 to 1=7 the prices offered to a medium-sized (hundreds or thousands of computers) datacenter. Further, the fixed costs of software development and deployment can be amortized over many more machines. Others estimate the price advantage as a factor of 3 to 5 . Thus, a sufficiently large company could leverage these economies of scale to offer a service well below the costs of a medium-sized company and still make a tidy profit.

2. Leverage existing investment. Adding Cloud Computing services on top of existing infrastructure provides a new revenue stream at (ideally) low incremental cost, helping to amortize the large investments of datacenters. Indeed, according to Werner Vogels, Amazon's CTO, many Amazon Web Services technologies were initially developed for Amazon's internal operations.

3. Defend a franchise. As conventional server and enterprise applications embrace Cloud Computing, vendors with an established franchise in those applications would be motivated to provide a cloud option of their own. For example, Microsoft Azure provides an immediate path for migrating existing customers of Microsoft enterprise applications to a cloud environment.

Table 2
Economies of scale in 2006 for medium-sized datacenter (_1000 servers) vs. very large datacenter (_50,000 servers)

Technology	Cost in Medium-sized DC	Cost in Very Large DC	Ratio
Network	95 per Mbit/sec/month	13 per Mbit/sec/month	7.1
Storage	2.20 per GByte / month	0.40 per GByte / month	5.7
Administration	140 Servers / Administrator	>1000 Servers / Administrator	7.1

Table 3
Price of kilowatt-hours of electricity by region

Price per KWH	Where	Possible Reasons Why
3.6¢	Idaho	Hydroelectric power; not sent long distance
10.0¢	California	Electricity transmitted long distance over the grid; limited transmission lines in Bay Area; no coal fired electricity allowed in California.
18.0¢	Hawaii	Must ship fuel to generate electricity

4. Attack an incumbent. A company with the requisite datacenter and software resources might want to establish a beachhead in this space before a single "800 pound gorilla" emerges. Google AppEngine provides an alternative path to cloud deployment whose appeal lies in its automation of many of the scalability and

load balancing features that developers might otherwise have to build for themselves.

5. Leverage customer relationships. IT service organizations such as IBM Global Services have extensive customer relationships through their service offerings. Providing a branded Cloud Computing offering gives those customers an anxiety-free migration path that preserves both parties' investments in the customer relationship.

6. Become a platform. Facebook's initiative to enable plug-in applications is a great fit for cloud computing, as we will see, and indeed one infrastructure provider for Facebook plug-in applications is Joyent, a cloud provider. Yet Facebook's motivation was to make their social-networking application a new development platform.

Several Cloud Computing (and conventional computing) datacenters are being built in seemingly surprising locations, such as Quincy, Washington (Google, Microsoft, Yahoo!, and others) and San Antonio, Texas (Microsoft, US National Security Agency, others). The motivation behind choosing these locales is that the costs for electricity, cooling, labor, property purchase costs, and taxes are geographically variable, and of these costs, electricity and cooling alone can account for a third of the costs of the datacenter. Table 3 shows the cost of electricity in different locales . Physics tells us it's easier to ship photons than electrons; that is, it's cheaper to ship data over fiber optic cables than to ship electricity over high-voltage transmission lines.

3 Clouds in a Perfect Storm: Why Now, Not Then?

Although we argue that the construction and operation of extremely large scale commodity-computer datacenters was the key necessary enabler of Cloud Computing, additional technology trends and new business models also played a key role in making it a reality this time around. Once Cloud Computing was "off the ground," new application opportunities and usage models were discovered that would not have made sense previously.

3.1 New Technology Trends and Business Models

Accompanying the emergence of Web 2.0 was a shift from "high-touch, high-margin, high-commitment" provisioning of service "low-touch, low-margin, low-commitment" self-service. For example, in Web 1.0, accepting credit card payments from strangers required a contractual arrangement with a payment processing service such as VeriSign or Authorize.net; the arrangement was part of a larger business relationship, making it onerous for an individual or a very small business to accept credit cards online. With the emergence of PayPal, however, any individual can accept credit card payments with no contract, no long-term commitment, and only modest pay-as-you-go transaction fees. The level of "touch" (customer support and relationship management) provided by these services is minimal to nonexistent, but 6 the fact that the services are now within reach of individuals seems to make this less important. Similarly, individuals' Web pages can now use Google AdSense to realize revenue from ads, rather than setting up a relationship with an ad placement company, such Double Click (now acquired by Google). Those ads can provide the business model for Wed 2.0 apps as well. Individuals can distribute Web content using Amazon Cloud Front rather than establishing a relationship with a content distribution network such as Akamai.

Amazon Web Services capitalized on this insight in 2006 by providing pay-as-you-go computing with no contract: all customers need is a credit card. A second innovation was selling hardware-level virtual machines cycles, allowing customers to choose their

own software stack without disrupting each other while sharing the same hardware and thereby lowering costs further.

3.2 New Application Opportunities

While we have yet to see fundamentally new types of applications enabled by Cloud Computing, we believe that several important classes of existing applications will become even more compelling with Cloud Computing and contribute further to its momentum. When Jim Gray examined technological trends in 2003 , he concluded that economic necessity mandates putting the data near the application, since the cost of wide-area networking has fallen more slowly (and remains relatively higher) than all other IT hardware costs. Although hardware costs have changed since Gray's analysis, his idea of this "breakeven point" has not. Although we defer a more thorough discussion of Cloud Computing economics to Section 6, we use Gray's insight in examining what kinds of applications represent particularly good opportunities and drivers for Cloud Computing.

Mobile interactive applications. Tim O'Reilly believes that "the future belongs to services that respond in real time to information provided either by their users or by nonhuman sensors." [38] Such services will be attracted to the cloud not only because they must be highly available, but also because these services generally rely on large data sets that are most conveniently hosted in large datacenters. This is especially the case for services that combine two or more data sources or other services, e.g., mash ups. While not all mobile devices enjoy connectivity to the cloud 100% of the time, the challenge of disconnected operation has been addressed successfully in specific application domains, 2 so we do not see this as a significant obstacle to the appeal of mobile applications.

