Cloud Computing: A Taxonomy of Platform and Infrastructure-level Offerings

David Hilley

College of Computing

Georgia Institute of Technology

April 2009

Cloud Computing: A Taxonomy of Platform and Infrastructure-level Offerings

David Hilley

1 Introduction

Cloud computing is a buzzword and umbrella term applied to several nascent trends in the turbulent landscape of information technology. Computing in the "cloud" alludes to ubiquitous and inexhaustible on-demand IT resources accessible through the Internet. Practically every new Internet-based service from Gmail [1] to Amazon Web Services [2] to Microsoft Online Services [3] to even Facebook [4] have been labeled "cloud" offerings, either officially or externally.

Although cloud computing has garnered significant interest, factors such as unclear terminology, non-existent product "paper launches", and opportunistic marketing have led to a significant lack of clarity surrounding discussions of cloud computing technology and products. The need for clarity is well-recognized within the industry [5] and by industry observers [6]. Perhaps more importantly, due to the relative infancy of the industry, currently-available product offerings are not standardized. Neither providers nor potential consumers really know what a "good" cloud computing product offering should look like and what classes of products are appropriate. Consequently, products are not easily comparable.

The scope of various product offerings differ and overlap in complicated ways – for example, Amazon's EC2 service [7] and Google's App Engine [8] partially overlap in scope and applicability. EC2 is more flexible but also lower-level, while App Engine subsumes some functionality in Amazon Web Services suite of offerings [2] external to EC2. On balance, App Engine incorporates that functionality at the expense of being less general-purpose and more tightly integrated. Differences such as these make comparisons between products difficult; it is unclear how the major products fit into the greater universe of offerings and what axes of comparison are appropriate.

The goal of this study is to perform a detailed survey of different offerings to classify and clarify the commonalities and differences along various product dimensions. The objects of analysis in this work are *products* and not their component technologies in a vacuum. Clarifying the relationship of the various cloud computing products will allow both consumers and service providers to assess their current and planned future offerings in light of their desired properties and marketplace positioning.

Scope: Since the landscape of cloud computing is so broad and nebulous, this work's scope is explicitly limited to lower-level offerings. In particular, "infrastructure" or "platform"-level services as well as relevant supporting services – this will include offerings like Amazon's EC2 [7] or Google's App Engine [8]. It will not cover application-level offerings such as GMail [1], Google Docs [9], or Microsoft Online Services [3]. As the industry is awash in startups and announced-but-not-delivered "vaporware", this analysis will not attempt to be exhaustive; instead it will cover the most important players with notable offerings.

Contributions: Broadly, this paper's contributions are as follows: 1) an analysis of cloud computing's value proposition and initial impact (Section 2), 2) a summary of major, directly-competing lower-level cloud products (Section 3), and 3) an in-depth analysis and taxonomy of products on identified axes of comparison (Section 4).

Roadmap: The structure of the remainder of this paper is as follows:

- Section 2 presents general cloud computing background. In particular, it covers the following major subtopics:

- General Background – This section explains why cloud computing is now feasible and the technological trends leading to its ascendancy.
- Business Context (Section 2.1) – This section discusses cloud's value proposition on both the supply and demand side.
- Business Cases (Section 2.2) – This section analyzes the experiences of some early adopters.
- Types of Products (Section 2.3) – This section summarizes the classification schemes the industry is using and the basic product distinctions.
- Major Players (Section 2.4) – This section briefly summarizes the biggest cloud players; it provides context by explaining how these companies entered the cloud computing field and how it fits their existing business competencies.

- Section 3 summarizes the major cloud product offerings used in this analysis.
- Section 4 identifies dimensions of comparison between cloud products and analyzes the products listed in Section 3 in context.
- Section 5 concludes.

2 Background

"Cloud Computing" is a broad but fledgling area spurred by several important technological trends. The most important trends underlying all cloud offerings are 1) ubiquitous network connectivity and 2) virtualization. Connectivity is critical for the viability of the cloud model because services are delivered and accessed over the Internet. *Virtualization* is technique of abstraction applied at different layers of the computing hardware or software stack, decoupling resources' interfaces from underlying low-level details. Applied properly, it can allow seamless scaling of resources with changes in demand. It also enables more efficient sharing of surplus capacity that might otherwise be underutilized. The abstraction afforded by virtualized services means resource users do not need to be aware of sharing or cooperate explicitly.

To understand virtualization in a non-technical context, consider teller lines at a bank: while some banks have independent lines per teller window, others provide a common feeder line. The latter is an example of virtualization. From a customer's perspective, there is a single line; behind the scenes, the bank can increase or decrease capacity to meet demand by adding or removing tellers. Furthermore, in the un-virtualized example where separate tellers are directly exposed to customers, lowering capacity by removing a teller is a more complex operation because customers may already be enqueued in a particular line. Virtualization abstracts away the number of tellers from the customers, creating one virtual teller line.

Virtualization is one enabler of *utility computing*, a service model of computing where computational resources (such as CPU time, storage or network bandwidth) are sold as on-demand and metered resources, like public utilities. The key difference (and benefit) of modern cloud computing over older managed server rental/hosting models is the billing regime. With utility-style computing, one may rent 1,000 servers for one hour; the resources do not have to be reserved in monthly contracts, and the user only pays for what he uses. In this manner, resource usage can simply scale and contract to meet current demand.

2.1 Business Context

The business and economical reasons for cloud computing are a direct result of various advantages enjoyed by Internet technology heavyweights such as Google, Amazon, Microsoft, IBM and others. On the provider side, the first and most obvious advantage is massive economies of scale. A recent research report on cloud computing by Berkeley's RADLab titled "Above the Clouds: A Berkeley View of Cloud Computing" [10] cited the figures in Table 1 to illustrate the magnitude of cost advantage enjoyed by the largest players – network, storage and administration costs can be better by a factor of five or more. The second advantage, which is a result of both scale and success, is significant expertise in managing and deploying large-scale IT services. BusinessWeek compared Amazon's cloud offerings to Wal-Mart's supply-chain prowess: "It's as if [Wal-Mart] had decided to turn itself inside out, offering its industry-leading supply chain and logistics systems to any and all outsiders, even rival retailers" [11]. By offering cloud services, these providers can partially subsidize their existing investments in infrastructure and expertise used for their core businesses and directly profit from excess capacity.

On the consumer side, the benefits are obvious. A painful reality of running IT services is the fact that peak demand is often an order of magnitude above average demand. Adam Selipsky, Vice President of Amazon Web Services, notes [12] that 10 to 15% average utilization of IT resources is common in enterprises [13]. This massive over-provisioning is capital-intensive and wasteful, but cloud computing can allow seamless scaling with demand changes. In this way, cloud computing is analogous to flexible or reactive capacity in manufacturing situations. Figure 1 illustrates the cost of over-provisioning versus the ideal of dynamically scaling capacity to meet demand. Furthermore, scaling services up is not a simple matter of acquiring and configuring extra hardware and software: the process often requires significant re-engineering at all levels of the software and hardware stack. Just like a bike shed design cannot be trivially scaled to produce a stable structure the size of a shopping mall, moving from 50 to 50,000 servers requires fundamental changes in administration, network design and software design. Such expertise is specialized and expensive and diverts attention from core competencies.

In a talk titled "A Head in the Cloud: The Power of Infrastructure as a Service" [14], Werner Vogels (CTO of Amazon.com) gives an illustrative example why the cost of running these services is so high: he notes that developers of high-scale services spend 70% of their time on generic service delivery and scaling problems and only 30% on actual differentiated product functionality. Given these numbers, it makes sense to outsource expertise to companies like Amazon and Google. He states that Amazon's cloud computing value proposition is allowing companies to address uncertainty by 1) acquiring resources on demand, 2) releasing resources when unneeded, 3) paying for use rather than provisioning, 4) leveraging others' core competencies and, 5) turning fixed capital investments into variable costs [14]. Russ Daniels of HP (Vice President and CTO of Cloud Services Strategy) analogizes Vogel's last point about eliminating costly capital investment as follows: "It's like renting a car. I commute back and forth from San Francisco on a train, and I used to keep a car here so that I could commute to Cupertino when I needed to, but it's really quite expensive, so I started using ZipCar. Now if I need to go to Cupertino I can arrange to rent a car for three hours and pay $8 an hour, but if you compare that to what it would cost me to have a car fulltime, it's cheap" [15].

Ultimately, most companies face a trade-off between cloud computing services and the level of internally maintained infrastructure. The Economist describes this decision as a "trade-off between sovereignty and efficiency," [16], but also notes that IT infrastructure growth trends are unsustainable for a majority of businesses [17].[1] Certainly many established businesses will want to maintain some internal IT infrastructure for sensitive data and redundancy. Startups and companies whose core com-

[1] For more about the massive costs of running large datacenters, see Section 2 in "The Cost of a Cloud: Research Problems in Data Center Networks" [18].

petencies are not in high technology are two areas where cloud computing is already making an impact [19, 20]. Startups do not have the capital to invest in significant infrastructure; they can also be much more nimble by starting with contracted services. Fundamentally, cloud computing lowers the barriers to entry for running Internet-facing services. For example, in the space of 48 hours (after adding a new feature), Animoto scaled from 50 servers to more than 3500 based on dynamic demand [14]. See Section 2.2 for a series of examples illustrating the value proposition of cloud services.

Although many early success stories do show the significant promise of cloud computing, with the exception of a few startups and independent projects, the uptake is not significant. Most established businesses using cloud computing are cautiously running test projects and waiting for more clarity, "next generation" products and more industry maturity.

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 1: Cloud economies of scale
Measured in 2006 for medium-sized datacenter (1000 servers) vs. very large datacenter (50,000 servers) [10]

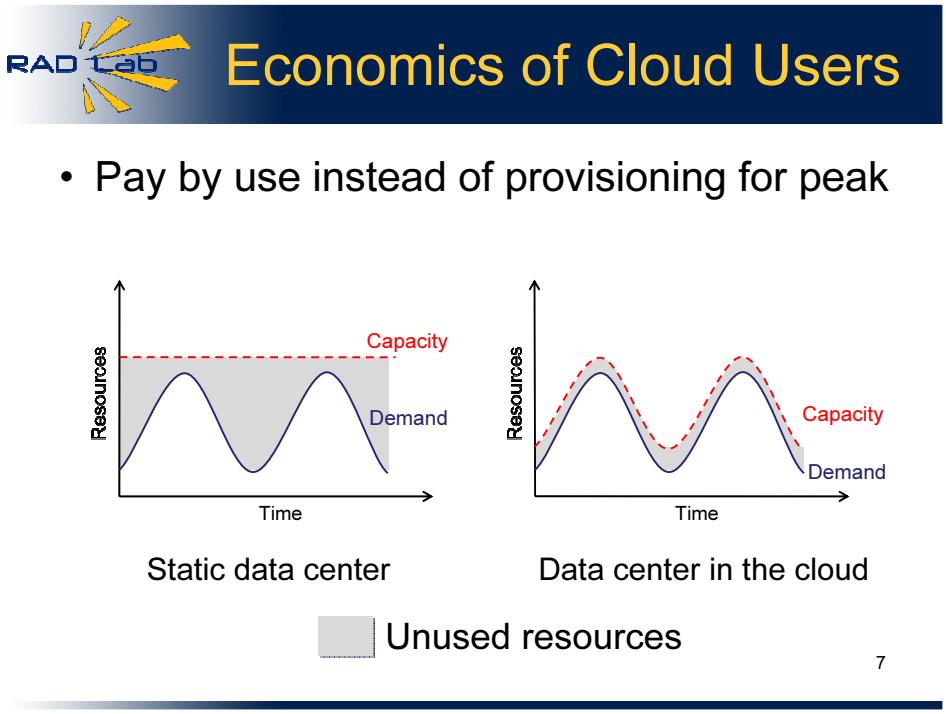

Figure 1: RADLab Presentation: Cloud Value Proposition
From RADLab Presentation "Above the Clouds: A Berkeley View of Cloud Computing" [10, 21]

2.2 Cases

This section contains a series of illustrative examples highlighting current uses of cloud computing in both established businesses and startups. These cases are meant to demonstrate the value proposition

of viable cloud computing offerings and the benefits companies have derived from such services.

New York Times: One of the most cited examples of cloud computing's promise comes from the New York Times [20]. The New York Times has a large collection of high-resolution scanned images of its historical newspapers, spanning 1851-1922. They wanted to process this set of images into individual articles in PDF format. Using 100 EC2 instances, they were able to finish the processing in 24 hours for a total cost of $890 ($240 for EC2 computing time and $650 for S3 data transfer and storage usage, storing and transferring 4.0TB of source images and 1.5TB of output PDFs) [22]. Derek Gottfrid noted, "In fact, it work so well that we ran it twice, since after we were done we noticed an error in the PDFs" [23]. The New York Times was able to use 100 servers for 24 hours at the low standard price of ten cents an hour per server.

If the New York times had purchased even a single server for this task, the likely cost would have exceeded the $890 for just the hardware, and they also need to consider the cost of administration, power and cooling. In addition, the processing would have taken over three months with one server. If the New York Times had purchased four servers, as Derek Gottfrid had considered [23], it would have still taken nearly a month of computation time. The rapid turnaround time (fast enough to run the job twice) and vastly lower cost strongly illustrates the superior value of cloud services.