Parallel batch processing. Although thus far we have concentrated on using Cloud Computing for interactive SaaS, Cloud Computing presents a unique opportunity for batch-processing and analytics jobs that analyze terabytes of data and can take hours to finish. If there is enough data parallelism in the application, users can take advantage of the cloud's new "cost associatively": using hundreds of computers for a short time costs the same as using a few computers for a long time. For example, Peter Harkins, a Senior Engineer at The Washington Post, used 200 EC2 instances (1,407 server hours) to convert 17,481 pages of Hillary Clinton's travel documents into a form more friendly to use on the WWW within nine hours after they were released [3]. Programming abstractions such as Google's Map Reduce [16] and its open-source counterpart Hadoop [11] allow programmers to express such tasks while hiding the operational complexity of choreographing parallel execution across hundreds of Cloud Computing servers. Indeed, Cloud era [1] is pursuing commercial opportunities in this space. Again, using Gray's insight, the cost/benefit analysis must weigh the cost of moving large datasets into the cloud against the benefit of potential speedup in the data analysis. When we return to economic models later, we speculate that part of Amazon's motivation to host large public datasets for free [8] may be to mitigate the cost side of this analysis and thereby attract users to purchase Cloud Computing
Cycles near this data.

The rise of analytics. A special case of compute-intensive batch processing is business analytics. While the large database industry was originally dominated by transaction processing, that demand is leveling off. A growing share of computing resources is now spent on understanding customers, supply chains, buying habits, ranking, and so on. Hence, while online

transaction volumes will continue to grow slowly, decision support is growing rapidly, shifting the resource balance in database processing from transactions to business analytics.

Extension of compute-intensive desktop applications. The latest versions of the mathematics software packages Matlab and Mathematica are capable of using Cloud Computing to perform expensive evaluations. Other desktop applications might similarly benet from seamless extension into the cloud. Again, a reasonable test is comparing the cost of computing in the Cloud plus the cost of moving data in and out of the Cloud to the time savings from using the Cloud. Symbolic mathematics involves a great deal of computing per unit of data, making it a domain worth investigating. An interesting alternative model might be to keep the data in the cloud and rely on having sufficient bandwidth to enable suitable visualization and a responsive GUI back to the human user. Offline image rendering or 3D animation might be a similar example: given a compact description of the objects in a 3D scene and the characteristics of the lighting sources, rendering the image is an embarrassingly parallel task with a high computation-to-bytes ratio.

"Earthbound" applications. Some applications that would otherwise be good candidates for the cloud's elasticity and parallelism may be thwarted by data movement costs, the fundamental latency limits of getting into and out of the cloud, or both. For example, while the analytics associated with making long-term financial decisions are appropriate 7 for the Cloud, stock trading that requires microsecond precision is not. Until the cost (and possibly latency) of wide area data transfer decrease (see Section 7), such applications may be less obvious candidates for the cloud.

4 Classes of Utility Computing

Any application needs a model of computation, a model of storage and, assuming the application is even trivially distributed, a model of communication. The statistical multiplexing necessary to achieve elasticity and the illusion of infinite capacity requires resources to be virtualized, so that the implementation of how they are multiplexed and shared can be hidden from the programmer. Our view is that different utility computing offerings will be distinguished based on the level of abstraction presented to the programmer and the level of management of the resources.

Amazon EC2 is at one end of the spectrum. An EC2 instance looks much like physical hardware, and users can control nearly the entire software stack, from the kernel upwards. The API exposed is "thin": a few dozen API calls to request and configure the virtualized hardware. There is no a priori limit on the kinds of applications that can be hosted; the low level of virtualization—raw CPU cycles, block-device storage, IP-level connectivity— allow developers to code whatever they want. On the other hand, this makes it inherently difficult for Amazon to offer automatic scalability and failover, because the semantics associated with replication and other state management issues are highly application-dependent.

AWS does offer a number of higher-level managed services, including several different managed storage services for use in conjunction with EC2, such as Simple DB. However, these offerings have higher latency and nonstandard API's, and our understanding is that they are not as widely used as other parts of AWS.

Table 4 summarizes how these three classes virtualize computation, storage, and networking. The scattershot offerings of scalable storage suggest that scalable storage with an API comparable in richness to SQL remains an open research problem (see

Section 7). Amazon has begun offering Oracle databases hosted on AWS, but the economics and licensing model of this product makes it a less natural fit for Cloud Computing. Table 4 summarizes how these three classes virtualize computation, storage, and networking. The scattershot offerings of scalable storage suggest that scalable storage with an API comparable in richness to SQL remains an open research problem (see

Section 7). Amazon has begun offering Oracle databases hosted on AWS, but the economics and licensing model of this product makes it a less natural fit for Cloud Computing.

Table 4

	Amazon Web Services	Microsoft Azure	Google AppEngine
Computation model (VM)	_ x86 Instruction Set Architecture (ISA) via Xen VM _ Computation elasticity allows scalability, but developer must build the machinery, or third party VAR such as Right Scale must provide it	_ Microsoft Common Language Runtime (CLR) VM; common intermediate form executed in managed environment _ Machines are provisioned based on declarative descriptions (e.g. which "roles" can be replicated); automatic load balancing	_ Predefined application structure and framework; programmer-provided "handlers" written in Python, all persistent state stored in Mega Store (outside Python code) _ Automatic scaling up and down of computation and storage; network and server failover; all consistent with 3-tier Web app structure
Storage model	_ Range of models from block store (EBS) to augmented key/blob store (Simple DB) _ Automatic scaling varies from no scaling or sharing (EBS) to fully automatic (Simple DB, S3), depending on which model used _ Consistency guarantees vary widely depending on which model used _ APIs vary from standardized (EBS) to proprietary	_ SQL Data Services (restricted view of SQL Server) _ Azure storage service	_Mega Store/Big Table
Networking model	Declarative specification of IP level topology; internal placement details concealed _ Security Groups enable restricting which nodes may communicate _ Availability zones provide abstraction of independent network failure _ Elastic IP addresses provide persistently routable network name	Automatic based on programmer's declarative descriptions of app components (roles)	Fixed topology to accommodate 3-tier Web app structure _ Scaling up and down is automatic and programmer invisible

Will one model beat out the others in the Cloud Computing space? We can draw an analogy with programming languages and frameworks. Low-level languages such as C and assembly language allow fine control and close communication with the bare metal, but if the developer is writing a Web application, the mechanics of managing sockets, dispatching requests, and so on are cumbersome and tedious to code, even with good libraries. On the other hand, high-level frameworks such as Ruby on Rails make these mechanics invisible to the programmer, but are only useful if the application readily fits the request/reply structure and the abstractions provided by Rails; any deviation requires diving into the framework at best, and may be awkward to code. No reasonable Ruby developer would argue against the superiority of C for certain tasks, and vice versa. Correspondingly, we believe different tasks will result in demand for different classes of utility computing.

Table 4: Examples of Cloud Computing vendors and how each provides virtualized resources (computation, storage, networking) and ensures scalability and high availability of the resources.

5 Cloud Computing Economics

In this section we make some observations about Cloud Computing economic models:

In deciding whether hosting a service in the cloud makes sense over the long term, we argue that the finegrained economic models enabled by Cloud Computing make tradeoff decisions more fluid, and in particular the elasticity offered by clouds serves to transfer risk.

As well, although hardware resource costs continue to decline, they do so at variable rates; for example, computing and storage costs are falling faster than WAN costs. Cloud Computing can track these changes—and potentially pass them through to the customer—more effectively than building one's own datacenter, resulting in a closer match of expenditure to actual resource usage.

In making the decision about whether to move an existing service to the cloud, one must additionally examine the expected average and peak resource utilization, especially if the application may have highly variable spikes in resource demand; the practical limits on real-world utilization of purchased equipment; and various operational costs that vary depending on the type of cloud environment being considered.

5.1 Elasticity: Shifting the Risk

Although the economic appeal of Cloud Computing is often described as "converting capital expenses to operating expenses" (CapEx to OpEx), we believe the phrase "pay as you go" more directly captures the economic benefit to the buyer. Hours purchased via Cloud Computing can be distributed non-uniformly in time (e.g., use 100 server-hours today and no server-hours tomorrow, and still pay only for what you use); in the networking community, this way of selling bandwidth is already known as usage-based pricing. 3 In addition, the absence of up-front capital expense allows capital to be redirected to core business investment.

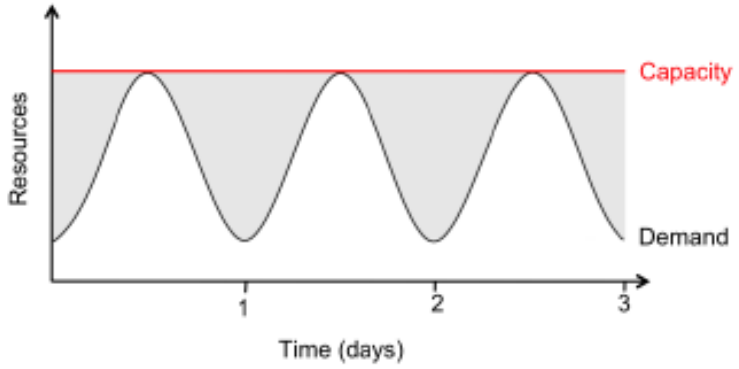
Therefore, even though Amazon's pay-as-you-go pricing (for example) could be more expensive than buying and depreciating a comparable server over the same period, we argue that the cost is outweighed by the extremely important Cloud Computing economic benefits of elasticity and transference of risk, especially the risks of over provisioning (underutilization) and under provisioning (saturation).

We start with elasticity. The key observation is that Cloud Computing ability to add or remove resources at a fine grain (one server at a time with EC2) and with a

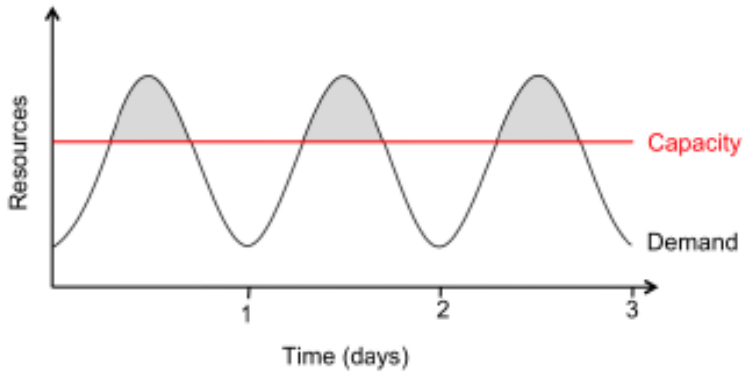
lead time of minutes rather than weeks allows matching resources to workload much more closely. Real world estimates of server utilization in datacenters range from 5% to 20% [37, 38]. This may sound shockingly low, but it is consistent with the observation that for many services the peak workload exceeds the average by factors of 2 to 10. Few users deliberately provision for less than the expected peak, and therefore they must provision for the peak and allow the resources to remain idle at nonpeak times. The more pronounced the variation, the more the waste. A simple example demonstrates how elasticity allows reducing this waste and can therefore more than compensate for the potentially higher cost per server-hour of paying-as-you-go vs. buying.

Example: Elasticity. Assume our service has a predictable daily demand where the peak requires 500 servers at noon but the trough requires only 100 servers at midnight, as shown in Figure 2(a). As long as the average utilization over a whole day is 300 servers, the actual utilization over the whole day (shaded area under the curve) is 300 _ 24 = 7200 server-hours; but since we must provision to the peak of 500 servers, we pay for 500 _ 24 = 12000 server-hours, a factor of 1.7 more than what is needed. Therefore, as long as the pay-as-you-go cost per server-hour over 3 years4 is less than 1.7 times the cost of buying the server, we can save money using utility computing.

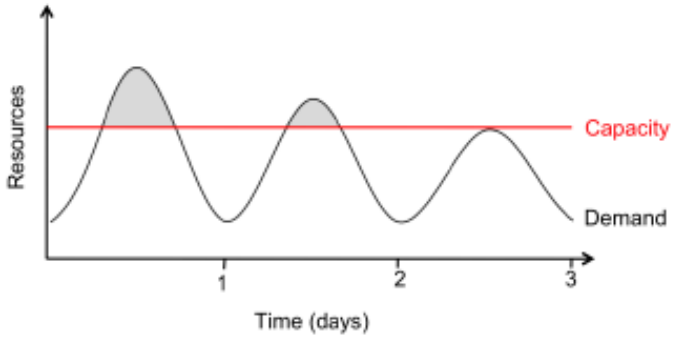
They may also underestimate the spike (Figure 2(b)), however, accidentally turning away excess users. While the monetary effects of over provisioning are easily measured, those of under provisioning are harder to measure yet potentially equally serious: not only do rejected users generate zero revenue; they may never come back due to poor service. Figure 2(c) aims to capture this behavior: users will desert an under provisioned service until the peak user.