Washington Post: In a similar but more recent story, the Washington Post was able to convert 17,481 pages of scanned document images into a searchable database in about a day using Amazon EC2 [24]. On March 19th at 10am, Hillary Clinton's official White House schedule from 1993-2001 was released to the public as a large collection of scanned images (in PDF format, but non-searchable). Washington Post engineer Peter Harkins used 200 Amazon EC2 instances to perform OCR (Optical Character Recognition) on the scanned files to create searchable text – "I used 1,407 hours of virtual machine time for a final expense of $144.62. We consider it a successful proof of concept."

DISA: Federal Computer Week reported that the Defense Information Systems Agency (DISA) compared the cost of using Amazon EC2 versus internally maintained servers [25]: "In a recent test, the Defense Information Systems Agency compared the cost of developing a simple application called the Tech Early Bird on $30,000 worth of in-house servers and software with the costs of developing the same application using the Amazon Elastic Compute Cloud from Amazon.com's Web Services. Amazon charged 10 cents an hour for the service, and DISA paid a total of $5 to develop an application that matched the performance of the in-house application. "

SmugMug: SmugMug [26], a photo sharing and hosting site like Flickr [27], stores a significant amount of its photo data in Amazon's S3 cloud storage service [28]. In 2006, they saved "$500,000 in planned disk drive expenditures in 2006 and cut its disk storage array costs in half" by using Amazon S3 [28]. According to the CEO of SmugMug, they could "easily save more than $1 million" in the subsequent year by using S3 [29]. The CEO noted that their current growth rate at the time of the article requires about $80,000 worth of new hardware per month, and the monthly costs increase even more significantly after adding "power, cooling, the data center space, and the manpower needed to maintain them." In contrast, Amazon S3 costs around $23,000 a month for equivalent storage and is all-inclusive (power, maintenance, cooling, etc. are all figured in to the cost of the storage service) [29].

Eli Lily: Eli Lily, one of the largest pharmaceutical companies, is starting to utilize Amazon's storage and compute clouds to provide on-demand high-performance computing for research purposes[2] [30]. John Foley points out, "it used to take Eli Lilly seven and a half weeks to deploy a server internally" whereas Amazon can provision a virtual server in three minutes. In addition "a 64-node Linux cluster can be online in five minutes (compared with three months internally)." Amazon's cloud services not

[2]Biomedical simulation software is a common application of high-performance computing (HPC).

only provide on-demand scaling and usage-based billing, they also allow Eli Lily to react with significantly increased agility, cutting out time-consuming equipment acquisition and deployment processes.

Best Buy's Giftag: Best Buy's Giftag [31] is a new online wish-list service hosted using Google's App Engine [32]. In a video interview, the developers indicated that they started developing the site with another technology and switched to Google App Engine for its superior development speed and scaling benefits. As one developer succinctly stated, "a lot of the work that none of us really want to do is [already] done for us." The developers also praised App Engine's design for enabling effortless scaling; web applications built on App Engine inherit Google's best-in-class technologies and experience operating large scale websites. Ultimately, App Engine allows developers to concentrate on creating site-specific differentiated functionality: "that worry about the operational aspects of an application going away really frees you up for writing great code or testing your code better" [32].

TC3: TC3 (Total Claims Capture & Control) is a healthcare services company providing claims management solutions. TC3 now uses Amazon's cloud services to enable on-demand scaling of resources and lower infrastructure costs [33]. TC3's CTO notes, "we are utilizing Amazon S3, EC2, and SQS to enable our claim processing system capacity to increase and decrease as required to satisfy our service level agreements (SLAs). There are times we require massive amounts of computing resources that far exceed our system capacities and when these situations occurred in the past our natural reaction was to call our hardware vendor for a quote. Now, by using AWS products, we can dramatically reduce our processing time from weeks or months down to days or hours and pay less than purchasing, housing and maintaining the servers ourselves" [33]. One unique aspect of TC3's operations is the fact that, since they handle US healthcare-related services, they are required to comply with HIPPA (Health Insurance Portability and Accountability Act). Regulatory compliance is one current major obstacle facing corporate adoption of cloud computing – the fact that TC3 is able to comply with HIPPA on Amazon's services is promising.

2.3 Types of Products

Cloud computing products are often roughly classified into a hierarchy of *-as a service* terms, presented here[3] in order of increasing specialization:

Infrastructure as a Service (IaaS) is providing general on-demand computing resources such as virtualized servers or various forms of storage (block, key/value, database, etc.) as metered resources. Sometimes called *Hardware as a Service (HaaS)*. This can often be seen as a direct evolution of *shared hosting* with added on-demand scaling via resource virtualization and use-based billing.

Platform as a Service (PaaS) is providing an existent managed higher-level software infrastructure for building particular classes of applications and services. The platform includes the use of underlying computing resources, typically billed similar to IaaS products, although the infrastructure is abstracted away below the platform.

Software as a Service (SaaS) is providing specific, already-created applications as fully or partially remote services. Sometimes it is in the form of web-based applications and other times it consists of standard non-remote applications with Internet-based storage or other network interactions.

As mentioned earlier, this report will focus on the infrastructure- and platform-level offerings since they are lower-level building blocks for companies. Software as a service – implying specific applications delivered in a non-traditional way – is quite different from lower-level offerings. Fundamentally, PaaS

[3] An entire glossary (with additional terms) is provided in Appendix A for reference.

and IaaS can be used as flexible substrates for a wide range of applications, including software-as-a-service applications.

No two experts (or laymen) seem to agree on the precise categories in cloud computing. In an introductory article on cloud computing on IBM DeveloperWorks [34], M. Tim Jones presents one such hierarchy with examples in Figure 2. The data-Storage-as-a-Service category (dSaaS) is not widely used as a separate term; in this report we will consider some data storage services, such as Amazon S3 or Amazon SimpleDB, but we will stick to the IaaS and PaaS rough classifications.

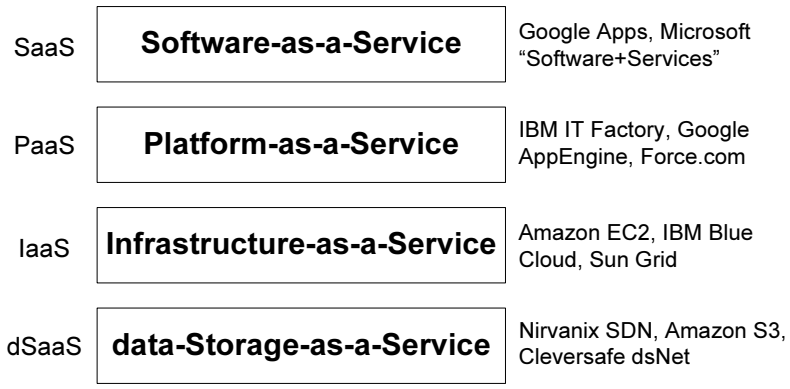

Figure 2: A Hierarchy of Cloud Offerings
From "Cloud Computing with Linux" [34].

Figure 3 shows Wikipedia's cloud hierarchy, which layers *storage* above *infrastructure*, and decomposes software into several distinct sub-layers. Figure 4 shows a cloud stack shown in a short "lightning presentation" given at Cloud Camp Seattle 2009 [35].

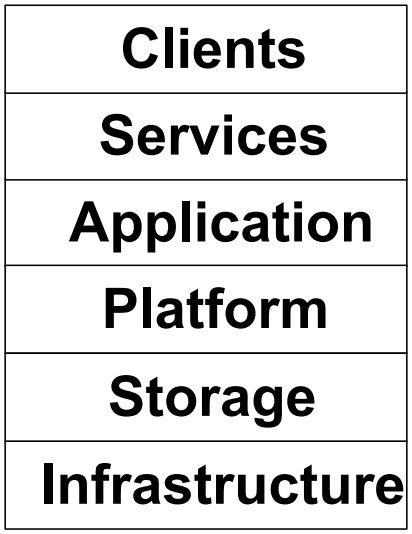

Figure 3: Wikipedia's Hierarchy of Cloud Offerings
From Wikipedia's Cloud Computing article: `http://en.wikipedia.org/wiki/Cloud_computing`

In reality, the lines are fuzzy and solutions actually exist on a continuum from low-level building blocks to prefabricated applications. The line between infrastructure-level and platform-level offerings is often particularly nebulous and a matter of taste. Additionally, some services are higher-level in certain

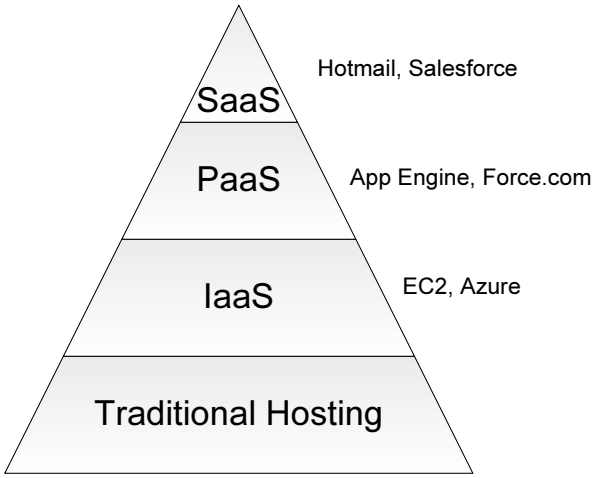

Figure 4: Cloud Camp Seattle Hierarchy of Cloud Offerings
From a lightning talk at Cloud Camp Seattle [36]

ways while being lower-level in others. In Berkeley's "Above the Clouds Report", Armbrust et al. [10] state, "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." Here we will use the *as-a-service* classifications roughly; part of the impetus behind this report is to clarify differences between products by more finely decomposing analysis into different traits. We identify and discuss these traits separately in Section 4.

Armbrust et al. [10] also differentiate *public clouds* and *private clouds*. *Public clouds* refer to the cloud-computing products which are the focus of this paper, while *private clouds* "refer to internal datacenters of a business or other organization." Many so-called "cloud computing" products are actually for internal use[4], but we will not consider them here.

2.4 Major Players

As an promising industry on the cusp of high growth, cloud computing is attracting many potential entrants. The following is a short overview of several representative major early industry players; the summaries provide context for the current state of the burgeoning industry by explaining how these companies entered the cloud computing field and how it fits their existing business competencies. Future entrants will likely share common rationales and similarly-positioned companies should be watching for potential business opportunities in the growing cloud computing industry.

2.4.1 Amazon.com

Amazon.com is a quintessential brand name in electronic commerce. In the decade since its founding, it has transformed once from an online book-seller to a general retail platform and again to the industry leading cloud computing provider. Starting in 2006 with its Simple Storage Service (S3) [37] and Elastic Compute Cloud (EC2) [7], Amazon has been a trailblazer in IaaS cloud offerings. Amazon's CTO Werner Vogels is a deep technical mind, holding a Ph.D. in Computer Science; Vogels is instrumental in directing Amazon's cloud strategy and often serves as the public face for it.

[4] or use by a cloud service provider in internal operations

Amazon has been praised for their cloud strategy and is considered the industry leader in the emerging market – in 2008, Vogels was awarded "Chief of the Year" by InformationWeek magazine [38] and accepted "Best Enterprise Startup" award from Tech Crunch's "Crunchies" [39] on behalf of Amazon. If cloud computing takes off as an industry, many observers predict Amazon's future retail revenues will be dwarfed by their cloud offerings. ZDNet's Larry Dignan quipped, "Amazon will be like a book store that sells cocaine out the back door. Books will be just a front to sell storage and cloud computing" [40].

2.4.2 Google

Google is the most important and most watched Internet company today. Search engine rankings indicate Google's market share is nearly two-thirds of the total search market, versus approximately 20% for Yahoo and 10% or less for Microsoft, the next largest competitors [41]. With its expertise running the world's most popular search engine and its vast, industry-leading infrastructure to support the world's most visited Internet site [42], expanding into cloud computing services is a natural fit.

While its search dominance presently appears bulletproof, Google's search prowess does not directly translate to automatic cloud computing leadership. So far, compared to Microsoft and Amazon, Google's single App Engine [8] offering is less ambitious and more limited in scope; additionally, external reception has been more reserved [43]. Although it has been met with early skepticism, Google is standing behind App Engine and continuously improving it. Their public road-map and feature updates may make it more attractive if they deliver their planned functionality [44].

2.4.3 IBM

In the computing industry, IBM has developed a sterling reputation for dependability – their stature is condensed into the aphorism, "nobody ever got fired for buying IBM." IBM has sustained product lines which predate the founding of the other companies in this section; this remarkable stability is a huge asset. After a tumultuous period in the early 90s [45], IBM refocused its core business from hardware-centric to software and service-centric offerings [46]. Despite IBM's technical pedigree, it is not an Internet hosting heavyweight like Google, Amazon, Yahoo or even Microsoft. Consequently, IBM's entry into cloud computing is more focused toward its core competencies and general focus on consulting.

IBM's "Blue Cloud" effort [47] has received significant press attention, but it is not clear from IBM's publicly available information what exactly Blue Cloud is [48]. James Staten, an analyst at Forrester research, contacted IBM directly to clear up this confusion. He summarizes Blue Cloud as follows:

> IBM's BlueCloud initiative isn't (at least not initially) an attempt to become a cloud services provider or to become a cloud computing platform, but rather to help their customers experiment with, try out, and custom design cloud solutions to fit their needs. Building off the IBM Innovation Center concept, IBM is providing Cloud centers that are places customers from enterprise and government accounts, as well as non-IBM customers can test out cloud computing concepts, mostly for deployment internal to their own data centers. Gerrit Huizenga, the technical solutions architect for BlueCloud for IBM's Systems & Technology Group (STG) said these efforts are helping them build out a series of cloud blueprints, or proven/standardized cloud infrastructures. "Our goal is to deliver solutions that make it much easier to deploy and manage these things," Huizenga said. [48]

In addition to Blue Cloud, IBM has forged partnerships with other cloud providers, such as Amazon [49], to offer its software and tools in applications hosted on other providers' clouds.