(a) Provisioning for peak load



(b) Underprovisioning 1



(c) Underprovisioning 2

Figure 2: (a) Even if peak load can be correctly anticipated, without elasticity we waste resources (shaded area) during nonpeak times. (b) Under provisioning case 1: potential revenue from users not served (shaded area) is sacrificed. (c) Under provisioning case 2: some users desert the site permanently after experiencing poor service; this attrition and possible negative press result in a permanent loss of a portion of the revenue stream.

load equals the datacenter's usable capacity, at which point users again receive acceptable service, but with fewer potential users.

Example: Transferring risks. Suppose but 10% of users who receive poor service due to under provisioning are "permanently lost" opportunities, i.e. users who would have remained regular visitors with a better experience. The site is initially provisioned to handle an expected peak of 400,000 users (1000 users per server _ 400 servers), but unexpected positive press drives 500,000 users in the first hour. Of the 100,000 who are turned away or receive bad service, by our assumption 10,000 of them are permanently lost, leaving an active user base of 390,000. The next hour sees 250,000 new unique users. The first 10,000 do fine, but the site is still over capacity by 240,000 users. This results in 24,000 additional defections, leaving 376,000 permanent users. If this pattern continues, after lg 500000 or 19 hours, the number of new users will approach zero and the site will be at capacity in steady state. Clearly, the service operator has collected less than 400,000 users' worth of steady revenue during those 19 hours, however, again illustrating the underutilization argument —to say nothing of the bad reputation from the disgruntled users.

Do such scenarios really occur in practice? When Animoto made its service available via Facebook, it experienced a demand surge that resulted in growing from 50 servers to 3500 servers in three days. Even if the average utilization of each server was low, no one could have foreseen that resource needs would suddenly double every 12 hours for 3 days. After the peak subsided, traffic fell to a level that was well below the peak. So in this real world example, scale-up elasticity was not a cost optimization but an operational requirement, and scale-down elasticity allowed the steady-state expenditure to more closely match the steady-state workload.

Even less-dramatic cases suffice to illustrate this key benefit of Cloud Computing: the risk of mis-estimating workload is shifted from the service operator to the cloud vendor. The cloud vendor may charge a premium (reflected as a higher use cost per server-hour compared to the 3-year purchase cost) for assuming this risk. We propose the following simple equation that generalizes all of the above cases. We assume the Cloud Computing vendor

employs 11 usage-based pricing, in which customers pay proportionally to the amount of time and the amount of resources they use. While some argue for more sophisticated pricing models for infrastructure services [28, 6, 40], we believe usage based pricing will persist because it is simpler and more transparent, as demonstrated by its wide use by "real" utilities such as electricity and gas companies. Similarly, we assume that the customer's revenue is directly proportional to the total number of user-hours. This assumption is consistent with the ad-supported revenue model in which the number of ads served is roughly proportional to the total visit time spent by end users on the service.

$$\text{UserHours}_{\text{cloud}} \times (\text{revenue} - \text{Cost}_{\text{cloud}}) \geq \text{UserHours}_{\text{datacenter}} \times \left(\text{revenue} - \frac{\text{Cost}_{\text{datacenter}}}{\text{Utilization}}\right)$$

The left-hand side multiplies the net revenue per user-hour (revenue realized per user-hour minus cost of paying Cloud Computing per user-hour) by the number of user-hours, giving the expected profit from using Cloud Computing. The right-hand side performs the same calculation for a fixed-capacity datacenter by factoring in the average utilization, including nonpeak workloads. Whichever side is greater represents the opportunity for higher profit.

Apparently, if Utilization = 1:0 (the datacenter equipment is 100% utilized), the two sides of the equation look the same. However, basic queuing theory tells us that as utilization approaches 1.0, system response time approaches infinity. In practice, the usable capacity of a datacenter (without compromising service) is typically 0.6 to 0.8.6 Whereas a datacenter must necessarily overprovision to account for this "overhead," the cloud vendor can simply factor it into Cost cloud. (This overhead explains why we use the phrase "pay-as-you-go" rather than rent or lease for utility computing. The latter phrases include this unusable overhead, while the former doesn't. Hence, even if you lease a 100 Mbits/second Internet link, you can likely use only 60 to 80 Mbits/second in practice.)

Finally, there are two additional benefits to the Cloud Computing user that result from being able to change their resource usage on the scale of hours rather than years. First, unexpectedly scaling down (disposing of temporarily underutilized equipment)—for example, due to a business slowdown, or ironically due to improved software efficiency— normally carries a financial penalty. With 3-year depreciation, a $2,100 server decommissioned after 1 year of operation represents a "penalty" of $1,400. Cloud Computing eliminates this penalty.

Second, technology trends suggest that over the useful lifetime of some purchased equipment, hardware costs will fall and new hardware and software technologies will become available. Cloud providers, who already enjoy economy-of-scale buying power as described in Section 3, can potentially pass on some of these savings to their customers. Indeed, heavy users of AWS saw storage costs fall 20% and networking costs fall 50% over the last 2.5 years, and the addition of nine new services or features to AWS over less than one year. 7 If new technologies or pricing plans become available to a cloud vendor, existing applications and customers can potentially benefit from them immediately, without incurring a capital expense. In less than two years, Amazon Web Services increased the number of different types of compute servers ("instances") from one to five, and in less than one year they added seven new infrastructure services and two new operational support options. 8

5.2 Comparing Costs: Should I Move to the Cloud?

Whereas the previous section tried to quantify the economic value of specific Cloud Computing benefits such as elasticity, this section tackles an equally important but larger question: Is it more economical to move my existing datacenter-hosted service to the cloud, or to keep it in a datacenter?

Table 5 updates Gray's 2003 cost data to 2008, allowing us to track the rate of change of key technologies for Cloud Computing for the last 5 years. Note that, as expected, wide-area networking costs have improved the least in 5 years, by less than a factor of 3. While computing costs have improved the most in 5 years, the ability to use the extra computing power is based on the assumption that programs can utilize all the cores on both sockets in the computer. This assumption is likely more true for Utility Computing, with many Virtual Machines serving thousands to millions of customers, than it is for programs inside the datacenter of a single company.

To facilitate calculations, Gray calculated what $1 bought in 2003. Table 5 shows his numbers vs. 2008 and compares to EC2/S3 charges. At first glance, it appears that a given dollar will go further if used to purchase hardware in 2008 than to pay for use of that same hardware. However, this simple analysis glosses over several important factors.

Pay separately per resource. Most applications do not make equal use of computation, storage, and network bandwidth; some are CPU-bound, others network-bound, and so on, and may saturate one resource while underutilizing others. Pay-as-you-go Cloud Computing can charge the application separately for each type of resource, reducing the waste of underutilization. While the exact savings depends on the application, suppose the CPU is only 50% utilized while the network is at capacity; then in a datacenter you are effectively paying for double the number of CPU cycles actually being used. So rather than saying it costs $2.56 to rent only $1 worth of CPU, it would be more accurate to say it costs $2.56 to rent $2 worth of CPU. As a side note, AWS's prices for wide-area networking are actually more competitive than what a medium-sized company would pay for the same bandwidth.

Table 5: We update Gray's costs of computing resources from 2003 to 2008, normalize to what $1 could buy in 2003 vs. 2008, and compare to the cost of paying per use of $1 worth of resources on AWS at 2008 prices.

Power, cooling and physical plant costs. The costs of power, cooling, and the amortized cost of the building are missing from our simple analyses so far. Hamilton estimates that the costs of CPU, storage and bandwidth roughly double when those costs are amortized over the building's lifetime [23, 26]. Using this estimate, buying 128 hours of CPU in 2008 really costs $2 rather than $1, compared to $2.56 on EC2. Similarly, 10 GB of disk space costs $2 rather than $1, compared to $1.20–$1.50 per month on S3. Lastly, S3 actually replicates the data at least 3 times for durability and performance, ensure durability, and will replicate it further for performance is there is high demand for the data. That means the costs are $6.00 when purchasing vs. $1.20 to $1.50 per month on S3.

Operations costs. Today, hardware operations costs are very low—rebooting servers is easy (e.g., IP addressable power strips, separate out of band controllers, and so on) and minimally trained staff can replace broken components at the rack or server level. On one hand, since Utility Computing uses virtual machines instead of physical machines, from the cloud user's point of view these tasks are shifted to the cloud provider. On the other hand, depending on the level of virtualization, much of the software management costs may remain—upgrades, applying patches, and so

on. Returning to the "managed vs. unmanaged" discussion of Section 5, we believe these costs will be lower for managed environments (e.g. Microsoft Azure, Google AppEngine, Force.com) than for hardware-level utility computing (e.g. Amazon EC2), but it seems hard to quantify these benefits in a way that many would agree with.

	WAN bandwidth/mo.	**CPU hours (all cores)**	**disk storage**
Item in 2003	1 Mbps WAN link	2 GHz CPU, 2 GB DRAM	200 GB disk, 50 Mb/s transfer rate
Cost in 2003	$100/mo.	$2000	$200
$1 buys in 2003	1 GB	8 CPU hours	1 GB
Item in 2008	100 Mbps WAN link	2 GHz, 2 sockets, 4 cores/socket, 4 GB DRAM	1 TB disk, 115 MB/s sustained transfer
Cost in 2008	$3600/mo.	$1000	$100
$1 buys in 2008	2.7 GB	128 CPU hours	10 GB
cost/performance improvement	2.7x	16x	10x
Cost to rent $1 worth on AWS in 2008	**$0.27–$0.40** ($0.10–$0.15/GB × 3 GB)	**$2.56** (128× 2 VM's@$0.10 each)	**$1.20–$1.50** ($0.12–$0.15/GB-month × 10 GB)

With the above caveats in mind, here is a simple example of deciding whether to move a service into the cloud.

Example: Moving to cloud. Suppose a biology lab creates 500 GB of new data for every wet lab experiment. A computer the speed of one EC2 instance takes 2 hours per GB to process the new data. The lab has the equivalent 20 instances locally, so the time to evaluate the experiment is 500_2=20 or 50 hours. They could process it in a single hour on 1000 instances at AWS. The cost to process one experiment would be just 1000_$0:10 or $100 in computation and another 500_$0:10 or $50 in network transfer fees. So far, so good. They measure the transfer rate from the lab to AWS at 20 Mbits/second. [19] The transfer time is (500GB _ 1000MB=GB _ 8bits=Byte)=20Mbits=sec = 4; 000; 000=20 = 200; 000 seconds or more than 55 hours. Thus, it takes 50 hours locally vs. 55 + 1 or 56 hours on AWS, so they don't move to the cloud. (The next section offers an opportunity on how to overcome the transfer delay obstacle.)

A related issue is the software complexity and costs of (partial or full) migrating data from a legacy enterprise application into the Cloud. While migration is a one-time task, the amount of effort can be significant and it needs to be considered as a factor in deciding to use Cloud Computing. This task is already spawning new business opportunities for companies that provide data integration across public and private Clouds.

Top 05 Obstacles and Opportunities for Cloud Computing

In this section, we offer a ranked list of obstacles to the growth of Cloud Computing. Each obstacle is paired with an opportunity—our thoughts on how to overcome the obstacle, ranging from straightforward product development to major research projects. Table 6 summarizes our top ten obstacles and opportunities. The first three are technical obstacles to the

adoption of Cloud Computing, the next five are technical obstacles to the growth of Cloud Computing once it has been adopted, and the last two are policy and business obstacles to the adoption of Cloud Computing.

Table 6: Top 10 Obstacles to and Opportunities for Adoption and Growth of Cloud Computing.