2.4.4 Microsoft

Microsoft is a strong business software vendor, but their Internet service efforts have largely been eclipsed by competitors such as Google and Yahoo [41]. Despite its inability to displace current Internet service market leaders, Microsoft has significant infrastructure and operational experience to run large Internet-facing services. Although entering cloud computing is an attractive proposition from the narrow perspective of utilizing existing capital investments, Microsoft is understandably concerned with the potential for cloud computing to cannibalize its other core businesses (i.e. traditional computer software and operating systems).

As usual, Microsoft's efforts in this new area are partially defensive and many observers were skeptical of Microsoft ever fully entering infrastructure and platform-level cloud offerings. Microsoft's initial foray into cloud computing was in the form of software-as-a-service offerings [50]. Microsoft started with subscription-based versions of existing Microsoft office and productivity products and moved into value-added online components [51]. The company's recent announcement of the Azure platform signals their full entry into lower-level cloud services. As noted in the "Above the Clouds" Berkeley presentation slides, Microsoft has carefully constructed Azure to complement their existing profitable software business. Azure partially exists to sell .NET developer tools [21] as well as Microsoft's own related operating systems and services. This is in contrast to Amazon's and Google's approaches, which do not currently monetize cloud application development, just hosting.[5]

2.4.5 Salesforce

Salesforce [52] started out as a Customer Relationship Management (CRM) software provider; founded in 1999, Salesforce is now an S&P 500 company and one of the top 50 largest software companies by software revenue [53]. Although Salesforce was an early pioneer in software-as-a-service [54], their Force.com platform-as-a-service launch in 2007 [55] put them in the business of lower-level cloud offerings.

Salesforce describes their Force.com product as a platform-as-a-service offering, and it provides a higher-level web application framework (and auxiliary services) to construct certain business-oriented software-as-a-service products hosted on Salesforce's cloud. Salesforce is somewhat unique in their target market, as most other major cloud offerings are not domain-specialized in the same way as Force.com. This narrower focus, combined with Salesforce's existing CRM prowess, may provide a unique advantage. SaaS applications constructed using Force.com's platform will integrate with Salesforce's existing popular CRM offerings. Salesforce is the only major cloud computing player to enter the market by generalizing an existing software-as-a-service product.

2.4.6 Rackspace

Rackspace is a major managed hosting provider. According to Gartner, Rackspace ranked third in managed hosting market share at the time of their mid-2008 IPO (only smaller than IBM and AT&T) [56]. Managed hosting providers typically have significant investments in datacenters, server hardware, IT management staff and Internet connectivity. Most providers like Rackspace offer standard services such as *colocation*, *dedicated hosting*, *virtual private servers* and *shared web hosting* as well as a variety of other custom services. With colocation, customer-provided server hardware is installed in server racks and powered and maintained by hosting staff, but the customer can install arbitrary software and handle software configuration. Dedicated hosting is similar, but the server hardware is rented from the hosting company. Virtual private server hosting rents virtualized dedicated servers. In

[5] Amazon does sell premium support and consulting services to help developers if necessary, but such services are optional.

these arrangements, several virtual servers are multiplexed on top of a single shared physical server; to each client, however, the virtual server looks like a dedicated physical machine (with potentially lower performance, due to sharing). In all cases, the client can install arbitrary software. With shared web hosting, the hosting provider installs and manages the operating system/server software and several clients may share one physical machine; each client can run certain kinds of websites, but does not have arbitrary control of the server software.

Since managed hosting providers have significant investments in datacenters and related hardware and staff, cloud computing is a natural fit. Additionally, cloud computing is a direct threat to their existing business model, providing more flexibility and often better prices. Cloud computing providers will increasingly poach customers from dedicated or shared hosting providers; why would companies rent servers by the month when they can rent them as needed by the hour for a similarly competitive rate? Rackspace's entry into cloud computing is as much a defensive move as it is a natural use of their existing infrastructure. Managed hosting providers must evolve or die.

3 Products

The following sections briefly summarize a variety of significant cloud computing offerings. These products will be used in the subsequent discussion of service comparison criteria (Section 4).

3.1 Amazon AWS Umbrella

Amazon's cloud offerings fall under a group of complementary products called "Amazon Web Services" [2]. Amazon lists the following as its infrastructure-level services:

- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon SimpleDB
- Amazon Simple Storage Service (Amazon S3)
- Amazon CloudFront
- Amazon Simple Queue Service (Amazon SQS)
- AWS Premium Support

In addition, EC2 has an optional, supplemental service called Amazon Elastic Block Store (EBS) [57].

Elastic Compute Cloud (EC2): EC2 is Amazon's flagship cloud offering. EC2 allows the metered, on-demand rental of virtual machine computing resources. EC2 is rented in units called *instances*, each of which represents a virtual server with particular hardware specifications. From a user's perspective, it is like renting physical servers by the hour in any quantity. There are five differentiated types of instances to rent with varying CPU power, memory, hard disk space and IO performance [58]. An application needing a significant amount of RAM or CPU performance can rent more expensive but more powerful instances, while a network-bound application, like a web-server, can use cheaper and less powerful instances.

While EC2 provides metered computing facilities with temporary local storage, three Amazon products provide metered permanent storage facilities: the Elastic Block Store (EBS), the Simple Storage Service (S3) and SimpleDB.

Elastic Block Store (EBS): Elastic Block Store works in conjunction with EC2 to provide extra high-performance, persistent storage to EC2 virtual machine instances. EC2 instances have local storage capacity, but such space is temporary and only available while an instance continues to run. EBS provides storage like a virtual disk (*block storage*) which can be attached to a given EC2 instance; the data will stay available independent of the EC2 instances currently running and can be moved from instance to instance instance without the need to explicitly build some sort of higher-level data transfer mechanism.

Simple Storage Service (S3): S3 was Amazon's first infrastructure-level web service, launched in early 2006. S3 provides robust *object storage* metered per gigabyte per month. While EBS provides a virtual disk-like *block storage* abstraction to attach to EC2 virtual machine instances, S3 provides a storage facility which can be accessed independent of EC2 instances. One can use S3 by itself as a storage repository without using EC2; one can also have many EC2 instances accessing the same data from S3. Fundamentally, the interface to storage is different – while *block storage* acts like a disk, *object storage* provides a higher-level interaction paradigm. Discrete objects (which are similar to files) are stored and retrieved by name.

SimpleDB: SimpleDB is a pseudo-relational data storage service. It stores data much like a relational database management system (RDBMS), providing a richer data query and manipulation interface than block or object storage. SimpleDB is also accessible independent of EC2 instances and presents higher-level database-like storage accessed using a SQL-like query language.

CloudFront: CloudFront is Amazon's newest service, introduced in November 2008 [59]. CloudFront is a *Content Delivery Network* (CDN) which works with data stored in S3. A CDN is designed to enhance the delivery of data (content) to data consumers (customers / end users) by providing closer "edge locations" for distribution. By providing many different edge locations, a content provider can provide end users with lower delivery latency and better performance. A CDN roughly analogous to strategically-located regional distribution centers in supply chain logistics. With CloudFront's launch, Amazon is now poised to compete with established CDN businesses [60] like Akamai [61] and Limelight Networks [62].

Simple Queue Service (SQS): Amazon's Simple Queue Service provides reliable messaging between distributed software components. It is often used in conjunction with EC2 to coordinate the actions of different instances or distinct components of a bigger application running on EC2.

Amazon provides the following example to show how their different services might fit together in a cloud-based application:

> For example, here is how a video transcoding website uses Amazon EC2, Amazon SQS, Amazon S3, and Amazon SimpleDB together. End users submit videos to be transcoded to the website. The videos are stored in Amazon S3, and a message ("the request message") is placed in an Amazon SQS queue ("the incoming queue") with a pointer to the video and to the target video format in the message. The transcoding engine, running on a set of Amazon EC2 instances, reads the request message from the incoming queue, retrieves the video from Amazon S3 using the pointer, and transcodes the video into the target format. The converted video is put back into Amazon S3 and another message ("the response message") is placed in another Amazon SQS queue ("the outgoing queue") with a pointer to the converted video. At the same time, metadata about the video (e.g., format, date created and length) can be indexed into Amazon SimpleDB for easy querying. During this whole workflow, a dedicated Amazon EC2 instance can constantly monitor the incoming queue and, based on the number of messages in the incoming queue, is able to dynamically

adjust the number of transcoding Amazon EC2 instances to meet customers' response time requirements. [63]

AWS Premium Support: AWS Premium Support is not a technical product offering itself; it is paid support and consulting related to Amazon's cloud services. Amazon will provide help with both operational support and technical issues related to software development using their cloud services.

These examples of infrastructure-level offerings are the lowest-level cloud offerings but also very general-purpose. Amazon also has a variety of higher-level, more application-specific platform-level offerings, mostly centered around merchant interactions. For instance, Amazon provides a Flexible Payments Service (FPS), which allows merchants to process payments using Amazon's existing payment system.

3.2 Google App Engine

Google's App Engine is one of the most prominent examples a platform-as-a-service cloud offering. App Engine provides a single, pre-built solution for constructing very large scale web-based applications hosted on Google's infrastructure. Unlike Amazon's array of individual components which are composed to build customized solutions, App Engine is a single integrated solution. For this reason, App Engine is typically classified as a platform-level offering rather than an infrastructure-level offering – it is a higher-level but less general solution. If your desired application fits App Engine's target model, it requires less upfront work because more of the solution software stack is already built; however, many potential applications simply do not match App Engine's pre-built platform well enough to consider it as a substrate. Although App Engine has not seen as wide uptake as Amazon's services, it is newer and has only been out of test phases for about a year [64]. Early users include Best Buy (for their Gifttag [31] service) and the White House website, WhiteHouse.gov [65]. The White House website itself does not run on App Engine, but App Engine-based Google Moderator is used to handle high-volume voting during online town hall meetings. During the first such event, the voting site handled up to 700 hits per second over a 48 hour period, serving one million unique users [65] – these impressive numbers highlight App Engine's scalability.

3.3 Microsoft Azure Services Platform

Like Amazon's offerings, Microsoft's Azure Services Platform consists of several components:

- Live Services
- SQL Services
- .NET Services
- SharePoint Services
- Dynamics CRM Services

Figure 5 shows Microsoft's diagram of the Azure Services Platform components and how they fit into Microsoft's product landscape. Although Microsoft's Azure Services Platform is still in limited release and some details are unclear, whitepapers [66] and industry observers' analysis [67] have shed some light on specifics.

Figure 5: Microsoft's Azure Services Platform
From "What is the Azure Services Platform?" [68].

The basic (foundational) Azure platform allows users to run managed code in a virtual machine on Microsoft-hosted and maintained servers. Users have to choose *Web* or *Worker* roles for application instances: *Web* roles are appropriate for hosted applications interacting with the outside world via the network, while *Worker* roles are appropriate for code that simply performs processing. The basic Azure platform also provides storage in three forms: 1) Blobs, 2) Tables and 3) Queues. Blob storage is similar to Amazon's S3; table storage is similar to Amazon's SimpleDB and queue storage is similar to Amazon's SQS.

In addition to the basic Azure platform, the additional services include the following:

SQL Services: SQL Data Services allow customers to host database-like storage to the cloud. The software is based on Microsoft's relational database management system, SQL Server, but it exposes a slightly different interface than the common relational database. This service is similar to Amazon's SimpleDB.

.NET Services: .NET services includes three components: the 1) Access Control Service, 2) Service Bus and 3) Workflow Service. These are auxiliary services used to construct complex applications using Azure.

Live Services: Live services include a variety of component services common to Microsoft's ecosystem of "Live" branded services [69], such as MSN Hotmail [70], Live Messenger [71], Live Search [72] and others. For instance, an application could access common identity information associated with a user's Live account.

Both SharePoint and Dynamics CRM services are larger, domain-specific components from existing Microsoft software from which developers can utilize functionality.

3.4 IBM Blue Cloud

IBM's Blue Cloud is not a commercially available cloud hosting service, but rather part of a consulting strategy [48]. Since it does not provide infrastructure or platform-level on-demand metered computing resources for sale, we will not consider it here in our comparison.

3.5 ServePath GoGrid

ServePath's GoGrid service [73] is an infrastructure-level offering substantially similar to Amazon's EC2 service. ServePath also offers a complementary Cloud Storage product [74] – their storage product is like having a remote network filesystem or file-server (*file storage*), so the storage paradigm is unlike any of Amazon's offerings.

3.6 XCalibre FlexiScale

Flexiscale [75] is also similar to GoGrid and EC2. Their cloud storage is a virtual dedicated disk abstraction similar to Amazon's Elastic Block Store (EBS). They also state "we can offer NFS shares on the same high-end storage back-end" [76], which would be similar to GoGrid's storage paradigm.

3.7 Sun Grid Compute Utility

Sun's Grid Compute Utility [77, 78] was one of the earliest cloud-like services. Announced in 2005, the now-defunct [79] infrastructure-level service was somewhat similar to Amazon's EC2, but with more limitations. Rather than run arbitrary software in virtual machines, Sun's Grid ran self-contained Solaris 10 applications for $1 per CPU-hour. Due to its architecture, it was mostly used for *batch*-style computation – batch computation involves large-scale, long non-interactive processing of data such as mathematical simulations, 3D rendering and various bulk data processing. For instance, the 3D animated film "Big Buck Bunny" was rendered on Sun's Grid [80]. In contrast, running a website requires interaction, as the web server must respond to incoming requests in a timely fashion.

According to the old Sun Grid website [77], Sun is planning on revamping its offerings to reflect the current state of cloud computing. In March 2009, Sun announced plans to unveil its "Open Cloud Platform" to compete with Amazon's EC2 and S3 offerings [81].

3.8 Mosso: The Rackspace Cloud

Rackspace started as a web hosting company and now offers several cloud-computing style products under the *Mosso* division [82]. Mosso offers three basic products: 1) Cloud Sites, 2) Cloud Files and 3) Cloud Servers. Cloud Sites [83] is best classified as a platform-as-a-service offering, because it provides pre-configured supported web application stacks like a shared web hosting provider, but with the billing regime and scaling of cloud offerings. Cloud Files [84] is a cloud storage service similar to Amazon's S3. Mosso also offers a CDN option [85] with Cloud Files which is analogous to Amazon's CloudFront. Unlike Amazon, however, Mosso does not operate their own CDN; instead they have partnered with Limelight Networks [62]. Cloud Servers [86] is an Amazon EC2 competitor.

3.9 Engine Yard Solo

Engine Yard Solo [87], introduced in January 2009, is a managed Ruby on Rails [88] platform offering hosted on top of Amazon EC2 service. Essentially it is a layered product creating a more specific and higher-level platform-as-a-service offering on top of Amazon's infrastructure-level cloud offering.

3.10 Heroku

Heroku [89] is another managed Ruby on Rails [88] web hosting platform hosted in Amazon's EC2 compute cloud.

3.11 Salesforce force.com

Salesforce [52] is a Customer Relationship Management (CRM) software vendor, delivering its software as a online service (SaaS). Force.com is a unique platform-as-a-service offering allowing vendors to create business applications delivered on Salesforce's existing infrastructure. Salesforce cites "enterprise resource planning (ERP), human resource management (HRM) and supply chain management (SCM)" as target force.com application domains [90]. This makes force.com relatively specialized among the cloud computing offerings detailed here, as it is more domain-specific and not targeted towards typical general web applications like Google App Engine or other similar platforms.

3.12 Bungee Connect

Like Force.com, Bungee Connect [91] provides a higher-level platform-as-a-service offering to develop and deploy web-based business applications. Bungee can host applications on the Bungee Grid and uses Amazon EC2 for extra flexible capacity. Part of Bungee's value proposition is extending into the application development process by providing a high-level development environment delivered as a web-based service.

3.13 ENKI

ENKI [92] offers cloud-like virtual server hosting with daily usage-based billing [93].

3.14 Cloud-like Hosting

Many web-hosting or *virtual private server* hosting companies describe their offerings as cloud computing or use "cloud" terminology for marketing purposes. Although there exists a continuum of products, the primary distinction between "proper" cloud services and these hosting packages is the billing granularity: utility computing offerings are billed on demand and allow quick scaling. The non-cloud services require monthly rental of virtual server instances and closer to traditional hosting.

For example, Joyent [94] offers hosting solutions for various web application stacks (Joyent Accelerators). Although Joyent calls its datacenter a "cloud" and can scale up quickly, the billing regime for server usage is monthly. Even if extra server instances are required only one day out of a month, they still must be reserved for the entire month [95]. Many hosting providers using 3Tera's AppLogic [96] software, such as GridLayer [97] and RightServers [98], also have similar monthly virtual server offerings marketed as cloud services.

3.15 Related Products

In addition to core cloud service hosting, an entire ecosystem of related products has evolved to capitalize on the new and growing market. Again, due to the market's immaturity and lack of standardized technology (as well as the technical nature of certain software), various related products are often conflated with true cloud hosting services. Here we will not attempt to enumerate an authoritative list of such products, but will instead mention a few illustrative examples encountered in our product analysis.

3Tera AppLogic [96] is described as a "Grid Operating System for Web Applications" – it is a software layer used by many cloud and cloud-like hosting companies (e.g. ENKI [92]) to manage their data center resources. 3Tera is often confused with cloud hosting providers, but they actually sell solutions to such providers [99]. RightScale [100, 101] is a startup providing software to help businesses utilize and manage resources spanning multiple cloud providers. Similarly, Engine Yard Vertebra [102] is a "cloud-agnostic" framework for service discovery and management in cloud applications.

4 Dimensions of Comparison & Discussion

The relationships between products listed in Section 3 are complicated and often subtle. Unlike familiar commoditized IT hardware products, cloud computing services are not mutually interchangeable and such services are not directly comparable on orthogonal dimensions. In this section, we will attempt to analyze different, but often interrelated, dimensions of comparison between cloud computing offerings. Note that the terms "user" or "developer" in this section refer to the same concept – direct users of cloud services. The direct users are typically software developers and technologists. For example, the developers of Best Buy's Gifttag are direct users of App Engine, while Gifttag's users are indirectly using App Engine.

4.1 Resource Offered

The most fundamental property of a cloud offering is the actual resource (or resources) being offered. The two significant categories of resources are 1) storage and 2) computational resources (via machine-level virtualization or hosted, higher-level application hosting).[6] Most cloud providers offer both "compute" and storage hosting, but they are generally billed separately, even for "integrated" services such as App Engine. They may also vary independently: for example, Amazon offers several different types of storage services, one of which works with EC2 (EBS) and two of which can be used independently. Ultimately, an actual deployed service typically requires both storage and computational resources, but some companies may choose to leverage the cloud for only one. For example, SmugMug initially used Amazon's S3 for storage without using EC2 for web hosting or other computation.

4.2 Level of Virtualization or Abstraction

For a given resource, one obvious dimension of comparison between cloud offerings is the level of resource virtualization (or level of "abstraction"). This property determines the kind of interface presented to a resource user – an interface could be low-level like a virtualized server or high-level like a custom application programming environment designed for specific kinds of business applications. For example, the "Above the Clouds: A Berkeley View of Cloud Computing" presentation [21] shows different levels of abstraction of computational offerings on a continuum spanning from "lower-level, less management" to "higher-level, more management" (see Figure 6).

Here we will briefly discuss levels of virtualization/abstraction in the context of the two major resource categories:

Compute: Figure 6 shows several offerings on a continuum from lower to higher levels of abstraction. The listed products are all "compute" offerings, although some services also have integrated storage solutions under the same product name. We will discuss the storage elements in the subsequent section.

At the lowest level, services like EC2, GoGrid, Mosso Cloud Servers and FlexiScale offer "instruction set" virtualization, which means that the resource offered looks like a regular dedicated server. To a developer, it is similar to having a dedicated physical computer, except the computer is a virtual machine hosted on an unknown physical machine potentially shared with other users. At a slightly higher level, offerings like Mosso Cloud Sites also provide a virtual machine-like abstraction but resource users do not have full control over the machine (as they would with a personally owned and administered

[6] Some cloud services may only concern more specialized resources like network bandwidth/data transfer (as in CDN-like services, for example). As another example, Amazon's Simple Queue Service (SQS) can be seen as more of a messaging mechanism than a storage system (although it has elements of storage since delivery is reliable and messages are retained for several days).

server). Each virtual machine has a pre-installed operating system and a set of centrally administered and pre-configured software packages for use. Typically, the software packages include web servers and related software, and this flavor of partially managed virtual machine is similar to traditional *shared web hosting* services.[7] To a developer, these services are similar to lower-level machine-level cloud offerings with two differences: 1) the resource user does not have to manage the system software (operating system, web server configuration, etc.) and 2) the resource user cannot run arbitrary software or computation. If the developer needs to run software not supported by Mosso Cloud Sites, then he would have to use a lower-level offering; on the other hand, if the Cloud Sites software matches the developer's own needs, then the additional benefit of not having to install and manage lower-level details of the software may be a net win.

Microsoft's Azure sits at a similar level but has slightly different characteristics. Azure's cloud provides "bytecode" level virtualization: the resource presented to the user is a virtual machine which does not appear like a physical machine. Instead Azure presents safe, higher-level virtual machines – the most ubiquitous example of this kind of VM is Sun's Java Virtual Machine [103], which provides a common-denominator platform to run applications on a variety of hardware. In addition to the virtual machine, Azure provides a set of APIs (Application Programmer Interface) which allow the cloud application code to communicate with other cloud services in Azure and the outside world. These APIs have low-level features similar to a operating system as well as higher-level features similar to web application frameworks.

Taking another step towards more management and higher-level abstractions, offerings like Heroku and Engine Yard Solo provide cloud-based hosting for Ruby on Rails [88], a specific web application software stack. Whereas services like Mosso Cloud Sites and standard shared web hosting providers attempt to be generically useful for most web hosting, offering a wide range of installed web server runtime support components (e.g. many languages such as PHP, Python, Perl), Heroku and Engine Yard Solo offer one specific web application framework (and programming language). This specialization allows such services to provide added value in the form of automatic or better monitoring, scaling, quicker development and many other benefits. Of course, the downside is that they can only host Ruby on Rails web applications, whereas lower-level virtual machine offerings are more general. In fact, both Heroku and Engine Yard Solo are actually hosted in Amazon's EC2 compute cloud.

Google App Engine is similar to hosted Ruby on Rails offerings in its level of specialization, but with a few key differences. An obvious difference is the fact that App Engine runs on Google's own cloud rather than being a "value-added" service on top of another cloud provider. App Engine was also designed from the ground up to be a cloud web application stack based on Google's web hosting expertise and proprietary technologies. The latter point is important: Ruby on Rails is a popular web application framework, but it was designed to be general purpose tool for building average websites. Seamless scaling of Ruby on Rails is non-trivial and it commonly has external dependencies on a relational database, which often becomes a scaling bottleneck [104]. Companies like Heroku, Engine Yard and Joyent augment Ruby on Rails with their own technologies to scale it to varying degrees. On the other hand, App Engine was designed from the ground up as a cloud framework using the techniques Google has perfected in its own operations, providing the capability to scale as needed.

Finally, at the far right side of the cloud spectrum in Figure 6, we have domain specific web application frameworks, like Force.com and Bungee Connect. While web frameworks like Ruby on Rails App Engine are designed to host a wide-variety of web applications, Force.com is specialized for certain kinds of business-oriented web applications. Consequently, the building blocks provided to developers already include useful domain-specific functionality, allowing quicker development of suitable applications.

[7] The main difference is the billing granularity.

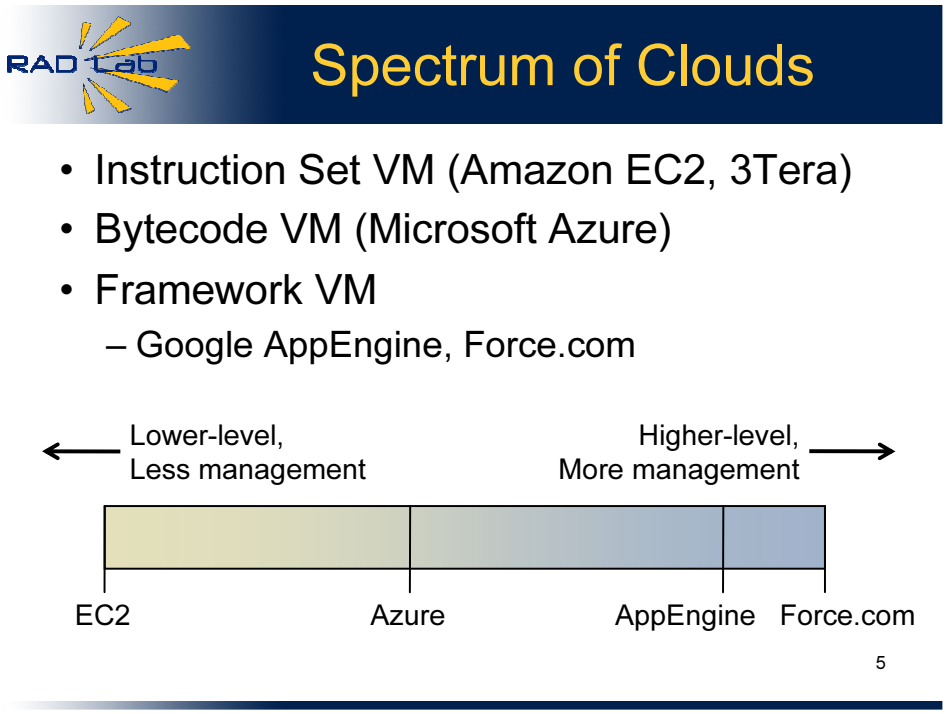

Figure 6: RADLab Presentation: Levels of Cloud Abstraction or Virtualization
From RADLab Presentation "Above the Clouds: A Berkeley View of Cloud Computing" [10, 21]

Storage: Storage solutions also exist at different levels of abstraction, but they do not necessarily follow a strict high-to-low hierarchy. At a lower level, Amazon's Elastic Block Store (EBS) provides block storage to EC2 virtual machines, and FlexiScale's virtual dedicated disks are similar. Block storage is very low-level unstructured storage mimicking a virtual hard drive of a desired size. A *filesystem* is often layered on top of a block device (virtual or physical) to provide structure: instead of just a linear series of bytes, a file system offers access in the form of named files typically structured into directory hierarchies.

At a higher level, GoGrid and FlexiScale offer file server-like storage. With this kind of storage, a filesystem-like abstraction is provided but the underlying block storage is not directly accessible. Ultimately, resources users can store and retrieve hierarchically organized files in a familiar paradigm. Services like Amazon's S3 or Azure's storage components provide "object" or "blob" storage, which is somewhat similar to filesystem-based storage. Fixed size entities can be stored and retrieved by name, but the organization may not be hierarchical like most filesystems, and there may be various limitations on object size or layout.