	Obstacle	Opportunity
1	Availability of Service	Use Multiple Cloud Providers to provide Business Continuity; Use Elasticity to Defend Against DDOS attacks
2	Data Lock-In	Standardize APIs; Make compatible software available to enable Surge Computing
3	Data Confidentiality and Auditability	Deploy Encryption, VLANs, and Firewalls; Accommodate National Laws via Geographical Data Storage
4	Data Transfer Bottlenecks	FedExing Disks; Data Backup/Archival; Lower WAN Router Costs; Higher Bandwidth LAN Switches
5	Performance Unpredictability	Improved Virtual Machine Support; Flash Memory; Gang Scheduling VMs for HPC apps
6	Scalable Storage	Invent Scalable Store
7	Bugs in Large-Scale Distributed Systems	Invent Debugger that relies on Distributed VMs
8	Scaling Quickly	Invent Auto-Scaler that relies on Machine Learning; Snapshots to encourage Cloud Computing Conservationism
9	Reputation Fate Sharing	Offer reputation-guarding services like those for email
10	Software Licensing	Pay-for-use licenses; Bulk use sales

Number 1 Obstacle: Availability of a Service

Organizations worry about whether Utility Computing services will have adequate availability, and this makes some wary of Cloud Computing. Ironically, existing SaaS products have set a high standard in this regard. Google Search is effectively the dial tone of the Internet: if people went to Google for search and it wasn't available, they would think the Internet was down. Users expect similar availability from new services, which is hard to do. Table 7 shows recorded outages for Amazon Simple Storage Service (S3), AppEngine and Gmail in 2008, and explanations for the outages. Note that despite the negative publicity due to these outages, few enterprise IT infrastructures are as good.

Table 7: Outages in AWS, AppEngine, and Gmail

Service and Outage	Duration	Date
S3 outage: authentication service overload leading to unavailability [39]	2 hours	2/15/08
S3 outage: Single bit error leading to gossip protocol blowup. [41]	6-8 hours	7/20/08
AppEngine partial outage: programming error [43]	5 hours	6/17/08
Gmail: site unavailable due to outage in contacts system [29]	1.5 hours	8/11/08

Just as large Internet service providers use multiple network providers so that failure by a single company will not take them off the air, we believe the only plausible solution to very high availability is multiple Cloud Computing providers. The high-availability computing community has long followed the mantra "no single source of failure," yet the management of a Cloud Computing service by a single company is in fact a single point of failure. Even if the company has multiple datacenters in different geographic regions using different network providers, it may have common software infrastructure and accounting systems, or the company may even go out of business. Large customers will be reluctant to migrate to Cloud Computing without a business-continuity

strategy for such situations. We believe the best chance for independent software stacks is for them to be provided by different companies, as it has been difficult for one company to justify creating and maintain two stacks in the name of software dependability.

Number 2 Obstacle: Data Lock-In

Software stacks have improved interoperability among platforms, but the APIs for Cloud Computing itself are still essentially proprietary, or at least have not been the subject of active standardization. Thus, customers cannot easily extract their data and programs from one site to run on another. Concern about the difficult of extracting data from the cloud is preventing some organizations from adopting Cloud Computing. Customer lock-in may be attractive to Cloud Computing providers, but Cloud Computing users are vulnerable to price increases (as Stallman warned), to reliability problems, or even to providers going out of business. For example, an online storage service called The Linkup shut down on August 8, 2008 after losing access as much as 45% of customer data . The Linkup, in turn, had relied on the online storage service Nirvanix to store customer data, and now there is finger pointing between the two organizations as to why customer data was lost. Meanwhile, The Linkup's 20,000 users were told the service was no longer available and were urged to try out another storage site. The obvious solution is to standardize the APIs so that a SaaS developer could deploy services and data across multiple Cloud Computing providers so that the failure of a single company would not take all copies of customer data with it. The obvious fear is that this would lead to a "race-to-the-bottom" of cloud pricing and flatten the profits of Cloud Computing providers. We offer two arguments to allay this fear. First, the quality of a service matters as well as the price, so customers will not necessarily jump to the lowest cost service. Some Internet Service Providers today cost a factor of ten more than others because they are more dependable and offer extra services to improve usability. Second, in addition to mitigating data lock-in concerns, standardization of APIs enables a new usage model in which the same software infrastructure can be used in a Private Cloud and in a Public Cloud. 9 Such an option could enable "Surge Computing," in which the public Cloud is used to capture the extra tasks that cannot be easily run in the datacenter (or private cloud) due to temporarily heavy workloads. 10

Number 3 Obstacle: Data Confidentiality and Audit ability

"My sensitive corporate data will never be in the cloud." Anecdotally we have heard this repeated multiple times. Current cloud offerings are essentially public (rather than private) networks, exposing the system to more attacks. There are also requirements for audit ability, in the sense of Sarbanes-Oxley and Health and Human Services Health Insurance Portability and Accountability Act (HIPAA) regulations that must be provided for corporate data to be moved to the cloud.

We believe that there are no fundamental obstacles to making a cloud-computing environment as secure as the vast majority of in-house IT environments, and that many of the obstacles can be overcome immediately with well understood technologies such as encrypted storage, Virtual Local Area Networks, and network middle boxes (e.g. firewalls, packet filters). For example, encrypting data before placing it in a Cloud may be even more secure than unencrypted data in a local data center; this approach was successfully used by TC3, a healthcare company with access to sensitive patient records and healthcare claims, when moving their HIPAA-compliant application to AWS Cloud computing gives SaaS providers and SaaS users greater freedom to place their

storage. For example, Amazon provides S3 services located physically in the United States and in Europe, allowing providers to keep data in whichever they choose. With AWS regions, a simple configuration change avoids the need to find and negotiate with a hosting provider overseas.

Number 4 Obstacle: Scalable Storage

Early in this paper, we identified three properties whose combination gives Cloud Computing its appeal: short-term usage (which implies scaling down as well as up when resources are no longer needed), no up-front cost, and infinite capacity on-demand. While it's straightforward what this means when applied to computation, it's less obvious how to apply it to persistent storage.

As Table 4 shows, there have been many attempts to answer this question, varying in the richness of the query and storage API's, the performance guarantees offered, and the complexity of data structures that are directly supported by the storage system (e.g., schema-less blobs vs. column-oriented storage).14 The opportunity, which is still an open research problem, is to create a storage system would not only meet these needs but combine them with the cloud advantages of scaling arbitrarily up and down on-demand, as well as meeting programmer expectations in regard to resource management for scalability, data durability, and high availability.