Another common type of storage is database-like storage. Traditional modern databases store many rows of relatively uniform tabular data. Although relational database management systems (RDBMSs) following the SQL standard have been the most popular general purpose databases for many years, full relational/SQL-supporting databases do not scale easily to serve large web applications. As a result, many cloud providers have pseudo-relational databases which store bulk structured data. Amazon's SimpleDB is one instance, providing database storage accessed using a SQL-like query language. Similarly, Google App Engine provides a "Datastore API" [105] to interact with Google's quasi-relational database BigTable [106] using a SQL-like language called GQL. Microsoft's Azure provides both basic tabular storage as well as the more powerful SQL Data Services.

4.3 Generality/Flexibility

In some sense, the generality and flexibility of a cloud resource is inversely related to its level of virtualization or abstraction. Lower-level resources are typically more general: two obvious illustrative examples are Heroku and Engine Yard Solo, which run Ruby on Rails cloud offerings on top of Amazon's lower-level EC2 service. EC2 can host these higher-level web application frameworks as well as other arbitrary software. The same is true to some extent with storage, as one can layer file storage, blob storage or database-like storage on top of virtual block storage.

In the earlier discussion of levels of abstraction (Section 4.2), we highlighted several different levels of compute resources:

- Arbitrary instruction-set virtual machine execution
- No administrative access, but generic managed virtual machines with a variety of common software – like shared web hosting
- Abstract safe virtual machines (like the Java VM) with low and high-level APIs/external services
- Specific general-purpose web frameworks
- Domain-specific web frameworks

The above list appears in order of increasing management/abstraction, and the lower-level options can usually support higher level solutions (potentially with more development/configuration effort). The lowest level services like EC2 are effectively equivalent to having a dedicated physical server where the hardware, Internet connection and power are managed by a contracted third party; the user can load arbitrary software. As abstraction increases, more of the administration and configuration of the software stack is provided by a third party, limiting generality. Since web applications with typical needs will probably constitute a significant portion of cloud applications, the lower generality of higher-level solutions will not be an impediment to business for such providers. Use cases like bulk processing at the New York Times or scientific simulation at Eli Lily (see Section 2.2) are not served by web applications at all, and require lower-level services like EC2 or GoGrid. Sun's defunct Grid Compute Utility product also specialized in such use-cases, including 3D rendering and scientific simulations [80].

4.4 Ease of Solution

While generality and flexibility are often inversely related to the level of resource abstraction/virtualization, ease of solution is directly related. Higher level resources trade generality for specialization, providing more pre-built common functionality. Assuming the higher-level offering matches the developer's needs, it usually requires less work to achieve the same or better results (if the developer is starting from scratch – see Section 4.5 for alternate considerations). Simply put, higher-level offerings hide the complexity present at lower levels of abstraction.

Consider the deployment of a typical web application on EC2 – the user would need to configure an operating system image, configure web server software as well as any web application framework and also develop mechanisms for load balancing, monitoring, failure handling and more. On the other hand, App Engine already handles all of these issues below the virtualized App Engine framework. Offerings like Mosso Cloud Sites at least manage the operating system and web software configuration (and typically monitoring and some load balancing as well). Recognizing this disadvantage, Amazon is starting to provide pre-built partial solutions for EC2 to match common use-cases: these solutions include pre-configured OS images with various software stacks, and upcoming load balancing, scaling,

monitoring and management solutions [107]. Finally, domain-specific offerings like Force.com provide even more useful, pre-built functionality over offerings like App Engine because they are specialized for a particular class of business-oriented web applications.

4.5 Legacy Support

Legacy issues are a recurrent thorn in the side (or at least constant considerations) of many IT-related projects. Legacy systems are existing computer systems or programs based on older technologies that continue to be used. Typical examples include mature applications that work well (or well enough) but are expensive to modify or replace. From the perspective of the cloud, almost any existing software can be considered "legacy" in some sense. Many applications which would benefit from migration to the cloud exist in some working form outside the cloud, and companies are hesitant to completely recreate systems that are already working well enough. The ability to reuse code from existing systems has both positive and negative aspects.

Legacy support is sometimes coupled with the level of generality – a low-level "instruction set" virtual machine can typically support arbitrary legacy software written for a given instruction set. On the other hand, App Engine and Ruby on Rails offerings are both at similar levels of generality, but App Engine was built from scratch for Google's cloud offering, so it has no meaningful legacy support. Developers must develop App Engine-based web applications mostly from scratch, while Ruby on Rails offerings could potentially allow existing Ruby on Rails-based web applications to migrate to the cloud. In this manner, legacy support also affects ease of solution (Section 4.4). If a system is built from scratch, higher-level and more specialized solutions are generally more convenient; if a significant existing application needs to be migrated to the cloud, the situation might be reversed. Fundamentally, legacy support in cloud scenarios is a complicated issue, because even if legacy applications can run in the cloud, they may not be able to scale well, so there is a trade-off between the effort required to fix an existing system and starting over.

On the other hand, when clouds are used to run massive computation rather than serve web requests, legacy support is often a huge boon. For example, the Washington Post document conversion project [24] utilized existing PDF and OCR (Optical Character Recognition) utilities run on a large number of EC2 instances to convert a large set of documents into searchable files. It would be a significant and unnecessary effort to re-write these utilities in .NET CLR-compatible code to run them on the Azure virtual machine platform.

4.6 Scaling

Scalability is a key issue for many cloud-hosted applications. One of the key benefits of cloud computing is the illusion of infinite and on-demand resources, so an important challenge is constructing an application that can actually use such flexibility effectively. Oftentimes some piece of the software architecture becomes a bottleneck which prevents the system from scaling up even when resources are available. Building web applications that can serve hundreds of thousands or even millions of clients is very difficult, but companies like Amazon and Google have the necessary experience and technology. One important property for assessing cloud offerings is whether a provider's product can help a user with scalability.

In some sense, the ability of a cloud offering to assist in scaling automatically is inversely related to the level of virtualization. Higher-level offerings, like Google App Engine, have certain restrictions and utilize domain knowledge and insights to scale hosted applications seamlessly. The higher level abstractions in App Engine were also designed specifically to fit a scalable development model. The techniques used to scale an application often come from specific insights and knowledge about the

problem; some details can be ignored when the scaling is limited to a specific application. Offerings like EC2 are so general that they cannot easily use this domain knowledge. EC2 can be running any kind of arbitrary application, from well-designed web services to games to encryption software. Consequently, Amazon is planning to provide pre-built partial solutions help users scale common use-cases [107].

Another more specialized consideration is a cloud provider's total available resources. While it is unlikely a client could exhaust Amazon's or Google's available servers, other cloud providers may have more limited total resources. A provider's maximum internal capacity is typically guarded as a secret.

4.7 Integration

Another trade-off for cloud services is the level of integration between potentially separable concerns: for example, Google App Engine is a single, conceptually unified offering, while Amazon S3 allows clients to use various forms of cloud storage independent of EC2 compute instances. In fact, SmugMug's use of S3 (see Section 2.2) predates the launch of EC2; SmugMug's success highlights the value of individual components for some clients. With the exception of integrated framework providers (i.e. App Engine or Ruby on Rails services), most cloud providers seem to offer at least separable storage and compute capabilities. On the other hand, developing separate services general enough to be independently useful may come at the cost of introducing additional complexity when such services are used together; integrated solutions can potentially be more coherent and thus simpler to use. Integration is also partially related to lock-in (see Section 4.9).

4.8 Standards

Standards are an important consideration in many IT projects. Technology users, particularly businesses, often prefer to use products conforming to well-known technical standards. Standardized products may provide several benefits to users: 1) price advantages and availability of substitutes due to competition (when many vendors provide conforming products), 2) interoperability with other products and 3) technical transparency for customization. From a vendor's perspective, the potential for competition is a disincentive to provide standardized products; however, vendors must balance the potential revenue benefits of proprietary products versus the potential for customers to prefer competing standardized products. Standardized products can follow a specific technical standard maintained by an organization such as ANSI or the ISO, or they can simply implement a commonly-used or familiar interface (*de facto* standards). Standards are closely related to lock-in (see Section 4.9).

The Cloud Computing Interoperability Forum (CCIF) [108] is an early industry coalition, including IBM, Sun, Cisco and Intel, focusing on standardization and interoperability issues. The CCIF mission statement summarizes their goals – "the CCIF will not condone any use of a particular technology for the purposes of market dominance and or advancement of any one particular vendor, industry or agenda. Whenever possible the CCIF will emphasis the use of open, patent-free and/or vendor-neutral technical solutions" [109]. Various major cloud providers and industry participants have different views on standards: Google, Amazon and IBM are major industry supporters of open-source software and open standards [110, 111, 112]. Smaller players often embrace open standards because they lack the clout to push proprietary products over standardized ones. On the other hand, Microsoft is often notorious for anti-competitive behavior involving proprietary products. In fact, one penalty in the European Union's ongoing anti-trust proceedings against Microsoft requires the company to release technical information about proprietary protocols to allow competitors to develop competing interoperable solutions [113].

Although in recent years Microsoft has attempted to appear more open to standards-conformant products, many industry insiders and observers find such efforts disingenuous [114]; standardization alone does not necessarily bring the potentially realizable benefits of increased competition or interoperability. In fact, Microsoft has also been accused of manipulating the ISO standards body approval process [115].

In the context of the products in our comparison, Amazon's offerings are largely built using open source technologies. For example, EC2 machine images are based on the open-source Xen virtualization software and can be created with freely available tools, and EC2 virtualizes the ubiquitous Intel x86 instruction set.[8] Similarly, interaction with S3 and other Amazon services occurs via a simple published HTTP interface. Services like GoGrid and FlexiScale tend to follow Amazon's example. Behind the scenes, however, S3 and SimpleDB rely somewhat on proprietary Amazon technology, but most cloud vendors are plagued by this in some of their storage products. For example, most database-like cloud storage offerings do not follow the standard SQL query language and relational data model for good technological reasons; consequently, every offering uses its own proprietary data model and query features. Offerings like Amazon's EBS, GoGrid's fileserver-based Cloud Storage and Flexiscale's virtual dedicated disk and fileserver-based storage offerings are all significantly more standard (and thus substitutable) in their interfaces. In these cases, "standard" does not mean following a single specific technical standard per se, but their interfaces are familiar and interchangeable with many commonly-used devices (block devices appear like regular hard disks, and filesystems are commonly used).

Google's App Engine is built around the open-source Python programming language[9], but App Engine relies on some proprietary Google technology. App Engine was also built from scratch for the cloud and is controlled solely by Google. Ruby on Rails, on the other hand, was originally built by one company (37signals [117]) but is now independently maintained as open source software. Although interested parties are still able to influence its direction, many vendors offer competing Rails products. Unless Google agrees to maintain certain compatible and stable App Engine interfaces, a competitor wishing to provide a compatible substitute products would face a difficult situation, subject to the whims of Google's changes. Offerings like Mosso's Cloud Sites provide a web stack of widely-used, standardized software, either open source (Linux, Apache, MySQL and Python, Perl, PHP, etc.) or Microsoft-based (Windows, IIS, SQL Server and .NET). Microsoft's Azure offering follows some technical standards, but the platform as a whole is tied to Microsoft's proprietary product line. Interfaces to some Azure components follow a documented HTTP-based interaction paradigm, like Amazon and many other cloud providers. On the other hand, Azure is very much like App Engine in that it was designed from the ground up as an ecosystem of complementary cloud offerings, so applications must be designed specifically for Azure and it is under the sole control of Microsoft. Higher-level, domain-specific offerings like Force.com are usually tied to a particular vendor, because standardization is more difficult in niche domains compared to general-purpose, widely-used technical artifacts.

4.9 Lock-in

Lock-in is one of the most cited and controversial obstacles to widespread adoption of cloud computing in enterprises [10, 118, 119, 120]. Simply put, businesses are wary of being tied to particular cloud computing vendors due to lack of competing, compatible products. As we've discussed extensively in this study, most cloud offerings are not directly substitutable. It is risky to be tied to a single vendor because that vendor might raise prices, go out of business, become unreliable or fail to keep up with

[8] EC2 will run software that runs on desktop and most server PCs.

[9] and Google has announced forthcoming support for Java-based App Engine development [116]

technological progress. As mentioned earlier, standards (Section 4.8) are closely related to lock-in but do not determine the full picture. Standards only mitigate certain technical obstacles to lock-in; even if a product is standardized, there is no guarantee competing products will emerge or that a competitor can match the desired level of service. For example, a compatible EC2 competitor might have difficulty matching Amazon's available resources. A company like Animoto, requiring 5000 EC2 server instances [121], may not be able to find a cloud provider with a compatible and affordable offering that can match the scale required.

As one example, Amazon and Google both use open standards (to some extent) for their offerings and although interfaces are public and relatively standardized, product implementations are private. Interface transparency allows compatible and interoperable substitutes, but there is no guarantee that substitutes will match the performance or scaling abilities of the original products. An open source project called AppDrop provides an App Engine compatible interface that runs on Amazon's EC2 [122], so applications for Google's App Engine can be run on Amazon's infrastructure. Unfortunately, however, the scalability of AppDrop is nowhere near comparable to App Engine. While AppDrop is just a proof-of-concept, it does illustrate the challenges in mitigating lock-in even when products use relatively open standards.

Development tools are also subject to lock-in concerns. Although vendors using open source software typically do not care how clients develop cloud applications to run on their infrastructure, some vendors explicitly offer value-added development utilities. For example, Force.com and Bungee Connect offer higher-level web-based graphical development tools. While it may be possible to use their platforms to run cloud applications without using their tools to develop said applications, the development tools are part of the value proposition. Also, as mentioned earlier, Microsoft's Azure is explicitly built around its ecosystem of developer tools – most of which have no viable substitutes.

Data lock-in is specific subset of the general *lock-in* concern [10]. While many discussions of lock-in concern whether a cloud application running on Amazon's EC2 (for example) could run on a competing provider, data lock-in concerns the potentially massive amount of data stored with a cloud provider. Since data is much harder to recreate when compared to an application, data lock-in issues are often the most important lock-in concerns. For example, a significant portion of SmugMug's data is stored in Amazon's S3 (see Section 2.2). If Amazon goes out of business, does SmugMug lose its data? If SmugMug only ran computation in the cloud and its provider went out of business, SmugMug could re-create its entire cloud-based application easier than it could re-create all customer-stored photo data. Alternately, if SmugMug needs to move to another service, how will it migrate all of the data? Satisfactory answers to these questions are critical to wholesale uptake of cloud services.

4.10 Interoperability

Interoperability is closely related to both standards (Section 4.8) and lock-in (Section 4.9); in this context, we will focus on interoperability between cloud providers. Interoperability solutions can often address lock-in – for example, RightScale [100] provides software to help clients manage resources across multiple cloud providers (EC2, FlexiScale and GoGrid). Software to abstract away a specific cloud provider's platform can be used to mitigate the potential for lock-in. In this case, an application would be written for a "cloud agnostic" software platform where provider-specific quirks are handled under a management layer. In addition, Armbrust et al. believe that businesses *must* use multiple cloud providers to eliminate singles sources of failure:

> 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. [10]

4.11 SLAs

Cloud users want assurances that their provider will remain reliable because service interruptions can cause significant financial harm. Service Level Agreements (SLAs) are contractual promises of certain levels of reliability; such provisions often include monetary compensation if the level of service provided is below the contractually specified level. For example, Amazon's EC2 SLA species the following: "AWS will use commercially reasonable efforts to make Amazon EC2 available with an Annual Uptime Percentage (defined below) of at least 99.95%. ... If the Annual Uptime Percentage for a customer drops below 99.95% for the Service Year, that customer is eligible to receive a Service Credit equal to 10% of their bill (excluding one-time payments made for Reserved Instances) for the Eligible Credit Period" [123].

Amazon is an industry-leading example in cloud computing SLAs; it was the first major provider to define SLAs for all of its products. Microsoft's Azure is not yet publicly launched and SLA details are unavailable. The Azure FAQ promises, "we will also give you access to robust service level agreements and guarantees on quality of service" [124]. Google App Engine does not yet provide an SLA and this is perceived as somewhat of a disadvantage, but Google promises SLA details are forthcoming [125]. GoGrid features a strong-sounding SLA, promising 100% uptime and a 100-fold refund for downtime. For example, "a Failure lasting seven (7) hours would result in credit of seven hundred (700) hours of free service for the feature in question; a Failure lasting fifteen (15) minutes would result in a 1500-minute, or 25-hour, credit" [126]. FlexiScale's SLA specifies 99.99% uptime [76] although it does not clearly specify what compensation is provided for failure to meet this target. Mosso's SLA provides a credit of "1 day's hosting fee for each 60 minutes of unscheduled downtime" [127]. ENKI provides a "10x money-back guarantee for unplanned outages" longer than 15 minutes [128]. Engine Yard Solo and Heroku are both hosted in Amazon's EC2 cloud (so they inherit its SLA), and neither Force.com nor BungeeConnect currently specify SLAs.

Although certain SLAs sound very strong (i.e. GoGrid's), evaluating SLAs is complicated. A 100-fold refund for seven hours of down time with ten servers would only amount to $560 of credit maximum (and the credit cannot exceed the monthly bill); an e-commerce website down for seven hours could easily lose thousands of dollars of sales. Since the financial loss from a provider breaking an SLA may far exceed the specified remuneration, SLAs are only one part of the larger picture. Actual observed provider performance and reliability are just as important as contractual guarantees. See Section 4.15 under "Reliability" for more on this point.

4.12 Redundancy

Redundancy is important in several contexts. Obviously redundancy is critical at the cloud provider level. A cloud provider should have multiple data centers, redundant networking, backup power, data backup plans and other redundant resources. Although customers generally trust that cloud providers

take such precautions, large scale provider failures have already occurred – one notorious example is an online cloud storage provider "The Linkup," which ceased operations after losing nearly half of customers' data [129].

At the level of individual cloud applications, service redundancy is often achieved by launching multiple "compute" instances in different *failure domains*. A failure domain is a set of resources that can experience correlated failures – for instance, one might run server instances on different racks in a single data center to protect against rack-wide failure modes (switch failure, network uplink failure, etc.), or even in multiple data centers to protect against data-center-wide failure modes (regional network failure, massive power failure, natural disasters, etc.). Data redundancy is perhaps even more important than service redundancy as data loss is much more difficult to fix than intermittent service interruption.

Like scalability, the feasibility of automatic redundancy in cloud offerings is often inversely related to the level of abstraction of the resource. Take Amazon's storage offerings: EBS and S3. S3 provides a higher-level storage interface than EBS, and S3 is also much more redundant. S3 data is replicated in multiple *availability zones*, which are essentially Amazon's engineered coarse-grained failure domains. On the other hand, EBS is only redundant within a single availability zone [130]. Amazon notes, "volumes that operate with 20 GB or less of modified data since their most recent Amazon EBS snapshot can expect an annual failure rate (AFR) of between 0.1% - 0.5%, where failure refers to a complete loss of the volume" [57]. Similarly, Google can automatically make a web site using App Engine span multiple failure domains, while an EC2 user would need to specifically launch multiple instances in different availability zones. Obviously some customers will need to layer extra redundancy on top of existing solutions that do not provide acceptable redundancy guarantees.

4.13 Security

Security is a continuous consideration in IT-related projects. Unlike many other traits in technological contexts, security is notoriously hard to quantify or even compare qualitatively. For this reason, security evaluation of cloud offerings will mostly hinge on company reputation and, eventually, real-world track records – but even real world track records are difficult to compare between companies, because security breaches may not be publicly disclosed unless compelled by regulation. Like SLAs, companies might specify contractual compensation for certain kinds of provider negligence leading to security failures, but such provisions may be worth very little since security failures are not as easily observable as service availability failures.

Businesses using cloud services want to ensure that their data is secure from both external attackers as well as internal snoopers (employees of the cloud provider). Although data theft and snooping is mitigated by properly encrypting data to be stored within the cloud[10], encryption cannot prevent *denial-of-service* attacks such as data deletion or corruption. Some early research users of Amazon S3 suggest, "users should employ some kind of data authentication technology to assure themselves that data returned by S3 is the same as the data that was stored there. Technology such as an HMAC or a digital signature would protect users from both accidental data modification by Amazon and from malicious modification by third parties who managed to crack a password or intercept a password-reset email message" [131]. Service integrity is another security issue: businesses want to ensure their running services are not subject to denial-of-service attacks or hijacked. The latter can be very insidious, as a third party might (for example) gain control of a business's e-commerce site and besmirch its reputation. This situation is the digital equivalent of identity theft. *Isolation* is a related concern – cloud providers serve many customers and they all share common hardware and infrastructure. Although

[10] It must be encrypted before it is sent to the cloud, not by the cloud provider.

resource virtualization prevents customers from having to explicitly coordinate resource sharing, the cloud provider must ensure that multiple customers do not interfere with each other, maliciously or otherwise (see Section 4.15 under "Resource Sharing & Performance" for more on isolation).

4.14 Resource Billing

Resource billing is one of the distinguishing characteristics of cloud computing services. As mentioned earlier, the model of *utility computing* separates cloud services from more traditional forms of hosting. With true cloud services, resources are billed based on dynamic use, and billing is generally "cost associative," so using 1,000 servers for one hour is the same as using one server for 1,000 hours [10]. Today, one distinction between different cloud offerings is not just the price of specific resources, but what resources are actually metered and billed. Although certain resources are obviously fundamental (i.e. storage or compute time used), customers must also be aware of secondary incurred usage charges related to primary resources.

Take Amazon, for example: the basic resources of "compute" and "storage" naturally admit per-hour and per-gigabyte (per month) costs, but Amazon also charges for ingoing and outgoing network data transfer, as well as a nominal fee per request to some services. Amazon's S3 pricing in the US is based on the following schedule [37]:

Storage	
$0.150 per GB	first 50 TB / month of storage used
$0.140 per GB	next 50 TB / month of storage used
$0.130 per GB	next 400 TB /month of storage used
$0.120 per GB	storage used / month over 500 TB
Data Transfer In	
$0.100 per GB	all data transfer in
Data Transfer Out	
$0.170 per GB	first 10 TB / month data transfer out
$0.130 per GB	next 40 TB / month data transfer out
$0.110 per GB	next 100 TB / month data transfer out
$0.100 per GB	data transfer out / month over 150 TB
Requests	
$0.01 per 1,000	PUT, COPY, POST, or LIST requests
$0.01 per 10,000	GET and all other requests (except DELETE)

The bottom category, "Requests," specifies a nominal fee for HTTP-based interaction with S3; although these charges are relatively small, they do impose certain constraints on how data is added to and fetched from S3. Consider storing and retrieving a single GB of data from S3. The storage costs 15 cents for a month, and the data transfer in and out will cost 10 cents and 17 cents, respectively. If we send the entire 1GB of data in a single PUT request and retrieve it with a single GET request, our request costs are low enough to ignore. If instead, we have a sensor producing 1KB of data every second and send each reading as a separate request, we will incur 1,048,576 PUT requests to create 1GB of data, costing us about $10.48 in request fees for that same GB of data, nearly 25 times the cost of the storage and data transfer.[11] For another example of auxiliary fees, consider SimpleDB. Although SimpleDB is ostensibly a data storage service and does not need to be used in conjunction with EC2

[11] This sensor reading scenario is not contrived to produce unreasonable overheads. Enterprise applications based in a local datacenter might work in this manner since the number of requests is largely irrelevant. Businesses must be aware of these secondary charges when moving an existing application to a cloud provider.

compute instances, SimpleDB billing also includes charges for processing time. This is because some database queries may require significant processing to execute, so SimpleDB CPU time is also metered.

With EC2, the main cost driver is instance hours, and Amazon offers instances with varying amounts of RAM, CPU power, IO capacity and operating systems for different prices. Amazon also charges for data transfer in and out of instances, and EBS-related charges are similar to S3 (price per GB-month as well as per million I/O requests and various request charges for putting data in and out of S3). Google App Engine's prices are comparable to Amazon's [132], but the compute metering is slightly different. Amazon's EC2 service charges 10 cents per instance hour for the cheapest instance type, and App Engine charges 10 cents per CPU hour, which is roughly equivalent. However, App Engine charges based on CPU time used to process App Engine requests, while EC2 charges for time a machine instance is running (even if it is idle and not processing requests). This disparity is due to the difference in level of virtualization/abstraction between the two services. If an instance is mostly idle and only processes requests occasionally, App Engine's pricing is more advantageous. Unlike Amazon, App Engine also meters and bills for email sending volume (in addition to the network bandwidth generated by sending email).

Compared to EC2, GoGrid's compute and network billing are slightly different. GoGrid charges for "server RAM hours," which means that the basic differentiator between server instances is the amount of RAM available. A single compute instance with twice the RAM of another instance will cost twice as much. EC2 offers instances with different amounts of RAM, but the price scaling does not directly track the amount of RAM. EC2 also offers instances varying in CPU power and I/O bandwidth, while GoGrid does not. GoGrid charges $0.50 per GB of outbound traffic compared to Amazon's $0.17 - $0.10 per GB. On balance, GoGrid does not charge for any inbound traffic, while Amazon charges $0.10 per GB. Like GoGrid, Mosso Cloud Servers's computer instance vary regularly with the amount of RAM, and FlexiScale's compute instance billing varies regularly by RAM amount and CPU count per instance. ENKI's billing is rather unusual and complex – ENKI bills based on an amortized monthly percentage of both CPU and RAM usage of a standard 16GB/4 core virtual server instance [93]. For example, a client can allocate an arbitrary percentage of a CPU core (say 7%) and some amount of RAM and the monthly billing is based on used resource hours per month. The CPU and RAM allocation can be changed as needed during the month.

Like App Engine, Mosso's Cloud Sites bills based on used compute cycles. Azure's pricing has not been announced, but it is likely to be a hybrid between App Engine's compute pricing and Amazon's non-compute resource pricing. Azure's Pricing & Licensing FAQ [133] lists the following metered resources:

- Compute time, measured in machine hours
- Bandwidth requirements (transmissions to and from the Azure data center), measured in GB
- Storage, measured in GB
- Transactions, measured as application requests such as Gets and Puts

Specialized business SaaS-oriented cloud providers like Force.com and Bungee Connect additionally meter usage per user of the constructed service. Bungee Connect charges six cents per "user-session-hour" and Force.com can charge either per-login or per-user per-month.