Number 5 Obstacle: Bugs in Large-Scale Distributed Systems

One of the difficult challenges in Cloud Computing is removing errors in these very large scale distributed systems. A common occurrence is that these bugs cannot be reproduced in smaller configurations, so the debugging must occur at scale in the production datacenters.

One opportunity may be the reliance on virtual machines in Cloud Computing. Many traditional SaaS providers developed their infrastructure without using VMs, either because they preceded the recent popularity of VMs or because they felt they could not afford the performance hit of VMs. Since VMs are de rigueur in Utility Computing, that level of virtualization may make it possible to capture valuable information in ways that are implausible without VMs.

8 Conclusion and Questions about the Clouds of Tomorrow

The long dreamed vision of computing as a utility is finally emerging. The elasticity of a utility matches the need of businesses providing services directly to customers over the Internet, as workloads can grow (and shrink) far faster than 20 years ago. It used to take years to grow a business to several million customers – now it can happen in months.

From the cloud provider's view, the construction of very large datacenters at low cost sites using commodity computing, storage, and networking uncovered the possibility of selling those resources on a pay-as-you-go model below the costs of many medium-sized datacenters, while making a profit by statistically multiplexing among a large group of customers. From the cloud user's view, it would be as startling for a new software startup to build its own datacenter as it would for a hardware startup to build its own fabrication line. In addition to startups, many other established organizations take advantage of the elasticity of Cloud Computing regularly, including newspapers like the Washington Post, movie companies like Pixar, and universities like ours. Our lab has benefited substantially from the ability to complete research by conference deadlines and adjust resources over the semester to accommodate course deadlines. As Cloud Computing users, we were relieved of dealing with the twin dangers of over-provisioning and under-provisioning our internal datacenters.

Some question whether companies accustomed to high-margin businesses, such as ad revenue from search engines and traditional packaged software, can compete in Cloud Computing. First, the question presumes that Cloud Computing is a small margin business based on its low cost. Given the typical utilization of medium-sized datacenters, the potential factors of 5 to 7 in economies of scale, and the further savings in selection of cloud datacenter locations, the apparently low costs offered to cloud users may still be highly profitable to cloud providers. Second, these companies may already have the datacenter, networking, and software infrastructure in place for their mainline businesses, so Cloud Computing represents the opportunity for more income at little extra cost.

Although Cloud Computing providers may run afoul of the obstacles summarized in Table 6, we believe that over the long run providers will successfully navigate these challenges and set an example for others to follow, perhaps by successfully exploiting the opportunities that correspond to those obstacles.

Hence, developers would be wise to design their next generation of systems to be deployed into Cloud Computing. In general, the emphasis should be horizontal scalability to hundreds or thousands of virtual machines over the efficiency of the system on a single virtual machine. There are specific implications as well:

Applications Software of the future will likely have a piece that runs on clients and a piece that runs in the Cloud. The cloud piece needs to both scale down rapidly as well as scale up, which is a new requirement for software systems. The client piece needs to be useful when disconnected from the Cloud, which is not the case for many Web 2.0 applications today. Such software also needs a pay-for-use licensing model to match needs of Cloud Computing.

Infrastructure Software of the future needs to be cognizant that it is no longer running on bare metal but on virtual machines. Moreover, it needs to have billing built in from the beginning, as it is very difficult to retrofit an accounting system.

While we are optimistic about the future of Cloud Computing, we would love to look into a crystal ball to see how popular it is and what it will look like in five years:

Change In Technology and Prices Over Time: What will billing units be like for the higher-level virtualization clouds? What will Table 5, tracking the relative prices of different resources, look like? Clearly, the number of cores per chip will increase over time, doubling every two to four years. Flash memory has the potential of adding another relatively fast layer to the classic memory hierarchy; what will be its billing unit? Will technology or business innovations accelerate network bandwidth pricing, which is currently the most slowly-improving technology?

Virtualization Level: Will Cloud Computing be dominated by low-level hardware virtual machines like Amazon EC2, intermediate language offerings like Microsoft Azure, or high-level frameworks like Google AppEngine? Or will we have many virtualization levels that match different applications? Will value-added services by independent companies like Right Scale, Heroku, or Engine Yard survive in Utility Computing, or will the successful services be entirely co-opted by the Cloud providers? If they do consolidate to a single virtualization layer, will multiple companies embrace a common standard? Will this lead to a race to the bottom in pricing so that it's unattractive to become a Cloud Computing provider, or will they differentiate in services or quality to maintain margins?

References

ABRAMSON, D., BUYYA, R., AND GIDDY, J. A computational economy for

grid computing and its implementation in theNimrod-G resource broker. Future Generation Computer Systems 18, 8 (2002), 1061–1074.

ADMINISTRATION, E. I. State Electricity Prices, 2006 [online]. Available from: http://www.eia.doe.gov/neic/rankings/stateelectricityprice.htm.

AMAZON AWS. Public Data Sets on AWS [online]. 2008. Available from: http://aws.amazon.com/publicdatasets/.

Amazon.com CEO Jeff Bezos on Animoto [online]. April 2008. Available from: http://blog.animoto.com/2008/04/21/amazon-ceo-jeff-bezos-on-animoto/.

AND TURNER, J. OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review38, 2 (April 2008).

BARROSO, L. A., AND HOLZLE, U. The Case for Energy-Proportional Computing. IEEE Computer 40, 12 (December2007).

BECHTOLSHEIM, A. Cloud Computing and Cloud Networking. talk at UC Berkeley, December 2008.

BIALECKI, A., CAFARELLA, M., CUTTING, D., AND O'MALLEY, O. Hadoop: a framework for running applications on large clusters built of commodity hardware. Wiki at http://lucene. apache. org/hadoop.

Black Friday traffic takes down Sears.com. Associated Press (November 2008).

BRODKIN, J. Loss of customer data spurs closure of online storage service 'The Linkup'. Network World (August 2008).

CARR, N. Rough Type [online]. 2008. Available from: http://www.roughtype.com.