The granularity and tiers of billing also vary significantly between cloud providers. Some providers will round up to the nearest hour or GB – for compute resources, Amazon rounds to the nearest instance hour, while Google rounds to the very fine grained 1/1200th of a CPU second; many providers have free thresholds above which resources are billed, and some offer tiered resource pricing. Amazon

is now offering "reserved instances" in EC2, "which provides customers who have predictable usage patterns with a way to even further reduce costs" [134] by buying EC2 instance hours in bulk for a discounted rate. ENKI has unusual monthly amortized billing, and complicated, opaque resource metering may make it difficult for businesses to predict costs. Ultimately, the granularity of cloud billing is what distinguishes "cloud" resources from more traditional hosting. Traditional hosting arrangements require rental of dedicated or shared servers in flat monthly increments. Cloud services use finer-grained billing to increase flexibility and allow resource reservation to more closely track actual demand.

4.15 Other Issues

In the diverse and evolving world of cloud products, many other potential dimensions of comparison exist. Some examples include:

- **Software** – Software is a broad issue in cloud computing. Major open questions exist concerning both software used within deployments of cloud applications and during development of such applications. For example, software used within an EC2 instance may include image converting software used by the New York Times or Washington Post. Although early uses of cloud computing are often powered by open-source and free software, traditional pay-per-license software does not translate well to the more dynamic environment of the cloud. Armbrust et al. name "Software Licensing" one of the top ten obstacles to the growth of cloud computing and note, "the licensing model for commercial software is not a good match to Utility Computing" [10]. Some providers have already attempted to address such issues by partnering with software vendors to offer rate-inclusive licenses (e.g. Amazon's Windows EC2 instances cost more than Linux instances because the licensing fee is built in). In addition, better development tools are needed for cloud software. Specialized providers like Salseforce and Bungee Connect have recognized and attempted to capture business by addressing this need.

- **Geo-diversity** – The location and geographical diversity of a cloud provider's data centers might be a concern for some users. Some services are "embarrassingly distributed" and benefit immensely from datacenters closer to clients (since this decreases latency of network access) [135]. Additionally, the location of a cloud provider's data may be critical for legal reasons; for example, Canadian universities cannot use Google's email hosting service due to regulatory concerns over conflicts with Canadian academic privacy laws and the USA PATRIOT Act [136].[12]

- **Regulatory Concerns & Auditing** – As mentioned above, regulations may limit adoption of the cloud for certain businesses. In particular, HIPPA and Sarbanes-Oxley compliance are frequently cited in concerns about cloud services and outsourced infrastructure [137, 138]. Providers are just starting to address such issues.

- **Reliability** – In the SLA section (Section 4.11), we mentioned that providers' actual reliability track records are just as important as contractual guarantees. Although cloud services are likely to have significantly better reliability than a small business's self-maintained IT infrastructure, high profile outages have occurred. For example, Armbrust et al. cite multi-hour outages of Amazon's S3, Google's App Engine and GMail [10]. As mentioned in Section 4.10, Armbrust et al. ultimately believe that businesses should use redundant cloud providers to guard against provider-wide outages (or even rarer and more devastating failures like providers suddenly going out of business) [10].

[12] Since GMail data is hosted partially in US, it may be subject to search by the provisions of the PATRIOT Act.

- **Resource Sharing & Performance** – As mentioned in Section 4.13, in the cloud, many customers share common physical computer hardware and network infrastructure. Isolation for security is one facet of resource sharing, but sharing can also cause performance problems. Since providers use statistical multiplexing, unreasonable levels of over-subscription will cause degraded service. Even without massive over-subscription, poor resource scheduling and management on the part of cloud providers could also degrade customer observed performance to unacceptable levels. In addition, some providers might misrepresent their total available capacity. Like reliability, the performance and quality of service in the face of resource sharing is a cloud attribute that will be assessed largely by providers' observed track records. An early academic analysis tracking seven months of Amazon EC2 and S3 performance data found that Amazon's services are a good value, but performance is highly variable [131]. For example, "[researchers] saw a large drop in system performance of S3 between April 9th and April 11th that we are unable to explain, and which Amazon has declined to explain" [131].

5 Conclusion

Cloud computing is a fledgling area with significant promise for revolutionizing certain business concerns. Both marketing opportunism and the relative immaturity of the industry have lead to muddled terminology and unclear product comparisons. In a frequently-quoted screed, Oracle founder and CEO Larry Ellison lamented these problems:

> The interesting thing about cloud computing is that we've redefined cloud computing to include everything that we already do. I can't think of anything that isn't cloud computing with all of these announcements. The computer industry is the only industry that is more fashion-driven than women's fashion. Maybe I'm an idiot, but I have no idea what anyone is talking about. What is it? It's complete gibberish. It's insane. When is this idiocy going to stop? [139]

Although scathing, Ellison's concerns are shared by industry insiders [5] and observers [6]. This analysis has attempted to provide some much-needed clarity by enumerating major projects, identifying dimensions of product comparison and analyzing different products in depth along the various dimensions of comparison. The comparative analysis presented in Section 4 is more fine-grained than unstandardized – but widely-used – industry categorical classification schemes (see Section 2.3). Ultimately this style of comparison is much more expressive and revealing than simplistic, one-dimensional PaaS or IaaS categorizations.

The cloud industry is immature, and currently available products are first-generation offerings. As the market matures, the huge variation in products will probably diminish with the industry converging on several popular types of second- and third-generation cloud offerings. Some early providers will leave, and industry observers are expecting heavyweights like Facebook, eBay, HP and Yahoo to enter at some point. While adoption is tenuous and its future success is not assured, cloud computing's market potential is very promising.

Appendix A – Glossary

The following is a glossary developed for a previous class project. It is presented here with additional entries.

Utility computing is the concept of selling computing services on-demand and in metered increments, like a public utility – this concept underlies most "cloud" offerings and is often used interchangeably, although some instances of utility computing predate or would not be properly classified as "cloud" computing.

Infrastructure as a Service (IaaS) is providing general on-demand computing resources such as virtualized servers or various forms of storage (block, key/value, database, etc.) as metered resources. Sometimes called *Hardware as a Service (HaaS)*. Some forms are similar to *shared hosting* but typically with added on-demand scaling via resource virtualization and use-based billing.

Platform as a Service (PaaS) is providing an existent managed higher-level software infrastructure for building particular classes of applications and services. The platform includes the use of underlying computing resources, typically billed similar to IaaS products, although the infrastructure is abstracted away below the platform.

Software as a Service (SaaS) is providing specific, already-created applications as fully or partially remote services. Sometimes this is in the form of web-based applications and other times it consists of standard non-remote applications with Internet-based storage or other network interactions.

Service Level Agreement (SLA) is a contract-specified level of service a provider will meet while providing some product or service. For example, an Internet service provider might specify 99.9% uptime per year – the provider is specifying in the contract that the service should not be out more than 8.76 hours per year (if the SLA is violated, the customer could be entitled to compensation).

Virtualization is a broad term for various abstraction techniques applied to different layers of the computing hardware or software stack. Generally speaking, virtualization hides resource complexities and decouples a resource's interface from lower-level details by presenting a virtual resource.

Failure Domains are sets of resources with the potential for experiencing correlated failures. For example, a set of computers on a single power source will all become unavailable if the power source fails.

Batch Computation is large-scale bulk data processing. Typical domains include mathematical simulations, data mining and 3D rendering. Batch computation is latency insensitive and is fundamentally different from interactive computation, which requires prompt responses. Google creates large search indices of the entire World Wide Web as a batch process, which may take several hours. Google search request processing, however, is not batch computation. Search queries use the pre-processed, stored web index data to perform data lookup and reply immediately.

Legacy Systems are existing computer systems or programs based on older technologies that continue to be used.

References

[1] Gmail: Email from Google. http://mail.google.com/.

[2] Amazon Web Services. http://aws.amazon.com/.

[3] Microsoft Online Services. http://www.microsoft.com/online/default.mspx.

[4] Facebook. http://www.facebook.com.

[5] James Urquhart. The need for a standard cloud taxonomy. CNet News, The Wisdom of Clouds. http://news.cnet.com/8301-19413_3-10148806-240.html.

[6] Gartner Says Contrasting Views on Cloud Computing Are Creating Confusion. http://www.gartner.com/it/page.jsp?id=766215.

[7] Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/.

[8] Google App Engine. http://code.google.com/appengine/.

[9] Welcome to Google Docs. http://docs.google.com/.

[10] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28, Electrical Engineering and Computer Sciences, University of California at Berkeley, February 2009.

[11] Jeff Bezos' Risky Bet. http://www.businessweek.com/magazine/content/06_46/b4009001.htm.

[12] Part II - Scaling into the Cloud with Amazon Web Services. Washington Technology Industry Association. http://www.washingtontechnology.org/pages/events/events_events_wsaevent.asp?EventID=779.

[13] James Hamilton. WTIA: Scaling into the Cloud with Amazon Web Services, March 2009. http://perspectives.mvdirona.com/2009/03/04/WTIAScalingIntoTheCloudWithAmazonWebServices.aspx.

[14] Werner Vogels. A Head in the Cloud: The Power of Infrastructure as a Service. Talk at Stanford University for Stanford EE Computer Systems Colloquium (EE380), April 2008.

[15] Jon Stokes. Into the cloud: a conversation with Russ Daniels, Part II, February 2009. http://arstechnica.com/business/news/2009/02/into-the-cloud-a-conversation-with-russ-daniels-part-ii.ars.

[16] Clouds and judgment. *The Economist*, October 2008. http://www.economist.com/opinion/displayStory.cfm?source=hptextfeature&story_id=12471098.

[17] Let it rise. *The Economist*, October 2008. http://www.economist.com/opinion/displaystory.cfm?story_id=12411882.

[18] Albert Greenberg, James Hamilton, David A. Maltz, and Parveen Patel. The Cost of a Cloud: Research Problems in Data Center Networks. *ACM SIGCOMM Computer Communication Review (CCR)*, 39(1):68–73, 2009.

[19] Jeremy Elson and Jon Howell. Handling Flash Crowds from Your Garage. In *USENIX 2008 Annual Technical Conference (ATC '08)*, pages 171–184, Berkeley, CA, USA, 2008. USENIX Association.

[20] Derek Gottfrid. The New York Times Archives + Amazon Web Services = TimesMachine, May 2008. http://open.blogs.nytimes.com/2008/05/21/the-new-york-times-archives-amazon-web-services-timesmachine/.

[21] Armando Fox, Anthony Joseph, Randy Katz, and David Patterson. Above the Clouds: A Berkeley View of Cloud Computing. Video Interview. http://www.youtube.com/watch?v=IJCxqoh5ep4, February 2009.

[22] Liza Daly. Processing the Deep Backlist at the New York Times. O'Reilly Tools of Change for Publishing, August 2008. http://toc.oreilly.com/2008/08/processing-the-deep-backlist-a.html.

[23] Derek Gottfrid. Self-service, Prorated Super Computing Fun!, November 2007. http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun/.

[24] Washington Post Case Study: Amazon Web Services. Amazon Web Services Business Case Studies. http://aws.amazon.com/solutions/case-studies/washington-post/.

[25] Federal Computer Week: Amazon.mil? DISA is intrigued by Web services model for creating systems, October 2006. http://aws.amazon.com/about-aws/media-coverage/2006/10/30/federal-computer-week-amazon-mil/.

[26] SmugMug Photo Sharing. Your photos look better here. http://www.smugmug.com/.

[27] Welcome to Flickr - Photo Sharing. http://www.flickr.com/.

[28] SmugMug Case Study: Amazon Web Services. Amazon Web Services Business Case Studies. http://aws.amazon.com/solutions/case-studies/smugmug/.

[29] James Rogers. Photo sharing site SmugMug shifts from RAID to Amazon's S3 service. Byte and Switch – Storage Networking and Beyond, January 2007. http://www.byteandswitch.com/document.asp?doc_id=111156.

[30] John Foley. Eli Lilly On What's Next In Cloud Computing. Information Week Business Technology Network: Plug Into the Cloud, January 2009. http://www.informationweek.com/cloud-computing/blog/archives/2009/01/whats_next_in_t.html.

[31] Giftag - Pick Anything | Welcome to Giftag. http://www.giftag.com/.

[32] App Engine Developers - Best Buy's Giftag. Google Code Blog, February 2009. http://google-code-updates.blogspot.com/2009/02/app-engine-developers-best-buys-giftag.html.

[33] TC3 Health Case Study: Amazon Web Services. Amazon Web Services Business Case Studies. http://aws.amazon.com/solutions/case-studies/tc3-health/.

[34] M. Tim Jones. Cloud Computing with Linux. http://www.ibm.com/developerworks/linux/library/l-cloud-computing/index.html.

[35] CloudCamp Seattle Feb 28th, 2009. http://cloudcamp-seattle-09.eventbrite.com/.

[36] James Hamilton. Seattle Cloud Camp, March 2009. http://perspectives.mvdirona.com/2009/03/12/SeattleCloudCamp.aspx.

[37] Amazon Simple Storage Service (Amazon S3), March 2008. http://aws.amazon.com/s3/.

[38] John Foley. Chief Of The Year: Amazon CTO Werner Vogels , December 2008. http://www.informationweek.com/news/management/interviews/showArticle.jhtml?articleID=212501217.

[39] Crunchies 2008 Award Winners, January 2009. http://crunchies2008.techcrunch.com/.

[40] Larry Dignan. Amazon's cloud computing will surpass its retailing business. http://blogs.zdnet.com/BTL/?p=8471.

[41] Eric Bangeman. 18-month beatdown: Google search crushing Microsoft, Yahoo, August 2008. http://arstechnica.com/microsoft/news/2008/08/18-month-beatdown-google-search-crushing-microsoft-yahoo.ars.

[42] Global Internet Audience Surpasses 1 billion Visitors, According to comScore, January 2009. http://www.comscore.com/press/release.asp?press=2698.

[43] James Urquhart. Is Google App Engine successful?, January 2009. http://news.cnet.com/8301-19413_3-10148273-240.html.

[44] Dare Obasanjo. Google App Engine on the road to becoming useful for building real web applications, February 2009. http://www.25hoursaday.com/weblog/2009/02/09/GoogleAppEngineOnTheRoadToBecomingUsefulForBuildingRealWebApplications.aspx.

[45] John Markoff. I.B.M. Faces a Harsher World in 90's. New York Times, March 1991. http://query.nytimes.com/gst/fullpage.html?res=9D0CE0D7173DF932A15750C0A967958260.

[46] Michael Kanellos and John G. Spooner. IBM's outsider: A look back at Lou. CNet News, February 2002. http://news.cnet.com/2100-1001-828095.html.

[47] IBM Introduces Ready-to-Use Cloud Computing. http://www-03.ibm.com/press/us/en/pressrelease/22613.wss.

[48] James Staten. IBM's Play In Cloud Computing? Listening Carefully, July 2008. http://blogs.forrester.com/it_infrastructure/2008/07/ibms-play-in-cl.html.

[49] James Hamilton. Another Step Forward for Utility Computing, February 2009. http://perspectives.mvdirona.com/2009/02/12/AnotherStepForwardForUtilityComputing.aspx.

[50] Jeremy Reimer. Microsoft tests "pay-as-you-go" software, February 2007. http://arstechnica.com/old/content/2007/02/8907.ars.

[51] Microsoft Office Live. http://www.officelive.com/.

[52] CRM - The Leader In Software-as-a-Service (SaaS) - salesforce.com CRM - salesforce.com. http://www.salesforce.com/.

[53] Software Top 100 ::: The World's Largest Software Companies, March 2009. http://www.softwaretop100.org/.

[54] SaaS and CRM on demand vendor guide. SearchCRM.com Learning Guide, October 2007. http://searchcrm.techtarget.com/generic/0,295582,sid11_gci1275211,00.html.

[55] Michael Arrington. Salesforce Enters Custom Application Market With Force.com, September 2007. http://www.techcrunch.com/2007/09/13/salesforce-enters-custom-application-market-with-forcecom/.

[56] Rich Miller. Rackspace IPO an Important Sector Indicator, April 2008. http://seekingalpha.com/article/74354-rackspace-ipo-an-important-sector-indicator.

[57] Amazon Elastic Block Store (Amazon EBS). http://aws.amazon.com/ebs/.

[58] Amazon EC2 Instance Types. http://aws.amazon.com/ec2/instance-types/.

[59] Werner Vogels. Expanding the Cloud: Amazon CloudFront, November 2008. http://www.allthingsdistributed.com/2008/11/amazon_cloudfront.html.

[60] Peter Bright. Amazon to take on Akamai with cloud delivery network, September 2008. http://arstechnica.com/old/content/2008/09/amazon-to-take-on-akamai-with-cloud-delivery-network.ars.

[61] Akamai: The Leader in Web Application Acceleration and Performance Management, Streaming Media Services and Content Delivery. http://www.akamai.com/.

[62] Limelight Networks. http://www.limelightnetworks.com/.

[63] Amazon Simple Queue Service (Amazon SQS). http://aws.amazon.com/sqs/.

[64] Garett Rogers. Google App Engine pricing a disappointment. ZDNet: Googling Google Blog, May 2008. http://blogs.zdnet.com/Google/?p=1058.

[65] Google Developer Products Help WhiteHouse.gov Connect With America. Google Code Blog, April 2009. http://google-code-updates.blogspot.com/2009/04/google-developer-products-help.html.

[66] David Chappell. Introducing the Azure Services Platform. David Chappel & Associates Whitepaper (Sponsored by Microsoft Corporation), October 2008. http://www.davidchappell.com/blog/2008/10/introducing-azure-services-platform.html.

[67] Jon Torresdal. Amazon Web Services compared to the Azure Services Platform, December 2008. http://blog.torresdal.net/Trackback.aspx?guid=81adc1a6-376a-420b-938c-4d551e5e9e34.

[68] About - What is the Azure Services Platform? http://www.microsoft.com/azure/whatisazure.mspx.

[69] Live Services. http://dev.live.com/.

[70] MSN Hotmail. http://www.hotmail.com/.

[71] Messenger - Windows Live. http://messenger.live.com.

[72] Live Search. http://www.live.com/.

[73] GoGrid :: Scalable Load-Balanced Windows and Linux Cloud-Server Hosting. http://www.gogrid.com/.

[74] Cloud Storage : Mountable, Infinitely Scalable Storage Solution for your Cloud Servers. http://www.gogrid.com/how-it-works/cloud-storage.php.

[75] FlexiScale - Utility Computing On-Demand. http://www.flexiscale.com/.

[76] FlexiScale FAQ. http://www.flexiscale.com/FlexiScale_FAQ.pdf. v1.7 PHUB.

[77] Network.com. http://www.network.com.

[78] Sun Utility Computing. http://www.sun.com/service/sungrid/index.jsp.

[79] Gavin Clarke. Sun closes 'future' pay-per-use utility computing service. The Register, December 2008. http://www.theregister.co.uk/2008/12/10/sun_closes_cloud/.

[80] Sun's Network.com Renders Computer-Animated Movie "Big Buck Bunny", June 2008. http://www.sun.com/aboutsun/pr/2008-06/sunflash.20080602.1.xml.

[81] Charles Babcock. Sun Unveils Open Cloud Computing Platform. InformationWeek News, March 2009. http://www.informationweek.com/news/services/hosted_apps/showArticle.jhtml?articleID=215901026.

[82] About Mosso. http://www.mosso.com/story.jsp.

[83] Cluster Server, Clustered Hosting, Hosting Space, Cloud Site @ Mosso. http://www.mosso.com/cloud.jsp.

[84] Cloud Storage, CDN, Online Storage, On Demand Storage @ Mosso. http://www.mosso.com/cloudfiles.jsp.

[85] Limelight Content Distribution. http://www.mosso.com/cloudfiles_cdn.jsp.

[86] Cloud Compute, Cloud Server, Cloud Servers, On Demand Server @ Mosso. http://www.mosso.com/cloudservers.jsp.

[87] Engine Yard Solo. http://www.engineyard.com/solo/.

[88] Ruby On Rails. http://rubyonrails.org/.

[89] Heroku – Instant Rails Platform. http://heroku.com/.

[90] Platform as a Service (PaaS) - Powering On-Demand SaaS Development - salesforce.com. http://www.salesforce.com/platform/.

[91] Platform as a Service: Bungee Connect is Platform as a Service. http://www.bungeeconnect.com/.

[92] ENKI - Managed Cloud Computing. http://www.enkiconsulting.net/.

[93] Understanding Computing Utility, Billing. ENKI Whitepaper, August 2008. http://www.enkiconsulting.net/images/stories/EnkiLLC/computing_utility_billing_tutorial.pdf.

[94] Joyent: Welcome to Joyent. http://www.joyent.com/.

[95] Joyent Pricing. http://www.joyent.com/accelerator/pricing/.

[96] Grid Computer Operating System For Web Applications - AppLogic | 3tera. http://www.3tera.com/AppLogic/.

[97] GridLayer. http://www.thegridlayer.com/.

[98] Right Servers. http://www.rightservers.com/packages/.

[99] Jon Udell. 3Tera clarification. InfoWorld Blog, August 2006. http://weblog.infoworld.com/udell/2006/08/31.html.

[100] Web-based Cloud Computing Management Platform by RightScale. http://www.rightscale.com/.

[101] Mary Hayes Weier. Cloud Computing Startup Gets $4.5M Venture Funding. http://www.informationweek.com/news/internet/web2.0/showArticle.jhtml?articleID=207401919.

[102] Engine Yard Vertebra. http://www.engineyard.com/vertebra/.

[103] The Java Virtual Machine Specification. http://java.sun.com/docs/books/jvms/.

[104] 5 Question Interview with Twitter Developer Alex Payne. http://www.radicalbehavior.com/5-question-interview-with-twitter-developer-alex-payne/.

[105] The Python Datastore API - Google App Engine - Google Code. http://code.google.com/appengine/docs/python/datastore/.

[106] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A Distributed Storage System for Structured Data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):1–26, 2008.

[107] New Features for Amazon EC2, March 2009. http://aws.amazon.com/contact-us/new-features-for-amazon-ec2/.

[108] Cloud Computing Interoperability Forum (CCIF). http://www.cloudforum.org/.

[109] CCIF Mission & Goals. http://groups.google.com/group/cloudforum/web/ccif-mission-goals?pli=1.

[110] Google Open Source Blog: Welcome to the Google Open Source Blog, February 2008. http://google-opensource.blogspot.com/2008/02/welcome-to-google-open-source-blog.html.

[111] Tim O'Reilly. Amazon and Open Source. O'Reilly Media – Ask Tim, February 2004. http://oreilly.com/pub/a/oreilly/ask_tim/2004/amazon_0204.html.

[112] David Berlind. Open source: IBM's deadly weapon. ZDNet Tech Update, April 2002. http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2860394,00.html.

[113] Ina Fried. EU closes in on Microsoft penalty. CNET News, August 2003. http://news.cnet.com/2100-1016_3-5060463.html.

[114] ECMA Standard C# - a clever trap set by Microsoft, March 2004. http://rollerweblogger.org/roller/entry/ecma_standard_c_a_clever.

[115] Michael Calore. Microsoft Allegedly Bullies and Bribes to Make Office an International Standard. Wired News, August 2007. http://www.wired.com/software/coolapps/news/2007/08/ooxml_vote.

[116] Ryan Paul. Google launches test of Java on App Engine, April 2009. http://arstechnica.com/open-source/news/2009/04/google-launches-test-of-java-on-app-engine.ars.

[117] Simple small business software, collaboration, CRM: 37signals. http://www.37signals.com/.

[118] Cloud Computing and Vendor Lock-In.
http://www.25hoursaday.com/weblog/2008/10/19/CloudComputingAndVendorLockIn.aspx.

[119] Cloud computing is a trap, warns GNU founder Richard Stallman.
http://www.guardian.co.uk/technology/2008/sep/29/cloud.computing.richard.stallman.

[120] Windows Azure Offers Developers Iron-Clad Lock-in.
http://developers.slashdot.org/article.pl?sid=08/10/31/2243245.

[121] James Hamilton. WTIA: Scaling into the Cloud with Amazon Web Services, March 2009.
http://perspectives.mvdirona.com/2009/03/04/
WTIAScalingIntoTheCloudWithAmazonWebServices.aspx.

[122] Exclusive: Google App Engine ported to Amazon's EC2, April 2008.
http://waxy.org/2008/04/exclusive_google_app_engine_ported_to_amazons_ec2/.

[123] Amazon EC2 SLA, March 2009. http://aws.amazon.com/ec2-sla/.

[124] Frequently Asked Questions | Azure Services Platform. http://www.microsoft.com/azure/faq.mspx.

[125] Phil Wainewright. Google: App Engine 'not fit for business'. ZDNet: Software as a Service blogs, April 2008. http://blogs.zdnet.com/SAAS/?p=489.

[126] GoGrid Cloud Hosting : Service Level Agreement, March 2009.
http://www.gogrid.com/legal/sla.php.

[127] Cloud Sites Service Level Agreement, March 2009. http://www.mosso.com/sla.jsp.

[128] ENKI versus the other guys: Amazon EC2. http://www.enkiconsulting.net/
computing-utility-services-faq/enki-versus-the-other-guys-amazon-ec2.html.

[129] Jon Brodkin. Loss of customer data spurs closure of online storage service 'The Linkup'. Network World News, August 2008. http://www.networkworld.com/news/2008/081108-linkup-failure.html.

[130] Amazon's Elastic Block Store explained. RightScale Blog, August 2008.
http://blog.rightscale.com/2008/08/20/amazon-ebs-explained/.

[131] Simson L. Garfinkel. An Evaluation of Amazon's Grid Computing Services: EC2, S3, and SQS. Technical Report TR-08-07, Center for Research on Computation and Society, School for Engineering and Applied Sciences, Harvard University, 2007.

[132] Google App Engine Billing FAQ, March 2008.
http://code.google.com/appengine/kb/billing.html.

[133] About - Pricing & Licensing | Azure Services Platform.
http://www.microsoft.com/azure/pricing.mspx.

[134] Werner Vogels. Introducing Amazon EC2 Reserved Instances - A way to further reduce IT costs, March 2009.
http://www.allthingsdistributed.com/2009/03/amazon_ec2_reserved_instances.html.

[135] Kenneth Church, Albert Greenberg, and James Hamilton. On Delivering Embarrassingly Distributed Cloud Services. In *Seventh ACM/SIGCOMM Workshop on Hot Topics in Networks (HotNets '08)*, October 2008.

[136] Simon Avery. Patriot Act haunts Google service. http://www.theglobeandmail.com/servlet/
story/RTGAM.20080324.wrgoogle24/BNStory/Technology/home.

[137] Vinnie Mirchandani. A threat to clouds?
http://dealarchitect.typepad.com/deal_architect/2008/07/a-threat-to-clouds.html.

[138] Ephraim Schwartz. The Dangers of Cloud Computing.
http://www.cio.com/article/426214/The_Dangers_of_Cloud_Computing.

[139] Dan Farber. Oracle's Ellison nails cloud computing. CNet News: Outside the Lines, September 2008.
http://news.cnet.com/8301-13953_3-10052188-80.html.