CHANG, F., DEAN, J., GHEMAWAT, S., HSIEH, W., WALLACH, D., BURROWS, M.,CHANDRA, T., FIKES, A., AND GRUBER, R. Bigtable: A distributed storage system for structured data. In Proceedings of the 7th USENIX Symposium onOperating Systems Design and Implementation (OSDI'06) (2006).

CHENG, D. PaaS-onomics: A CIO's Guide to using Platform-as-a-Service to Lower Costs of Application Initiatives WhileImproving the Business Value of IT. Tech. rep., LongJump, 2008.

Cloudera, Hadoop training and support [online]. Available from: http://www.cloudera.com/.

DEAN, J., AND GHEMAWAT, S. Mapreduce: simplified data processing on large clusters. In OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation (Berkeley, CA, USA, 2004), USENIXAssociation, pp. 10–10.

DECANDIA, G., HASTORUN, D., JAMPANI, M., KAKULAPATI, G., LAKSHMAN, A., PILCHIN, A., SIVASUBRAMANIAN, S., VOSSHALL, P., AND VOGELS, W. Dynamo: Amazon's highly available key-value store. In Proceedings of twenty-firstACM SIGOPS symposium on Operating systems principles (2007), ACM Press New York, NY, USA, pp. 205–220.

DEMERS, A. J., PETERSEN, K., SPREITZER, M. J., TERRY, D. B., THEIMER, M. M., AND WELCH, B. B. The bayou architecture: Support for data sharing among mobile users. In Proceedings IEEE Workshop on Mobile Computing Systems &Applications (Santa Cruz, California, August-September 1994), pp. 2–7.22

GARFINKEL, S. An Evaluation of Amazon's Grid Computing Services: EC2, S3 and SQS . Tech. Rep. TR-08-07, HarvardUniversity, August 2007.

GHEMAWAT, S., GOBIOFF, H., AND LEUNG, S.-T. The google file system. In SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles (New York, NY, USA, 2003),

ACM, pp. 29–43.Available from:http://portal.acm.org/ft_gateway.cfm?id=945450&type=pdf&coll=Portal&dl=GUIDE&CFID=19219697&CFTOKEN=50259492.

GRAY, J. Distributed Computing Economics. Queue 6, 3 (2008), 63–68. Available from: http://portal.acm.org/ft_gateway.cfm?id=1394131&type=digital%20edition&coll=Portal&dl=GUIDE&CFID=19219697&CFTOKEN=50259492.

GRAY, J., AND PATTERSON, D. A conversation with Jim Gray. ACM Queue 1, 4 (2003), 8–17.

H¨O LZLE, U. Private communication, January 2009.

HAMILTON, J. Cooperative Expendable Micro-Slice Servers (CEMS):Low Cost, Low Power Servers for Internet-ScaleServices. In Conference on Innovative Data Systems Research (CIDR '09) (January 2009).

HAMILTON, J. Cost of Power in Large-Scale Data Centers [online]. November 2008. Available from:http://perspectives.mvdirona.com/2008/11/28/CostOfPowerInLargeScaleDataCenters.aspx.

HAMILTON, J. Internet-Scale Service Efficiency. In Large-Scale Distributed Systems and Middleware (LADIS) Workshop(September 2008).

HAMILTON, J. Perspectives [online]. 2008. Available from: http://perspectives.mvdirona.com.

HOSANAGAR, K., KRISHNAN, R., SMITH, M., AND CHUANG, J. Optimal pricing of content delivery network (CDN) services. In The 37th Annual Hawaii International Conference onSystem Sciences (2004), pp. 205–214.

JACKSON, T. We feel your pain, and we're sorry [online]. August 2008. Available from: http://gmailblog.blogspot.com/2008/08/we-feel-your-pain-and-were-sorry.html.

KISTLER, J. J., AND SATYANARAYANAN, M. Disconnected operation in the coda file system. In Thirteenth ACM Symposiumon Operating Systems Principles (Asilomar Conference Center, Pacific Grove, U.S., 1991), vol. 25, ACM Press, pp. 213–225.

KREBS, B. Amazon: Hey Spammers, Get Off My Cloud! Washington Post (July 2008).

MCCALPIN, J. Memory bandwidth and machine balance in current high performance computers. IEEE Technical Committeeon Computer Architecture Newsletter (1995), 19–25.MCKEOWN, N., ANDERSON, T., BALAKRISHNAN, H., PARULKAR, G., PETERSON, L., REXFORD, J., SHENKER, S., ,

NURMI, D., WOLSKI, R., GRZEGORCZYK, C., OBERTELLI, G., SOMAN, S., YOUSEFF, L., AND ZAGORODNOV, D. Eucalyptus: A Technical Report on an Elastic Utility Computing Archietcture Linking Your Programs to Useful Systems .Tech. Rep. 2008-10, University of California, Santa Barbara, October 2008.

PARKHILL, D. The Challenge of the Computer Utility. Addison-Wesley Educational Publishers Inc., US, 1966.

PAXSON, V. private communication, December 2008.

RANGAN, K. The Cloud Wars: $100+ billion at stake. Tech. rep., Merrill Lynch, May 2008.

SIEGELE, L. Let It Rise: A Special Report on Corporate IT. The Economist (October 2008).

STERN, A. Update From Amazon Regarding Friday's S3 Downtime. CenterNetworks (February 2008). Available from:http://www.centernetworks.com/amazon-s3-downtime-update.

STUER, G., VANMECHELEN, K., AND BROECKHOVE, J. A commodity market algorithm for pricing substitutable Grid resources. Future Generation Computer Systems 23, 5 (2007), 688–701.

TC3 Health Case Study: Amazon Web Services [online]. Available from: http://aws.amazon.com/solutions/ case-studies/tc3-health/.

THE AMAZON S3 TEAM. Amazon S3 Availability Event: July 20, 2008 [online]. July 2008. Available from: http://status.aws.amazon.com/s3-20080720.html.

VOGELS, W. A Head in the Clouds—The Power of Infrastructure as a Service. In First workshop on Cloud Computing andin Applications (CCA '08) (October 2008).

Washington Post Case Study: Amazon Web Services [online]. Available from: http://aws.amazon.com/solutions/case-studies/washington-post/.

WILSON, S. AppEngine Outage. CIO Weblog (June 2008). Available from: http://www.cio-weblog.com/50226711/appengine_outage.php